Delft University of Technology
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft Institute of Applied Mathematics

The Heston model with Term Structure: Option Pricing and Calibration

A thsis submitted to the
Delft Institute of Applied Mathematics
in partial fulfillment of the requirements

for the degree

MASTER OF SCIENCE
in
APPLIED MATHEMATICS

by

Thomas van der Zwaard

Delft, the Netherlands
August 2016

MSc THESIS APPLIED MATHEMATICS

"The Heston model with Term Structure: Option Pricing and Calibration"

Thomas van der Zwaard

Delft University of Technology

Supervisors

Prof.dr.ir. C.W. Oosterlee

Dr. J. du Toit

Responsible professor

Prof.dr.ir. C.W. Oosterlee

Other thesis committee members

Dr. D. Kurowicka

Dr.ir. L.A. Grzelak

August 2016

Delft, the Netherlands

Abstract

This thesis addresses the calibration of the Heston model with term structure (i.e. with piecewise constant parameters) to a set of European option prices from the FX market. Several option pricing methods are discussed and compared, among which the COS method, Lewis' method and the Andersen QE Monte Carlo scheme. Several modifications are proposed in order to improve the practical usability of the COS method in terms of speed, accuracy and robustness. The calibration of the Heston model with term structure is chosen as a benchmarking test-case for comparing several optimization techniques, that are both open-source as well as from licensed products. The performance the optimizers is measured in terms of speed of the calibration. In addition, a simple hedge test using the calibrated model is used as a secondary performance metric. The combined effort of finding the fastest optimization techniques and fastest pricing method has the potential of speeding up daily FX calibrations performed in many financial institutions.

Keywords: Option pricing, foreign exchange (FX) market, COS method, Heston model with term structure, calibration, optimization, benchmarking, hedging.

Acknowledgements

This thesis has been submitted for the the degree of Master of Science in Applied Mathematics at Delft University of Technology, the Netherlands. The academic supervisor of this thesis was prof.dr.ir. Kees Oosterlee of the Numerical Analysis group at the Delft Institute of Applied Mathematics. The research conducted for this thesis has taken place at the Numerical Algorithms Group (NAG) in the United Kingdom, under the supervision of dr. Jacques du Toit. NAG is a software engineering company that is specialized in numerical engineering, consulting services and High Performance Computing services.

First of all I would like to thank Kees Oosterlee and Jacques du Toit for their guidance and support during the course of this thesis and also for being part of the examination committee. Secondly, I would like to thank dr. Dorota Kurowicka and dr.ir. Lech Grzelak for being part of the examination committee. Thirdly, I would like to thank the people from NAG for the pleasant working environment and for the many fruitful discussions we have had. In particular I would like to thank Jan Fiala for his guidance on the optimization part of this thesis. In addition I would like to thank Alan Bain from BNP Paribas for our conversations and discussions. Furthermore, I would like to express my gratitude towards the anonymous contributor of foreign exchange data that has been used for this thesis. Last but not definitely not least, I would like to thank my family and friends for their support for the duration of my studies.

Thomas van der Zwaard
Delft, August 2016

Contents

List of Abbreviations

AD	Algorithmic differentiation.
ATM	At-the-money.
bp	Basis point.
BF	Butterfly.
BS	Black-Scholes.
cdf	Cumulative distribution function.
cgf	Cumulant generating function.
chf	Characteristic function.
EMG	Exponentially modified Gaussian.
FD	Finite differencing.
FFT	Fast Fourier Transform.
FX	Foreign exchange.
GN	Gauss-Newton.
IG	Inverse Gaussian.
ITM	In-the-money.
KKT	Karush-Kuhn-Tucker.
LICQ	Linear independence constraint qualification.
LM	Levenberg-Marquardt.
MC	Monte Carlo.
mgf	Moment generating function.
NAG	Numerical Algorithms Group.
NIG	Normal inverse Gaussian.
ODE	Ordinary differential equation.
OTM	Out-of-the-money.
PDE	Partial differential equation.
PIDE	Partial integro differential equation.
pdf	Probability density function.
QE	Quadratic-Exponential.
RR	Risk-reversal.
SDE	Stochastic differential equation.
SQP	Sequential Quadratic Programming.
volvol	Volatility of volatility.

Chapter 1

Introduction

For many years already, financial markets have been places in which loads of money has been moved between the numerous players. Examples of the products being traded on those markets are financial securities such as stocks and bonds, commodities such as precious metals and food, and derivative contracts. The latter are contracts on underlying entities such as stock, an index, exchange rates or interest rates. The most common type of derivative contracts are European options, that give the buyer of the contract the right, but not the obligation to buy or sell the underlying at a specified price at a predefined point in time.

Since 1973, the year in which the CBOE (Chicago Board Options Exchange) was established, trading activities on financial markets and the academic interest therein has increased remarkably. The establishment of this organization marked the start of the trade in standardized option contracts through a clearing house, called exchange-traded options. Note that this is not the only type of option contract that is trade-able, there are also the so-called over-the-counter options, which are not listed on any exchange and the trade takes place between two private parties. In both situations a very important question is the process of establishing a fair price of the contract.

1973 is also the year in which Fischer Black and Myron Scholes published their paper [6] on "The Pricing of Options and Corporate Liabilities". They used the stock price, its volatility, the interest rate and the duration of the option contract to derive a mathematical option pricing formula: the renowned Black-Scholes formula. They had to make numerous assumptions in order to obtain this elegant analytic expression for the value of option contracts. Some of these assumptions lead to several disadvantages of their model. Take for instance the assumption of constant volatility over time. This is in direct contradiction with the financial markets, so in this aspect the model fails to mimic the reality. As an attempt to fix this issue, several stochastic volatility models have been developed over the years, of which the Heston model [37] developed in 1993 by Steven Heston is a well-known example.

Every mathematical model that attempts to describe real-life phenomena using mathematics has its shortcomings, and so does the Heston model. Even though the change of volatility over time is now governed by a random process, it is assumed that the way in which this happens between now and one month will be the same as between nine and ten years from now. In other words, the parameters of the volatility process are assumed to be constant over time. In 2003, Mikhailov and Nögel [48] have proposed a version of the Heston model with time-dependent parameters, which assumes parameters to be piece-wise constant. They use a computer algebra package to obtain the solution for the characteristic function, which can then be used for

the pricing of option contracts. This notion of piece-wise constant parameters for the Heston model shall from now on be reffered to as the Heston model with term structure. In 2007, Elices [25] presents a full mathematical derivation of the characteristic function and defines the characteristic function in a recursive way. Note that this has not been the only way in which time-dependence of parameters can be incorporated into the Heston model. For example, in 2006 Piterbarg [58] derives so-called 'effective' time-independent parameters for stochastic volatility models with time-dependent parameters. In essence these 'effective' parameters are a sort of average parameters that imply time-dependence of the model.

After having obtained the characteristic function, several different types of pricing methods can be used for pricing option contracts under the Heston model with term structure. This can be done using either Monte Carlo techniques, Partial Differential Equation (PDE) methods or numerical integration methods. The in 2008 introduced COS method by Fang and Oosterlee [26] is an example of the last category, and uses a Fourier cosine approximation. Another example is the method by Lewis [43], which is based on the so-called fundamental transform, making use of the (generalized) Fourier transform. Lastly, in the class Monte Carlo methods, Andersen's Quadratic-Exponential scheme [2] can be considered as a well-known example. This method aims to improve other well-known schemes such as the Euler scheme.

In financial markets not only European option contracts are being priced, but also more complicated contracts called exotic options. These exotic options are typically valued using Monte Carlo techniques. Before these techniques can be applied, parameter values of a model must be known. To obtain these parameter values, the model is calibrated to a set of market data that contains European option quotes. This calibration procedure involves many evaluations of a pricing technique, so a quick and accurate way of calculating prices is essential. In this thesis the methods described above are compared, together with some newly proposed modifications of the COS method that aim to improve the practical usability of the method in terms of speed, accuracy and robustness.

The way such a calibration is typically performed is using a numerical minimization of an objective function containing both the model prices as well as the market prices. The choice of optimization technique could possibly play a crucial role when it comes to speed: one solver may for example require fewer price evaluations, both resulting in a faster calibration. Therefore it is useful to know which of the available optimization techniques is the most suitable for the calibration of the Heston model with term structure. The comparison of various optimization techniques in terms of calibration performance will be referred to as the process of benchmarking. This calibration is performed on a set of European option quotes from the foreign exchange (FX) market, which is the market for exchanging currencies. It turns out that a speed-up of a factor 12 is a possibility when choosing the most appropriate optimization technique and option pricing method.

In addition to speed of calibration, the correctness of the solution is also very important. Next to the goodness of fit of the model to the market data, an argument from risk analysis point of view might provide useful additional information on the calibration results. After calibration, when exotic options are typically priced, one would not like the possible mispricing of option contracts. A way to measure this exposure in terms of risk can be obtained using a hedging argument.

1.1 Structure of the thesis

After having briefly introduced the ins an outs of the foreign exchange market and its many different styles of quoting option prices in Chapter 2, the data being used for this thesis is presented.
In Chapter 3, the introduction of the Heston model takes place including its pricing PDE and characteristic function. Subsequently, the earlier mentioned term structure will be introduced into the Heston model, and the results from Elices [25] on the characteristic function are summarized and applied to the current situation.
In order to get familiar with several option pricing methods, in Chapter 4 a summary of the COS method [26], Lewis' method [43] and Andersen's Quadratic-Exponential method [2] is given.
In Chapter 5, several newly developed modifications of the COS method are presented, among which the way the truncation range is computed and the amount of terms that have to be used in the Fourier cosine expansion. At the end of this chapter, in Section 5.4, all the different pricing methods including the modifications of the COS method will be compared during a set of numerical tests.
Next, in Chapter 6, the calibration procedure is discussed. Derivatives appear to be crucial during the calibration, so some information on the technique of algorithmic differentiation as an alternative to finite differences is presented. Furthermore, a short introduction into numerical optimization can be found, accompanied by a description of the various optimization techniques used during the benchmarking. Naturally, the results of the benchmarking will finally be presented.
In Chapter 7, as an alternative measure of performance of the calibration next to accuracy and speed, a hedge test is developed and performed on the market data.
Finally in Chapter 8 all findings will be summarized and concluded, and possible topics of future research are proposed.

Chapter 2

Foreign exchange market

The data that has been used during the course of the project, is financial data from the foreign exchange (FX) market. There are some differences between this market and for example the stock market that cannot be left unexposed. Those characteristics of the foreign exchange market that are important for the rest of this thesis will be highlighted in the various sections of this chapter: an introduction to the FX market and its terminology can be found in Section 2.1. Section 2.2 discusses the pricing of option contracts in the FX world under the famous Black-Scholes model [6]. Finally, in Section 2.3, the quoting style within the FX market is discussed and brief description of the data used for this thesis is given.

Note that the reader is required to have some background in the domain of financial mathematics, and option pricing in particular. If the reader wishes to get more acquainted with this theory before continuing with reading this chapter, the books by Higham [39] and Seydel [62] can be consulted.

2.1 Foreign exchange terminology

The foreign exchange market, the market for trading currencies, is the largest financial market in terms of traded volumes. According to the financial news agency Reuters [59], *"Foreign exchange, the world's biggest market, is shrinking"*. During the end of the year 2014 the activity in the FX market was the highest: *"close to $6 trillion changed hands on an average day"*. Despite the fact that this market is shrinking, it is still the biggest, with many large international banks participating.
The FX market is widely used as a *"snapshot of global trade and economic activity"*. But this is not the only use of the market, as many businesses use it to control the risk of future payments in a foreign currency to lose their value in their domestic currency. Another popular feature of the FX market is speculation of many forms.

The relation between currencies is the basis of this market. Two of these currencies together are called a foreign exchange pair, which is represented as CC1CC2 where CC1 and CC2 are the foreign and domestic currencies respectively. The domestic currency is the numeraire in quoting exchange rate. Generally speaking, when S_t is the spot exchange rate at time t for CC1CC2, then at that instant, one is able to buy one unit of CC1 for S_t units of CC2.
For example, a quote of EURUSD 1.7500 in the spot market means that one Euro can be exchanged for 1.7500 US dollars. If the quote would change to EURUSD 1.7900, the Euro has relatively speaking increased in value. This can for example be caused by a weakening in US dollar buying strength, or an increase in Euro buying strength, or perhaps even both. The

inverse of this argument holds when for example the EURUSD quote changes from 1.7500 to 1.7351. Then the Euro has relatively speaking become weaker with respect to the US dollar.

2.2 Options in the FX market

In the stock option market, the value of stock is taken as the underlying asset. Similarly, in the FX option market, the exchange rate will be taken as an underlying. An FX option gives the buyer the right but not the obligation to change a fixed amount of CC1 for another currency CC2 at a predefined fixed exchange rate at a fixed point in time. The predefined fixed exchange rate is also known as the option strike K. The fixed point in the future at which the settlement of the contract will take place is called maturity T. Note that T is always given in years, so for example an option contract with a maturity of 2 months has maturity value $T = 0.16667$ in case no correction is done for the amount of trading days in one year. The options that will be considered in this thesis are European calls and puts. The first gives the holder the right to buy the underlying and the second gives the holder the right to sell the underlying at maturity.

The value of European options in a general option market setting are typically priced using the famous Black-Scholes formula [6]. This framework can be extended to the situation of the foreign exchange market, an example of which can be found in the work by Garman and Kohlhagen [29]. They state that a European option with underlying S and payoff function at maturity:

$$C(S_T, T) = \max\left[\alpha(S_T - K), 0\right],$$

has the price $C(S_0, t_0)$ at time $t = t_0$, which is defined as:

$$C(S_0, t_0) = \alpha\left(e^{-r_f(T-t_0)} S_0 N(\alpha d_1) - e^{-r_d(T-t_0)} K N(\alpha d_2)\right), \tag{2.1}$$

$$d_1 = \frac{\log\left(\frac{S_0}{K}\right) + \left(r_d - r_f + \frac{\sigma^2}{2}\right)(T - t_0)}{\sigma\sqrt{T - t_0}}, \tag{2.2}$$

$$d_2 = d_1 - \sigma\sqrt{T - t_0}. \tag{2.3}$$

Here, S_t denotes the spot exchange rate at time t for the pair CC1CC2, where r_d is the constant risk-free rate of the domestic currency and r_f the constant risk-free rate of the foreign currency. σ stands for the constant volatility rate of the process of S_t. Parameter α denotes the put-call type, where $\alpha = 1$ stands for a call option and $\alpha = -1$ indicates a put option. Finally, $N(\cdot)$ denotes the cumulative normal distribution function. For simplicity, the notation $C_t := C(S_t, t)$ is introduced. Note that the solution (2.1) is similar to Merton's [47] proportional dividend model, where the dividend yield d on the stock is replaced by the foreign interest rate r_f.

It is common practice in FX markets to quote prices in terms of forward prices. Garman and Kohlhagen [29] show that when the forward is defined as:

$$F_t = e^{(r_d - r_f)(T-t)} S_t,$$

a substitution of this definition in option price formula (2.1) results in the following pricing formula:

$$C(F_0, t_0) = \alpha \cdot e^{-r_d(T-t_0)}\left[F_0 N(\alpha d_1) - K N(\alpha d_2)\right], \tag{2.4}$$

$$d_1 = \frac{\log\left(\frac{F_0}{K}\right) + \frac{\sigma^2}{2}(T - t_0)}{\sigma\sqrt{T - t_0}},$$

$$d_2 = d_1 - \sigma\sqrt{T - t_0}.$$

Using the pricing formula (2.4), the option value depends only upon the forward rate F_t and the domestic risk-free rate r_d and it does not depend directly on asset price S_t and foreign risk-free rate r_f. Thus, when the current domestic risk-free interest rate is given, all market information needed is represented by the forward price together with the volatility.

2.3 FX market quotes

The option quotes of the Europeans are given in terms of implied volatility. In the FX market, these implied volatility quotes are not quoted in terms of strike but in terms of option delta's. With the delta and implied volatility of an option in hand, one can extract the corresponding strike value and it becomes straightforward to calculate the option price according to (2.1) or (2.4).

2.3.1 Implied volatility

When looking at Black-Scholes option pricing equation (2.1), the only parameter that cannot be directly observed from the market is volatility σ. Implied volatility is the value for σ in (2.1) that sets $C(S_0, t_0) = C_0^{BS}(S_0, t_0, r_d, r_f, \sigma_{imp}, K)$ equal to the market price of the option C_0^{Market}, where the implied volatility is denoted as σ_{imp}. It is often said that the implied volatility is the wrong number to put in the wrong formula to obtain the right option price. This is due to the many assumptions and features of the Black-Scholes model that are unrealistic when compared to the market reality.
The goal is to choose σ_{imp} such that the market price and the model price are exactly equal, resulting in the following equation:

$$C_0^{Market} - C_0^{BS}(S_0, t_0, r_d, r_f, \sigma_{imp}, K) \quad = \quad 0. \tag{2.5}$$

Equation (2.5) has to be solved numerically for σ_{imp} using a root-finding algorithm such as the bisection method, Newton-Raphson's algorithm, secant method or Brent's method.
When such a method has found a solution for (2.5) in terms of σ_{imp}, it is useful to know if this solution is unique. One can imagine that in case of possible multiple solutions, there could be two (or more) different prices for the same financial product. This would inevitably create an arbitrage opportunity, therefore a unique solution is required. To verify uniqueness, people look at the option vega ν:

$$\nu \quad = \quad \frac{\partial C_0}{\partial \sigma} \quad = \quad \dots \quad = \quad \frac{S_0 e^{-r_f(T-t_0)-\frac{1}{2}d_1^2}\sqrt{T-t_0}}{\sqrt{2\pi}} \quad > \quad 0.$$

In conclusion, due to the strict positivity of the option vega, the option price is a strictly increasing function in the volatility parameter. This means that any solution found to (2.5) is a unique solution.

2.3.2 Delta types

As stated before, the implied volatilities are not quoted in terms of strikes but in terms of option delta's. In the FX market, the delta of an option represents the amount of foreign currency one needs to buy when trying to hedge the risk of a change in the spot exchange rate. There are many different delta quoting conventions in the various FX markets. In this and following paragraphs only the most important information on the delta types will be stated, as this delta quoting is only needed to obtain formula's (2.14) up and till (2.19) for the strikes in terms of the market quotes. In case more background information on the delta conventions is required

by the reader, the book by Clark [13] can be consulted.

For options with a short maturity (up to and including one year), the spot delta is used. For options with a maturity longer than that, the forward delta is used. Furthermore, there are two conventions about which currency is used as payment currency of the option. In case the option price is paid in domestic currency, the delta is called the pips delta. If the option price is paid in foreign currency, the delta is called the premium-adjusted delta. All these conventions together imply there are four different possible formats of delta's available in the market:

$$\Delta_{pips,S} = \frac{\partial C_0}{\partial S_0} = \alpha e^{-r_f(T-t_0)} N(\alpha d_1), \tag{2.6}$$

$$\Delta_{pips,F} = e^{r_f(T-t_0)} \frac{\partial C_0}{\partial S_0} = \alpha N(\alpha d_1), \tag{2.7}$$

$$\Delta_{pa,S} = \Delta_{pips,S} - \frac{C_0}{S_0} = \alpha \frac{K}{S_0} e^{-r_d(T-t_0)} N(\alpha d_2), \tag{2.8}$$

$$\Delta_{pa,F} = e^{r_f(T-t_0)} \left[\Delta_{pips,S} - \frac{C_0}{S_0} \right] = \alpha \frac{K}{S_0} e^{-(r_d - r_f)(T-t_0)} N(\alpha d_2). \tag{2.9}$$

The underlying currency pair determines which type of delta quoting is used. Once it is known which delta convention has to be applied, the strikes can be calculated using the delta's.

2.3.3 ATM straddle, risk-reversal and butterfly

It is said that the implied volatilities from the market are quoted in terms of delta's but that is not all there is to it. The quotes are in terms of three option strategies: the at-the-money (ATM) straddle, the risk-reversal (RR) and the butterfly (BF). The latter two can be in terms of different delta's, e.g. 25-Δ RR or 10-Δ BF.

The ATM straddle consists of a long position in both a call and a put option (with the same maturity T and strike K), such that the delta of this strategy equals zero:

$$\Delta_C(\sigma_{ATM}) + \Delta_P(\sigma_{ATM}) = 0. \tag{2.10}$$

Thus, the delta of the call has the opposite sign of the put delta. Hence from equations (2.6) till (2.9) the relation $N(d_1) = N(d_2) = \frac{1}{2}$ can be obtained. This means that the ATM strikes can be calculated for the different different delta types. These ATM strikes are given in Table 2.1.

Delta type	ATM Delta	ATM Strike
$\Delta_{pips,S}$	$\frac{1}{2}\alpha e^{-r_f(T-t_0)}$	$Fe^{\frac{1}{2}\sigma^2(T-t_0)}$
$\Delta_{pips,F}$	$\frac{1}{2}\alpha$	$Fe^{\frac{1}{2}\sigma^2(T-t_0)}$
$\Delta_{pa,S}$	$\frac{1}{2}\alpha e^{-\left(r_f+\frac{1}{2}\sigma^2\right)(T-t_0)}$	$Fe^{-\frac{1}{2}\sigma^2(T-t_0)}$
$\Delta_{pa,F}$	$\frac{1}{2}\alpha e^{-\frac{1}{2}\sigma^2(T-t_0)}$	$Fe^{-\frac{1}{2}\sigma^2(T-t_0)}$

Table 2.1: ATM Delta and strike for different delta quoting styles.

For now assume that the 25-Δ RR and 25-Δ BF are quoted, but the results that follow can naturally be extended for any delta.
The 25-Δ risk-reversal strategy consists of a long position in a 25-Δ call and a short position in a 25-Δ put with the same absolute value of delta. Thus:

$$\sigma_{25\Delta RR} = \sigma_{25\Delta Call} - \sigma_{25\Delta Put}. \tag{2.11}$$

The risk-reversal captures the slope of the implied volatility smile, so it is a measure of the volatility skew.
The 25-Δ butterfly strategy consists of half a long call position, half a long put position and a short ATM position. Thus:

$$\sigma_{25\Delta BF} = \frac{1}{2}(\sigma_{25\Delta Call} + \sigma_{25\Delta Put}) - \sigma_{ATM}. \tag{2.12}$$

The butterfly captures the curvature of the implied volatility smile.
Equations (2.10), (2.11) and (2.12) can be combined to extract the implied volatilities of the 25-Δ call and put:

$$\begin{aligned}\sigma_{25\Delta Call} &= \sigma_{ATM} + \sigma_{25\Delta BF} + \frac{1}{2}\sigma_{25\Delta RR},\\ \sigma_{25\Delta Put} &= \sigma_{ATM} + \sigma_{25\Delta BF} - \frac{1}{2}\sigma_{25\Delta RR}.\end{aligned}$$

In conclusion, from the market quotes it is possible to obtain a σ_{ATM}, $\sigma_{25\Delta Call}$ and $\sigma_{25\Delta Put}$. Plugging these values into (2.1) will give the corresponding market prices.

2.3.4 Market data

For this thesis, datasets of the currency pair EURUSD from 11-3-2011 till 19-5-2016 will be used. See Figure 2.1 for a plot of the spot exchange rate per day. This resulted in a total of 1355 different datasets that could be used for testing purposes. The datasets contain information about the initial spot exchange rate S_0 and for each different maturity a EUR rate (r_f), USD rate (r_d), 25-Δ and 10-Δ RR quotes ($\sigma_{25\Delta RR}$ and $\sigma_{10\Delta RR}$), 25-Δ and 10-Δ BF quotes ($\sigma_{25\Delta BF}$ and $\sigma_{10\Delta BF}$) and ATM quote (σ_{ATM}). For each date there are 7 different maturities with option quotes: 2m, 3m, 6m, 1y, 2y, 3y and 5y.

In this dataset, the so-called premium currency is the US dollar, which means that the premium is paid in the domestic currency and therefore the pips delta is used. For maturities up to and including one year the spot delta will be used and for maturities larger than that the forward delta.

$$\begin{aligned}\Delta_{pips,S} &= \alpha e^{-r_f(T-t_0)} N(\alpha d_1),\\ &= \alpha e^{-r_f(T-t_0)} N\left(\alpha \frac{\log\left(\frac{F_0}{K}\right) + \frac{\sigma^2}{2}(T-t_0)}{\sigma\sqrt{T-t_0}}\right),\\ \Rightarrow \qquad K &= F_0 \exp\left\{-\alpha N^{-1}\left(\alpha \Delta_{pips,S} e^{r_f(T-t_0)}\right)\sigma\sqrt{T-t_0} + \frac{1}{2}\sigma^2(T-t_0)\right\}. \end{aligned}\tag{2.13}$$

Note that $N^{-1}(\cdot)$ denotes the inverse of the normal cumulative distribution function (cdf). Using (2.13), implied volatility quotes σ_{ATM}, $\sigma_{25\Delta Call}$, $\sigma_{25\Delta Put}$ and their corresponding delta values, the following equations give the strike values for the spot delta strikes:

$$K_{ATM,S} = F_0 \exp\left\{\frac{1}{2}\sigma_{ATM}^2(T-t_0)\right\}, \tag{2.14}$$

$$K_{25\Delta Call,S} = F_0 \exp\left\{-N^{-1}\left(\frac{1}{4}e^{r_f(T-t_0)}\right)\sigma\sqrt{T-t_0} + \frac{1}{2}\sigma_{25\Delta Call}^2(T-t_0)\right\}, \tag{2.15}$$

$$K_{25\Delta Put,S} = F_0 \exp\left\{N^{-1}\left(\frac{1}{4}e^{r_f(T-t_0)}\right)\sigma\sqrt{T-t_0} + \frac{1}{2}\sigma_{25\Delta Put}^2(T-t_0)\right\}. \tag{2.16}$$

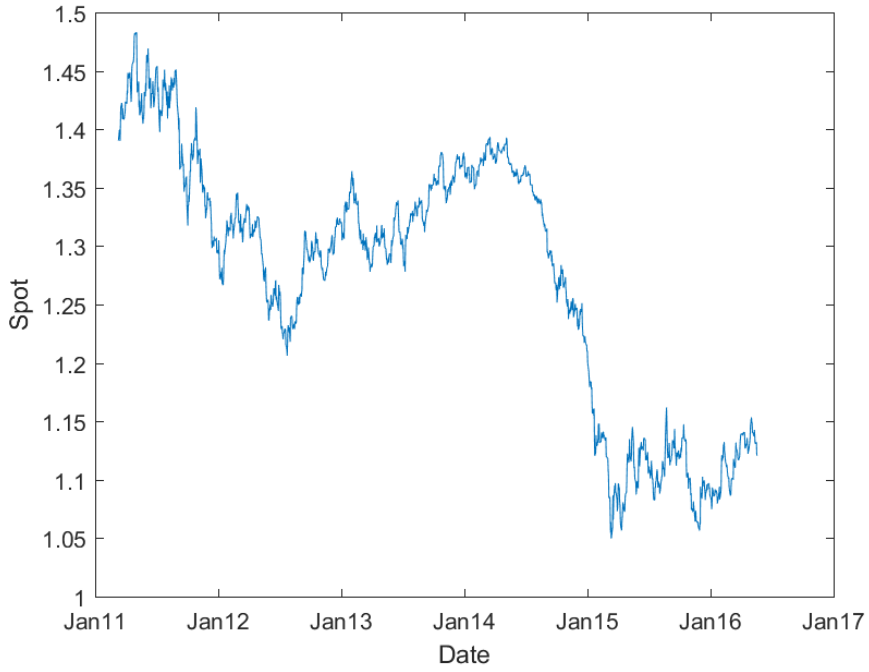


Figure 2.1: EURUSD spot exchange rate from 11-3-2011 till 19-5-2016.

In a similar fashion, using the pips forward delta

$$\Delta_{pips,F} = \alpha N(\alpha d_1),$$

obtain the following results for the strike values corresponding to the forward delta quotes:

$$K_{ATM,F} = F_0 \exp\left\{\frac{1}{2}\sigma_{ATM}^2(T-t_0)\right\}, \tag{2.17}$$

$$K_{25\Delta Call,F} = F_0 \exp\left\{-N^{-1}\left(\frac{1}{4}\right)\sigma\sqrt{T-t_0} + \frac{1}{2}\sigma_{25\Delta Call}^2(T-t_0)\right\}, \tag{2.18}$$

$$K_{25\Delta Put,F} = F_0 \exp\left\{N^{-1}\left(\frac{1}{4}\right)\sigma\sqrt{T-t_0} + \frac{1}{2}\sigma_{25\Delta Put}^2(T-t_0)\right\}. \tag{2.19}$$

Chapter 3

Heston Model

One of the known shortcomings of the Black-Scholes model for option pricing [6] is the constant volatility. This clearly comes forward when looking at a market implied volatility curve. Take for example data from the S & P 500 Index at a fixed point in time and plot the implied volatilities as a function of the strikes given in the market. According to the Black-Scholes model this implied volatility curve should be a straight line. Generally, the implied volatility curve from the market not a horizontal line. Therefore, the market mechanism does not work with a constant volatility but a volatility that varies over time. This effect is called volatility smile.

There are several ways to model the dependence of volatility on time. Popular choices to achieve this time dependence are local volatility models [21, 24] and stochastic volatility models, or a combination of the two. Examples of stochastic volatility models are the SABR model [34], the CEV model [18, 19], the GARCH model [7], the 3/2 model [38] and the Heston model [37]. In this thesis we will focus on the latter, which was introduced by Steven Heston in 1993 [37].

The reason the Heston model is chosen here can be summarized by the following quote by Gatheral [30]: *"although the dynamics of the Heston model are not realistic, with appropriate choices of parameters, all stochastic volatility models generate roughly the same shape of implied volatility surface ... Given the relative cheapness of Heston computations, it's easy to see why the model is so popular."*

In Section 3.1 a full description of the Heston model will be given and all necessary results and properties needed later on will be presented. In Section 3.2 a term structure will be added to the Heston model, where each parameter will be piecewise constant over the different terms in the term structure. For this new model the same results and properties as for the single-term model will be given.

3.1 Heston stochastic volatility model

In this section, the Heston model will be formulated. Furthermore, a pricing PDE is derived, and an analytic expression for the characteristic function (chf) is given.

3.1.1 Formulation of the Heston model

In equation (3.1), the formulation of Heston's stochastic volatility model [37] is presented. Here, S_t is the FX spot price at time t, r_d and r_f are the risk-free rates is domestic and foreign currency

respectively:

$$\begin{cases} dS_t &= (r_d - r_f)S_t dt + \sqrt{v_t} S_t dW_t^1, \\ dv_t &= \kappa(\theta - v_t)dt + \gamma\sqrt{v_t} dW_t^2, \\ dW_t^1 \cdot dW_t^2 &= \rho dt. \end{cases} \tag{3.1}$$

κ is the speed of mean reversion, θ is the level of mean reversion, γ is the volatility of the variance process (often called the vol-vol), ρ is the correlation between Brownian motions (BM) W_t^1 and W_t^2. Note that all the parameters that are mentioned are time-independent, so constant over the entire timespan that is considered. The variance process is known as the square-root process introduced by Cox, Ingersoll and Ross [16].

Given the nature of the Heston dynamics it is useful to ensure that the variance process will be non-negative with positive probability. The Feller condition [27] in (3.2) makes sure that the upward drift in the variance process is sufficiently large to ensure that the process cannot reach zero:

$$2\kappa\theta \quad > \quad \gamma^2. \tag{3.2}$$

It is common practice to rewrite the model (3.1) in terms of the logarithm of the asset spot price:

$$x_t = \log(S_t).$$

Using the above and substituting $\mu = r_d - r_f$ one gets the following model dynamics:

$$\begin{cases} dx_t &= (\mu - \frac{1}{2}v_t)dt + \sqrt{v_t} dW_t^1, \\ dv_t &= \kappa(\theta - v_t)dt + \gamma\sqrt{v_t} dW_t^2, \\ dW_t^1 \cdot dW_t^2 &= \rho dt. \end{cases} \tag{3.3}$$

This is done to achieve an affine model formulation. What this is and why it is needed will be explained later on in this section.

The model formulation from equation (3.3) is not the model that shall be used in this thesis. As proposed in [54], the Heston formulation of equation (3.4) shall be used from now on:

$$\begin{cases} dS_t &= \mu S_t dt + \sigma\sqrt{v_t} S_t dW_t^1, \\ dv_t &= \lambda(1 - v_t)dt + \alpha\sqrt{v_t} dW_t^2, \\ dW_t^1 \cdot dW_t^2 &= \rho dt. \end{cases} \tag{3.4}$$

The difference between the models in (3.1) and (3.4) has primarily to do with the variance process v_t. If one scales and normalizes (setting long-term variance θ equal to 1) the variance process from (3.1) the result will be the variance process from (3.4). The reason this is done is to make the variance process so to say 'dimensionless', meaning that the effect of the variance process on the evolution of the asset spot price is scaled by parameter σ. Thus, this parameter σ will tell how much the process of S_t is influenced by the change in variance over time.

The Feller condition (3.5) for the model formulation (3.4) is similar to the Feller condition (3.2) for the original model formulation (3.1):

$$2\lambda \;>\; \alpha^2. \tag{3.5}$$

Rewriting the model from (3.4) in terms of the log asset-price results in the model formulation in (3.6):

$$\begin{cases} dx_t &= (\mu - \frac{1}{2}\sigma^2 v_t)dt + \sigma\sqrt{v_t}dW_t^1, \\ dv_t &= \lambda(1-v_t)dt + \alpha\sqrt{v_t}dW_t^2, \\ dW_t^1 \cdot dW_t^2 &= \rho dt. \end{cases} \tag{3.6}$$

With help of the multi-dimensional Ito lemma and the martingale pricing approach indeed, the Heston pricing PDE is derived for the new Heston model formulation from (3.4) and (3.6). The full derivation of the pricing PDE can be found in Appendix A.1. The final result is stated below in (3.7):

$$\begin{aligned} -\frac{\partial C_t}{\partial \tau} + \left(\mu - \frac{1}{2}\sigma^2 v_t\right)\frac{\partial C_t}{\partial x_t} + \lambda(1-v_t)\frac{\partial C_t}{\partial v_t} + \frac{1}{2}\sigma^2 v_t \frac{\partial^2 C_t}{\partial x_t^2} \\ + \sigma\rho\alpha v_t \frac{\partial^2 C_t}{\partial x_t \partial v_t} + \frac{1}{2}\alpha^2 v_t \frac{\partial^2 C_t}{\partial v_t^2} - \mu C_t \;=\; 0, \end{aligned} \tag{3.7}$$

with C_t the option value at time t.

3.1.2 Characteristic function for option pricing

For many option pricing methods, Fourier techniques are used nowadays. This involves knowing the characteristic function (chf) of the process. For the Heston model there exists an analytic expression of the chf. This implies that European option values can be calculated in a relatively easy and computationally efficient way using for example Fourier inversion techniques. The joint chf of the process stated in (3.6) over time interval $[0, T]$ is defined as:

$$\phi_{0T}(X, V|x_0, v_0) \;=\; \mathbb{E}^{\mathbb{Q}}\left[e^{iXx_T + iVv_T}|x_0, v_0\right].$$

As one might imagine, this formula cannot be used straight away for option pricing because this is a joint chf, which corresponds to a joint density. European option pricing is always done using the marginal density of the spot asset price or in this case the logarithm of the spot asset price. This implies that option pricing techniques involving chf, require that the marginal chf of the log-asset price is known:

$$\phi_{0T}(X|x_0, v_0) \;=\; \mathbb{E}^{\mathbb{Q}}\left[e^{iXx_T}|x_0, v_0\right] \;=\; \phi_{0T}(X, V|x_0, v_0)|_{V=0}.$$

In his article, Heston gives an analytic expression for this marginal chf. The difference with the approach taken in this section is that here an analytic expression of the joint chf is required, after which the marginal chf can be retrieved by evaluating the joint version at $V = 0$. This is done because of the extension of the Heston model by including a term structure which will follow later on in this section. The joint chf of the log-asset price corresponding to Heston formulation (3.3) is given by Mikhailov and Nögel [48] and Elices [25], where the last one provides a proper

mathematical derivation.

Since the model formulation in this thesis is slightly different, and for purpose of full understanding, a derivation of the chf corresponding to dynamics (3.6) is presented here. The famous Feynman-Kac formula tells that given a final condition for C_t (so in this case an initial condition due to the transformation $\tau = T - t$), the value C_t that satisfies the Heston pricing PDE (A.4) is the unique solution of risk-neutral pricing formula:

$$\begin{aligned} C(x,t) &= e^{-\mu\tau}\mathbb{E}^{\mathbb{Q}}\left[C(y,T)|x\right], \\ &= e^{-\mu\tau}\int_{\mathbb{R}} C(y,T)f(y|x)\ dy. \end{aligned}$$

Unfortunately, the probability density function (pdf) $f(y|x)$ is not known in most cases. On the other hand, often an analytic expression of the chf is known, and it relates to the pdf by a Fourier transformation. Duffie, Pan and Singleton [23] have designed a framework to find a chf by probing a solution.
A requirement for using this method is that the process at hand should be in the affine diffusion class. In general, consider the following system of SDE's:

$$d\mathbf{X}_t = \underline{\mu}(\mathbf{X}_t)dt + \underline{\sigma}(\mathbf{X}_t)d\tilde{\mathbf{W}}_t,$$

where $\mathbf{X}_t = \left[X_t^1, X_t^2, \ldots, X_t^n\right]^T$ is the vector indicating the different processes and $\tilde{\mathbf{W}}_t = \left[\tilde{W}_t^1, \tilde{W}_t^2, \ldots, \tilde{W}_t^n\right]^T$ is a vector of independent Brownian motions. If the following equations hold, the processes are in the affine diffusion class:

$$\begin{aligned} \underline{\mu}(\mathbf{X}_t) &= a_0 + a_1\mathbf{X}_t, \\ \underline{\sigma}(\mathbf{X}_t)\underline{\sigma}(\mathbf{X}_t)^T &= (c_0)_{ij} + (c_1)_{ij}^T\mathbf{X}_t, \\ r(\mathbf{X}_t) &= r_0 + r_1^T\mathbf{X}_t, \end{aligned}$$

were $(a_0, a_1) \in \mathbb{R}^n \times \mathbb{R}^{n\times n}$, $(c_0, c_1) \in \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times n\times n}$ and $(r_0, r_1) \in \mathbb{R} \times \mathbb{R}^n$.

Duffie, Pan and Singleton state that for affine diffusion processes, the discounted chf as in (3.8) with boundary condition (3.9) at $t = T$ has a solution of the form (3.10):

$$\phi_{0T}(X,V|x_0,v_0) = \mathbb{E}^{\mathbb{Q}}\left[e^{-\int_t^T \mu_s\ ds}\cdot e^{iXx_T+iVv_T}\middle|\ x_0, v_0\right], \tag{3.8}$$

$$\phi_{TT}(X,V|x_T,v_T) = e^{iX\cdot x_T+iV\cdot v_T}, \tag{3.9}$$

$$\phi_{0T}(X,V|x_0,v_0) = e^{C(X,V,\tau)+D_1(X,V,\tau)x_0+D_2(X,V,\tau)v_0}. \tag{3.10}$$

The goal is to find coefficients $C(X,V,\tau)$ (not to be confused with the $C(x,t)$ that stands for the price of an option), $D_1(X,V,\tau)$ and $D_2(X,V,\tau)$. These coefficients should satisfy the following system of Riccati-type ODE's where $\mathbf{D}(X,V,\tau) = [D_1(X,V,\tau), D_2(X,V,\tau)]^T$:

$$\begin{aligned} \frac{dC(X,V,\tau)}{d\tau} &= -r_0 + \mathbf{D}a_0 + \frac{1}{2}\mathbf{D}^T c_0\mathbf{D}, \\ \frac{d\mathbf{D}(X,V,\tau)}{d\tau} &= -r_0 + a_1^T\mathbf{D} + \frac{1}{2}\mathbf{D}^T c_1\mathbf{D}. \end{aligned}$$

Solving these Riccati-type ODE's results in the required analytic expression of the chf.

Using matrix-vector notation for the Heston model as in (A.1), it is easy to see that the affine diffusion assumption is satisfied and that the following Riccati-type ODE's have to be satisfied:

$$\begin{aligned}
\frac{\partial C}{\partial \tau} &= \mu D_1 + \lambda D_2, && (3.11)\\
\frac{\partial D_1}{\partial \tau} &= 0, && (3.12)\\
\frac{\partial D_2}{\partial \tau} &= -\frac{1}{2}\sigma^2 D_1 - \lambda D_2 + \frac{1}{2}\sigma^2 D_1^2 + \sigma\alpha\rho D_1 D_2 + \frac{1}{2}\alpha^2 D_2^2, && (3.13)
\end{aligned}$$

with initial conditions $C(X,V,0) = 0$, $D_1(X,V,0) = iX$, $D_2(X,V,0) = iV$. Solution to (3.12) is trivial:

$$D_1(X,V,\tau) = iX.$$

This solution can be substituted into (3.11) and (3.13) to get the following system of Riccati-type ODE's:

$$\begin{aligned}
\frac{\partial C}{\partial \tau} &= \mu i X + \lambda D_2, && (3.14)\\
\frac{\partial D_2}{\partial \tau} &= -\frac{1}{2}\sigma^2 X(i+X) - (\lambda - \sigma\alpha\rho i X) D_2 + \frac{1}{2}\alpha^2 D_2^2. && (3.15)
\end{aligned}$$

Solving these ODE's by separation of variables and plugging them into the joint chf results in the following:

$$\begin{aligned}
\phi_{0T}(X,V|x_0,v_0) &= e^{C(X,V,T)+D_2(X,V,T)v_0+iXx_0}, && (3.16)\\
C(X,V,\tau) &= i\mu X\tau + \frac{\lambda}{\alpha^2}\left(-2\log\left(\frac{1-\tilde{g}e^{-d\tau}}{1-\tilde{g}}\right) + (\lambda - \rho\sigma\alpha i X - d)\tau\right) && (3.17)\\
&\quad + C(X,V,0),\\
D_2(X,V,\tau) &= \frac{\lambda - \rho\sigma\alpha i X + d}{\alpha^2}\left(\frac{g-\tilde{g}e^{-d\tau}}{1-\tilde{g}e^{-d\tau}}\right), && (3.18)\\
\tilde{g} &= \frac{\lambda - \rho\sigma\alpha i X - d - D_2(X,V,0)\alpha^2}{\lambda - \rho\sigma\alpha i X + d - D_2(X,V,0)\alpha^2},\\
g &= \frac{\lambda - \rho\sigma\alpha i X - d}{\lambda - \rho\sigma\alpha i X + d},\\
d &= \sqrt{(\lambda - \rho\sigma\alpha i X)^2 + \alpha^2\sigma^2 X(i+X)},\\
C(X,V,0) &= 0,\\
D_2(X,V,0) &= iV.
\end{aligned}$$

This formulation is not the only one going round in literature. For now the fact that a formulation including the market price of volatility risk is often incorporated is neglected. The main difference of interest is the fact that the complex root d has two possible values which have opposite signs. The formulation by Elices picks the main value of the complex square root, whereas Heston picked the other one. Albrecher et al [1] argue that the formulation that does not consider the main value of the complex square root could cause numerical issues when pricing European options using Fourier techniques. The chf belonging to this approach is unstable under certain conditions. The main issue is that when the time to maturity increases, nearly any combination of parameters in the Heston model leads to instabilities. On the other hand, when choosing the main value of the complex square root as Elices did, the chf is proven to be stable under the full parameter space.

3.2 Heston model with term structure

Even though the Heston model deals with one of the many pitfalls of the famous Black-Scholes model, the Heston model still has some shortcomings itself. The main issue is that all parameters are assumed to be constant over the entire timespan. There is a good reason for this though. Because the chf is a solution of a Riccati-type ODE, most parameters have to be constant in order to still end up with an analytic expression for the chf.
Mikhailov and Nögel [48] propose an approach that uses piecewise constant parameters over different time intervals. So the total timespan is cut into smaller time intervals, over which the parameters are constant. This results in a similar Heston model formulation as in (3.6), but now each parameter depends on time t:

$$\begin{cases} dx_t &= (\mu_t - \frac{1}{2}\sigma_t^2 v_t)dt + \sigma_t\sqrt{v_t}dW_t^1, \\ dv_t &= \lambda_t(1-v_t)dt + \alpha_t\sqrt{v_t}dW_t^2, \\ dW_t^1 \cdot dW_t^2 &= \rho_t dt. \end{cases} \tag{3.19}$$

The coefficients $C(X,V,\tau)$, $D_1(X,V,\tau)$ and $D_2(X,V,\tau)$ of the chf over each time period are calculated by taking the coefficients of the previous time interval as initial conditions for the new time interval.
Another approach is proposed by Elices [25], in which the chf can be computed in a recursive way. This is preferred when making an implementation of the model. Elices first provides a general framework which is then applied to the Heston model, which will be summarized below.

Consider a general N-dimensional Markov process $\mathbf{x}(t) = [x_1(t), \ldots, x_N(t)]^T$. It is obvious that the interest goes to the chf of this N-dimensional process, which is the Fourier transform of the pdf:

$$\phi_{uv}(\mathbf{X}|\mathbf{x}_u) \;=\; \mathbb{E}\left[e^{\mathbf{X}\cdot\mathbf{x}_v}\right] \;=\; \int_{\mathbb{R}^N} e^{\mathbf{X}\cdot\mathbf{x}_v} f_{uv}(\mathbf{X}|\mathbf{x}_u)\; d\mathbf{x}.$$

$\phi_{uv}(\mathbf{X}|\mathbf{x}_u)$ and $f_{uv}(\mathbf{X}|\mathbf{x}_u)$ are respectively the chf and pdf of the joint distribution of the process $\mathbf{x}_v = \mathbf{x}(t_v)$ at time t_v, conditional on the initial value $\mathbf{x}_u = \mathbf{x}(t_u)$ at time t_u. Note that the notation $\mathbf{X}\cdot\mathbf{x}_v$ refers to the inner product between the two vectors $\mathbf{X} = [X_1, \ldots, X_N]^T$ and $\mathbf{x}_v$.

Similar to what is done in Section 3.1.2, the following chf's are considered with exponents linear in stochastic processes $\mathbf{x}_v$ at time t_v:

$$\phi_{uv}(\mathbf{X}|\mathbf{x}_u) \;=\; \exp(C_{uv}(\mathbf{X}) + \mathbf{D}_{uv}\cdot\mathbf{x}_u).$$

Here, the functions $C_{uv}(\mathbf{X})$ and $\mathbf{D}_{uv} = [D_{uv,1}(\mathbf{X}), \ldots, D_{uv,N}(\mathbf{X})]^T$ depend both on $\mathbf{X}$ as well as on parameters of the model under consideration in time interval $[t_u, t_v]$.

Now consider the case in which the time range of interest $[0, t_v]$ will be split up into two subintervals that do not overlap: $[0, t_u]$ and $[t_u, t_v]$ such that $0 < t_u < t_v$. On the first interval, $[0, t_u]$, the process is described by the chf $\phi_{0u}(\mathbf{X}|\mathbf{x}_0)$ conditional on $\mathbf{x}_0$. Similarly, on the second interval, $[t_u, t_v]$, the process is described by the chf $\phi_{uv}(\mathbf{X}|\mathbf{x}_u)$ conditional on $\mathbf{x}_u$. Given that the two chf's $\phi_{0u}(\mathbf{X}|\mathbf{x}_0)$ and $\phi_{uv}(\mathbf{X}|\mathbf{x}_u)$ are known, the goal is to find the joint chf $\phi_{0v}(\mathbf{X}|\mathbf{x}_0)$ of the joint distribution at time t_v conditional on $\mathbf{x}_0$ using these two known chf's. Elices has shown that the following holds:

$$\phi_{0v}(\mathbf{X}|\mathbf{x}_0) \;=\; \exp(C_{0v}(\mathbf{X}) + \mathbf{D}_{0v}\cdot\mathbf{x}_0),$$

where

$$\begin{aligned} C_{0v}(\mathbf{X}) &= C_{uv}(\mathbf{X}) + C_{0u}\left(\frac{1}{i}\mathbf{D}_{uv}(\mathbf{X})\right), \\ \mathbf{D}_{0v}(\mathbf{X}) &= \mathbf{D}_{0u}\left(\frac{1}{i}\mathbf{D}_{uv}(\mathbf{X})\right). \end{aligned}$$

Of course this concept can be extended by cutting the time interval of interest into M non-overlapping time intervals. The outcome of that will result in the same formulation, only more recursive steps have to be taken before the chf $\phi_{0,M}$ is obtained.

As mentioned above, this general framework will now be used in the context of the Heston model. This means that for the time interval $[t_u, t_v]$ one has $\mathbf{X} = [X, V]^T$, $\mathbf{x}_u = [x(t_u), v(t_u)]^T = [x_u, v_u]^T$, and

$$\begin{aligned} \phi_{uv}(\mathbf{X}|\mathbf{x}_u) &= \exp(C_{uv}(\mathbf{X}) + \mathbf{D}_{uv} \cdot \mathbf{x}_0), \\ &= \exp(C_{uv}(\mathbf{X}) + D_{uv,1}(\mathbf{X})x_u + D_{uv,2}(\mathbf{X})v_u). \end{aligned} \tag{3.20}$$

where $C_{uv}(\mathbf{X})$ and $D_{uv,2}(\mathbf{X})$ are given by equations (3.17) and (3.18) respectively where time-indexes are added to all the parameters. Furthermore, $D_{uv,1}(\mathbf{X}) = iX$ and $\tau = t_v - t_u$. The variable X corresponds to the log-asset price and V to the variance process. If time dependent parameters across several periods are considered, a similar result as above holds for obtaining the joint chf (3.21) from 0 to any time:

$$\phi_{0v}(X, V|x_0, v_0) = \exp(C_{0v}(X, V) + D_{0v,2}(X, V)v_0 + D_{0v,1}(X, V)x_0), \tag{3.21}$$

where

$$\begin{aligned} C_{0v}(X, V) &= C_{uv}(X, V) + C_{0u}\left(X, \frac{1}{i}D_{uv,2}(X, V)\right), \\ D_{0v,2}(X, V) &= D_{0u,2}\left(X, \frac{1}{i}D_{uv,2}(X, V)\right), \\ D_{0v,1}(X, V) &= iX. \end{aligned}$$

Note that $C_{uv}(X, V)$ and $D_{uv,2}(X, V)$ are defined in respectively (3.17) and (3.18). Once again, the marginal chf of the log-asset price is obtained by evaluation the joint chf at $V = 0$.

At this moment, with model dynamics, pricing PDE and chf in hand, it is possible to start the pricing of option contracts under the assumption that the market can be modelled by the Heston model with term structure. Several methods for pricing these option contracts are stated and explained in chapter 4.

Chapter 4

Pricing methods

Many methods exists for pricing options. There are methods that give exact solutions or semi-exact solutions, but unfortunately this is only possible in a limited amount of cases. In order to price all products that need to be priced, numerical methods exist. These methods can be classified into different categories: Monte Carlo techniques, Partial (Integro) Differential Equation (PIDE) methods and finally numerical integration methods. Since the numerical integration methods are often used for calibration processes, this is the domain of methods that is of interest for this thesis.

All these numerical integration methods are based on a transformation into the Fourier domain. These Fourier-based algorithms exploit the fact that even though for complicated models there is usually no analytic expression for the pdf available, it is often possible to derive a (semi-)analytic expression of the corresponding chf. Using the Fast Fourier Transform (FFT), integration can be done at a computational complexity of order $N \log_2 N$. Here N represents the number of integration points used in the FFT method. A classic example of this FFT application is the Carr-Madan method [12]. The order of complexity is already a good improvement compared to the order N^2 of arithmetic operations without cleverly calculating the density.

Three option pricing methods are described in this chapter. Firstly, in Section 4.1, the COS method by Fang and Oosterlee [26] will be summarized. This Fourier-based method is an alternative to the FFT-based methods. This method is able to improve the speed of computation for European options and some exotic options. The basis of this method lies in the use of Fourier-cosine expansions.
Secondly, in Section 4.2, the pricing method proposed by Lewis [43] will be discussed. This method is based on the so-called fundamental transform, making use of the (generalized) Fourier transform.
The last method that has been used is the Monte Carlo method by Andersen [2]. This method is called Andersen's Quadratic-Exponential scheme and aims to be an improvement to other well-known schemes such as the Euler scheme. This method is of less importance for this thesis and therefore a description of this method can be found in Appendix B.

4.1 COS method

An elaborate summary of the COS method [26] is presented in this section, which will then be applied to the Heston model with term structure.

4.1.1 Fourier integrals and cosine series

Generally speaking, when involved in European option pricing with numerical integration techniques, the risk-neural valuation formula (4.1) is the basis of most methods:

$$C(x,t_0) \;=\; e^{-r_d\tau}\mathbb{E}^{\mathbb{Q}}\left[C(y,T)|x\right], \;=\; e^{-r_d\tau}\int_{\mathbb{R}} C(y,T)f(y|x)\;dy. \tag{4.1}$$

Here, C is the value of the option, $\tau = T-t_0$, $\mathbb{E}^{\mathbb{Q}}[\cdot]$ is the expectation under risk-neutral measure $\mathbb{Q}$. x and y are two state variables at respectively times $t=t_0$ and $t=T$. $f(y|x)$ is the pdf of y conditional on x. Finally, r_d denotes the domestic risk-free rate.

The goal is to find a (semi-)analytic expression for the chf since it is known that the chf $\phi(\cdot)$ and pdf $f(\cdot)$ have the following relationship:

$$f(x) \;=\; \frac{1}{2\pi}\int_{\mathbb{R}} e^{-i\omega x}\phi(\omega)\;d\omega. \tag{4.2}$$

The idea is to reconstruct the integral in (4.2) from its Fourier-cosine series expansion, where the cosine series coefficients are extracted directly from the integrand. When a function has finite support, the Fourier-cosine expansion of that function typically gives an optimal approximation. For functions supported on a finite interval $[a,b]\subset\mathbb{R}$ the Fourier-cosine expansion reads:

$$f(x) \;=\; \sum_{k=0}^{\infty}{}' A_k\cdot\cos\left(k\pi\frac{x-a}{b-a}\right),$$

where

$$A_k \;=\; \frac{2}{b-a}\int_a^b f(x)\cos\left(k\pi\frac{x-a}{b-a}\right)\;dx,$$

where by $\sum'$ is meant that the first term in the summation is multiplied by 0.50. The integrand in equation (4.2) has to decay towards zero at $\pm\infty$ for the truncation of the integral to be justified without loss of accuracy. This is because of existence conditions of the Fourier transform. Now suppose that the truncation range $[a,b]$ is chosen such that the truncation error of the integrals small enough, i.e.

$$\phi^1(\omega) \;:=\; \int_a^b e^{i\omega x}f(x)\;dx, \;\approx\; \int_{\mathbb{R}} e^{i\omega x}f(x)\;dx \;=\; \phi(\omega).$$

The superscript in $\phi^i(\cdot)$ denotes the i-th consecutive approximation of $\phi(\cdot)$. From the approximation above, derive the following:

$$\begin{aligned} A_k \;&=\; \frac{2}{b-a}\Re\left\{\phi_1\left(\frac{k\pi}{b-a}\right)\cdot\exp\left(\frac{-ika\pi}{b-a}\right)\right\}, \\ &\approx\; \frac{2}{b-a}\Re\left\{\phi\left(\frac{k\pi}{b-a}\right)\cdot\exp\left(\frac{-ika\pi}{b-a}\right)\right\} \;=\; F_k, \end{aligned}$$

where $\Re\{\cdot\}$ denotes the real part. Now replace A_k by F_k in the series expansion of the density on interval $[a,b]$ and afterwards truncate the summation:

$$f^1(x) \;=\; \sum_{k=0}^{\infty}{}' F_k\cdot\cos\left(k\pi\frac{x-a}{b-a}\right), \tag{4.3}$$

$$f^2(x) \;=\; \sum_{k=0}^{N-1}{}' F_k\cdot\cos\left(k\pi\frac{x-a}{b-a}\right). \tag{4.4}$$

4.1.2 Pricing European options

Using the approximation of $f(x)$ by $f^2(x)$, it is time to derive the COS formula for European options by replacing the density in the pricing equation by its Fourier-cosine series. Because the density in risk-neutral valuation formula (4.1) rapidly decays to zero as $y \to \pm\infty$, the infinite integration range can be truncated to $[a, b] \subset \mathbb{R}$ without loosing significant accuracy. This results in approximation C^1 of C:

$$C^1(x, t_0) \quad = \quad e^{-r_d \tau} \int_a^b C(y, T) f(y|x)\, dy.$$

Next, replace the density $f(y|x)$ by its cosine expansion:

$$f(y|x) \quad = \quad \sum_{k=0}^{\infty}{}' A_k(x) \cdot \cos\left(k\pi \frac{y-a}{b-a}\right),$$

where

$$A_k(x) \quad = \quad \frac{2}{b-a} \int_a^b f(y|x) \cos\left(k\pi \frac{y-a}{b-a}\right)\, dy,$$

resulting in:

$$C^1(x, t_0) \quad = \quad e^{-r_d \tau} \int_a^b C(y, T) \sum_{k=0}^{\infty}{}' A_k(x) \cdot \cos\left(k\pi \frac{y-a}{b-a}\right)\, dy. \tag{4.5}$$

Now interchange summation and integration, and rewrite using:

$$V_k \quad = \quad \frac{2}{b-a} \int_a^b C(y, T) \cos\left(k\pi \frac{y-a}{b-a}\right)\, dy, \tag{4.6}$$

resulting in:

$$C^1(x, t_0) \quad = \quad \frac{1}{2}(b-a) e^{-r_d \tau} \sum_{k=0}^{\infty}{}' A_k(x) V_k.$$

Here, the V_k coefficients are the cosine-series coefficients of $C(y, T)$ in y. Due to the rapid decay of these coefficients, the series are truncated to obtain approximation $C^2(x, t_0)$:

$$C^2(x, t_0) \quad = \quad \frac{1}{2}(b-a) e^{-r_d \tau} \sum_{k=0}^{N-1}{}' A_k(x) V_k.$$

As before, approximate the $A_k(x)$ coefficients by $F_k(x)$ and replace them in order to get the final COS approximation $C^3(x, t_0)$ of European option value $C(x, t_0)$:

$$C(x, t_0) \quad \approx \quad C^3(x, t_0) \quad = \quad e^{-r_d \tau} \sum_{k=0}^{N-1}{}' \Re\left\{\phi\left(\frac{k\pi}{b-a}; x\right) \cdot \exp\left(\frac{-ika\pi}{b-a}\right)\right\} V_k. \tag{4.7}$$

Fang and Oosterlee have shown that the coefficients V_k can be obtained analytically for European options. Since the assumption is made that the chf of log-asset price x is known, they represent the payoff as a function of the log-asset price. They use the following definitions:

$$x = \log\left(\frac{S_0}{K}\right), \quad y = \log\left(\frac{S_T}{K}\right).$$

In terms of log-asset price, the payoff for European options is given by:

$$C(y,T) = [\alpha \cdot K(e^y - 1)]^+ \quad \text{where} \quad \alpha = \begin{cases} 1, & \text{for a call,} \\ -1, & \text{for a put.} \end{cases}$$

Using the cosine-series coefficients $\chi_k(c,d)$ of $g(y) = e^y$ on $[c,d] \subset [a,b]$ and the cosine-series coefficients $\psi_k(c,d)$ of $g(y) = 1$ on $[c,d] \subset [a,b]$, one can derive V_k using its definition (4.6). Coefficients χ_k and ψ_k are given by the following:

$$\begin{aligned}
\chi_k &= \int_c^d e^y \cos\left(k\pi \frac{y-a}{b-a}\right) dy, \\
&= \frac{1}{1+\left(\frac{k\pi}{b-a}\right)^2}\left[\cos\left(k\pi\frac{d-a}{b-a}\right)e^d - \cos\left(k\pi\frac{c-a}{b-a}\right)e^c\right. \\
&\quad \left. + \frac{k\pi}{b-a}\sin\left(k\pi\frac{d-a}{b-a}\right)e^d - \frac{k\pi}{b-a}\sin\left(k\pi\frac{c-a}{b-a}\right)e^c\right], \\
\psi_k &= \int_c^d \cos\left(k\pi \frac{y-a}{b-a}\right) dy, \\
&= \begin{cases} \frac{b-a}{k\pi}\left[\sin\left(k\pi\frac{d-a}{b-a}\right) - \sin\left(k\pi\frac{c-a}{b-a}\right)\right], & k \neq 0, \\ d-c, & k = 0. \end{cases}
\end{aligned}$$

Using the above, V_k as defined in (4.6) can be rewritten as follows for respectively a European call and put:

$$\begin{aligned}
V_k^{call} &= \frac{2}{b-a}K(\chi_k(0,b) - \psi_k(0,b)), \\
V_k^{put} &= \frac{2}{b-a}K(\chi_k(a,0) - \psi_k(a,0)).
\end{aligned}$$

4.1.3 Truncation range $[a,b]$

For the interval of integration $[a,b]$ for the COS method, Fang and Oosterlee propose to use the following:

$$[a,b] = \left[x_0 + \kappa_1 - L\sqrt{\kappa_2 + \sqrt{\kappa_4}},\ \ x_0 + \kappa_1 + L\sqrt{\kappa_2 + \sqrt{\kappa_4}}\right], \quad L = 10. \tag{4.8}$$

Here κ_n denotes the n-th cumulant of $\log(S_T/K)$ and $x_0 = \log(S_0/K)$. These cumulants are defined by the cumulant-generating function $K(t)$:

$$K(t) = \log\left(\mathbb{E}\left[e^{tX}\right]\right) = \log(\phi_X(-it)).$$

for some random variable X and its chf ϕ_X. The cumulants are given by the derivatives of $K(t)$ evaluated at zero. Furthermore, the value of L is chosen in the range of around 8 to 10 for most

models to give accurate results.

In their article, Fang and Oosterlee state that when trying to price a call option, the accuracy of the method is highly sensitive to the choice of L in (4.8). The nature of the issue lies in the exponential growth of the call payoff with the log-stock price, which introduces significant cancellation errors for large values of L. Payoffs of put options are bounded by the strike level, and therefore no issues arise when pricing this type of options. This leads to the choice of using the put-call parity when call options are to be priced:

$$C_{call}(x, t_0) \quad = \quad C_{put}(x, t_0) + S_0 e^{-r_f(T-t_0)} - K e^{-r_d(T-t_0)}.$$

Anywhere in this thesis where call options are priced using the COS method the put-call parity will be used.

In case of the Heston model, the values of κ_1 and κ_2 are given in Table 11 of [26]. According to Fang and Oosterlee, since the analytic expression of κ_4 is too lengthy it is omitted and a larger value of L is proposed. In cases where the Feller condition ($2\lambda > \alpha^2$) is not satisfied, the value of κ_2 may become negative. Looking at equation (4.8) this tells that the argument of the square root will become negative. Since $[a, b] \subset \mathbb{R}$ must hold, this issue is solved by taking the absolute value of cumulant κ_2.
However, as can be read in the next chapter in equation (5.5), the second cumulant is equal to the variance σ^2. By definition this cannot be a negative value, and therefore it can be concluded that there is a mistake in the article by Fang and Oosterlee. This claim is supported by the article by Fabien Le Floc'h [41], which also states that the analytic formula for the second cumulant is incorrect and a new formula based on a Taylor expansion is proposed.
If the Heston model with term structure is used during pricing with the COS method, the value of κ_4 cannot be omitted as in the case of the multi-term Heston model. Unfortunately, the derivation of these cumulants becomes more and more difficult when a term structure is introduced. Therefore, a finite difference approximation of the cumulants is proposed. In Chapter 5 the finite difference approximation of the cumulants is discussed in more detail. Furthermore it is also a possibility to replace finite differences by the technique of algorithmic differentiation (AD), which will be introduced in Section 6.3.

4.2 Lewis' method

As shown in Chapter 3, the Heston model defined in (3.1) has an associated PDE (3.7) that can be used for pricing options, which can be solved with a transformation technique. Lewis defines a way of transformation called the fundamental transform $\hat{H}(k, v, \tau)$. This transform depends on the (generalized) Fourier transform variable k, volatility v and time to maturity $\tau = T - t$. Once obtained, the same function $\hat{H}(k, v, \tau)$ is used to determine the value of any European option, since the function does not depend on all the properties of the option contract. This last step requires an integration in the complex k-plane.
The fundamental transform has a lot of special properties, many of which follow from the fact that there is an analytically known chf.
The starting point is European option payoff $C(x, v, T) = [\alpha \cdot (e^x - K)]^+$, where as before $x = \log(S)$. Since the put-call parity can be used to determine the price of a European put when the corresponding call-price is known, consider only the European call for now. This means setting $\alpha = 1$, resulting in $C(x, v, T) = [e^x - K]^+$. Now apply the Fourier transform to

the option payoff:

$$\begin{aligned}\hat{C}(x,v,t) &= \int_{-\infty}^{\infty} e^{ikx} C(x,v,t)\ dx, \\ &= \int_{-\infty}^{\infty} e^{ikx}[e^x - K]^+\ dx, \\ &= \left[\frac{e^{(ik+1)x}}{ik+1} - K\frac{e^{ikx}}{ik}\right]\Bigg|_{x=\log(K)}^{x=\infty}.\end{aligned}$$

This upper limit of the integral does not exist unless $\Im(k) > 1$ holds, where $\Im(\cdot)$ denotes the imaginary part. Assuming this holds, it is possible to write:

$$\hat{C}(x,v,T) = -\frac{K^{1+ik}}{k^2 - ik}. \tag{4.9}$$

In general, the Fourier transform of the payoff exists only when $\Im(k)$ is bounded to the strip $\alpha < \Im(k) < \beta$. Given the transform from equation (4.9), it is possible to compute the inverse transformation:

$$C(x,v,t) = \frac{1}{2\pi}\int_{i\cdot\Im(k)-\infty}^{i\cdot\Im(k)+\infty} e^{-ikx}\hat{C}(x,v,t)\ dx. \tag{4.10}$$

When $\Im(k) > 1$, the PDE satisfied by $\hat{C}(x,v,t)$ based on Heston formulation (3.1) is:

$$\frac{\partial \hat{C}_t}{\partial t} = [r_d + ik\mu]\hat{C}_t + \frac{v(k^2-ik)}{2}\hat{C}_t - (\kappa(\theta - v) - ik\rho\gamma v)\frac{\partial \hat{C}_t}{\partial v} - \frac{\gamma^2 v}{2}\frac{\partial^2 \hat{C}_t}{\partial v^2}. \tag{4.11}$$

Now define $\hat{H}(k,v,\tau)$ as:

$$\hat{C}(k,v,t) = e^{[-r_d - ik\mu]\tau}\hat{H}(k,v,\tau). \tag{4.12}$$

Using (4.11) and (4.12), note that $\hat{H}(k,v,\tau)$ satisfies the following PDE:

$$\frac{\partial \hat{H}_\tau}{\partial \tau} = \frac{v(k^2-ik)}{2}\hat{H}_\tau + (\kappa(\theta - v) - ik\rho\gamma v)\frac{\partial \hat{H}_\tau}{\partial v} + \frac{\gamma^2 v}{2}\frac{\partial^2 \hat{H}_\tau}{\partial v^2}, \tag{4.13}$$

where

$$\hat{H}(k,v,0) = 1. \tag{4.14}$$

Lewis gives this solution in analytic form for the Heston model:

$$\begin{aligned}\hat{H}(k,v,T) &= \exp\left(\frac{2\kappa\theta}{\gamma^2}\left[t\cdot g - \log\left(\frac{1-he^{-\xi t}}{1-h}\right)\right] + vg\left[\frac{1-e^{-\xi t}}{1-he^{-\xi t}}\right]\right), \qquad (4.15)\\ g &= \frac{b-\xi}{2}, \\ h &= \frac{b-\xi}{b+\xi}, \\ t &= \frac{\gamma T}{2}, \\ \xi &= \sqrt{b^2 + 4\frac{k^2-ik}{\gamma^2}}, \\ b &= \frac{2(ik\rho\gamma + \kappa)}{\gamma}.\end{aligned}$$

The PDE from (4.13) combind with initial condition (4.14) is called regular if there exists a fundamental solution of (4.13) that is regular as a function of k in the fundamental strip of regularity. The latter is a strip $\alpha < \Im(k) < \beta$ in the complex half-plane where $\alpha, \beta \in \mathbb{R}$. If a fundamental strip of regularity is chosen, one can apply the following procedure to obtain the solution $C(x, v, t)$ given the fundamental transform:

1. Multiply $\hat{H}(k, v, \tau)$ by the Fourier transform of the payoff.
2. Multiply by $e^{[-r_d - ik\mu]\tau}$.
3. Perform the inverse transformation as in (4.10). When keeping $\Im(k)$ in a suitable strip of the complex k-plane, there will be a solution $C(x, v, t)$.
4. Transform the solution into terms of S instead of x.

For a European call one knows that the initial-value problem (4.13) is regular in a strip $\alpha < \Im(k) < \beta$ ánd $\beta > 1$ because $\Im(k) > 1$ needs to hold. Applying these steps gives:

$$\begin{aligned} C(S, v, \tau) &= -\frac{Ke^{-r_d\tau}}{2\pi} \int_{i\cdot\Im(k)-\infty}^{i\cdot\Im(k)+\infty} e^{-ikX} \frac{\hat{H}}{k^2 - ik}\, dk, \\ &= -\frac{Ke^{-r_d\tau}}{\pi} \Re\left\{ \int_{i\cdot\Im(k)}^{i\cdot\Im(k)+\infty} e^{-ikX} \frac{\hat{H}}{k^2 - ik}\, dk \right\}, \end{aligned} \tag{4.16}$$

where

$$X = \log\left(\frac{S_t}{K}\right) + \mu\tau. \tag{4.17}$$

In practice, the k-plane integrations are typically done along $\Im(k) = 1/2$, so the same choice is made here, and it is known that often $\hat{H}$ is free of singularities in this strip. The formula for the call option with this choice of $\Im(k)$ is obtained by using the put-call parity:

$$C(S, v, \tau) = Se^{-r_f\tau} - \underbrace{\left[Ke^{-r_d\tau} - P(S, v, \tau)\right]}_{cash-secured\ put},$$

where as indicated the expression between brackets is a cash-secured put. Furthermore, $P(S, v, \tau)$ is the value of the put option. The Fourier transform of the payoff function for the cash-secured put is defined as:

$$\hat{H}(k, v, T) = \frac{K^{ik+1}}{k^2 - ik},$$

which is the same as the Fourier transform of the call option in (4.9) apart from a minus sign. Also note the different bound $0 < \Im(k) < 1$ compared to $\Im(k) > 1$ for a call. Assuming that $\hat{H}$ is regular in a fundamental strip intersecting with $0 < \Im(k) < 1$, the call-price equivalent to the definition in (4.16) is as follows:

$$C(S, v, \tau) = Se^{-r_f\tau} - \frac{Ke^{-r_d\tau}}{\pi} \Re\left\{ \int_{i/2}^{i/2+\infty} e^{-ikX} \frac{\hat{H}}{k^2 - ik}\, dk \right\}. \tag{4.18}$$

Since in the markets one is usually able to observe prices in terms of forwards, the call-price from equation (4.18) is rewritten in terms of forwards. Using the definition of a forward from

Section 2.2, equations (4.18) and (4.17) are rewritten as:

$$\begin{aligned} C(F,v,\tau) &= e^{-r_d\tau}\left[F - \frac{K}{\pi}\Re\left\{\int_{i/2}^{i/2+\infty} e^{-ikX}\frac{\hat{H}}{k^2-ik}\,dk\right\}\right], \\ X &= \log\left(\frac{F_t}{K}\right). \end{aligned}$$

The Lewis method described above is implemented in NAG's s30nac [53]. Since the formulation of the Heston model is slightly different in NAG's s30ncc [54], namely the formulation in (3.19), than the one used above, the method described above holds with a slightly different PDE as in (4.13) resulting in a different solution for $\hat{H}(k,v,\tau)$ as in (4.15). This new solution is not presented here, but it can be easily derived if necessary.

Note that Lewis' method [43] is used in NAG option pricing routines s30nac [53] for the single-term Heston model and s30ncc [54] for the multi-term Heston model. Both routines have been used plentifully during the coding process of this thesis.

Chapter 5

Modifications of the COS method

The COS method by Fang and Oosterlee [26] that has been discussed in Section 4.1 is a novel option pricing method, and has proven to be fast and accurate. However, from a practical point of view, while implementing the method, there are some parameters whose values are not explicitly given for every situation that one might find him/herself in. The choices of values for L and N in [26] are examples of this. Furthermore, in addition to the improvement of parameter choices, there appears to be a way in which the choice of truncation range can be improved.

If a practitioner chooses to implement the COS method using the values of L and N as proposed in the article, there is no guarantee that these parameters will be appropriate for every situation that might be encountered. Thinking of a calibration setting, where typically the pricing model is part of objective function for numerical optimization, the optimizer can go in directions that a user would not expect it to go. This means that certain 'extreme' and unrealistic parameter combinations can be encountered during the optimization. It is obvious the prices for those options must be just as accurate as the more 'moderate' parameter combinations. In order to be certain the pricing method performs equally well over the entire parameter space, additional work is required.

In this chapter, new approaches will be developed for setting the truncation range $[a, b]$ (Section 5.1) and for determining the amount of terms N in the cosine series expansion (see Section 5.2). For the choice of parameter L an adaptive approach is proposed in Section 5.3.

All these modifications aim to be an improvement of the ability to use the COS method in practice. Finally in Section 5.4 the modifications of the COS method from the current chapter will be put to the test along with all the methods described in Chapter 4, and the accuracy and speed of the methods will be compared.

5.1 Truncation range

As described in Section 4.1.3, in order to determine the truncation range, the cumulants of $\log(S_T/K)$ are used. A more thorough introduction to the concept of cumulants will be given in Section 5.1.1. Once this has been done, the newly proposed version of the truncation range can be found in Section 5.1.2. To see if this new range can be considered superior to the original one, some numerical tests are performed in Section 5.1.3.

5.1.1 Theory of cumulants

In general, given a random variable X, the moment generating function (mgf) is defined as:

$$M_X(t) \;=\; \mathbb{E}\left[e^{tX}\right] \;=\; \int_{-\infty}^{\infty} f_X(x)e^{tx}\,dx, \quad t \in \mathbb{R},$$

where $f_X(x)$ is the pdf of X. When all moments are finite, the mgf can be rewritten using a Maclaurin series expansion:

$$M_X(t) \;=\; \sum_{n=0}^{\infty} \frac{m_n \cdot t^n}{n!},$$

where the raw moments m_n for $n = 0, 1, \ldots$ are defined as:

$$m_n \;=\; \mathbb{E}[X^n] \;=\; \frac{d^n M_X}{dt^n}(0). \tag{5.1}$$

Furthermore, the chf of this random variable X is defined as :

$$\phi_X(t) \;=\; \mathbb{E}\left[e^{itX}\right].$$

Note that the following relationship exists between the mgf and the chf by combining the definitions of both functions:

$$\phi_X(-it) \;=\; \mathbb{E}\left[e^{i(-it)X}\right] \;=\; \mathbb{E}\left[e^{itX}\right] \;=\; M_X(t).$$

The cumulant-generating function (cgf) is defined as:

$$K_X(t) \;=\; \log\left(M_X(t)\right) \;=\; \log(\phi_X(-it)), \tag{5.2}$$

and it has a similar Maclaurin expansion as the mgf:

$$K_X(t) \;=\; \sum_{n=1}^{\infty} \frac{\kappa_n \cdot t^n}{n!},$$

where the cumulants κ_n for $n = 1, 2, \ldots$ are defined as:

$$\kappa_n \;=\; \frac{d^n K_X}{dt^n}(0). \tag{5.3}$$

Using the results from (5.2), the first four derivatives of the cgf can be expressed as follows:

$$\begin{aligned}
\frac{dK_X(t)}{dt} &= \frac{1}{\phi_X(-it)}\frac{d\phi_X(-it)}{dt},\\
\frac{d^2K_X(t)}{dt^2} &= -\frac{1}{\phi_X(-it)^2}\left[\frac{d\phi_X(-it)}{dt}\right]^2 + \frac{1}{\phi_X(-it)}\frac{d^2\phi_X(-it)}{dt^2},\\
\frac{d^3K_X(t)}{dt^3} &= \frac{2}{\phi_X(-it)^3}\left[\frac{d\phi_X(-it)}{dt}\right]^3 - \frac{3}{\phi_X(-it)^2}\frac{d\phi_X(-it)}{dt}\frac{d^2\phi_X(-it)}{dt^2} + \frac{1}{\phi_X(-it)}\frac{d^3\phi_X(-it)}{dt^3},\\
\frac{d^4K_X(t)}{dt^4} &= -\frac{6}{\phi_X(-it)^4}\left[\frac{d\phi_X(-it)}{dt}\right]^4 + \frac{12}{\phi_X(-it)^3}\left[\frac{d\phi_X(-it)}{dt}\right]^2\frac{d^2\phi_X(-it)}{dt^2}\\
&\quad -\frac{3}{\phi_X(-it)^2}\left[\frac{d^2\phi_X(-it)}{dt^2}\right]^2 - \frac{4}{\phi_X(-it)^2}\frac{d\phi_X(-it)}{dt}\frac{d^3\phi_X(-it)}{dt^3} + \frac{1}{\phi_X(-it)}\frac{d^4\phi_X(-it)}{dt^4}.
\end{aligned}$$

Using these results on the derivatives of the cgf and the identity $\phi_X(0) = 1$, the first four cumulants are defined as:

$$\begin{aligned}
\kappa_1 &= \left.\frac{dK_X(t)}{dt}\right|_{t=0} = \left.\frac{d\phi_X(-it)}{dt}\right|_{t=0}, \\
\kappa_2 &= \left.\frac{d^2K_X(t)}{dt^2}\right|_{t=0} = \left[-\left(\frac{d\phi_X(-it)}{dt}\right)^2 + \frac{d^2\phi_X(-it)}{dt^2}\right]\Bigg|_{t=0}, \\
\kappa_3 &= \left.\frac{d^3K_X(t)}{dt^3}\right|_{t=0} = \left[2\left[\frac{d\phi_X(-it)}{dt}\right]^3 - 3\frac{d\phi_X(-it)}{dt}\frac{d^2\phi_X(-it)}{dt^2} + \frac{d^3\phi_X(-it)}{dt^3}\right]\Bigg|_{t=0}, \\
\kappa_4 &= \left.\frac{d^4K_X(t)}{dt^4}\right|_{t=0} = \left[-6\left(\frac{d\phi_X(-it)}{dt}\right)^4 + 12\left(\frac{d\phi_X(-it)}{dt}\right)^2\frac{d^2\phi_X(-it)}{dt^2} - 3\left(\frac{d^2\phi_X(-it)}{dt^2}\right)^2\right. \\
&\qquad \left. -4\frac{d\phi_X(-it)}{dt}\frac{d^3\phi_X(-it)}{dt^3} + \frac{d^4\phi_X(-it)}{dt^4}\right]\Bigg|_{t=0}.
\end{aligned}$$

The derivatives of the chf can be approximated using finite differences (see Appendix A.3 for a scheme) or can be computed using the technique of algorithmic differentiation described in Section 6.3.

Using the known relationship between the cumulants and moments, the following can be stated:

$$\kappa_1 = m_1 = \mu, \tag{5.4}$$
$$\kappa_2 = m_2 - m_1^2 = \sigma^2, \tag{5.5}$$
$$\kappa_3 = 2m_1^3 - 3m_1m_2 + m_3 = \gamma_3\sigma^3, \tag{5.6}$$
$$\kappa_4 = -6m_1^4 + 12m_1^2m_2 - 3m_2^2 - 4m_1m_3 + m_4 = \gamma_4\sigma^4, \tag{5.7}$$

where μ is the mean of X, σ^2 is the variance of X, γ_3 is the skewness of X and γ_4 is the kurtosis of X. These properties will be needed later on.

Recall what skewness is, namely a measure of asymmetry around the mean of a distribution. A normal distribution, which is symmetric, has a skewness of $\gamma_3 = 0$. The value of γ_3 can be both positive and negative, resulting in distributions that are skewed respectively right and left (see Figure 5.1).

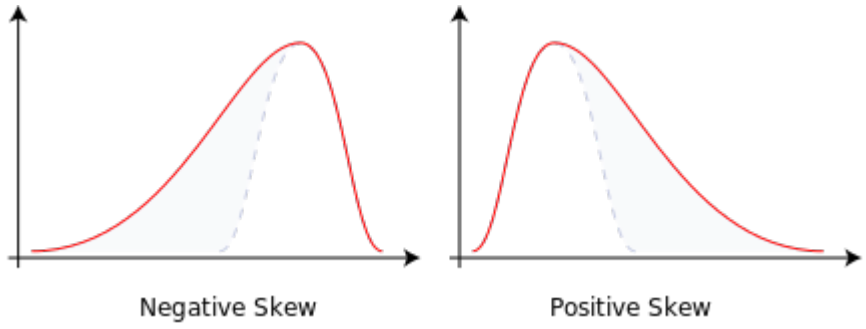


Figure 5.1: Skewness.
Source: "http://bit.ly/2agzSgQ".

In the case of positive skewness, the right tail of the distribution is thicker. For negative skewness the opposite holds, namely a thick left tail of the distribution.

5.1.2 New truncation range

Now that an introduction into cumulants has been given, it is possible to turn to the matter of developing an alternative truncation range. Before doing so, first consider the original truncation range (4.8), which will be rewritten using the results from (5.5) and (5.7) as:

$$[a_{old}, b_{old}] = [x_0 + \kappa_1 - L \cdot f(\kappa_2, \kappa_4), \quad x_0 + \kappa_1 + L \cdot f(\kappa_2, \kappa_4)], \tag{5.8}$$

$$f(\kappa_2, \kappa_4) = \sqrt{\kappa_2 + \sqrt{\kappa_4}} = \sigma\sqrt{1 + \sqrt{\gamma_4}}, \tag{5.9}$$

where for now the choice of L is arbitrary. Following the logic the way truncation range (5.8) is put together, the following new truncation range is proposed:

$$a_{new} = x_0 + \kappa_1 - L\sqrt{\kappa_2 + \sqrt{\kappa_4} + \mathbb{1}_{\{\kappa_3<0\}} \cdot |\kappa_3|^{2/3}}, \tag{5.10a}$$

$$b_{new} = x_0 + \kappa_1 + L\sqrt{\kappa_2 + \sqrt{\kappa_4} + \mathbb{1}_{\{\kappa_3>0\}} \cdot |\kappa_3|^{2/3}}. \tag{5.10b}$$

Note that for now the value of L is assumed to be known. The newly proposed truncation range $[a_{new}, b_{new}]$ is very similar to the original form (5.8), with the difference that the third cumulant has been included. This has been done to include the possible skewness γ_3 of a distribution in the truncation range. Note that skewness relates to the third cumulant according to (5.6).

In order to inform the reader where the new truncation range originates from, a similar approach is taken as above. The difference is that $f(\kappa_2, \kappa_4)$ is replaced by new but similar functions $g(\kappa_2, \kappa_3, \kappa_4)$ and $h(\kappa_2, \kappa_3, \kappa_4)$ for the definitions of respectively a_{new} from (5.8) and b_{new} from (5.9). When truncating a positively skewed distribution, the right hand side of the truncation range should be further out to the right than when considering the symmetric equivalent of the distribution at hand. Exactly the other way round, negatively skewed distributions should have a left side of the truncation range that is further out to the left. Going back to the cumulants, from (5.6) it is clear that the third cumulant is the skewness scaled by a factor that is dependent on the variance. This is similar to the fourth cumulant, which is the kurtosis scaled by a factor that is dependent on the variance. This means that the third cumulant can be incorporated into the definition of the truncation range in a similar way as the fourth cumulant has been in (5.8). Taking into account all the features of skewness, the functions $g(\cdot)$ and $h(\cdot)$ can be defined as:

$$\begin{aligned} g(\kappa_2, \kappa_3, \kappa_4) &= \sqrt{\kappa_2 + \sqrt{\kappa_4} + \mathbb{1}_{\{\kappa_3<0\}} \cdot |\kappa_3|^{2/3}}, \\ &= \sigma\sqrt{1 + \sqrt{\gamma_4} + \mathbb{1}_{\{\gamma_3<0\}} \cdot |\gamma_3|^{2/3}}, \end{aligned} \tag{5.11}$$

$$\begin{aligned} h(\kappa_2, \kappa_3, \kappa_4) &= \sqrt{\kappa_2 + \sqrt{\kappa_4} + \mathbb{1}_{\{\kappa_3<0\}} \cdot |\kappa_3|^{2/3}}, \\ &= \sigma\sqrt{1 + \sqrt{\gamma_4} + \mathbb{1}_{\{\gamma_3>0\}} \cdot |\gamma_3|^{2/3}}. \end{aligned} \tag{5.12}$$

This leads to the following definition of the modified truncation range in (5.13), which is simply a reformulation of (5.10a) and (5.10a):

$$[a_{new}, b_{new}] = [x_0 + \kappa_1 - L \cdot g(\kappa_2, \kappa_3, \kappa_4), \quad x_0 + \kappa_1 + L \cdot h(\kappa_2, \kappa_3, \kappa_4)]. \tag{5.13}$$

Note that the changes in (5.11) and (5.12) of the indicator function from κ_3 to γ_3 are justified because by definition $\sigma > 0$, which implies that γ_3 and κ_3 have the same sign. In addition, note that by only extending one of the two sides of the truncation range (depending on the sign of κ_3), the other side of the truncation range still has the same value as in the original case (5.8).

This means that adding the third cumulant only enlarges the range on one side and leaves the other side unchanged. This implies that the error due to truncating the density with the new truncation range (5.13) should be smaller than when using of the original formula (5.8).

5.1.3 Numerical tests

To support the claim that the new truncation range will be more appropriate than the original range, the performance of the two truncation ranges has been tested for a certain given value of L. The idea is to pick a set of known probability distributions with known analytic expressions for the pdf, chf, cumulants and in some cases the cdf. For this set of distributions several tests are performed on the recovery of the pdf using a Fourier cosine expansion.

Recall from equation (4.1) that the price of an option is directly linked to the pdf of the underlying process. Furthermore recall from (4.5) that after the recovery of the pdf by a cosine series expansion, the only unknowns before summation and integration can be performed are coefficients V_k. These coefficients can be recovered analytically for several types of contracts, including European option contracts. Therefore the choice to test the recovery of several known pdf's will provide useful information on the quality of the truncation ranges. Due to the reasoning above these results will also be applicable to an option pricing situation.

The choice is made to consider the following set of distributions:

1. Gamma distribution,
2. Normal inverse Gaussian distribution,
3. Inverse Gaussian distribution,
4. Exponentially modified Gaussian distribution.

The relevant information on the various distributions being used can be found in Appendix A.2. For a function with support $\mathbb{R}$, define ϵ_1 to be the error measure for the choice of truncation range:

$$\epsilon_1 \quad = \quad \int_{\mathbb{R}\setminus[a,b]} f(y)\ dy. \tag{5.14}$$

Because there are two tails that contribute to this error, the choice is made to split up the error ϵ_1 into two pieces, each addressing one of the two tails:

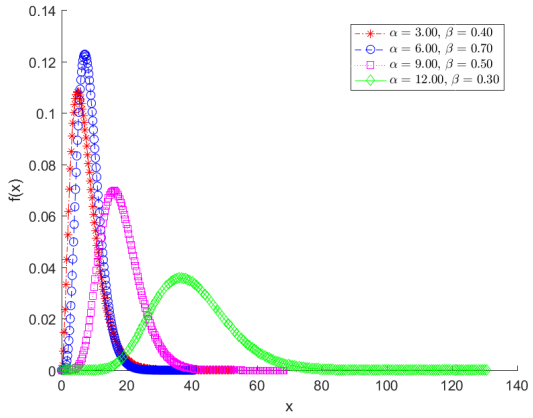
$$\epsilon_1 \quad = \quad \underbrace{\int_{-\infty}^{a} f(y)\ dy}_{\epsilon_{1,L}} \quad + \quad \underbrace{\int_{b}^{+\infty} f(y)\ dy}_{\epsilon_{1,R}}. \tag{5.15}$$

Note that these integrals can be evaluated using either the cdf if it is available or a numerical integration rule. The natural extension of this error definition for densities with support $(0, \infty)$ is:

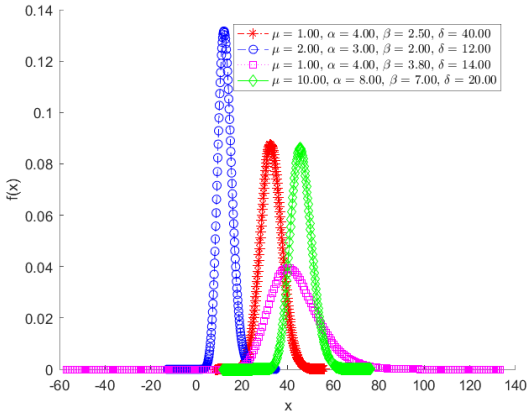
$$\epsilon_1 \quad = \quad \underbrace{\int_{0}^{a} f(y)\ dy}_{\epsilon_{1,L}} \quad + \quad \underbrace{\int_{b}^{+\infty} f(y)\ dy}_{\epsilon_{1,R}}. \tag{5.16}$$

In this situation, the formulae for the truncation ranges could give a value of a_{old} and/or a_{new} lower than zero, depending on the choice of L. In case this situation arises, the left side of the truncation range is fixed to a small positive value close to machine precision, resulting in $\epsilon_{1,L} = 0$. In those cases, any results concerning the left tail of the distribution are left out of results.

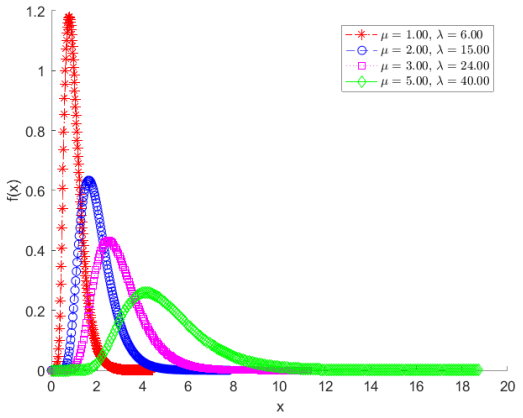
For each of the four distributions that will be used for testing purposes, four different parameter combination sets will be considered. Note that this choice is arbitrary, the only goal that needs to be achieved was four different pdf's for the same distribution type. In Figure 5.2 plots can be found of the densities that are truncated using the COS method for each of the four parameter combinations. For each of the distribution types, L (necessary for computing the truncation range) is fixed to a certain value.



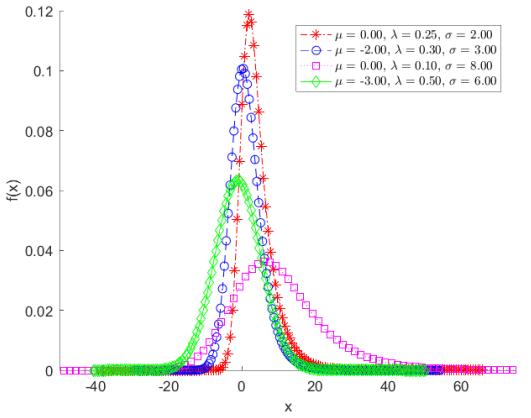
(a) Gamma, $L = 10$.



(b) Normal inverse Gaussian, $L = 10$.



(c) Inverse Gaussian, $L = 10$.



(d) Exponentially modified Gaussian, $L = 8$.

Figure 5.2: Density plots for various parameter combinations.

Note that in the COS method one must also set the value N, the amount of terms used in the Fourier cosine expansion. Since the goal is to test the performance of the truncation ranges, here a sufficiently large value of N is chosen in order to eliminate possible error contributions from truncation the infinite summation, i.e. the change from (4.3) to (4.4).
In Tables 5.1, 5.2, 5.3 and 5.4 the results of the tests are displayed. For each parameter set, the truncation ranges $[a_{old}, b_{old}]$ and $[a_{new}, b_{new}]$ and the relevant corresponding error values are reported.

Param. set	$[a_{old}, b_{old}]$	$\epsilon_{1,old,R}$	$[a_{new}, b_{new}]$	$\epsilon_{1,new,R}$
1	[0.0, 74.780]	4.887e-11	[0.0, 88.681]	2.617e-13
2	[0.0, 58.059]	2.349e-12	[0.0, 67.890]	5.218e-15
3	[0.0, 98.866]	3.572e-13	[0.0, 114.368]	4.441e-16
4	[0.0, 190.869]	9.071e-14	[0.0, 218.903]	1.110e-16

Table 5.1: Gamma, $L = 10$.

Param. set	$[a_{old}, b_{old}]$	$\epsilon_{1,old,L}$	$\epsilon_{1,old,R}$	$[a_{new}, b_{new}]$	$\epsilon_{1,new,L}$	$\epsilon_{1,new,R}$
1	[8.183, 57.869]	1.864e-10	1.093e-06	[8.183, 68.475]	1.865e-10	8.217e-11
2	[-12.694, 38.160]	3.415e-28	1.049e-08	[-12.694, 46.773]	3.415e-28	3.984e-12
3	[-56.830, 144.019]	1.957e-194	3.254e-08	[-56.830, 181.393]	1.957e-194	8.029e-11
4	[11.619, 80.677]	2.762e-33	9.914e-09	[11.619, 92.793]	2.762e-33	9.298e-13

Table 5.2: Normal inverse Gaussian, $L = 10$.

Param. set	$[a_{old}, b_{old}]$	$\epsilon_{1,old,R}$	$[a_{new}, b_{new}]$	$\epsilon_{1,new,R}$
1	[0.0, 7.599]	4.534e-09	[0.0, 8.880]	8.416e-11
2	[0.0, 13.347]	1.576e-09	[0.0, 15.617]	2.234e-11
3	[0.0, 19.326]	1.151e-09	[0.0, 22.584]	1.507e-11
4	[0.0, 32.210]	1.151e-09	[0.0, 37.641]	1.507e-11

Table 5.3: Inverse Gaussian, $L = 10$.

Param. set	$[a_{old}, b_{old}]$	$\epsilon_{1,old,L}$	$\epsilon_{1,old,R}$	$[a_{new}, b_{new}]$	$\epsilon_{1,new,L}$	$\epsilon_{1,new,R}$
1	[-51.384, 59.384]	1.361e-147	4.043e-07	[-51.384, 79.151]	1.361e-147	2.888e-09
2	[-39.733, 42.399]	9.260e-38	2.461e-06	[-39.733, 64.851]	9.260e-38	2.924e-09
3	[-95.000, 115.000]	4.976e-34	1.395e-05	[-95.000, 237.294]	4.976e-34	6.816e-11
4	[-52.039, 50.039]	3.960e-17	2.736e-10	[-52.039, 61.824]	3.960e-17	7.552e-13

Table 5.4: Exponentially modified Gaussian, $L = 8$.

Looking at the results in Tables 5.1, 5.2, 5.3 and 5.4 it is indeed the case that the error values for the modified formula of the truncation range are less or equal to the corresponding error values for the original formula of the truncation range. In many cases the differences between the two ranges are minimal, or at a level where the original truncation range is already very accurate. However, there exist cases in which the new truncation range performs significantly better. For example look at parameter set 1 in Table 5.2, there $\epsilon_{1,old,R} \approx 10^{-6}$ and $\epsilon_{1,new,R} \approx 10^{-11}$. Other examples of significantly more accurate results of the new truncation range can be found for parameter sets 1, 2 and 3 for the exponentailly modiffied Gaussian distribution in Table 5.4.
All in all, it can be concluded that the truncation range from (5.13) is an improvement of the original formula in (5.8), and in some cases the differences are significant.

5.2 Number of terms in Fourier cosine expansion

Remember that until this moment the focus has been on improving the truncation of the density using the COS method for given values of L and N. Another possible way of improving the COS method lies in the choice of N, the amount of terms in the Fourier cosine expansion, i.e. the point at which the infinite summation in (4.3) is truncated. In the article by Fang and Oosterlee there is no clear guidance on the choice of N for practical applications that are different from the test-case situations used in the article. Increasing the value of N will result in a higher accuracy of the method, but this is at the cost of a slowdown of the method. Furthermore, increasing N only leads to improvement up to the point where the error due to the truncation of the summation has become negligible and the error due to truncation of the density remains. This means that the value of N depends on $[a, b]$. Thus it would be useful to have a lower bound for N depending on the model and truncation range for a given level of accuracy that must be reached.

The search for a bound on N will be performed for a given truncation range $[a, b]$. On this interval the goal is to minimize the series truncation error $\epsilon_2(x)$:

$$\epsilon_2(x) \quad = \quad \left| \sum_{k=N}^{+\infty} F_k \cdot \cos\left(k\pi \frac{x-a}{b-a}\right) \right|. \tag{5.17}$$

$\epsilon_2(x)$ is the difference between (4.3) and (4.4) due to the truncation of the summation. One thing that should be kept in mind is that a larger value of N will result in a lower value of $\epsilon_2(x)$, but this comes at the cost of slowing down the computation by the COS method. Therefore it is desirable to have an expression for N that addresses this trade-off between speed and accuracy. On assumption that the terms in the sum are convergent towards zero in the tail, it appears sufficient to say something about the size of the N-th term in the summation of (4.4). In [26], with help of [8], it is stated that the cosine series expansion exhibits exponential convergence. Combining this property with the decaying properties of a pdf, it can be said that all the assumptions are met and therefore the choice to bound the N-th term in the summation of (4.4) is a reasonable approach. The choice to work with N instead of $N-1$ is without loss of generality, it is simply more convenient for notation and derivations.

Using the framework presented in Appendix A.4 and a given distribution with known analytic expression of the chf (note that the same distributions as in Section 5.1.3 are used) and a given truncation range $[a, b]$, a lower bound for N can be computed. Note that this framework in Appendix A.4 addresses the computation of N for the Heston model, while currently the four testing densities from above are used to obtain results. Naturally the framework presented can be extended to the test-cases. To summarize the main idea from Appendix A.4, the objective is to bound the absolute value of the N-th term of the summation in (4.4) by a given accuracy level ε which can be set as desired by a user. This gives rise to the following inequality:

$$\left| F_N \cdot \cos\left(k\pi \frac{x-a}{b-a}\right) \right| \quad \leq \quad \varepsilon. \tag{5.18}$$

Solving this inequality for the relevant testing distributions, the following lower bounds on N can be obtained:

1. Gamma distribution:

$$N \quad \geq \quad \frac{(b-a)\beta}{\pi} \left[\frac{(b-a)\varepsilon}{2}\right]^{-1/\alpha}. \tag{5.19}$$

2. Normal inverse Gaussian (NIG) distribution:

$$N \geq \frac{b-a}{\pi}\left[-\frac{1}{\delta}\log\left(\frac{(b-a)\varepsilon}{2}\right)-\gamma\right]. \tag{5.20}$$

3. Inverse Gaussian (IG) distribution:

$$N \geq \frac{b-a}{\pi}\frac{\lambda}{2\mu^2}\sqrt{\left[2\left(1-\frac{\mu}{\lambda}\log\left(\frac{(b-a)\varepsilon}{2}\right)\right)^2-1\right]^2-1}. \tag{5.21}$$

4. Exponentially modified Gaussian (EMG) distribution:

$$N \geq \frac{b-a}{\pi}\sqrt{-\frac{2}{\sigma^2}\log\left(\frac{(b-a)\varepsilon}{2}\right)}. \tag{5.22}$$

In the test results, which can be found in Tables 5.5 till 5.8, the truncation ranges are calculated using the modified formula from (5.13) for a fixed value of L per distribution. Furthermore, the tests are performed for only one of the four parameter sets that can be found in Figure 5.2, the choice of parameter set is arbitrary. Also, ε is fixed to 10^{-15} for all distributions. Now the lower bound on the amount of terms per test case can be calculated from (5.19) till (5.22), and this value will be denoted as $\tilde{N}$. This will be the starting value of the summation in (5.17), and the value of 2^{15} is taken to act as a finite but large upper bound. From the fact that the maximal $\tilde{N}$ in the results is 536, an upper bound of 2^{15} appears a legitimate choice in order to still have a sufficient amount of terms in the summation of (5.17). Finally, the tests are performed 10000 times, and values in the tables are the averages over those runs.

After having computed $\tilde{N}$, the error $\epsilon_2(x)$ is calculated at both the left and right endpoints of the truncation range, a and b. As can be seen in the row $N = 1.0 \cdot \tilde{N}$ from for example Table 5.5, the error values are roughly of the order ε. In the fourth column, the average runtime is normalized by $t_{\tilde{N}}$, thus resulting in the value 1.0 in the row $N = 1.0 \cdot \tilde{N}$. The next step is to take different values of N, ranging from $N = 0.2 \cdot \tilde{N}$ to $N = 1.8 \cdot \tilde{N}$ and fill the table.

N	$\epsilon_2(a)$	$\epsilon_2(b)$	$t_N/t_{\tilde{N}}$
$0.2\tilde{N}$	1.68e-07	4.24e-09	0.606
$0.4\tilde{N}$	3.13e-10	2.06e-11	0.655
$0.6\tilde{N}$	3.38e-13	1.44e-14	0.766
$0.8\tilde{N}$	1.03e-13	2.16e-15	0.879
$1.0\tilde{N}$	1.34e-14	2.64e-16	1.000
$1.2\tilde{N}$	2.17e-15	4.08e-17	1.362
$1.4\tilde{N}$	4.37e-16	8.67e-18	1.431
$1.6\tilde{N}$	9.24e-17	4.77e-18	1.523
$1.8\tilde{N}$	2.21e-17	4.34e-19	1.621

Table 5.5: Gamma distribution, param. set 4, $\tilde{N} = 252$, $L = 10$, $[a, b] = [0.00, 218.90]$, $t_{\tilde{N}} = 1.482 \cdot 10^{-4}$.

N	$\epsilon_2(a)$	$\epsilon_2(b)$	$t_N/t_{\tilde{N}}$
$0.2\tilde{N}$	4.60e-07	5.74e-08	0.632
$0.4\tilde{N}$	1.10e-11	1.10e-11	0.751
$0.6\tilde{N}$	2.52e-15	3.60e-16	0.919
$0.8\tilde{N}$	3.25e-19	3.25e-19	1.081
$1.0\tilde{N}$	0	0	1.000
$1.2\tilde{N}$	0	0	1.089
$1.4\tilde{N}$	0	0	1.152
$1.6\tilde{N}$	0	0	1.200
$1.8\tilde{N}$	0	0	1.261

Table 5.6: NIG distribution, param. set 3, $\tilde{N} = 256$, $L = 10$, $[a, b] = [-56.83, 181.39]$, $t_{\tilde{N}} = 1.556 \cdot 10^{-4}$.

N	$\epsilon_2(a)$	$\epsilon_2(b)$	$t_N/t_{\tilde{N}}$
$0.2\tilde{N}$	2.10e-06	5.06e-07	0.586
$0.4\tilde{N}$	1.54e-08	3.85e-10	0.762
$0.6\tilde{N}$	5.76e-11	1.25e-13	0.850
$0.8\tilde{N}$	6.25e-13	3.07e-14	0.919
$1.0\tilde{N}$	5.08e-15	2.43e-16	1.000
$1.2\tilde{N}$	3.85e-16	3.47e-18	1.110
$1.4\tilde{N}$	0	0	1.186
$1.6\tilde{N}$	0	0	1.254
$1.8\tilde{N}$	0	0	1.323

Table 5.7: IG distribution, param. set 4, $\tilde{N} = 460$, $L = 10$, $[a, b] = [0.00, 37.64]$, $t_{\tilde{N}} = 1.793 \cdot 10^{-4}$.

N	$\epsilon_2(a)$	$\epsilon_2(b)$	$t_N/t_{\tilde{N}}$
$0.2\tilde{N}$	2.34e-03	8.19e-04	0.707
$0.4\tilde{N}$	4.98e-05	2.61e-06	0.748
$0.6\tilde{N}$	1.74e-08	4.31e-08	0.830
$0.8\tilde{N}$	1.81e-11	1.32e-11	0.906
$1.0\tilde{N}$	1.04e-16	5.90e-17	1.000
$1.2\tilde{N}$	0	0	1.078
$1.4\tilde{N}$	0	0	1.154
$1.6\tilde{N}$	0	0	1.239
$1.8\tilde{N}$	0	0	1.315

Table 5.8: EMG distribution, param. set 2, $\tilde{N} = 86$, $L = 8$, $[a, b] = [-39.73, 64.85]$, $t_{\tilde{N}} = 8.511 \cdot 10^{-5}$.

The overall pattern is that the choice of $N = 1.0 \cdot \tilde{N}$ gives error values that are sufficiently small, namely approximately of order ε. In some cases it would have been possible to choose $N = 0.8 \cdot \tilde{N}$, and still have error values that are roughly of the order ε. However, since this does not lead to a significant improvement in computational time, the choice of $N = 1.0 \cdot \tilde{N}$ seems to be appropriate. Also, increasing the value of N does not lead to a significant decrease of $\epsilon_2(a)$ and $\epsilon_2(b)$. So the bound is neither too low nor too high, so it is a safe lower bound but not too safe. Therefore it can be concluded that calculating N using the approach described in (5.18) and Appendix A.4 is suitable and can be used for general types of distributions. The distribution of the Heston model with term structure will now be considered separately.

In Appendix A.4 a full derivation of a lower bound on N or the Heston model with n different terms in the term structure can be found. The following formula is the main result:

$$N \geq \frac{b-a}{\pi} \log\left(\frac{(b-a)\varepsilon}{2}\right) \left[-\sum_{i=1}^{n} \frac{\lambda_i \tau_i \sigma_i}{\alpha_i} \sqrt{1-\rho_i^2} - \frac{\sigma_1}{\alpha_1}\sqrt{1-\rho_1^2} v_0\right]^{-1}. \tag{5.23}$$

Next to the derivation of (5.23), asymptotic results for the lower bound can be found in Appendix A.4. Note that in an option pricing setting the terms in the summation (4.7) are multiplied by the payoff coefficients V_k. As these coefficients exhibit rapid decay as k increases, the terms in the summation of (4.7) become even smaller. This makes the value of N from (5.23) a safe value to use in an option pricing setting, as the already small terms in the summation for large k become smaller because of the multiplication by decaying coefficients V_k.

5.3 Adaptive choice of L

In the article by Fang and Oosterlee a value of $L = 10$ is proposed. However, the experience from practical situations is that this value is not always sufficiently large. Therefore an adaptive way for the choice of L will be discussed here. This adaptive way of choosing L will ensure that an implementation of the COS method will be robust. This so-called adaptive COS method is based on the COS method with truncation range (5.13) and value of N from (5.23). The

adaptive method is an iterative way of calculating the option price up to a desired level of accuracy TOL. During the first iteration an initial guess L_0 has to be given which will be incremented by ΔL every iteration. As soon as the relative absolute difference of the option price in the current iterate, C_{L_i}, compared to that of the previous iterate is below the tolerance level TOL the method has finished. In other words:

1. Initialize the following values:
   (a) $i = 0$, the current iterate.
   (b) L_0, the initial value of L.
   (c) ΔL, the size for updating L each iteration.
   (d) TOL, a tolerance level.
2. Compute N according to (5.23).
3. Calculate the option price C for L_0, which is denoted by C_{L_0}.
4. Go to the next iterate, $i = i + 1$, and perform the following steps:
   (a) Update $L_i = L_{i-1} + \Delta L$.
   (b) Compute C_{L_i} and
   $$\epsilon_{L_i,L_{i-1}} = \frac{\left|C_{L_i} - C_{L_{i-1}}\right|}{\max\left\{C_{L_i}, C_{L_{i-1}}\right\}}.$$
5. Now if:
   (a) $\epsilon_{L_i,L_{i-1}} \leq TOL$, then the sequence of option prices has converged and the series of computations has finished.
   (b) $\epsilon_{L_i,L_{i-1}} > TOL$, then go back to step 4 and repeat until convergence.

For an implementation of the adaptive method, the initial values are chosen as follows: $L_0 = 8$, $\Delta L = 2$ and $TOL = 10^{-8}$. Note that the optimal combination of these values strongly depends on the situation. Therefore this adaptive way will not give the fastest implementation of the COS method. On the other hand, it will guarantee robustness and accuracy up to a pre-defined level. In some application this can be more important than speed.

5.4 Numerical results

In this section numerical tests will be performed on the pricing of European options under the multi-term Heston model. The methods that will compete in the tests are those described in Chapter 4, together with all the proposed modifications of the COS method from this chapter.

In the paper of the COS method [26], a value of $L = 10$ said to give truncation error of around 10^{-12} in the range $T \in [0.1, 10]$. The value of $N = 160$ for the Heston model is said to *"price all options highly accurately"*. Furthermore, it is noted that *"larger values of L would require larger N to reach the same level of accuracy."* To put this claim to the test, options have been priced with an implementation of the COS method that uses these choices of N and L.

All the prices computed are compared to a reference value. As a method for creating reference values, the adaptive approach for the COS method from Section 5.3 will be used, with the

difference that the value of N will be set manually to 2^{17} instead of computing it using the lower bound from (5.23). Furthermore, the values $L_0 = 30$, $\Delta L = 2$ and $TOL = 10^{-14}$ are chosen. As a measure of accuracy of prices the relative absolute difference of a method compared to the reference value will be used. This can be defined as:

$$\epsilon_{i,j} \quad = \quad \frac{|C_i - C_j|}{\max\{C_i, C_j\}},$$

where the choice of $j = 9$ will represent the reference value, i.e. C_9 is the reference price.

During the tests the following methods have been compared:

(COS1). COS method with truncation range (5.8), $L = 10$, $N = 160$.

(COS2). COS method with truncation range (5.13), $L = 10$, $N = 160$.

(COS3). COS method with truncation range (5.13), $L = 10$, N from (5.23) with ε equal to machine precision.

(COS4). COS method with truncation range (5.13), $L = 24$, N from (5.23) with ε equal to machine precision.

(COS5). COS method with truncation range (5.13), L adaptive, N from (5.23) with ε equal to machine precision.

(Lewis). Lewis' method.

(QE). Monte Carlo using Anderson's QE scheme with 10^6 simulations and 10^3 time steps per year.

(QEM). Monte Carlo using Anderson's QE martingale corrected scheme with 10^6 simulations and 10^3 time steps per year.

The results (prices, errors and normalized runtimes) reported for methods 1 up and till 6 are averages over 1000 runs. The values of L reported for the COS method are the final values of L in case the adaptive COS method has been used. Generally speaking, the average prices are reported, but for the Monte Carlo methods a 95% confidence interval is given instead. When $\epsilon_{7,9}$ and $\epsilon_{8,9}$ are reported, the mean price values over all 10^6 simulations have been used to compare with the reference value.

In Tables 5.9 till 5.12 the results of the numerical tests can be found for four different Heston term structures of various lengths and using different parameter combinations. All parameters used for these tests can be found in Appendix C. Note that index i represents the i-th row of a table.

The results for COS1, which is the implementation of the COS method with parameter values according to the original article, confirm the claims that have been made in Section 5.1 when comparing to the results of COS2. These were the claims that the newly proposed truncation range (5.13) will lead to more accurate option prices than the original one (5.8). For each of the four test-cases the results of COS2 are more accurate than those of COS1 for the same values of L and N.

When comparing the results of COS3 and COS2, all the claims made in Section 5.2 are supported. This tells that the formula (5.23) gives an appropriate and usable lower bound on the

Method	C_i	$\epsilon_{i,9}$	t_i/t_4	N	L	$[a,b]$
COS1	4.003806670	1.42e-05	0.338	160	10	[-2.779,2.764]
COS2	4.003774830	2.22e-05	0.604	160	10	[-3.220,2.764]
COS3	4.003863536	2.10e-08	0.655	870	10	[-3.220,2.764]
COS4	4.003863620	1.78e-15	1.000	2034	24	[-7.717,6.643]
COS5	4.003863620	2.53e-11	1.914	1206	14	[-4.505,3.872]
Lewis	4.003863625	1.21e-09	13.148	-	-	-
QE	[3.992, 4.018]	2.81e-04	42081.609	-	-	-
QEM	[4.004, 4.031]	3.46e-03	60331.270	-	-	-
Reference	4.003863620	0	75.434	131072	32	[-10.287,8.860]

Table 5.9: Comparison of all the solvers for parameter set 1. $t_4 = 1.146 \cdot 10^{-2}$.

Method	C_i	$\epsilon_{i,9}$	t_i/t_4	N	L	$[a,b]$
COS1	1.852601295	6.47e-03	0.146	160	10	[-2.278,2.271]
COS2	1.853748970	7.10e-03	0.272	160	10	[-2.419,2.271]
COS3	1.840678100	2.35e-06	0.558	2886	10	[-2.419,2.271]
COS4	1.840682426	2.56e-13	1.000	6752	24	[-5.801,5.456]
COS5	1.840682425	2.37e-10	3.233	5106	18	[-4.352,4.091]
Lewis	1.840686703	2.32e-06	13.120	-	-	-
QE	[1.831, 1.856]	1.59e-03	19848.066	-	-	-
QEM	[1.830, 1.857]	1.46e-03	28391.717	-	-	-
Reference	1.840682426	0	37.470	131072	32	[-7.734,7.275]

Table 5.10: Comparison of all the solvers for parameter set 2. $t_4 = 2.509 \cdot 10^{-2}$.

amount of terms that should be used in the Fourier cosine expansion for a given truncation range and value of L.
A predictable consequence of increasing the value $L = 10$ to $L = 24$ is that the results of COS4 are superior to the previous three (i.e. COS1, COS2 and COS3) in terms of accuracy. Note that runtime normalization has taken place with respect to the runtime of COS4.

Looking at the adaptive version of the COS method, COS5, the accuracy is always lower than that of COS4. When looking at the final values of L for COS5 it becomes clear why this is the case: the value is always smaller than 24, which is the L-value used for COS4. This is the result of the choices of $TOL = 10^{-8}$ and $L_0 = 8$. If for example $TOL = 10^{-12}$ and $L_0 = 12$ would have been chosen, it would definitely lead to more accurate results, perhaps even more accurate than COS4. However, for the choices of $TOL = 10^{-8}$ and $L_0 = 8$, generally speaking COS5 is slower than COS4, and changing the parameter values to $TOL = 10^{-12}$ and $L_0 = 12$ would probably lead to a slowdown of the method. Once again the trade-off between speed and accuracy becomes visible. All in all it can be said that COS4 is a more suitable choice of method than COS5 in terms of speed and accuracy. However, in case robustness is an important requirement for the application or by the user, or if a user is inexperienced and does not fully understand the problem at hand, the adaptive implementation of the COS method can prove to

Method	C_i	$\epsilon_{i,9}$	t_i/t_4	N	L	$[a, b]$
COS1	3.382122779	1.52e-11	0.589	160	10	[-1.243,1.236]
COS2	3.382122779	3.02e-13	1.076	160	10	[-1.425,1.236]
COS3	3.382122779	3.02e-13	0.712	168	10	[-1.425,1.236]
COS4	3.382122779	2.40e-14	1.000	396	24	[-3.416,2.971]
COS5	3.382122779	3.02e-13	0.822	168	10	[-1.425,1.236]
Lewis	3.382122779	4.69e-11	19.271	-	-	-
QE	[3.368, 3.387]	1.44e-03	97257.615	-	-	-
QEM	[3.369, 3.388]	1.02e-03	101877.194	-	-	-
Reference	3.382122779	0	235.582	131072	36	[-5.122,4.459]

Table 5.11: Comparison of all the solvers for parameter set 3. $t_4 = 4.828 \cdot 10^{-3}$.

Method	C_i	$\epsilon_{i,9}$	t_i/t_4	N	L	$[a, b]$
COS1	37.81483822	1.73e-01	0.061	160	10	[-205.6,202.5]
COS2	38.23898291	1.86e-01	0.114	160	10	[-205.6,225.1]
COS3	32.23260086	1.75e-08	0.486	8316	10	[-205.6,225.1]
COS4	32.23260143	4.76e-14	1.000	19374	24	[-491.3,542.3]
COS5	32.23260143	7.22e-11	1.950	11512	14	[-287.2,315.7]
Lewis	32.23260143	2.01e-11	9.164	-	-	-
QE	[16.353, 80.404]	5.01e-01	7201.227	-	-	-
QEM	[31.350, 32.203]	1.41e-02	11139.080	-	-	-
Reference	32.23260143	0	172.148	131072	72	[-1470.8,1629.9]

Table 5.12: Comparison of all the solvers for parameter set 4. $t_4 = 6.861 \cdot 10^{-2}$.

be useful because of its simplicity.

The implementation of Lewis' method is both less accurate and significantly slower than COS4. Even COS5 is faster and on occasion more accurate. Note that the COS method and both Monte Carlo methods have been implemented in Matlab, whereas Lewis' method is a Matlab interface to the source code written in Fortran. Fortran is a compiled language, as opposed to Matlab, which is an interpreted language using a so-called 'Just In Time' compiler. Fortran is known to be a much faster language than Matlab. Thus, if a Matlab implementation of the COS method is faster than a Matlab interface to a routine in Fortran it means that a Fortran implementation of the COS method is likely to be even faster than Lewis' method called directly from Fortran.

Both Monte Carlo methods confirm that the prices reported by COS and NAG are correct, guaranteeing the correctness of implementations. Looking at speed and size of the confidence interval of both QE schemes, there is possible room for improvement of performance. This can be achieved by for example reusing random samples in a clever way or using variance reductions techniques such as antithetic variates, control variates, importance sampling and stratified sampling. Furthermore, note that the martingale corrected version of Andersen's QE scheme does not always perform better than the original QE scheme. This is in line with the statements

from the original article by Andersen [2]. There he states the following about enforcing the martingale condition: *"The practical relevance of this is often minor, as the net drift away from the martingale is typically very small and controllable by reduction in the time-step. Also, the ability to hit the mean of the distribution for X does not necessarily translate itself into better prices for options."*

In conclusion, everything depends on the desired accuracy. The goal is to then achieve this level of accuracy as fast as possible. Of all the methods proposed in Chapters 4 and 5, COS4 can be considered to be the best method in terms of addressing the trade-off between speed and accuracy. So manually setting a large value of L and using the formula for N from (5.23) with ε equal to machine precision is the most suitable choice. Note that the choice of L must not be too large, because at some point that could cause the runtime to increase while not gaining more accuracy. This requires some experience from the user. A good alternative for the less experienced user or in case robustness is a must, the adaptive implementation of the COS method is a suitable alternative. This method allows one to set the amount of digits of the price that have to be correct by the choice of TOL.

Chapter 6

Calibration

Up to this point the subject of interest has been the pricing of European options in a fast, accurate and robust way. As one might expect this is not the only type of option contract being sold in the markets. Other, more complicated, products called exotic options are an example of other types of contracts. Typically theses exotic options have a more complicated payoff than the standard European options, and therefore different techniques must be applied for the pricing of these options, as the old techniques are not applicable. What is done in practice is the valuation of exotic options using Monte Carlo techniques. However, before one can start applying this technique, all parameter values of a model must be known. To achieve this, the model is calibrated to a set of market data (FX data in this thesis) that contains European option quotes. This calibration procedure involves many evaluations of a pricing technique, so a quick and accurate way of calculating prices is essential. This issue has been addressed in Chapters 4 and 5.

Another issue that could play a crucial role in the speed and accuracy is the type of calibration procedure one chooses. The way this is typically done is by performing a numerical minimization of a least-squares objective function containing both the model prices and the market prices. The choice of numerical optimization technique could possibly play a crucial role when it comes to speed: one method may be faster than the other or may require fewer price evaluations, both resulting in a faster calibration. Therefore it is useful to know which optimization technique is the most suitable for the calibration of the Heston model with term structure. This will be done by setting up a so-called benchmarking suite for testing the performance of several different numerical optimization techniques. The 1355 separate dates with option quotes from the FX market will form 1355 different test-cases that each of the optimizers will calibrate the Heston model with term structure to. A comparison in terms of speed and accuracy will be done to see which of the optimization techniques is the most suitable for calibrating this model to this type of data.

In this chapter the calibration of the Heston model with term structure is performed. During this calibration procedure, the model parameter values are chosen such that the model replicates the quoted market prices as good as possible. Before doing this it is useful to have an idea what the effect of each parameter is on the option price. Note that an option price has a one to one relationship with the implied volatility: a higher implied volatility implies a higher option price. Therefore, in Section 6.1 the effect of each parameter on the implied volatility curve will be studied. In Section 6.2 the methodology of calibration will be explained, including the choice of objective function, initial guess and stopping criterion for optimization. Furthermore, in Section 6.3, the use of derivatives during the calibration will be addressed, including an introduction of the concept of algorithmic differentiation (AD) as an alternative to finite difference approxima-

tions of the derivatives. In Section 6.4 several basics of numerical optimization are given, after which a list of numerical optimization techniques is given that take part in the benchmarking suite. Methods that use derivatives as well as ones that do not will be considered. Finally in Section 6.5 the coding framework will be addressed. Furthermore, results of the calibration can be found, and all the different optimization techniques are compared.

6.1 Parameter effects on implied volatility

Before starting the calibration of a model right away, it is useful to gain some feeling for how the model parameters affect the market prices. Remember that an option price has a one to one relationship with the implied volatility: a higher implied volatility implies a higher option price. What is thus usually done is study the effects of model parameters on implied volatilities. The typical approach for this is setting all but one parameter constant, and vary this parameter over a range of numbers and see what effect this has on the implied volatility curve (i.e. plot of implied volatility σ_{imp} versus different strike values K).

Note that all these model prices come from an implementation of the Heston model with term structure where a flat term structure is taken, i.e. a term structure with parameters constant such that the model is behaving the same as the single-term Heston model. The reason to do so is because currently the only goal is to find out what the effects of the different parameters are on the implied volatility curve given all the other parameter values. These effects will be used later on at the end of Section 6.2.1 to decide which of the parameters to fix constant during the calibration to avoid an underdetermined optimization problem. The results of this investigation can be found in Figure 6.1.

In each of the plots all parameters but one are kept constant. If a parameter is not changed, it is set to its default setting:

$$t_0 = 0,\ T = 1,\ r_d = 0.05,\ r_f = 0,\ S_0 = 1,\ v_0 = 1,\ \alpha = 0.5,\ \sigma = 0.2,\ \rho = 0,\ \lambda = 1.$$

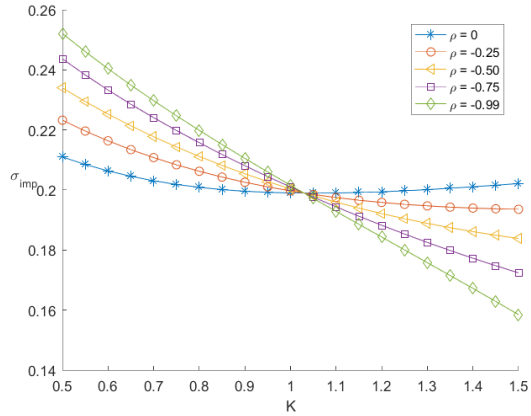
From Figure 6.1 it is clear that parameter ρ affects the symmetry, α affects the curvature of the volatility smile, λ affects the curvature as well and finally σ determines the level of the smile. Note that v_0 is deliberately not changed here because a normalization has been applied to the Heston model to end up with the model formulation (3.4). In this formulation, the value of σ determines the influence of the volatility process on the stock process.

6.2 Calibration procedure

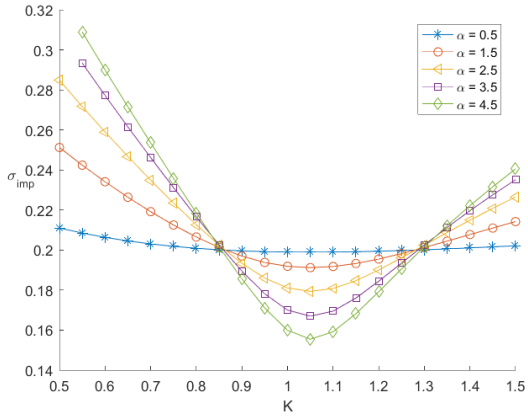
In this section an explanation will be given of the methodology of calibration. The choice of objective function, term structure and which parameters to calibrate can be found in Section 6.2.1. Subsequently in Section 6.2.2 the parameter constraints and initial guesses are discussed. Afterwards in Section 6.2.3 the choice of stopping criterion for the optimization will be justified. Finally in Section 6.2.4 the choice not to impose the Feller condition during calibration will be explained.

6.2.1 Type of calibration

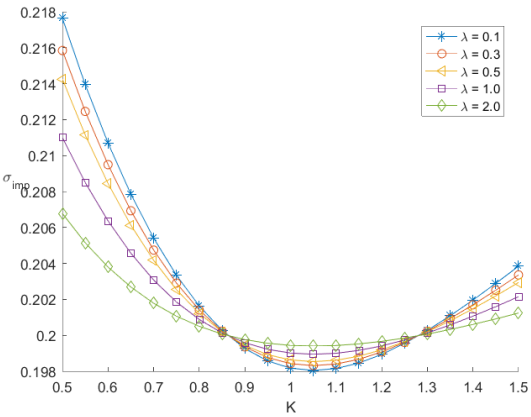
As mentioned before, the calibration of a model is typically done by making a model reproduce current market prices. However, this is not the only approach that is possible, as it is for example also an option to match historical data. In this case a time series of historical data of for



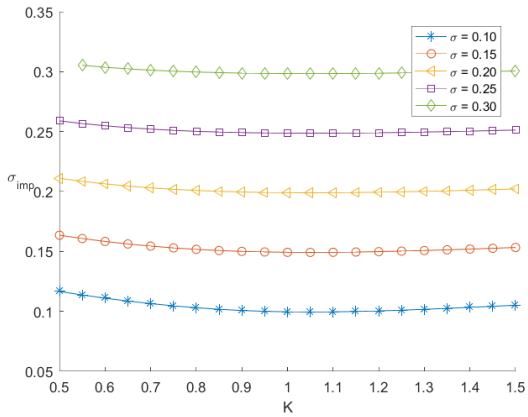
(a) σ_{imp} for different values of ρ.



(b) σ_{imp} for different values of α.



(c) σ_{imp} for different values of λ.



(d) σ_{imp} for different values of σ.

Figure 6.1: Implied volatility curves for different parameter settings.

example asset prices or option prices is formed from the market. It is then possible to use for example GARCH model [7] to fit the data and try to extract some information of for example the variance and its mean reversion. Others decide to use for instance filtering methods, Bayesian methods or a maximum likelihood approach. This approach can prove to be very useful from a risk management point of view. However, when pricing exotic options after calibrating a model this historical approach is never used. This has to do with the fact that options are always priced under the risk-neutral measure $\mathbb{Q}$, and the historical approach aims to mimic the process over time under the physical $\mathbb{P}$-measure. Furthermore, even though a perfect fit for the historical dataset on option prices can be obtained, there is still a chance that the European option quotes from today will be mispriced. Once this happens there is possible risk of arbitrage, which is of course not desired. Therefore the earlier mentioned market-implied approach of matching current option quotes will be taken as the method of calibration. In this way, a $\mathbb{Q}$-measure model is fitted to data from the real world, which are associated with the $\mathbb{P}$-measure. This results in parameters for a model under the $\mathbb{Q}$-measure, which can then be used for the pricing of exotic options under the $\mathbb{Q}$-measure. The approach of matching current market prices typically involves the minimization of an objective function using numerical optimization techniques.

Before starting the calibration all the necessary information has to be extracted from the supplied market data. First of all, strikes are calculated according to the quoting convention used,

as explained in Section 2.3.4. After that, the corresponding Black-Scholes option prices C_i^{market} are calculated by plugging in the strikes and the implied volatility quotes $\sigma_{imp,\,i}^{market}$ into the Black-Scholes formula (2.1). Note that this conversion from implied volatility option quotes to option prices using the Black-Scholes formula is a market convention. Now that the market option prices C_i^{market} have been calculated, the goal is to find the parameter regime under which model prices C_i^{model} are possibly close to the market prices. This is typically formulated as a least-squares minimization problem, which from now on will be referred to as the minimization of an objective function.

It might seem natural to formulate the minimization problems in terms of implied volatilities as there is a one to one relationship between this and the market prices. This would mean an objective that looks like:

$$\min \frac{1}{2} \sum_{i=1}^{m} \left(\sigma_{imp,\,i}^{market} - \sigma_{imp,\,i}^{model} \right)^2, \tag{6.1}$$

where the attempt is to fit a total of m market quotes. The choice is made not to take this approach because pricing methods as for example the ones described in Chapter 4 return the value of an option contract in terms of a price and not in terms of implied volatility. Thus, in order to use objective function (6.1), a conversion from model prices to implied volatilities has to be performed. This typically takes place using a root-finding algorithm such as Newton-Raphson's method or Brent's method. This would imply a lot of computation work to perform a root-finding for every market quote. As speed of calibration is of high importance, this does not seem a suitable choice of objective.

On the other hand, looking at objective (6.2):

$$\min \frac{1}{2} \sum_{i=1}^{m} \left(C_i^{market} - C_i^{model} \right)^2, \tag{6.2}$$

the model prices are retrieved immediately by the pricing method and the market prices can be computed using the closed form solution of the Black-Scholes model. It is trivial that this conversion using a closed form solution will take less computing time than using a root-finding algorithm. Therefore the second objective function (6.2) is chosen to be used as an objective function for the numerical optimization procedure.

Note that up to now only least-squares objectives have been considered. A possible argument against such a formulation is given by Bakshi et al. [5], who state that the objective function defines as the sum of squared errors may force the assignment of more weight to relatively expensive options (in-the-money (ITM, means that the derivative makes money if it were to expire today) and long-term options) and less weight to short-term and out-of-the money (OTM, means that the derivative makes no money if it were to expire today) options.

Remember that a call option is OTM if the strike is larger than the forward value of the option. If the two are equal, the option is said to be ATM, and finally the case the strike is smaller than the forward is characterized as ITM. For a put option these results are the other way round, so a strike larger than the forward results in a ITM option and a strike smaller than the forward results in a OTM option. See Figure 6.2 for an illustration of the situation.

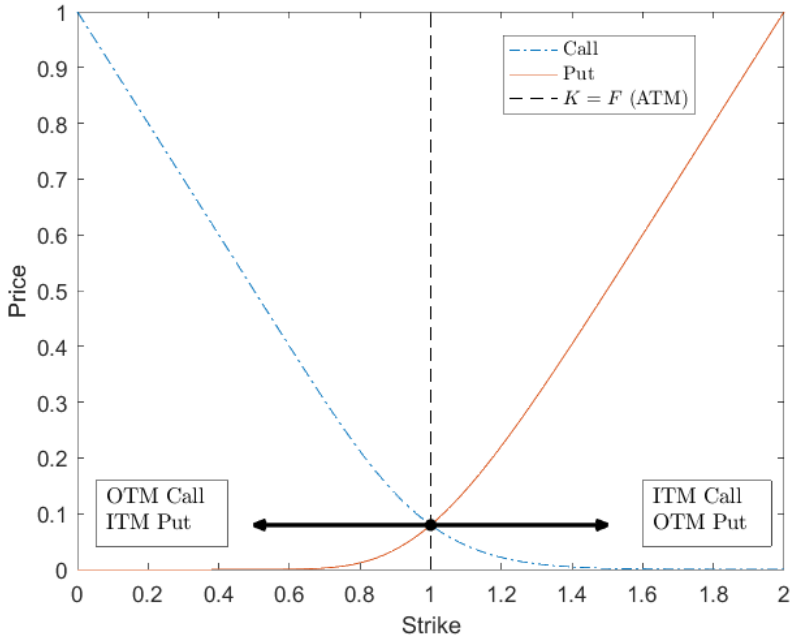


Figure 6.2: Visualization of OTM and ITM for calls and puts.

An alternative choice of objective function could be:

$$\min \sum_{i=1}^{m} \frac{\left|C_i^{market} - C_i^{model}\right|}{C_i^{market}}.$$

However, this would favour the cheaper option prices, so this does not seem a suitable objective function. Another attempt to fix the issue of favouring the more expensive options is to introduce weighting into the objective function (6.2) as follows:

$$\min \frac{1}{2} \sum_{i=1}^{m} w_i \left(C_i^{market} - C_i^{model}\right)^2, \tag{6.3}$$

where $w_i > 0$ is a weight. The reason to make this choice is that the prices further out into the wings of the implied volatility curve do not have to be replicated with an equal level of accuracy as the ATM price. Note that this has to do with liquidity as stated at the start of this section. The ATM quote is typically given the largest weight and the further away from this quote the less weight is assigned. A possible way of assigning weight values is using the so-called Black-Scholes vega weighting as proposed in [35] by Hamida and Cont. Other possible ways of objectives involve adding a regularization term to the least-squares formulation which is a convex penalty function. An example of such a regularization term can be found in the article by Cont and Tankov [17].

In this thesis, an implicit type of weighting is used. Typically when one uses objective function (6.3), OTM options are being priced as those are in general far more liquid than ITM prices. Furthermore, OTM prices are smaller than ITM prices, resulting in smaller numbers used in the objective function and therefore in implicit weights. The ATM quote is implicitly given the most weight and therefore is considered as the most important. In this way, the objective function (6.2) can be used and later on in Section 6.4 it will become clear that the least-squares

structure can be exploited during the minimization of the objective function, resulting in a faster calibration.

As stated in Section 2.3.4, the FX datasets contain information about the initial spot exchange rate S_0 and for each different maturity a foreign and domestic rate, several different x-Δ RR and BF quotes and an ATM quote. The most liquid quotes are the 25-Δ RR, 25-Δ BF and the ATM quote. When calibrating a model to a set of market data, a minimal requirement is to have a very good fit for the most liquid quotes, since those are traded the most. Therefore the choice has been made to use three quotes per maturity. Using maturities 2m, 3m, 6m, 1y, 2y, 3y and 5y this results in 21 different option quotes the model can be calibrated to. Note that even though there are more option quotes available than these seven, for example for shorter maturities, the Heston model is not capable of matching the extreme skews that typically exist for short term FX quotes [30]. Furthermore, for maturities larger than 5 years, interest rates come to play a more significant role, so a multi-factor model would be more suitable in those situations.

When looking at the Heston model with term structure in Section 3.2, this will imply the following term structure from $t_0 = 0$ to $T = 5$:

$$[0,5] = \underbrace{[0,0.1667]}_{\tau_1} \cup \underbrace{[0.1667,0.25]}_{\tau_2} \cup \underbrace{[0.25,0.5]}_{\tau_3} \cup \underbrace{[0.5,1]}_{\tau_4} \cup \underbrace{[1,2]}_{\tau_5} \cup \underbrace{[2,3]}_{\tau_6} \cup \underbrace{[3,5]}_{\tau_7}. \tag{6.4}$$

The choice has been made to minimize the objective function (6.2) separately for all different terms τ_i from the term structure. The way the calibration is set up is called a bootstrap calibration, or simply a term-by-term calibration. The way this is done is choosing an initial guess for all parameters for the first term and minimize objective function (6.2) for the three quotes for maturity $T = 0.1667$ (so $m = 3$ in (6.2)). After this first calibration, the initial guess for the calibration of the second term will be the calibrated parameters of the first term. The initial guess for the calibration of the third term will be the calibrated parameters of the first and second term together. This procedure continues until the calibration of the seventh term has finished. At this point a set of 21 parameters has been found that make sure the Heston prices match the market prices. Note that calibrating in this fashion reduces the dimension of the calibration from 21 to 7 separate calibrations of dimension 3, which will most likely lead to a speed-up of the calibration. One should keep in mind however that by doing this, it is assumed that the current term is independent of all other terms. This assumption is just for the forward dependence in time: the option price of maturity T_1 will definitely not depend on the price of an option with maturity $T_2 > T_1$. On the other hand, the other way around this argument is not necessarily true. In order to see if the bootstrap calibration is a suitable way of doing a calibration, the results from this calibration will be compared with a calibration where backward dependence in time is possible, so a calibration of dimension 21. These results can be found in Section 6.5.

Before continuing, note that the Heston model with term structure has four unknown parameters per maturity $t = T$, namely σ_t, λ_t, α_t and ρ_t. Since the choice has been made to calibrate only to three market quotes, one of the four parameters must be fixed constant while keeping the others to be determined during the calibration. The reason this must be done is that calibrating a model with four unknowns to a set of three market quotes leads to an underdetermined optimization problem.
In Section 6.1 it has been shown that parameters λ_t and α_t have the same effect on the volatility smile, namely the effect on curvature. Therefore the choice has been made to set one of the two parameters constant over all periods of the term structure. The choice is made to set λ constant

and not α because the latter is the volatility of volatility (a.k.a. volvol) parameter which is crucial for the model to be a useful stochastic volatility model at all. If one were to fix α instead of λ, the scaling of the stochastic part of the variance process would be constant, which would result in a model that would not be able to capture the curvature from the market in a good enough manner.

6.2.2 Parameter constraints and initial guesses

Note that some of the Heston parameters have bounds that must be imposed during calibration. Correlation parameter ρ must by definition be in the interval $[-1, 1]$. Furthermore the following two bounds must hold: $\sigma > 0,\ \alpha > 0$. Note that for these two parameters there is only a lower bound that must be imposed. However, the decision is made to also set an upper bound during the calibration on those two parameters to make sure that the calibration does not take forever if the solver is not able to fit the model to the market without letting a parameter reach very large values. Therefore an upper bound of 20 is set for both σ and α. Note that this bound is large enough for both parameters, as these values are typically not observed from market data.

Depending on the optimizer that is used, the result of the calibration may depend heavily on the initial parameter values used before starting to calibrate the first term. It can be observed from market data that there is a relation between model parameters and the BF and RR. This can be explained using equations (2.11) and (2.12). Looking at the equation for the butterfly one can see that this simply measures the curvature of the volatility smile since it measures the difference in implied volatility between the delta-neutral straddle and the average of the two 25-Δ options. Recall from Figure 6.1b that α had an affect on the curvature of the smile as well. In the same way, the risk reversal measures the difference between the two 25-Δ options, thus it is a measure of symmetry, just as correlation ρ is (see Figure 6.1a).
Since a normalization has taken place to get from Heston formulation (3.1) to (3.4), the choice of initial variance level $v_0 = 1$ is straightforward.
Furthermore, from Figure 6.1d it is clear that parameter σ affects the level of the volatility smile. Therefore the choice is made to set the initial guess for this equal to the ATM volatility quote for the first maturity.
Finally for parameter λ, which will be fixed constant over all the maturities as described in Section 6.2.1, the choice has been made to set it to 2.5 [4].

This choice might seem somewhat arbitrary but with help of market practitioners this value has been chosen.

A question that might come to mind after having set the initial guess for the first term is the sensitivity of the calibration results to the initial guess. In order to investigate whether different initial guesses lead to different minima of the objective function during the numerical minimization some numerical tests have been performed. For this a grid of initial guesses has been made for each parameter, resulting in a three-dimensional grid of starting points. For each of the set of initial guesses the calibration of the first term has been performed, and the outcome is that different initial guesses still lead to the same minimum. The only difference between the various calibrations is the speed: an initial guess further away from the actual solution will lead to a slower calibration than an initial guess that is close. Therefore it is crucial for the speed of computations to have an initial guess close to the solution, but one does not need to worry about the solver ending up at a different solution.

6.2.3 Stopping criterion

When doing a numerical minimization of objective function (6.2), a measure of accuracy must be defined in order to know when the result can be considered as satisfactory. When a certain level of accuracy is achieved during the optimization, the optimization algorithm is allowed to stop and continue with the calibration of the next term in the term structure. This level of accuracy should display the requirements of a user in terms of accuracy of matching prices and the speed of the calibration, depending on the application of the calibrated results. Possible applications are the pricing of exotic options using Monte Carlo techniques and calculating option price sensitivities for risk purposes.
A possible goal would be to replicate prices in units of notional with an accuracy of 1 basis point (bp), i.e. 0.01%. Translating this into an accuracy criterion gives:

$$\max_{1\leq i\leq 3}\left|\frac{C_i^{market}}{S_0}\cdot 100\% - \frac{C_i^{model}}{S_0}\cdot 100\%\right| \quad < \quad 0.01. \tag{6.5}$$

This measure of accuracy has been chosen as a stopping criterion for the solvers. Of course there are other possible choices of accuracy, such as demanding that the sum of squares in (6.2) is below a certain tolerance level. Since market practitioners find a matching of prices on the bp level sufficient for for example the pricing of exotic options, the choice of the criterion in (6.5) seems more appropriate than imposing a condition on the sum of squares.

One must keep in mind that on assumption that a solver has fully converged, i.e. the minimum has been found, all measures of accuracy should give an error value of zero. Since the criterion (6.5) states that a solver can stop the search for a minimum at the point the market prices fit the model prices with an accuracy of one basis point, this error is not equal to zero. Therefore, different optimization algorithms might give slightly different answers at the same level of accuracy. In Chapter 7 this is studied in more detail in terms of the error made during a simple hedge test.

6.2.4 Feller condition

In the FX market it is typically the case that the Feller condition (3.5) is not satisfied when trying to calibrate the Heston model. Even though from a modelling point of view this condition is essential, practitioners do not worry about this too much. Spreij et al [64] state that the differences between imposing or not imposing the Feller condition in practice are economically negligible. The reason the Feller condition is typically not imposed is because the FX smiles are very convex, and a lot of curvature is required from the Heston model to capture the convexity of the smile. Clark [13] states the following: *"The correlation ρ isn't part of the Feller condition, but in order to introduce a volatility skew, the volvol needs to be large enough to allow the correlated part of W_t^2 to feed through into a volatility skew for x_t. ... typical FX market conditions will require this volvol parameter to be large enough that the Feller condition is often violated in practice. ... you can only get so much convexity out of a Feller condition compliant Heston model."* Therefore the choice is made to ignore the Feller condition (3.5) during the calibration of the Heston model.

6.3 Derivatives

Typically during the calibration of a model an optimization technique is used to make sure that the model fits the market data with sufficient accuracy. Most of those techniques use derivatives

of the objective to compute search directions. For the Heston model without term structure it was possible to manually derive an analytic expression of the derivatives needed for the search directions of the optimizer. Unfortunately, for the Heston model with term structure this is not possible. Therefore, many practitioners approximate derivatives using finite differences (FD). As will be explained later on in this section, the method of algorithmic differentiation (AD), or sometimes it is referred to as automatic differentiation, is sometimes preferred over finite differences. A brief summary of the concept of algorithmic differentiation will be given in this section based on the book by Uwe Naumann [55].

There are numerous reasons why derivatives are needed. In the case of the Heston model with term structure calibration, derivatives are needed to determine search directions during the optimization process of the objective function. But in a more general sense, derivatives are required in many other fields next to Finance. A popular application that is present in many different fields of expertise is the calculation of parameter sensitivities using derivatives.

Derivatives can be obtained in various ways. The following methods are often used:

1. an analytical derivation of the derivatives using calculus,

2. finite difference approximation,

3. algorithmic differentiation.

See Sections 6.3.1, 6.3.2 and 6.3.3 respectively for more information.

6.3.1 Analytic derivatives

Let it be clear that if the first of the three methods is applicable, it is preferred to explore whether this method is useful in a practical application. However, this is not always possible. Even if this is possible it can prove to be quite troublesome to do in practice. Typically the function that needs to be differentiated is not given in closed form but implemented in many lines of code. Deriving an analytical derivative can be difficult in those cases because:

- The function can be implemented using lots of lines of code. This implies that the user must fully understand what is happening at any point in the code. Even when that requirement is fulfilled, it takes a lot of time and effort to derive an analytic expression in most cases.

- The function might not be given explicitly but implicitly for example as a solution of an optimization problem.

- The function could be a simulation of some process.

- Once a derivative is obtained, both the original and derivative code must be maintained: if something is changed in the original code, the derivative code must be changed as well.

All in all, this is not what a user would want to go through every time a derivative is needed. Therefore, in many cases finite differences are used to approximate the derivatives one needs. See Section 6.3.2 for more information.

6.3.2 Finite differences

Generally speaking, in optimization forward and central finite differences are used for approximating the first derivatives $f'(x)$ of a certain function $f(x)$ with respect to x.

$$\text{Forward:} \quad f'(x) = \frac{f(x+h) - f(x)}{h} + \mathcal{O}(h), \tag{6.6}$$

$$\text{central:} \quad f'(x) = \frac{f(x+h) - f(x-h)}{2h} + \mathcal{O}(h^2). \tag{6.7}$$

However, these approximations are known to cause problems in some cases. At a first glance it might seem that a smaller perturbation size $h > 0$ would lead to a better approximation of the required derivative. This is due to the mathematical definition of a derivative:

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}.$$

However, precision is lost during computations because computers only have finite precision arithmetic. When a number cannot be represented exactly by a number within the floating-point number system on the computer, rounding takes place. Rounding simply means that a nearby floating-point number is chosen as an approximation of the actual number. When two floating-point numbers are almost equal, that is they agree on all digits except for the last few, the difference between the two numbers may only have a few digits of accuracy due to the so-called catastrophic cancellation. When this difference is used in further calculations, errors will occur due to rounding and will propagate in the next set of computations. For finite differences this concept is applicable due to the subtraction of two or more numbers. The latter means that a first order finite difference has been used, and when higher order finite difference approximations are used, the effect of catastrophic cancellation and rounding will increase dramatically [55].
Another problem caused by the use of finite difference approximations is that the computational cost of computing first order derivatives is linear in the number of inputs. In an application where the amount of inputs is large, this will lead in many function evaluations to approximate all the derivatives.

6.3.3 Algorithmic differentiation

In an attempt to cope with the many drawbacks of finite difference approximations, algorithmic differentiation (AD) can be chosen as a method for computing derivatives. One of the fundamental assumptions of AD is that at runtime a computer program can be regarded as a sequence of assignments with arithmetic operations (i.e. the addition, subtraction, multiplication and division of floating point numbers) or intrinsic functions on their right-hand side (for example sin, cos or exp). The method assumes that functions are continuously differentiable, otherwise there is no point in considering this method at all. Using this differentiability, AD is in essence an application of the chain-rule.
Two types of AD exist: tangent-linear version and adjoint version. First of all, consider the following definition of a tangent-linear model: the Jacobian of operator F:

$$\nabla F = \nabla F(\mathbf{x}) = \left(\frac{\partial y_j}{\partial x_i} \right)_{i=1,\ldots,n}^{j=1,\ldots,m},$$

where $\mathbf{x} = [x_1, x_2, \ldots, x_n]$, induces a linear mapping $\nabla F : \mathbb{R}^n \to \mathbb{R}^m$ defined by:

$$\mathbf{x}^{(1)} \mapsto \nabla F \cdot \mathbf{x}^{(1)}.$$

Now the function $F^{(1)} : \mathbb{R}^{2n} \to \mathbb{R}^m$ defined as:

$$\mathbf{y}^{(1)} \quad = \quad F^{(1)}(\mathbf{x}, \mathbf{x}^{(1)}) \quad = \quad \nabla F(\mathbf{x}) \cdot \mathbf{x}^{(1)},$$

is referred to as the tangent-linear model of F. The directional derivative $\mathbf{y}^{(1)}$ can be regarded as the partial derivative of $\mathbf{y}$ with respect to an auxiliary scalar variable s, where:

$$\mathbf{x}^{(1)} \quad = \quad \frac{\partial \mathbf{x}}{\partial s}.$$

By the chain rule, the following holds:

$$\mathbf{y}^{(1)} \quad = \quad \frac{\partial \mathbf{y}}{\partial s} \quad = \quad \frac{\partial \mathbf{y}}{\partial \mathbf{x}} \cdot \frac{\partial \mathbf{x}}{\partial s} \quad = \quad \nabla F(\mathbf{x}) \cdot \mathbf{x}^{(1)}.$$

The entire Jacobian can be accumulated by letting $\mathbf{x}^{(1)}$ range over the Cartesian basis vectors in $\mathbb{R}^n$.

The computational complexity of the tangent-linear approach to computing the first derivative of a multivariate vector function $F : \mathbb{R}^n \to \mathbb{R}^m$ grows linearly in the number of independent variables n. Note that this is the same order of computational complexity as finite differences, namely $\mathcal{O}(n)$. This number can possibly get very large for many real-world applications. Exact (up to machine precision) Jacobians can be computed using the tangent-linear mode of AD. Finite differences on the other hand, even though the computational costs are the same, inherently end up giving an inaccurate result for the Jacobian.

In the case where $m = 1$, i.e. $y \in \mathbb{R}$, an adjoint model can return the same gradient as a tangent-linear model significantly faster. Adjoints of $\mathbf{x}$ and $\mathbf{y}$ are defined as partial derivatives of an auxiliary scalar variable t with respect to $\mathbf{y}$ and $\mathbf{x}$ where

$$\mathbf{y}_{(1)} \quad = \quad \frac{\partial t}{\partial \mathbf{y}} \quad \text{and} \quad \mathbf{x}_{(1)} \quad = \quad \frac{\partial t}{\partial \mathbf{x}}.$$

First of all, take a look at the following definition of the adjoint of a linear operator $\nabla F : \mathbb{R}^n \to \mathbb{R}^m$, which is defined as:

$$(\nabla F)^* : \mathbb{R}^m \to \mathbb{R}^n,$$

where

$$\langle (\nabla F)^* \cdot \mathbf{y}_{(1)}, \mathbf{x}^{(1)} \rangle_{\mathbb{R}^n} \quad = \quad \langle \mathbf{y}_{(1)}, \nabla F \cdot \mathbf{x}^{(1)} \rangle_{\mathbb{R}^m},$$

and where $\langle \cdot, \cdot \rangle_{\mathbb{R}^n}$ and $\langle \cdot, \cdot \rangle_{\mathbb{R}^m}$ denote the scalar products in $\mathbb{R}^n$ and $\mathbb{R}^m$ respectively.
It can be proven that the following holds:

$$(\nabla F)^* \quad = \quad (\nabla F)^T.$$

An immediate consequence of the definition of an adjoint linear operator is that when the adjoint of the output, $\mathbf{y}_{(1)}$, is chosen orthogonal to the directional derivative $\mathbf{y}^{(1)}$, then the adjoint of the input $\mathbf{x}_{(1)} = \nabla F(\mathbf{x})^T \cdot \mathbf{y}_{(1)}$ is orthogonal to $\mathbf{x}^{(1)}$.
The next step is to study the definition of an adjoint model. The Jacobian $\nabla F = \nabla F(\mathbf{x})$ induces a linear mapping $\mathbb{R}^m \to \mathbb{R}^n$ defined by:

$$\mathbf{y}_{(1)} \mapsto (\nabla F)^T \cdot \mathbf{y}_{(1)}.$$

Now the function $F_{(1)} : \mathbb{R}^{n+m} \to \mathbb{R}^n$ defined as:

$$\mathbf{x}_{(1)} \quad = \quad F_{(1)}(\mathbf{x}, \mathbf{y}_{(1)}) \quad = \quad \nabla F(\mathbf{x})^T \cdot \mathbf{y}_{(1)}, \tag{6.8}$$

is referred to as the adjoint model of F.
An application of the chain rule results in:

$$\mathbf{x}_{(1)} \quad = \quad \left(\frac{\partial t}{\partial \mathbf{x}}\right)^T \quad = \quad \left(\frac{\partial \mathbf{y}}{\partial \mathbf{x}}\right)^T \cdot \left(\frac{\partial t}{\partial \mathbf{y}}\right)^T \quad = \quad \nabla F(\mathbf{x})^T \cdot \mathbf{y}_{(1)}.$$

These definitions and statements are all very technical and it is not straightforward why it is worth the effort. Therefore consider the following example where a sequence of function calls $f_1, f_2, \ldots, f_{m+1}$ transforms $\mathbf{x} \in \mathbb{R}^n$ to $\mathbf{x_1} \in \mathbb{R}^n$ to $\mathbf{x_2} \in \mathbb{R}^n$ and so on. Finally f_{m+1} will transform $\mathbf{x_{m+1}} \in \mathbb{R}^n$ to $y \in \mathbb{R}$:

$$\mathbf{x} \xrightarrow{f_1} \mathbf{x_1} \xrightarrow{f_2} \mathbf{x_2} \rightarrow \cdots \rightarrow \mathbf{x_m} \xrightarrow{f_{m+1}} y.$$

When interested in the gradient $\dfrac{\partial y}{\partial \mathbf{x}}$, apply the chain-rule:

$$\frac{\partial y}{\partial \mathbf{x}} \quad = \quad \frac{\partial \mathbf{x_1}}{\partial \mathbf{x}} \frac{\partial \mathbf{x_2}}{\partial \mathbf{x_1}} \frac{\partial \mathbf{x_3}}{\partial \mathbf{x_2}} \cdots \frac{\partial \mathbf{x_m}}{\partial \mathbf{x_{m-1}}} \frac{\partial y}{\partial \mathbf{x_m}}. \tag{6.9}$$

The expression at the right-hand-side of (6.9) is usually evaluated from left to right, even though mathematically speaking from right to left would result in the same answer. The choice from left to right seems natural because this corresponds to the order in which the program computes intermediate values to go from $\mathbf{x}$ to y. The unfortunate consequence of this choice is that all but the last multiplication are matrix-matrix multiplications, and the last one is a matrix-vector product. Looking from a computational perspective, it is obvious that matrix-matrix multiplications are much more expensive than matrix-vector products. Therefore consider doing the multiplications in (6.9) from right to left. This will turn all the multiplications into matrix-vector products, which gives a much more efficient computation of the derivatives. One must be careful however, since this right-to-left approach implies that the entire program must be run in reverse order. This provides a challenge since the data to compute $\frac{\partial y}{\partial \mathbf{x_m}}$ is only available at the end of the calculation. The solution is to run the programme forwards, store all relevant intermediate values and then step backwards. During every step, the Jacobian $\dfrac{\partial \mathbf{x_{i+1}}}{\partial \mathbf{x_i}}$ is computed from the data that has been stored during the forward run of the programme. The required matrix-vector product is performed and one can move on to the next iteration.

Looking back at the definition of an adjoint model in equation (6.8), when setting $\mathbf{y}_{(1)} = 1$, a single call of the adjoint model $F_{(1)}$ will give the complete vector of first order partial derivatives. So comparing the computational complexity of the adjoint model, which has complexity $\mathcal{O}(1)$, with that of the tangent-linear model, which has complexity $\mathcal{O}(n)$, the gradient is obtained much cheaper. However, there is a downside of the adjoint model which originates from the fact that before one can start iterating backwards, a forward run of the programme has to be done and all relevant data has to be stored. This dataflow problem that needs to be solved in order to implement $F_{(1)}$ unfortunately dominates the computational costs. On the other hand, one does get exact derivatives, so it will always be a method worth considering.

6.4 Optimization techniques

The calibration described in Section 6.2 is performed using numerical optimization techniques. Up to this point, minimal information on this way of optimization has been given. As an introduction to numerical optimization the book by Nocedal and Wright [57] can be consulted. Note that throughout this section all the material is composed of excerpts from various sources, such

as the book by Nocedal and Wright, and summarized to give a clear overview of all the relevant different optimization techniques.

First of all in Section 6.4.1 a guide for the classification of optimization problems is provided. Secondly, in Section 6.4.2 an introduction to derivative-based optimization methods is given, after which in Section 6.4.3 some examples can be found of solvers that will be used in this thesis belonging to the class of derivative-based methods. Finally in Sections 6.4.4 and 6.4.5 the same is done for derivative-free methods.

6.4.1 Classification of optimization problems

When optimization is performed, one must first ask what objective must be accomplished and make sure this is a quantitative formulation for the performance of a given system of interest. The objective usually depends on several variables that influence the system. The goal is to find the correct variable values such that the objective is optimized. Usually there are some constraints imposed on the variables of the system. All of the above combined results in a model formulation, for which an optimization algorithm can be used to solve the problem.
After a certain algorithm has been applied, the user must be sure that the solution is actually an optimal solution. In many cases there are the so-called optimality conditions that help to verify that the current set of variables indeed gives an optimal solution to the problem.

Before continuing the introduction of optimization techniques, the mathematical definition of an optimization problem is given. In mathematical terms, optimization means maximizing/minimizing a function subject to certain constraints. The following notation is used from now on:

- $\mathbf{x}$ denotes the *vector of variables*.
- f denotes the *objective function* which is a function of $\mathbf{x}$ and needs to be minimized/maximized during the optimization.
- c_i denotes the i-th *constraint function*. These functions are scalar functions of $\mathbf{x}$ that define certain (in)equalities that the vector of variables $\mathbf{x}$ must satisfy.
- $\mathcal{E}$ and $\mathcal{I}$ denote the set of indices for equality and inequality constraints, respectively.

Using this notation, a general optimization problem is defined as:

$$\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x}) \quad \text{subject to} \quad \begin{cases} c_i(\mathbf{x}) = 0, & i \in \mathcal{E}, \\ c_i(\mathbf{x}) \geq 0, & i \in \mathcal{I}. \end{cases} \tag{6.10}$$

When spoken of a feasible region, the set of points satisfying all the constraints is meant.

The choice of optimization algorithm is not always the same. There are many different categories of algorithms that address different types of optimization problems. The following choices help to classify different types of optimization.

- **Discrete versus continuous optimization.**
  In some optimization problems the variables only make sense if they take on values from a finite set. Typically these are integer values. This type of problem is called a discrete optimization problem. On the other hand, continuous optimization involves models containing variables that are able to take on any real value (within the bounds of any constraints imposed), which involves an uncountably infinite set. This type of problem

is called a continuous optimization model. The latter type of problems is usually easier to solve because the functions involved are smooth, making the use of objective and constraint information at a particular point $\mathbf{x}$ possible to obtain information about the function's behaviour in the neighbourhood of that point. On the other hand, in discrete problems, the behaviour of the functions may change drastically when changing from one feasible point to another.

- **Constrained versus unconstrained optimization.**
  When looking at definition (6.10), unconstrained optimization problems are those for which $\mathcal{E} = \mathcal{I} = \emptyset$. This type of problem can be found in many practical applications. One can also find this type of problem when reformulating a constrained optimization problem where constraints can be taken away and replaced by adding a penalty term to the objective function. Another way of reformulating a constrained optimization problem into an unconstrained one is to do a parameter conversion such that the bounds disappear. Possible parameter conversions are as proposed in the paper by Elices [25]: when $p_{\min} \leq p \leq p_{\max}$ then

$$p = \frac{p_{\max} - p_{\min}}{2} \left(1 + p_{\min} + \tanh \left(\frac{\tilde{p}}{100} \right) \right). \tag{6.11}$$

  Here p is the original, constrained parameter and $\tilde{p}$ is the new, unconstrained parameter. Another possible parameter conversion is[1]:

$$\begin{aligned} p_{middle} &= \frac{p_{\max} + p_{\min}}{2}, \quad p_{width} = \frac{p_{\max} - p_{\min}}{2}, \\ p &= p_{middle} + p_{width} \cdot \tanh \left(\frac{\tilde{p} - p_{middle}}{p_{width}} \right). \end{aligned} \tag{6.12}$$

  According to Gill et al. [31] one must always watch out with such parameter conversions, because extra non-linearity is introduced into the problem. This might result in very strange behaviour of the optimization.
  Constrained optimization problems arise when constraints play a crucial role in a model. These constraints can take the form of simple bounds, general linear constraints or non-linear inequalities representing a complex relationship among the different variables.
  When both the objective function and all constraints are linear, the problem is classified as a linear programming problem. Non-linear programming problems are those in which some parts of the constraints or the objective function are non-linear.

- **Global versus local optimization**.
  Typically an algorithm for a non-linear problem only seeks local solutions (a point at which the objective is smaller than that at all feasible points in the neighbourhood). A global solution (a point with the lowest objective of all feasible points) is often not found by many algorithms.
  In a convex programming setting, and more specific for linear programs, a local solution is a global solution as well. In the case of a general non-linear problem on the other hand, both for the constrained and unconstrained cases, there may be local solutions which are not global solutions.

- **Deterministic versus stochastic optimization.**
  In some optimization problems the model has some unknown quantities at the time it is formulated, so the model can not be fully specified. In stochastic optimization, additional

[1] From the CMinpack documentation [20].

information about the uncertain quantities in the model is being used to obtain a solution which optimizes the expected performance of the model. In deterministic optimization on the other hand, the model is completely known.
Note that the case of an optimization algorithm with randomized search methods cannot be considered as stochastic optimization but as a stochastic algorithm. This is a very subtle but crucial difference. The randomness in these stochastic algorithms can reduce the effect of modelling errors and help escape a local optimum and converge towards the global optimum.

- **Derivative-free versus derivative-based optimization.**
Traditional numerical optimization algorithms make frequent use of derivatives. However, as discussed in Section 6.3, analytic derivatives are not always available. One of the unfortunate characteristics of the Heston model with term structure is that there are no analytical expressions of the option price with respect to the different parameters. So an alternative way of retrieving the derivatives has to be found in order to still be able to use derivative-based optimization methods. Possible solutions are applying algorithmic differentiation or using finite difference approximations (see Section 6.3 for more information).
On the complete opposite side there is the category of derivative-free optimization algorithms. The reason for choosing this type of optimization method could be the cost of computing the objective function value, therefore making a finite difference approximation of the derivatives expensive. Also inaccuracy of finite differences could be a motivator for choosing this type of algorithm.

Using all these different classifications, the optimization problem relevant to this research can generally be classified as continuous, constrained, local and deterministic. Note that some different methods can be tried by reformulating the model, as for example described in the change from constrained to unconstrained problems. The choice for a derivative-free or derivative-based optimization method depends on the availability, accuracy and cost of computing the derivatives. Because the availability of derivatives plays a crucial role in determining the final outlook of the optimization algorithms, assume that the general classification described above holds and that the availability of derivatives determines the two classes of derivative-based methods (see Section 6.4.2) and derivative-free methods (see Section 6.4.4).

6.4.2 Introduction to derivative-based optimization methods

Knowing in which of the classes the optimization problem of interest lies, it is convenient to reformulate the definition in (6.10) in a more compact way. For this consider the following definition of the *feasible set* Ω, which defines the set of points $\mathbf{x}$ satisfying all the constraints:

$$\Omega \quad = \quad \{\mathbf{x} \mid c_i(\mathbf{x}) = 0, \quad i \in \mathcal{E};\ c_i(\mathbf{x}) \geq 0, \quad i \in \mathcal{I}\}.$$

This allows one to rewrite (6.10) as:

$$\min_{\mathbf{x} \in \Omega} f(\mathbf{x}) \tag{6.13}$$

When searching for a solution of (6.13), one must be able to tell when a current approximation is in fact also the optimal solution $\mathbf{x}^*$. For this there are optimality conditions, which come in two types:

- *Necessary conditions.*
  These conditions must be satisfied by any solution point.

- *Sufficient conditions.*
  These conditions on the objective and constraints are those that guarantee that a point $\mathbf{x}^*$ is in fact the optimal solution when satisfied.

Typically the optimality conditions are formulated in terms of the first and second derivatives. See Appendix D.1 for more information on the first and second order optimality conditions.

In optimization there exist two basic iterative approaches to find a minimum of (6.13). The first is the trust region approach and the second is the line search approach.
A trust region method uses a certain region around current iterate $\mathbf{x}_k$. In that region the method trusts a model m_k (for example a quadratic model) to be a good approximation of the objective function f. The step size is chosen to be the minimizer of this quadratic model in the trust region. So in effect the step length and direction are chosen simultaneously. If the step does not turn out to be as good as expected, the size of the trust region is reduced and the process of finding a minimizer of the quadratic model is repeated. One could say that the direction of the step changes when a change in trust region size takes place. The size of the trust region plays an important role in the effectiveness of a step. A too small region means that the algorithm misses the chance to take a significant step towards the minimizer of the objective function. On the other hand, in case of a trust region that is too large, the minimizer of the quadratic model may be far away from the actual minimizer of the objective function in the region, meaning the region size has to be decreased. Typically in algorithms, the region size is chosen according to how well the algorithm performs during the previous iterations. In case the model is reliable, produces good steps and accurately predicts the behaviour of the objective function along these steps, the region size may be increased, allowing for longer steps. Once a step fails, one knows that the quadratic model is a bad representation of the objective function in the current trust region. The natural thing to do is reduce the size of the trust region and try again. Usually the trust region framework is used when derivatives are available. In this case a quadratic model of the objective can be established using for example a Taylor-series expansion. In this case, if the size of the trust region is small enough, the approximation is accurate enough and the algorithm will make a successful step.
A line search method on the other hand does not use a region around the current iterate $\mathbf{x}_k$ but computes a search direction $\mathbf{p}_k$ and then decides on step length α_k. The next iterate is then defined as $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$. It is common for a line search algorithm to only consider search directions $\mathbf{p}_k$ that are descent directions, i.e. $\mathbf{p}_k^T \nabla f_k < 0$ must hold, where $f_k := f(\mathbf{x}_k)$. This guarantees that a reduction of f is possible when moving in this direction $\mathbf{p}_k$. Popular choices of search directions are:

- the steepest-descent direction $\mathbf{p}_k = -\nabla f_k$,

- the Newton direction, which is derived from a second-order Taylor series approximation, resulting in $\mathbf{p}_k = -(\nabla^2 f_k)^{-1} \nabla f_k$,

- quasi-Newton methods that replace the Hessian $\nabla^2 f_k$ in the Newton step by an approximation B_k. Examples of quasi-Newton methods are the symmetric-rank-one (SR1) formula [14] and the Broyden-Fletcher-Goldfarb-Shanno (BFGS) formula [10, 28, 33, 63].

- non-linear conjugate gradient methods of the form $\mathbf{p}_k = -\nabla f_k + \beta_k \mathbf{p}_{k-1}$.

The challenge in deciding on step length α_k lies in the trade-off between choosing it such that there is a large enough reduction of f and the fact that it must not take too long to choose the step length. Typically a set of possible choices for α_k is tried and the algorithm stops as soon as one of these values satisfies a certain condition. A popular example of these conditions are the Wolfe conditions [66, 67] conditions on sufficient decrease of the objective function and curvature. Remember that in the trust region framework the direction and step are computed simultaneously, but during a line search method the direction is computed first, and afterwards the step is computed.

The next step is to look at a special type of optimization problem, namely least-squares problems. In this type of problem the objective function f is of a special form:

$$f(\mathbf{x}) = \frac{1}{2}\sum_{j=1}^{m} r_j^2(\mathbf{x}). \tag{6.14}$$

Here, $r_j : \mathbb{R}^n \to \mathbb{R}$ is a smooth function that is referred to as a *residual*. From here on it is assumed that $m \geq n$. Define the *residual vector* $\mathbf{r} : \mathbb{R}^n \to \mathbb{R}^m$ as $\mathbf{r}(\mathbf{x}) = [r_1(\mathbf{x}), \ldots, r_m(\mathbf{x})]^T$. Using this definition, rewrite the objective function (6.14) as:

$$f(\mathbf{x}) = \frac{1}{2}||\mathbf{r}(\mathbf{x})||_2^2. \tag{6.15}$$

Now the *Jacobian* J is defined as the $m \times n$ matrix containing the first partial derivatives of the residuals:

$$J(\mathbf{x}) = \begin{bmatrix} \nabla r_1(\mathbf{x})^T \\ \nabla r_2(\mathbf{x})^T \\ \vdots \\ \nabla r_m(\mathbf{x})^T \end{bmatrix}.$$

Here, ∇r_j is the gradient of residual r_j. Using the Jacobian, the gradient and Hessian of the objective function are defined as follows:

$$\nabla f(\mathbf{x}) = J(\mathbf{x})^T \mathbf{r}(\mathbf{x}), \tag{6.16}$$

$$\nabla^2 f(\mathbf{x}) = J(\mathbf{x})^T J(\mathbf{x}) + \sum_{j=1}^{m} r_j(\mathbf{x}) \nabla^2 r_j(\mathbf{x}). \tag{6.17}$$

Typically, the Jacobian is cheap to compute, making the computation of the gradient (6.16) relatively easy. This implies that the first term of the Hessian (6.17) is cheap to compute as well, and for this part of the Hessian no second derivatives of the residuals are necessary. This first term of the Hessian is in many applications the most important term. The second term of the Hessian is usually negligible because either the residuals are close to affine near the solution, meaning that the $\nabla^2 r_j(\mathbf{x})$-term is small, or the residuals themselves are small. Typically in non-linear least-squares algorithms these properties of the Hessian are plentifully exploited.

A simple first method for solving the minimization problem of the non-linear objective function (6.14) is the Gauss-Newton method. In short, this method is a modified Newton method using line search. The idea is to minimize (6.14) while cleverly making use of the gradient and Hessian structures in respectively (6.16) and (6.17).
Recall that the 'standard' Newton method solves the equations:

$$\nabla^2 f_k \cdot \mathbf{p}_k^N = -\nabla f_k, \tag{6.18}$$

$$\Rightarrow \quad \mathbf{p}_k^N = -(\nabla^2 f_k)^{-1} \cdot \nabla f_k. \tag{6.19}$$

Note that the subscript k denotes the k-th iteration, whereas the subscript denotes the i-th entry of a vector. The Newton method is derived from a second-order Taylor approximation for $f(\mathbf{x}_k + \mathbf{p}_k)$ where $\mathbf{p}_k$ is a search direction:

$$f(\mathbf{x}_k + \mathbf{p}_k) \approx f(\mathbf{x}_k) + \mathbf{p}_k^T \nabla f_k + \frac{1}{2}\mathbf{p}_k^T \nabla^2 f_k \mathbf{p}_k \stackrel{\text{def}}{=} m_k(\mathbf{p}_k).$$

Assuming that $\nabla^2 f_k$ is positive definite, one obtains the Newton direction by searching for the vector $\mathbf{p}_k$ that minimizes $m_k(\mathbf{p}_k)$. Setting the derivative of $m_k(\mathbf{p}_k)$ gives the equation (6.18) with Newton step solution $\mathbf{p}_k^N$ in (6.19). Note that the assumption of positive definiteness of $\nabla^2 f_k$ is crucial because if this does not hold, $(\nabla^2 f_k)^{-1}$ may not exist.
Instead of solving (6.18), consider the following system of equations for obtaining the Gauss-Newton step solution $\mathbf{p}_k^{GN}$:

$$J_k^T J_k \mathbf{p}_k^{GN} = -J_k^T \mathbf{r}_k \tag{6.20}$$

Note that (6.20) is derived from (6.18) by using both the gradient of the objective function as defined in (6.16) and the Hessian approximation:

$$\nabla^2 f_k \approx J_k^T J_k. \tag{6.21}$$

As stated before, this approximation circumvents the trouble of having to calculate the residual Hessians $\nabla^2 r_j$, saving a significant amount of computational time.
When the approximation in (6.21) is close to perfect, meaning that the first term in (6.17) dominates the second, the convergence rate of the Gauss-Newton method is equivalent to that of Newton's method. This means that locally the Gauss-Newton method converges rapidly.
Whenever the Jacobian of the residuals J_k has full rank and $\nabla f_k \neq \mathbf{0}$, $\mathbf{p}_k^{GN}$ is a descent direction for f (i.e. a direction making an angle strictly less than $\frac{\pi}{2}$ radians with $-\nabla f_k$, which is guarantees a decrease in f), making it a proper direction for line search.
Implementations of the Gauss-Newton method often perform a line search in the direction $\mathbf{p}_k^{GN}$ where it is required that the step length satisfies some conditions guaranteeing sufficient decrease in f and some curvature conditions.

Next to the Gauss-Newton method there are many other methods for solving this type of problem, many of which are very similar to the Gauss-Newton method. One of those other methods is the Levenberg-Marquardt method [42, 46]. In the Levenberg-Marquardt method the same Hessian approximation is used as for the Gauss-Newton method, but instead of using line search, this method uses a trust region. This different approach guarantees that one of the pitfalls of the Gauss-Newton method is circumvented, namely the behaviour of the method when the Jacobian is (nearly) rank-deficient. Due to the same use of Hessian approximation, the local convergence properties of both methods are similar. More information on this method can be found in Appendix D.2.

Another type of method that is frequently used in practice is the class of Sequential Quadratic Programming methods. The general idea of a Sequential Quadratic Programming (SQP) method is to model the minimization problem at the current iterate $\mathbf{x}_k$ by a quadratic programming sub-problem, then use the minimizer of this sub-problem to define a new iterate $\mathbf{x}_{k+1}$. This means that a quadratic sub-problem needs to be designed such that it produces good new iterates. The SQP approach can be used in both line search and trust region frameworks and is appropriate for any problem size. SQP methods are especially effective when the minimization problem includes strong non-linearities in the constraints. For further reading on this method, consult Appendix D.3.

6.4.3 Examples of derivative-based solvers

As described in Section 6.4.1, certain classifications of optimization problems can be made. Assuming that one is dealing with a deterministic, continuous and derivative-based optimization problem, there is still some freedom of choice for (un)constrained and local versus global optimization. Next to these two possible choices there is also the matter of derivatives. Up to this moment, derivatives play a crucial role in the optimization methods discussed. Knowing that a derivative-based method is being used, one needs to use either finite difference approximations for the derivatives or apply AD in case no analytic expression of the derivatives will be available, which is assumed for now. In optimization, forward and central differences are typically used (see (6.6) and (6.7)). For the choice between FD and AD, the optimization algorithms within the derivatives-based class themselves do not change. For this comparative study several different derivative-based solvers have been used. From the NAG library the numerical optimizers e04fcc (unconstrained Gauss-Newton algorithm) [49], e04gbc (unconstrained Gauss-Newton algorithm) [50], e04unc (constrained least-squares SQP algorithm) [51] and e04wdc (constrained general SQP algorithm) [52] have been used. Furthermore the solvers IPOPT (interior point algorithm) [65], CMinpack (unconstrained Levenberg-Marquardt algorithm) [20], Levmar (constrained Levenberg-Marquardt algorithm) [44], Ceres (constrained Levenberg-Marquardt algorithm) [3] and Klaus Schittkowski's NLPLSQ (constrained least-squares SQP algorithm) [61] have been used. For each of the solvers a brief description can be found in Appendix D.5 on which algorithm it is based and what other relevant characteristics it has.

6.4.4 Introduction to derivative-free optimization methods

Derivative-free optimization methods are typically used when the objective function is expensive to compute and thus making finite difference approximations expensive as well. One would say at this point, why not use AD? The issue is that in some cases the objective function is like a black box, making the application of AD cumbersome. All the background information on derivative-free optimization is a summary of various chapters from Conn et al[15], Nocedal and Wright [57], Scheinberg [60] and Zhang et al [68].

There are three main classes of derivative-free optimization methods:

1. *"The first is a class of direct search methods that explore the variable space by sampling points from a predefined class of geometric patterns or that involve some random process. Some of these methods do not assume smoothness of the objective function and therefore can be applied to a broad class of problems"* [68].
   One of the pitfalls of this class of derivative-free optimization is that typically a relatively large number of objective function evaluations needed.

2. The second class combines finite differences with quasi-Newton methods.
   One of the pitfalls of this class is that finite difference approximations are being used, possibly resulting in situations where the solver is not robust due to the presence of noise in the objective function.

3. The third class involves the sequential minimization of quadratic or linear models based upon evaluations of the objective function at sample sets. Recent research has shown that this class is often superior compared to the first two classes.

6.4.5 Example of a derivative-free solver

The derivative-free method that has been chosen for this thesis, the DFBOLS solver by Zhang, Conn and Scheinberg [68], is in short a combination of a trust region framework with quadratic interpolation of the objective function. This method falls into the third class of derivative-free optimization methods (see Section 6.4.4 for a description of the classes).

In order to apply the trust region framework in the derivative-free case, a different type of approximation has to be chosen, which does not use any derivatives. A possible alternative is quadratic interpolation, of which a definition is given here and more information can be found in Appendix D.6. Consider the objective function $f(\mathbf{x})$, $\mathbf{x} \in \mathbb{R}^n$, which is to be interpolated by a (quadratic) polynomial $Q(\mathbf{x})$ at the set of interpolation points $Y = \{\mathbf{y}^j\}_{j=0}^p \subset \mathbb{R}^n$. This means that one needs to find the function $Q(\mathbf{x})$ such that:

$$Q(\mathbf{y}^j) \quad = \quad f(\mathbf{y}^j) \quad \forall\ j = 0, \ldots, p. \tag{6.22}$$

However, for this method a guarantee that the local approximation model is good enough is needed, meaning that a successful step is made after a sufficient reduction in the size of the trust region. As can be read in Appendix D.6 this turns out to be the case when the interpolation said is well-poised within the trust region.

One of the other defining characteristics of DFBOLS is the fact that the algorithm makes efficient use of the least-squares objective function. The DFBOLS algorithm can be considered as an adapted version of the Levenberg-Marquardt algorithm, which can be seen as a regularized version of the Gauss-Newton method, which is specially designed for small residual problems. The DFBOLS algorithm uses the following Hessian definition:

$$H \quad = \quad \begin{cases} \nabla Q(y)^T \nabla Q(y), & \text{small residual,} \\ \nabla Q(y)^T \nabla Q(y) + \kappa_H^3 I, & \text{otherwise.} \end{cases}$$

Typically the value of κ_H^3 is around 0.01. See Appendix D.6 and [68] for more information.
In order to deal with the special structure of the least-squares objective function, a linear (or possibly quadratic) interpolation model is built for each residual individually. All these models together are combined into a single model mimicking the structure of the original objective function. This can be considered as more efficient because although building several models requires more work and memory, it does not need more function evaluations and it better captures the structure of the problem. Since a function evaluation is typically speaking expensive to compute in derivative-free optimization, this approach is favourable. This is the key feature that distinguishes DFBOLS from other derivative-free solvers.

6.5 Numerical results

The concept of benchmarking in this context refers to the comparison of performance of the various optimization techniques from Section 6.4 on calibrating the Heston model with term structure to the set of 1355 datasets by bootstrapping, as described in Section 6.2. This implies that a clear definition of performance of a solver must be given. In this thesis overall CPU time of computations will be used as a performance metric. Note that the CPU time is dominated by the evaluations of the Heston model rather than time spent in the solvers themselves. Therefore this metric indirectly translates to the number of function evaluations needed to solve the calibration problem. The various optimizers will try to calibrate the model to all the datasets

from the market up to the 1bp condition (6.5) as rapidly as possible. All the CPU times and the successes of each solver for each dataset will be recorded and compared afterwards. If an optimizer fails to calibrate one or more terms in the term structure to the 1bp level, the optimizer is said to have failed the calibration for this specific dataset. From now on the calibration of a specific dataset may also be referred to as solving one of the problems. Note that if not stated otherwise, derivatives are approximated using finite differences.

All the libraries containing the various solvers have been build on the same machine such that everything was compatible with each other. The next step was to build a unifying framework in C++ in which the benchmarking would take place. A general solver interface has been developed to act as a wrapper to the various solvers. In this way the choice of a specific solver for the calibration is made relatively easy. The only component still left to be specified by the user are the type of derivatives (FD by the user/FD by the optimizer/AD) the solver has to use and the solver-dependent options such as tolerance levels.

After some initial tests the decision has been made to not include the unconstrained optimizers during the benchmarking. The motivation behind this is the strange performance of the three solvers as a consequence of parameter conversion (6.11) or (6.12) to move from constrained to unconstrained optimization. It seems as if the suspicions by Gill et al. [31] were correct: the parameter conversion introduces extra non-linearity into the problem resulting in strange behaviour and bad performance of the optimization.

To analyse the results, performance profiles from Dolan and Maré [22] have been used. A performance profile is way to visualize benchmarking results of optimization software. What has typically been done before for the analysis of benchmarking results, is the display of tables with the performance per solver per problem in terms of for example runtime, number of iterations or number of function evaluations. These are examples of so-called performance metrics. The challenge is to find a convenient and comprehensive way of displaying benchmarking results when the amount of test problems increases to the point where printing tables stops to be a good way to do so.
Other ways of comparing solvers are average or cumulative totals of the relevant performance metrics. In this way, it is possible for some of the problems that are hard to solve to influence the results such that they do not display the reality any more. In addition, the problems that any solver fails to solve must be omitted from the averaged or cumulative totals. This results in a bias of the solvers that are the most robust. A possible fix for this issue is to penalize a failed attempt to solve a problem, but a penalty value must be chosen, which will always be a subjective choice.
As an improvement of the previous method, sometimes the solvers are ranked according to their performance. This means that for each problem there is a 1st place, 2nd place, etcetera. The ranking consists of the frequency of the solver ending in the k-th place. This fixes the issue of a small set of difficult problems influencing the results. However, all information on the difference between the places is lost, there is only an indication which of two solvers is the best and not how much better it is.
Some academics give their preference in comparing medians and quartiles of performance metrics. This method appears to eliminate the risk of a small set of problems dominating the results. There are however some drawbacks to this method, among which once more the lack of display of relative size of improvement.
Performance profiles are said to be the perfect solution according to [22]. This performance profile is a (cumulative) distribution function of a chosen performance metric, which in this case

will be the ratio of runtime of the solver versus the minimal runtime over all the solvers on each dataset. The elegance of this method lies in the fact that no solver failures have to be discarded, and at the same time a visualization of the differences in performance between the solvers takes place.

Below, a more formal definition of the performance profile is given. Given are a problem set $\mathcal{P}$ of n_p problems and a set $\mathcal{S}$ of n_s solvers. Define the runtime of problem p and solver s as $t_{p,s}$. The performance ratio $r_{p,s}$, defined as:

$$r_{p,s} \quad = \quad \frac{t_{p,s}}{\min\{t_{p,s} : s \in \mathcal{S}\}},$$

is a way to compare the performance of solver s on problem p with the best performance of all the solvers on this particular problem p. Furthermore, define

$$r_M \quad = \quad 2 \cdot \max_{p \in \mathcal{P}} \left(\max_{s \in \mathcal{S}} r_{p,s} \right) \quad \geq \quad r_{p,s} \quad \forall\, p, s, \tag{6.23}$$

where equality holds if and only if problem p is not solved by solver s. Dolan and Moré prove that the choice of r_M does not effect the performance evaluation, thus the somewhat arbitrary choice of r_M in (6.23) is fine.
In order to obtain an overall view of the performance of a solver, define ρ_s to be the (cumulative) distribution function of the performance ratio:

$$\rho_s(\tau) \quad = \quad \frac{1}{n_p} \text{size} \left\{ p \in \mathcal{P} : r_{p,s} \leq \tau \right\}.$$

In other words, $\rho_s(\tau)$ is the probability for a solver s to have a performance ratio $r_{p,s}$ within a factor $\tau \in \mathbb{R}$ of the best possible ratio. The function $\rho_s : \mathbb{R} \to [0, 1]$ will from now on be called the performance profile.
$\rho_s(1)$ is thus the probability that a specific solver will win over all the solvers. So if one only cares about the number of wins and not about the time it took to achieve the wins, all that needs to be done is compare $\rho_s(1)$ for all the solvers.
Another value of interest, the value $1 - \rho_s(\tau)$, is the fraction of problems that a solver cannot solve within a factor τ of the best solver, where failed problems are included.
Typically, one is interested in the behaviour of τ around 1, a scaling can be done by plotting

$$\rho_s^{\log}(\tau) \quad = \quad \frac{1}{n_p} \text{size} \left\{ p \in \mathcal{P} : \log_2(r_{p,s}) \leq \tau \right\}.$$

Firstly, the performance profiles of all the solvers can be found in Figure 6.3. What is clear at once is the relatively bad performance of the interior point method (IPOPT) and the general SQP solver (e04wdc) compared to the other optimization techniques. Recall from Appendix D.5 that these were the only two solvers that were not able to exploit any extra information provided by a least-squares formulation of the objective function. As expected, this lack of ability to recognize such a situation leads to an unsatisfactory performance.
Secondly, because the plots of the other five solvers are very close to each other in Figure 6.3, consider the plots in Figure 6.4 to compare the performance of those five solvers. Note that this is a zoomed-in version of Figure 6.3. From this figure it is clear that the Levenberg-Marquardt method Levmar performs the best on just over 80% of the datasets. However, from Table 6.1 it turns out that it is not able to solve 15.42% of the problems. The performance profile of derivative-free method DFBOLS is the second best on the majority of the problems.

From Table 6.1 it is clear that eventually this solver is not able to solve 11.29% of the problems, which is less than the percentage for the Levenberg-Marquardt algorithm in Levmar. The Levenberg-Marquardt algorithm in Ceres on the other hand is even better in the end: it fails on only 10.26% of the problems. In addition, it does not perform a lot worse than the other Levenberg-Marquardt solver and the derivative-free solver. Finally, the two least-squares SQP solvers e04unc and Schittkowski's NLPLSQ are performing very similar to each other, both quite robust, failing to solve only roughly 10% of the problems. There is an explanation for these two SQP solvers to perform slightly worse in terms of runtime than the other solvers. The SQP solvers are aimed at more general models than the other solvers, as box-bounds as well as linear and non-linear constraints are allowed. This makes the solvers more complex and results in a relatively small amount of extra CPU time compared to the other solvers.

Note that there is an explanation for all the solvers to fail on 10 or more percent of the problems. Recall that a solver is said to fail on a problem if one or more of the terms in the term structure cannot be calibrated up to the 1bp level. It turns out that in many cases the solvers succeed in calibrating the first term, but then hit one of the bounds on correlation on the second term and fail to calibrate up to a one basis point level. This is most likely a consequence of particular market features that the model is not able to capture.

Depending on the specific requirements of a user in terms of speed and accuracy, either the Levenberg-Marquardt technique in Ceres or the derivative-free method DFBOLS can be considered as the best for this specific test-case of calibrating the Heston model with term structure to a set of European option quotes from the EURUSD FX market.

When combining this knowledge with the results from Section 5.4 on the choice of the best option pricing method, it is possible to see the speed-up of the entire calibration procedure. Say that one used to calibrate the model using the Lewis method with one of the two least-squares SQP optimizers. Now replace the Lewis method by the modified version of the COS method that had the best performance in Section 5.4. For an arbitrary date from the dataset (for now choose 28-3-2012) this leads to a speed-up of roughly a factor 7.75. This means that replacing the Lewis method by the modified COS method makes that the calibration up to a 1bp level is almost 8 times as fast. If now the SQP optimization method is replaced by the derivative-free method, this leads to an additional speed-up of roughly a factor 1.55. So moving from initially the Lewis method with a least-squares SQP optimization technique to finally the modified COS method with a derivative-free optimizer will lead to a speed-up of roughly a factor 12. This implies that calibrating for an hour in the initial situation is reduced to roughly 5 minutes in the final situation. This is a significant speed-up of the calibration procedure.

Solver name	Ceres	DFBOLS	Levmar	e04unc	Schittkowski	e04wdc	IPOPT
Percentage not solved	10.26	11.29	15.42	10.55	10.33	10.26	10.18

Table 6.1: Percentage of problems that each optimizer fails to solve.

Recall the discussion from Section 6.2.1 on the choice of using a bootstrapping approach during the calibration by calibrating term-by-term. The question posed there was whether the reduction in dimension of optimization by ignoring the backward dependence in time across the terms in the term structure could be justified. In order to put this to the test, the results from the bootstrap calibration for the NAG least-squares SQP solver have been compared with an implementation of a so-called full-force or non-bootstrap calibration where all 21 parameters

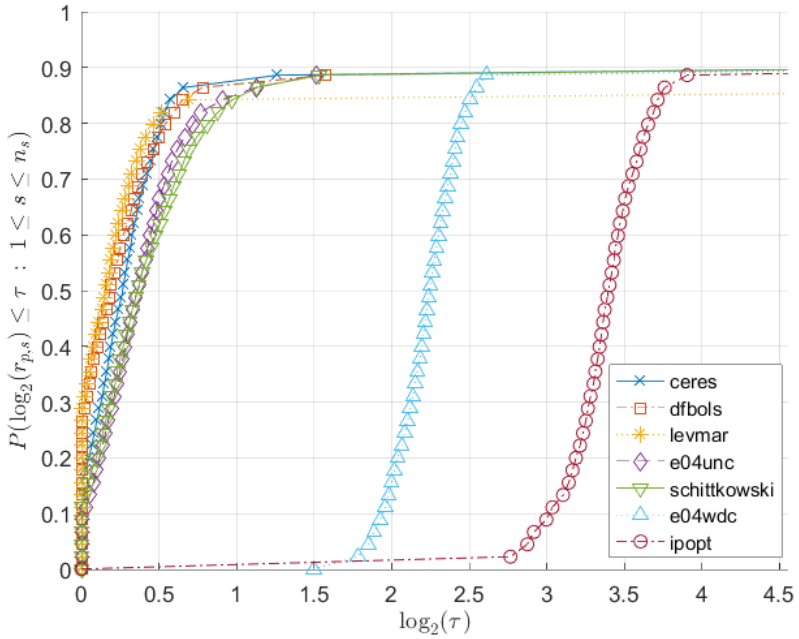


Figure 6.3: Performance profiles of all the solvers.

are calibrated at once. Once again a solver is said to have failed on a problem when one of the terms in the term structure cannot be calibrated up to the 1bp condition (6.5). Note that for the non-bootstrap calibration this implies that either all terms have reached the 1bp condition or none.
From Table 6.2 it is clear that the non-bootstrap calibration is able to solve around 5% more problems than the bootstrap calibration. However, from Figure 6.5 it is clear the two lines intersect roughly at $\log_2(\tau) = 6.75$, which implies that the non-bootstrap is roughly $2^{6.75} \approx 108$ times slower than the bootstrap calibration.

This slowdown is due to the size of the non-bootstrap problems. Recall that for a single term in the bootstrap calibration one has got to do 3 evaluations of the Heston model to obtain the objective value (i.e. sum of squares) and another 9 Heston evaluations (when doing forward finite differences) to obtain the full 3×3 gradient. Note that in the bootstrapping case there is no backward or forward dependence in time. This implies a total of 12 Heston evaluations per term per iteration. For the non-bootstrap calibration on the other hand, one has got to do 21 evaluations of the Heston model to obtain the objective value. In this case there is only backward dependence in time, resulting in an extra 252 Heston evaluations when doing forward finite differences. This is in total 273 Heston evaluations for a full gradient per iteration. So in the bootstrap case 7 problems of 3 variables are solved, resulting in 12 Heston evaluations per iteration. On the other hand, in the non-bootstrap case, one problem of 21 variables is solved, resulting in 273 Heston evaluations per iteration. It thus appears that in the case of the bootstrap calibration relatively a small amount of iterations is needed to converge to the desired level of accuracy.

It can be concluded that the non-bootstrap calibration is the method that should be used from a mathematical point of view, since this method gives the best results. Note that the 10% problems not solved by the bootstrap calibration is still an acceptable result, so the speed-up

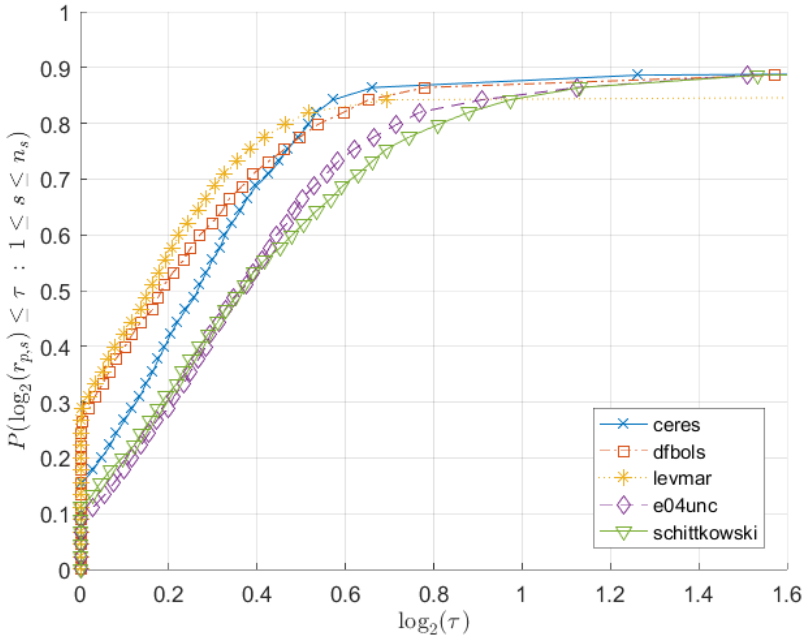


Figure 6.4: Performance profiles of the five best solvers.

of roughly a factor 108 at the cost of an extra 5% of the problems that are not solved is reasonable.

Method	bootstrap	non-bootstrap
Percentage not solved	10.55	5.83

Table 6.2: Percentage of problems that each method fails to solve.

Recall the discussion from Section 6.3 on the use of derivatives. AD is proposed as an alternative technique for computing derivatives as opposed to finite difference approximations. For the tests up to this point finite differences have been used, but in order to see whether the claims about AD being a more suitable way of calculating derivatives also holds for the current case, some tests have been done. From this point onward only consider the bootstrap calibration procedure again. The choice has been made to use the tangent-linear form of AD, as an adjoint model only makes sense when the number of inputs of a function is much higher than the Heston model with term structure. The current implementation of this AD version of the Heston pricing method is not very fast yet, therefore instead of comparing CPU times, for this test a comparison based on the amount of Heston pricing evaluations that will be done. This means a slight change in definition is needed for the performance profiles is needed. Define the number of function evaluations of problem p and solver s as $feval_{p,s}$. Redefine the performance ratio $r_{p,s}$ accordingly:

$$r_{p,s} = \frac{feval_{p,s}}{\min\{feval_{p,s} : s \in \mathcal{S}\}},$$

Using this modified definition of the performance profile, consider the results of the tests in Figures 6.6a and 6.6b. In Figure 6.6a the calibration has been performed up to the 1bp condition (6.5), whereas in Figure 6.6b the calibration is not stopped until the convergence criteria of the solver are reached. Note that these solver stopping criteria are the same for both FD as well

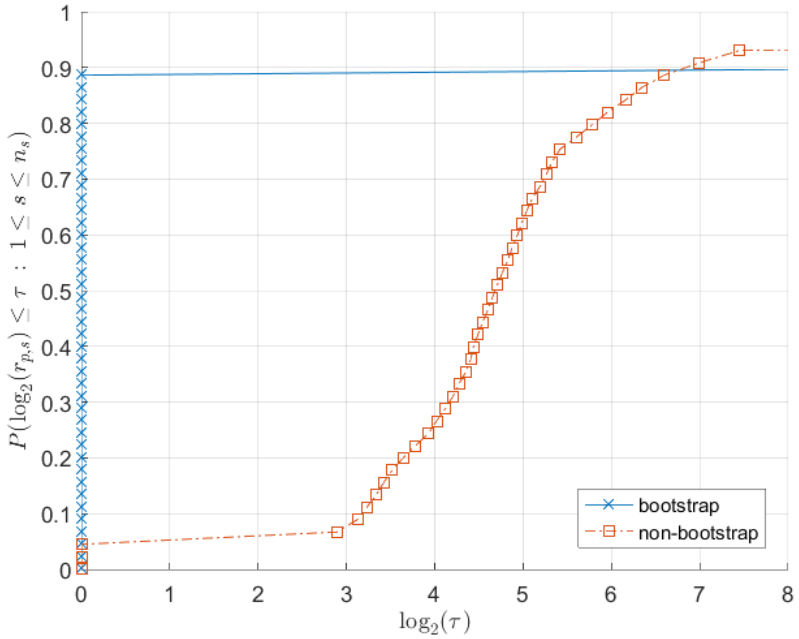
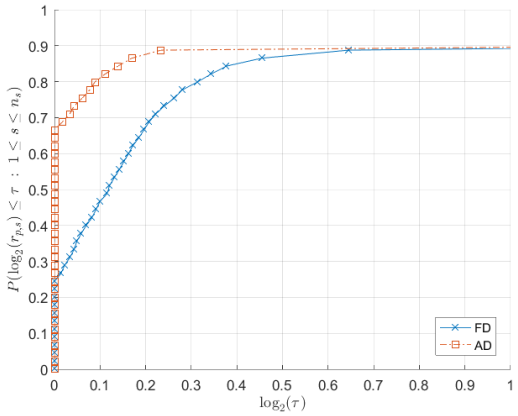


Figure 6.5: Performance profiles of bootstrap v. non-bootstrap.

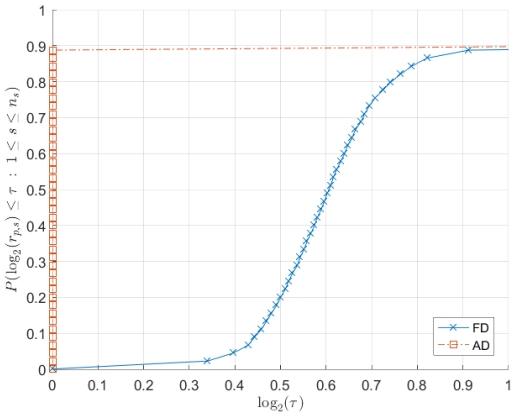
as AD. It is clear from Figure 6.6a that in order to reach the 1bp condition, the use of AD is more suitable than FD in terms of function evaluations, as the AD line is always more to the left than the FD line. An interesting observation can be made from Figure 6.6b, which says that if one desires a more accurate calibration, the use of AD make an even bigger difference in terms of smaller amount of Heston model evaluations. As expected, from Figures 6.6c and 6.6d it becomes clear that even though the AD calibration requires fewer function evaluations, in terms of CPU time the FD calibration is still faster. Note that for these two plots the original definition of the performance profile has been used again. When comparing Figures 6.6c and 6.6d, it can be said that AD is relatively speaking faster when doing a calibration until the stopping criterion of the solver itself, thus having a more accurate calibration than the 1bp condition provides. The relatively good performance of AD versus FD in Figure 6.6d in terms of CPU time is due to the fact that much fewer function evaluations are required (see Figure 6.6b). Also note from Table 6.3 that the percentage of the problems that cannot be solved is more or less equivalent for FD and AD. AD performs even slightly better. All in all, on assumption that a faster implementation of AD can be developed, this technique is indeed an improvement compared to using finite difference approximations.

Method	FD	AD
Percentage not solved	10.55	10.13

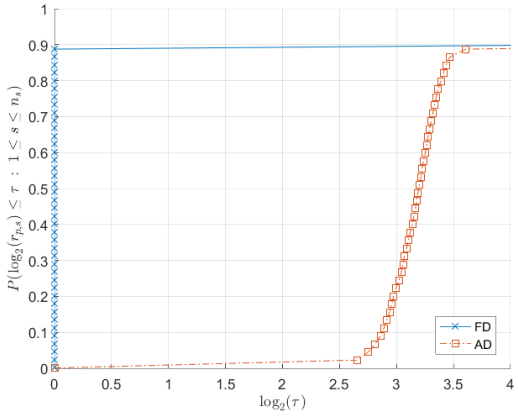
Table 6.3: Percentage of problems that each method fails to solve.



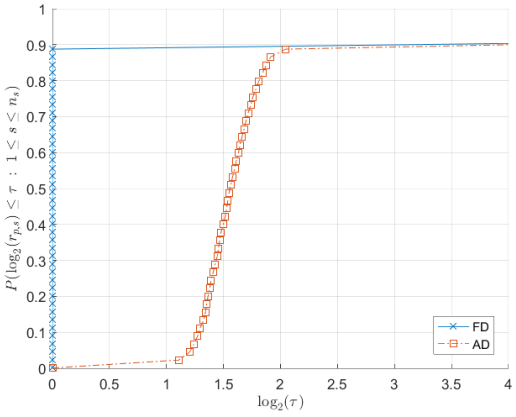
(a) Nr. of function eval. up to the 1bp condition.



(b) Nr. of function eval. up to full convergence.



(c) CPU time up to the 1bp condition.



(d) CPU time up to full convergence.

Figure 6.6: Implied volatility curves for different parameter settings.

Chapter 7

Hedging

In Chapter 6, the benchmarking of the various solvers on the case of the calibration of the Heston model with term structure has been performed. The calibration of every term in the term structure has been performed up to a certain level of accuracy: the 1bp condition (6.5). This means that all solvers calibrate to the same level of accuracy. This allowed for the comparison of runtimes of the calibration by the solvers on all the datasets in Section 6.5. One of the questions that remains is whether the fitting of the model to the market up to 1bp gives a sufficiently good fit if one decides to use the calibrated parameters for exotic option pricing or the calculation of sensitivities of option prices for risk analysis purposes. In this chapter this is put to the test using a hedging argument. In addition, there might be solvers that performed worse than others in terms of speed, but it might be that in terms of a hedging argument they can be considered superior to those others.

Using an example, the concept of hedging will be introduced. Say that a certain financial party wants to sell a European option with a maturity of 6 months for a certain fair price. From now on this party will be referred to as the writer of the option. If a third party buys the option and at the expiration date the option is ITM, the writer would lose money. In an attempt to reduce the risk of losing money, the writer will however buy/sell additional financial products in order to create a self-financing portfolio (i.e. a portfolio with no in- or outflow of money, thus a closed system) that would be ideally be free of risk. This idea of reducing the risk of a portfolio by buying/selling financial products is in essence hedging.

In the example above, at the time the option is sold, the writer of the option decides to buy for example some shares of stock and an additional option to eliminate the possible risk of losing money at maturity. Say that the writer decides to think carefully about the portfolio he assembles at the time of writing the option and thereafter does not adjust the portfolio at all. In such a situation a hedge can be considered as a static hedge. This would be an ideal situation for the writer, as changing positions within the portfolio will result in more transaction costs. However, there still exists the possibility of the underlying to behave in such a way that the writer will lose money at maturity. A possible fix to eliminate this risk is the concept of dynamic hedging, that requires constant re-balancing. By adding and removing products from the portfolio, the writer will most likely be able to reduce the risk of losing money significantly. In this case the reduction of risk is worth the transaction costs that have to be paid. Naturally, an ideal situation would be to have a dynamic hedge where re-balancing does not need to take place so often, thus a sort of static version of a dynamic hedge.

In an ideal world the writer would apply dynamic hedging and completely reduce the risk of

losing money. However, there are some practical limitations that make this impossible. The constant rebalancing of the portfolio, i.e. at every infinitesimal point in time, is practically impossible. It is on the other hand possible to re-balance a portfolio on for example a daily basis. This is called discrete dynamic hedging as discrete points in time are used for re-evaluating the positions of the portfolio. A thorough description of this concept of discrete dynamic hedging can be found in the book by Higham [39].

To give a more precise example of an hedging argument, in Section 7.1, the concept of Black-Scholes delta hedging is introduced according to the discrete dynamic hedging framework by Higham [39]. After the concept of hedging is more clear it is time to move on to see how one should deal with hedging under the Heston dynamics. Remember that under the Heston model an extra source of randomness is introduced by making the volatility stochastic. This will probably cause a delta hedge to be insufficient to reduce all risk and an extra tool is needed to eliminate the extra source of uncertainty introduced by the Heston model. This can be found in Section 7.2. Furthermore, in Section 7.3, some of the so-called Greeks (i.e. sensitivity of price w.r.t underlying parameters) are introduced and more important the way to compute them. Finally in Section 7.4 the results of the hedge tests can be found.

7.1 Black-Scholes hedging

Say that the writer of a European option C_t, where the underlying S_t follows a Black-Scholes process, wants to hedge his position (a short position). A possible approach could be to create a self-financing replicating portfolio $\Pi_t := \Pi(S_t, t)$ consisting of:

- A cash deposit $B_t := B(S_t, t)$ with dynamics $dB_t = r_d B_t dt$.
- $X_t := X(S_t, t)$ units of underlying asset S_t.

This implies the following definition of the replicating portfolio:

$$\Pi_t = X_t S_t + B_t. \tag{7.1}$$

The goal is to let portfolio Π_t replicate the position $-C_t$ at any point in time, where the focus is on making sure that Π_t has the same risk as $-C_t$ at any point in time. This implies that $\Pi_t + C_t$ must be risk-less at all times. The logical next step is to look at the change in $\Pi_t + C_t$ over a little time interval dt:

$$\begin{aligned} d(\Pi_t + C_t) &= d\Pi_t + dC_t, \\ &= X_t dS_t + dB_t + dC_t. \end{aligned} \tag{7.2}$$

Combining the fact that the dynamics of S_t under the Black-Scholes model [6] are given by:

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

[6] and the Ito expansion for option value C_t is given by:

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

The dynamics in (7.2) can then be rewritten as follows:

$$\begin{aligned} d(\Pi_t + C_t) &= X_t \left(\mu S_t dt + \sigma S_t dW_t\right) + r_d B_t dt \\ &\quad + \left(\frac{\partial C_t}{\partial t} + \mu S_t \frac{\partial C_t}{\partial S_t} + \frac{1}{2}\sigma^2 S_t^2 \frac{\partial^2 C_t}{\partial S_t^2} \right) dt + \sigma S_t \frac{\partial C_t}{\partial S_t} dW_t, \\ &= (\ldots)dt + \sigma S_t dW_t \left(X_t + \frac{\partial C_t}{\partial S_t} \right). \end{aligned} \tag{7.3}$$

The goal is to make the portfolio replicate the short position in the option, in a way that the difference between the two is predictable. In order to achieve that goal, remove all randomness from (7.3) by eliminating the dW_t-term by setting:

$$X_t = -\frac{\partial C_t}{\partial S_t} = -\frac{\partial C_t}{\partial S_0} = \Delta_{BS}.$$

Note that the chance of deriving the option values from with respect to S_t into S_0 is justified because the European option values are not path-dependent and therefore only depend on the initial values of the asset and variance process.
This type of hedging is called delta-hedging, since the units X_t of underlying asset S_t is defined as the option delta. See Section 7.3 for more information on how to compute this quantity.

7.2 Hedging under the Heston dynamics

Opposed to the constant volatility in the Black-Scholes model, the Heston model is a model with stochastic volatility (for the time being the single-term Heston model is considered). This introduces an extra source of randomness that has to be eliminated while hedging. Therefore only hedging with asset S_t as an instrument to remove risk will not suffice and an extra tool has to be introduced to eliminate the extra randomness. This tool will be another European option contract $C_{t,1}$ different from the option that is being hedged C_t.
Say that the writer of a European option C_t, where the underlying S_t follows a Heston process, wants to hedge his position (a short position). A possible approach could be to create a self-financing replicating portfolio $\Pi_t := \Pi(S_t, C_{t,1}, t)$, consisting of:

- A cash deposit $B_t := B(S_t, C_{t,1}, t)$ with dynamics $dB_t = r_d B_t dt$.
- $X_t := X(S_t, C_{t,1}, t)$ units of underlying asset S_t.
- $Y_t := Y(S_t, C_{t,1}, t)$ units of option $C_{t,1}$.

Note that the approach taken to solve the current hedging exercise is very similar to the approach taken in Section 7.1.
This implies the following definition of the replicating portfolio:

$$\Pi_t = X_t S_t + Y_t C_{t,1} + B_t. \tag{7.4}$$

The goal is to let portfolio Π_t replicate the position $-C_t$ at any point in time, where the focus is on making sure that Π_t has the same risk as $-C_t$ at any point in time. This implies that $\Pi_t + C_t$ must be risk-less at all times. The logical next step is to look at the change in $\Pi_t + C_t$ over a little time interval dt:

$$\begin{aligned} d(\Pi_t + C_t) &= d\Pi_t + dC_t, \\ &= X_t dS_t + Y_t dC_{t,1} + dB_t + dC_t. \end{aligned} \tag{7.5}$$

Combining the fact that the dynamics of S_t are given in (3.4) and the multi-dimensional Ito expansion for option values C_t and $C_{t,1}$ given in (A.2), the dynamics in (7.5) can be rewritten

as follows:

$$\begin{aligned} d(\Pi_t + C_t) &= \left(\left(\frac{\partial C_t}{\partial t} + \mu S_t \frac{\partial C_t}{\partial S_t} + \ldots + \frac{1}{2}\alpha^2 v_t \frac{\partial^2 C_t}{\partial v_t^2} \right) dt + \sigma\sqrt{v_t} S_t \frac{\partial C_t}{\partial S_t} dW_t^1 + \alpha\sqrt{v_t} \frac{\partial C_t}{\partial v_t} dW_t^2 \right) \\ &\quad + Y_t \left(\left(\frac{\partial C_{t,1}}{\partial t} + \ldots + \frac{1}{2}\alpha^2 v_t \frac{\partial^2 C_{t,1}}{\partial v_t^2} \right) dt + \sigma\sqrt{v_t} S_t \frac{\partial C_{t,1}}{\partial S_t} dW_t^1 + \alpha\sqrt{v_t} \frac{\partial C_{t,1}}{\partial v_t} dW_t^2 \right) \\ &\quad + X_t \left(\mu S_t dt + \sigma\sqrt{v_t} S_t dW_t^1 \right) + r_d B_t dt, \\ &= (\ldots)\, dt + \sigma\sqrt{v_t} S_t \cdot dW_t^1 \left[\frac{\partial C_t}{\partial S_t} + X_t + Y_t \frac{\partial C_{t,1}}{\partial S_t} \right] \\ &\quad + \alpha\sqrt{v_t} \cdot dW_t^2 \left[\frac{\partial C_t}{\partial v_t} + Y_t \frac{\partial C_{t,1}}{\partial v_t} \right]. \end{aligned} \tag{7.6}$$

The goal is to make the portfolio replicate the short position in the option, in a way that the difference between the two is predictable. In order to achieve that goal, remove all randomness from (7.6) by eliminating the $d\tilde{W}_t^i$ terms:

$$\begin{aligned} \frac{\partial C_t}{\partial S_t} + X_t + Y_t \frac{\partial C_{t,1}}{\partial S_t} &= 0, \\ \frac{\partial C_t}{\partial v_t} + Y_t \frac{\partial C_{t,1}}{\partial v_t} &= 0. \end{aligned}$$

Now it is possible to retrieve the strategy that makes the difference predictable in terms of X_t and Y_t:

$$X_t = -\frac{\partial C_t}{\partial S_t} - Y_t \frac{\partial C_{t,1}}{\partial S_t} = -\frac{\partial C_t}{\partial S_0} - Y_t \frac{\partial C_{t,1}}{\partial S_0}, \tag{7.7}$$

$$Y_t = -\frac{\partial C_t}{\partial v_t} \left(\frac{\partial C_{t,1}}{\partial v_t} \right)^{-1} = -\frac{\partial C_t}{\partial v_0} \left(\frac{\partial C_{t,1}}{\partial v_0} \right)^{-1}. \tag{7.8}$$

Note that the chance of deriving the option values from with respect to S_t and v_t into S_0 and v_0 is justified because the European option values are not path-dependent and therefore only depend on the initial values of the asset and variance process.

This type of hedging is called delta-vega hedging. See Section 7.3 for more information on the Greeks. Next to delta-vega hedging under a Heston regime, there are also other types of hedging. The simplest alternative is an extension of the Black-Scholes delta hedging described in Section 7.1, where for the Heston model the same results hold as for the Black-scholes model:

$$X_t = -\frac{\partial C_t}{\partial S_0}. \tag{7.9}$$

The analysis is continued based on delta-vega hedging. Naturally all that will follow can be adapted to match any type of hedging mentioned. Up till now, the implicit assumption has been made that the progress in time has been continuous and that at every infinitesimally small time instance the positions X_t and Y_t can be adjusted. However, in practice this is not possible due to for example transaction costs and other practical limits. Therefore the decision is made to update the hedging strategy daily. When considering the time-interval $[t_0, T]$, an equidistant time discretization of $N+1$ points in time can be made such that $t_0 < t_1 < \ldots < t_{N-1} < t_N = T$ and $t_{i+1} - t_i = dt$ holds for all i. This means that over a small time interval dt the positions X_t and Y_t are kept constant. To simplify notation, from now on a subscript i will indicate that the current time is t_i, meaning that Π_{t_i} will be denoted as Π_i. Furthermore, the difference of a

value over a time step will be denoted as $d\Pi_i = \Pi_{i+1} - \Pi_i$.
So if one is allowed to update the strategy daily, at the start of time instance t_{i+1} the following must hold:

$$\begin{aligned} \Pi_{i+1} &= d\Pi_i + \Pi_i, \\ &= (X_i dS_i + Y_i dC_{i,1} + D_i) + (A_i S_i + Y_i C_{i,1} + B_i), \\ &= X_i S_{i+1} + Y_i dC_{i+1,1} + e^{r_d \cdot dt} B_i. \end{aligned} \tag{7.10}$$

By definition the following must hold:

$$\Pi_{i+1} = X_{i+1} S_{i+1} + Y_{i+1} C_{i+1,1} + B_{i+1}. \tag{7.11}$$

Since the portfolio was chosen to be self-financing (meaning that no money can enter/leave the portfolio), equations (7.10) and (7.11) can be set equal and the following definition of B_{i+1} can be extracted:

$$B_{i+1} = (X_i - X_{i+1}) S_{i+1} + (Y_i - Y_{i+1}) C_{i+1,1} + e^{r_d \cdot dt} B_i. \tag{7.12}$$

Coming back to the fact that X_i and Y_i have been chosen such that $\Pi_i + C_i$ is risk-less, this must grow as a risk-free bond $e^{r \cdot dt}(\Pi_0 + C_0)$. This leads to the following definition of the hedging error (HE) made at each point in time:

$$HE_i = \left| \Pi_i + C_i - e^{r_d \cdot dt}(\Pi_0 + C_0) \right|. \tag{7.13}$$

As a measure of the overall error, the hedging error at the final time is considered, i.e. HE_N. In the ideal case HE_N would be equal to zero.

The following is a summary of the hedging strategy described above:

(Step 1) For time t_0:

- (Step 1.1) Initialize X_0 and Y_0 according to (7.7) and (7.8) respectively.
- (Step 2.2) Furthermore, set the initial amount of cash s.t. $\Pi_0 = C_0$
- (Step 3.3) Finally set Π_0 according to (7.11).

(Step 2) For each new point in time $t_{i+1} = t_i + dt$ up and till $t_{i+1} = t_N = T$:

- (Step 2.1) Observe new asset price S_{i+1}.
- (Step 2.2) Compute Π_{i+1} according to (7.10).
- (Step 2.3) Compute X_{i+1} and Y_{i+1} according to (7.7) and (7.8) respectively.
- (Step 2.4) Compute B_{i+1} according to (7.12).
- (Step 2.5) The new portfolio value is Π_{i+1} according to (7.11).
- (Step 2.6) Compute HE_{i+1} according to (7.13).

(Step 3) Observe HE_N, which is the hedging error at maturity.

7.3 Hedging and the Greeks

Now that a basic understanding of hedging has been established, it is time to see how to compute all the derivatives of the European option prices that have been mentioned. As a start, look into the definitions of the delta Δ_{BS}, gamma Γ_{BS} and vega ν_{BS} under the Black-Scholes model:

$$\begin{aligned}
\Delta_{BS} &= \frac{\partial C_t}{\partial S} = \alpha e^{-r_f(T-t)} N(\alpha d_1), \\
\Gamma_{BS} &= \frac{\partial^2 C_t}{\partial S^2} = \frac{e^{-r_f(T-t)-\frac{1}{2}d_1^2}}{S_t \sigma \sqrt{T-t}\sqrt{2\pi}}, \\
\nu_{BS} &= \frac{\partial C_t}{\partial \sigma} = \frac{S_0 e^{-r_f(T-t)-\frac{1}{2}d_1^2}\sqrt{T-t_0}}{\sqrt{2\pi}}.
\end{aligned}$$

Note that these results hold only for the Black-Scholes model, and not for the Heston model with term structure for example. This means that an alternative way of obtaining the Greeks must be thought of in case of a different model than the Black-Scholes model. For some models an analytic expression of the Greeks exist, unfortunately this is not the case for the Heston model with term structure. Another possibility to obtain the desired derivatives is using finite difference approximations. Due to the sensitivity of this method to noise and other causes described in Section 6.3, another method is preferred. AD is a tool that could prove to be useful for this particular application. The method that will be used however is of a different nature. Some useful results from the COS method by Fang and Oosterlee [26] allow for the following representation of the Greeks in terms of the Fourier cosine series expansion:

$$\begin{aligned}
\Delta_{cos} &= \frac{\partial C_t}{\partial S_0} \approx e^{-r_d\tau} {\sum_{k=0}^{N-1}}' \Re\left\{ \phi\left(\frac{k\pi}{b-a}; x\right) \cdot \exp\left(\frac{-ika\pi}{b-a}\right) \cdot \frac{ik\pi}{b-a} \right\} \frac{V_k}{S_0}, \\
\Gamma_{cos} &= \frac{\partial^2 C_t}{\partial S_0^2} \approx e^{-r_d\tau} {\sum_{k=0}^{N-1}}' \Re\left\{ \phi\left(\frac{k\pi}{b-a}; x\right) \cdot \exp\left(\frac{-ika\pi}{b-a}\right) \cdot \left[-\frac{ik\pi}{b-a} + \left(\frac{ik\pi}{b-a}\right)^2 \right] \right\} \frac{V_k}{S_0^2}, \\
\nu_{cos} &= \frac{\partial C_t}{\partial v_0} \approx e^{-r_d\tau} {\sum_{k=0}^{N-1}}' \Re\left\{ \frac{\partial \phi\left(\frac{k\pi}{b-a}; x\right)}{\partial v_0} \cdot \exp\left(\frac{-ika\pi}{b-a}\right) \right\} V_k.
\end{aligned}$$

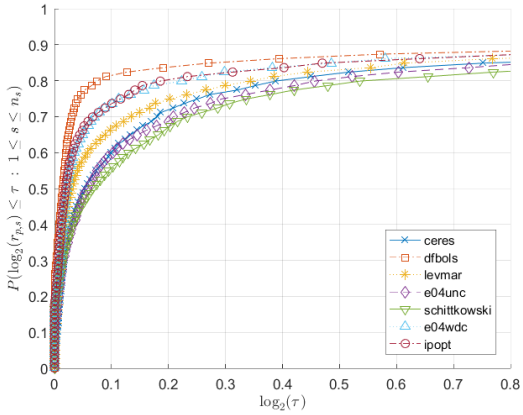
7.4 Numerical results

The delta and delta-vega hedging strategies described in Section 7.2 will now be used to test the calibration results for the solvers that competed in the benchmark. In order to display the results, once more performance profiles have been chosen. As a performance metric, the hedging error at maturity has been used. Define the hedging error at maturity of problem p and solver s as $(HE_N)_{p,s}$. The corresponding performance ratio $r_{p,s}$ is now defined as:

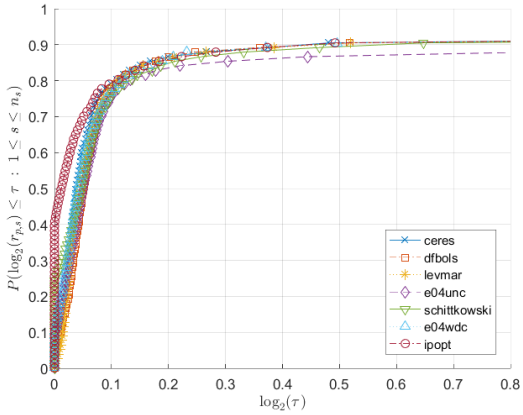
$$r_{p,s} = \frac{(HE_N)_{p,s}}{\min\{(HE_N)_{p,s} : s \in \mathcal{S}\}},$$

The tests are performed for two different types of option maturities, namely two months (2M) and six months (6M). As a strike for the European call option that is being sold by the writer use $K = 0.95 \cdot S_0$, s.t. the money is out of the money. If the option is OTM, then the writer does not lose any money. There remains the chance that during the course of the two or six months the underlying moves such that the option becomes ITM at maturity. This is the risk that needs

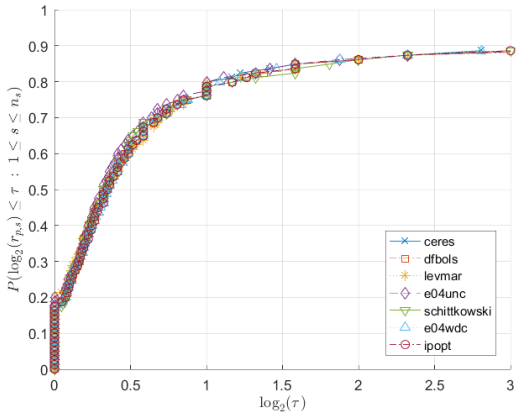
to be hedged. As an extra instrument needed for the delta-vega hedge, a put option with the same maturity and strike level has been used. All option prices and Greeks are calculated with the COS method.



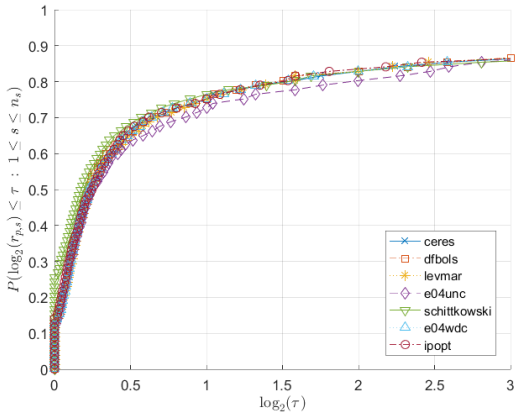
(a) Δ hedge, $T = 0.1667$.



(b) Δ hedge $T = 0.5$.



(c) Δ-ν hedge $T = 0.1667$.



(d) Δ-ν hedge $T = 0.5$.

Figure 7.1: Performance profiles of the various hedge tests.

Solver name	Ceres	DFBOLS	Levmar	e04unc	Schittkowski	e04wdc	IPOPT
Δ hedge 2M	5.076e-03	5.042e-03	4.803e-03	5.048e-03	5.059e-03	5.035e-03	5.029e-03
Δ-ν hedge 2M	3.071e-16	3.046e-16	2.871e-16	3.039e-16	3.076e-16	3.078e-16	3.066e-16
Δ hedge 6M	3.025e-02	2.955e-02	2.863e-02	3.893e-02	3.157e-02	3.008e-02	2.989e-02
Δ-ν hedge 6M	1.199e-15	1.181e-15	1.139e-15	1.371e-15	1.202e-15	1.200e-15	1.197e-15

Table 7.1: Average hedging error for the various hedge tests and solvers.

The evidence in Figure 7.1 and Table 7.1 is overwhelming that the delta-vega hedge is better than the delta hedge: the delta-vega hedging error is $\mathcal{O}(10^{-16})$ and $\mathcal{O}(10^{-15})$ for respectively a maturity of two and six months. On the other hand, the delta hedging error is $\mathcal{O}(10^{-3})$ and $\mathcal{O}(10^{-2})$ for respectively a maturity of two and six months, which is significantly larger than the delta-vega hedging errors. This confirms the statements made earlier on in this chapter, so an additional instrument is definitely required to remove the risk as result of the stochastic volatility

in the Heston model. Note that the delta-vega hedges for both maturities give hedging errors that are practically zero. The errors for the delta hedges on the other hand are significantly larger, and a bit more spread-out over the plots. All in all it seems as if the hedging argument provides little extra information on the choice of the best solver for the calibration of the Heston model with term structure, as the writer of the option can choose to do a delta-vega hedge and almost completely remove all the risk from the portfolio. This can be considered as a good thing, as the choice of optimization technique now only affects the performance in terms of CPU time.

Chapter 8

Conclusions and outlook

8.1 Summary and conclusion

This thesis has used the Heston model with term structure in an option pricing setting as well as in a calibration setting. Data from the EURUSD FX market has been used for calibration purposes. Descriptions of all the relevant features of the FX market (such as currency pairs, European option prices, delta conventions and the content of the dataset) have been provided in Chapter 2.

In Chapter 3, an introduction into the Heston model has been given, including the extension of the model by making the model parameters piecewise constant, resulting in the Heston model with term structure. An option pricing PDE and the relevant characteristic functions have been derived. This was a requirement for the option pricing methods that have been given in Chapter 4. In this thesis the COS method, Lewis' method and Andersen's QE Monte Carlo schemes have been used as option pricing methods. In an attempt to improve the practical usability of the COS method in terms of speed, accuracy and robustness, some modifications of the COS method have been proposed and put to the test in Chapter 5.

In Section 5.1 a new formula of the truncation range for the COS method has been proposed. The original formula from the paper used the first, second and fourth cumulants of the underlying distribution, thus taking into account properties of the mean, variance and kurtosis of the distribution. This formula assumes that the underlying distribution is symmetric because no information about the skewness has been used while truncating the density. What is thus proposed is a formula that extends one of the two sides of the truncation range from the original formula with some information on the skewness, by using the third cumulant. Numerical tests have been performed on some skewed distributions for which a lot of properties such as the chf, cdf and cumulants have analytic expressions. The distributions that have been considered are the gamma distribution, the normal inverse Gaussian distribution, the inverse Gaussian distribution and the exponentially modified Gaussian distribution. These numerical tests have confirmed the suspicions that the new way of computing the truncation range outperforms the original one in terms of the truncation error that is made.

Next, in Section 5.2, a new technique is proposed for the choice of number of terms N in the Fourier cosine series expansion of the COS method. This new technique makes sure that the value of N takes into account information of the underlying distribution and its truncation range (that has at this stage already been computed). In short, the error due to truncating an infinite summation is bounded by an accuracy parameter. Next, a leading order approx-

imation is performed to obtain a lower bound on the value of N. Numerical tests using the same test-set of distributions as in Section 5.1 show that this lower bound on N is indeed an appropriate value to use. By this it is meant that the value of N is large enough to provide sufficiently accurate results, but not too large in order to prevent a loss of speed of computations.

As a final addition to the COS option pricing framework, an adaptive method for determining the appropriate value of L in the truncation range has been proposed in Section 5.3. The idea is that a user can specify a tolerance level up to which options should be priced accurately. The methods sets up a converging series of option prices by incrementing the value of L every iteration. The method terminates as soon as the relative improvement of the last increment of L is below the specified tolerance level. In this way robustness of the implementation of the COS method is guaranteed.

All the methods from Chapter 4 and the modifications proposed in Chapter 5 have been put to the test in an option pricing setting in Section 5.4. For four different sets of Heston parameter combinations options are priced using the various methods. The first conclusion that can be drawn is that the new formula for the truncation range from Section 5.1 is an improvement also in this option pricing setting. The same holds for the way of computing the value for N in Section 5.2. Furthermore, it became clear that even though the adaptive way of determining the right value of L from Section 5.3 provides a robust pricing method, simply choosing a large value of L in advance for the COS method without any adaptive strategy for L can provide at least as much accuracy and will definitely be faster. Therefore, if speed is an important requirement, the latter approach is advised. Note that this approach does take into account the improvements from Sections 5.1 and 5.2. Compared to the Lewis implementation, the latter implementation of the COS method is significantly better in terms of both speed and accuracy. Finally the two Monte Carlo schemes by Andersen confirm the correctness of implementation of the COS method, but are obviously tremendously slow compared to the COS method. Also in terms of accuracy this method is not preferable.

Next to option pricing, the other main pillar of this thesis is the calibration of the model, of which a detailed description can be found in Chapter 6. The approach that has been chosen is to match current market prices with model prices and uses numerical optimization. Therefore an objective function for minimization must be formulated. A least-squares approach has been chosen, where current market prices have to be replicated by the model. A bootstrapping approach for the calibration is chosen to reduce the dimensionality of the problem by calibrating term-by-term. Since the model has four parameters per term but only three option prices are used for each term, one of the parameters must be fixed constant during the calibration to avoid an underdetermined optimization problem. In addition, it must be clear for the numerical optimization technique when a certain solution is satisfactory and the search for a better approximation of the solution can be stopped. The choice has been made to calibrate the model up to the point where the market prices are replicated at a one basis-point level. Derivatives are often a crucial component of numerical optimization, so the technique of algorithmic differentiations is discussed including its benefits compared to finite difference approximations of derivatives.

Part of this thesis was to compare the performance of several optimization techniques on the problem of calibrating the Heston model with term structure to the set of market data. This comparison can also be called a benchmark. Note that several optimization techniques took part in the benchmark, among which several are derivative-based and one is derivative-free. As the calibration problem involves constraints, a parameter conversion could be done in order to

transform the problem such that unconstrained optimization can be performed. Unfortunately this does not work due to the introduction of more non-linearity to the problem. This resulted in a strange behaviour of the unconstrained methods, therefore these are excluded from the results.

In an attempt to visualize all the results of all the solvers on the 1355 different datasets that were available, performance profiles have been used (see Section 6.5). As a performance metric the CPU runtime of the calibration has been chosen. It was immediately clear that the interior point method and the general SQP method performed a lot less than the other five solvers. This was caused by the fact that these two solvers do not recognize a least-squares structure of the objective function, and are therefore not able to use all the available information on the objective. Of the other five solvers, the two least-squares SQP methods performed slightly less than the other three methods. This can be explained by the fact that these solvers are aimed at more general optimization problems including highly non-linear constraints. This makes a more complicated solver and makes the two optimizers slightly slower than the others. Finally the two Levenberg-Marquardt methods and the derivative-free method performed the best. Note that for a lot of datasets one of the two Levenberg-Marquardt methods is the fastest, but in the end it fails to solve an extra 5% compared to the other two methods. Depending on the requirements a practitioner has, one of these last three methods can be considered as the best choice for the calibration.

Combining the results from the comparison of option pricing methods with the results from the comparison of optimization techniques for calibration it turns out that a speed-up of a factor 12 can be realized. This means that days become hours, and hours become minutes.

In addition to picking the best optimization technique for the calibration problem, some additional numerical tests have been performed. The first is on the choice of a bootstrap calibration, where backwards dependence in time is neglected. In order to test whether this was the right thing to do, all datasets have been calibrated again but now including this backward dependence in time (so a non-bootstrap calibration is performed). The conclusion is that this non-bootstrap approach gives better results: the bootstrap calibration fails to calibrate roughly 10.5% of the datasets as opposed to roughly 5.8% for the non-bootstrap calibration. Note that this increase in accuracy is at the cost of a slowdown of roughly a factor 108. So if one is willing to accept a worse performance of roughly five percent at the cost of a speed-up of roughly a factor 108, the bootstrap calibration is an appropriate approach of calibration. Furthermore the use of FD approximations for the derivatives has been compared with the use of AD for computing derivatives. The conclusion was that if a faster implementation of AD can be developed, the technique of AD is indeed an improvement compared to using FD approximations.

Finally a hedge test has been developed in order to see whether the best solver in terms of CPU time is also the best solver in terms of this hedge test. Unfortunately however it seems as if the hedge test that has been used provides little additional information on the choice of the best solver. From this it can be concluded that is does not make a significant different which solver is chosen during calibration from a hedging point of view.

8.2 Future research

As in every piece of work there are always additional things that can be done or things that can be done differently. A number of these possible topics for future research are:

- In Chapter 5 the choice has been made to stick with the same type of formulation for the

truncation range as originally proposed in [26]. This requires a choice for the parameter L, and in Section 5.3 an adaptive way of determining an appropriate value of L has been proposed. A possibility for future research for this aspect of the COS method is to try and find a new way of determining the value of L that possible takes into account some of the properties of the underlying distribution. Another possibility could be to think of a complete new way of determining the truncation range that still takes into account characteristics of the underlying distribution.

- The in Section 5.2 proposed method for determining the value of N used for the amount of terms in the Fourier cosine series approximation in the COS method could be tested for different distribution types that are often used in option pricing settings. One could for example think of the Variance Gamma process (introduced by Madan, Carr and Chang [45]) or the CGMY model (introduced by Carr, German, Madan and Yor [11]).

- In Section 6.2.2 the choice of initial guess has been addressed. There it is stated that the choice of initial guess does not influence the results in terms of different minima, but it does influence the speed of calibration. In order to speed up the calibration an initial guess for the first term that is close to the actual solution is desired. This is a topic of possible future research.

- As stated before, what is typically done after calibration is the pricing of exotic option contracts. A possible topic of future research could be to try and get hands on some exotic option prices such as barriers and see how accurately the calibrated model will be able to replicate the prices of those exotics using Monte Carlo techniques.

- Another thing that is typically done after calibration is the computation of option price sensitivities for risk analysis purposes. A new thing could be to try and use adjoint AD techniques to make a derivative code of the entire calibration process. In other words, this will result in sensitivities of the outcome parameters of the calibration with respect to the input parameters of the calibration. The latter are the market conditions that are extracted from the market data. In this way it is possible to see the effect of a changing market on the calibrated parameters. In turn, these parameters can be used for the pricing of exotic options. If one also makes a derivative code of that pricing technique, it is possible to obtain sensitivities of the exotic prices with respect to the market input parameters of the calibration. If required, more information on applications of AD in computational finance can be found in the article by Naumann and du Toit [56].

Bibliography

[1] H. Albrecher , P. Mayer, W. Schoutens and J. Tistaert , *'The Little Heston Trap'*, Wilmott, 6: 83-92, 2006.

[2] L.B.G. Andersen, *'Efficient Simulation of the Heston Stochastic Volatility Model'*, 2007. Available at SSRN: `http://ssrn.com/abstract=946405`.

[3] S. Agarwal, K. Mierle et al., *'Ceres Solver'*. See: `http://ceres-solver.org`.

[4] A. Bain, private communication, January 18, 2016.

[5] G. Bakshi, C. Cao and Z. Chen, *'Empirical Performance of Alternative Option Pricing Models'*, J. of Fin., 52(5): 2003-2049, 1997.

[6] F. Black and M. Scholes, *'The Pricing of Options and Corporate Liabilities'*, J. of Political Econ., 81(3): 385-408, 1973.

[7] T. Bollerslev, *'Generalized autoregressive conditional heteroskedasticity'* J. of Econometrics, 31(3): 307-327, 1986.

[8] J.P. Boyd, *'Chebyshev & Fourier spectral methods'*, Springer-Verlag, Berlin, 1989.

[9] M. Broadie and Ö. Kaya, *'Exact simulation of stochastic volatility and other affine jump diffusion processes'*, Operations Research, 54(2): 217-231, 2006.

[10] C.G. Broyden, *'The convergence of a class of double-rank minimization algorithms'*, J. Inst. Math. Appl., 6: 76-90, 1969.

[11] P.P. Carr, H. German, D.B Madan and M. Yor, *'The Fine Structure of Asset Returns: An Empirical Investigation'*, J. of Business, 75(2): 305-332, 2002.

[12] P.P. Carr and D.B. Madan, *'Option valuation using the fast Fourier transform'*, J. of Comp. Fin., 2: 61-73, 1999.

[13] I.J. Clark, *'Foreign Exchange Option Pricing: A Practitioners Guide'*, Wiley Finance, 2011.

[14] A.R. Conn, N.I.M. Gould, P.L. Toint, *'onvercenge of quasi-Newton matrices generated by the symmetric rank one update'*, J. of Math. Prog., 50(2): 177-195, 1991.

[15] A.R. Conn, K. Scheinberg and L.N. Vicente, *'Introduction to Derivative-Free Optimization'*, SIAM, 2009.

[16] J.C. Cox, J.E. Ingersoll and S.A. Ross, *'A Theory of the Term Structure of Interest Rates'*, Econometrica, 53(2): 385-408, 1985.

[17] R. Cont and P. Tankov, *'Non-parametric calibration of jump-diffusion option pricing models'*, J. of Comp. Fin., 7(3), 149, 2004.

[18] J.C. Cox, *'Notes on Option Pricing I: Constant Elasticity of Variance Diffusions'*, Working paper, Stanford University, 1975.

[19] J.C. Cox and S. Ross, *'The valuation of options for alternative stochastic processes'*, J. of Fin. Econ., 3: 145-166, 1976.

[20] F. Devernay, *'C/C++ Minpack'*, `http://devernay.free.fr/hacks/cminpack/`, 2007.

[21] E. Derman and I Kani, *'Riding on a Smile'*, Risk, 7(2): 139-145, 1994.

[22] E.D. Dolan and J.J. More, *'Benchmarking optimization software with performance profiles'*, Math. Prog., 91: 201-213, 2002.

[23] D. Duffie, J. Pan and K. Singleton, *'Transform Analysis and Asset Pricing for Affine Jump-Diffusions'*, Econometrica, 68(6): 1343-1376, 2000.

[24] B. Dupire, *'Pricing with a Smile'*, Risk, 7(1): 18-20, 1994.

[25] A. Elices, *'Models with time-dependent parameters using transform methods: application to Heston's model'*, 2007. See: `https://arxiv.org/pdf/0708.2020`.

[26] F. Fang and C.W. Oosterlee, *'A Novel Pricing Method for European Options based on Fourier-cosine Series Expansion'*, SIAM J. Sci. Comp., 2008.

[27] W. Feller, *'Two singular diffusion problems'*, Ann. of Math., 54(2): 173-182, 1951.

[28] R. Fletcher, *'A new approach to variable metric algorithms.'*, Computer Journal, 13(3): 317-322, 1969.

[29] M.B. Garman and S.W. Kohlhagen, *'Foreign Currency Option Values'*, J. of International Money and Fin., 2: 231-237, 1983.

[30] J. Gatheral, *'The Volatility Surface: A Practitioner's Guide'*, Wiley Finance, 2006.

[31] P.E. Gill, W. Murray and M.H. Wright, *'Practical Optimization'*, Academic Press, London, 1981.

[32] P.E. Gill, W. Murray and M.A. Saunders, *'Users' SNOPT 7.1: a Fortran package for large-scale linear nonlinear programming'*, Report NA 05-2 Dep. of Math., University of California, San Diego, 2005.

[33] D. Goldfarb, *'A Family of Variable Metric Updates Derived by Variational Means'*, Math. of Comp., 24: 23-26, 1970.

[34] P. Hagan, D. Kumar, L. Lesniewski and D.E. Woodward, *'Managing Smile Risk'*, Wilmott Magazine, pages 84-108, 2002.

[35] S.B. Hamida and R. Cont, *'Recovering volatility from option prices by evolutionary optimization'*, J. of Comp. Fin., 8(4): 1-34 2005.

[36] J.M. Harrisson and D.M. Kreps, *'Martingales and Arbitrage in Multiperiod Securities Markets'*, J. of Econ. Theory, 20: 381-408, 1979.

[37] S.L. Heston, *'A simple new formula for options with stochastic volatility'*, The Review of Fin. Stud., 6(2): 327-343, 1993.

[38] S.L. Heston, *'A Closed-Form Solution for Options with Stochastic Volatility with Applications to Bond and Currency Options'*, Technical report, Washington University of St. Louis, 1997.

[39] D.J. Higham, *'An Introduction to Financial Option Valuation'*, Cambridge University Press, 2004.

[40] C. Kahl and P. Jackel, *'Fast strong approximation Monte-Carlo schemes for stochastic volatility models'*, Working Paper, ABN-AMRO and University of Wuppertal, 2005.

[41] F. Le Floc'h, *'Fourier Integration and Stochastic Volatility Calibration'*, 2014. Available at SSRN: `http://ssrn.com/abstract=2362968`.

[42] K. Levenberg, *'A Method for the Solution of Certain Problems in Least Squares'*, J. of App. Math., 2(2): 164:168, 1944.

[43] A.L. Lewis, *'Option Valuation Under Stochastic Volatility: With Mathematica Code'*, Finance Press, 2000.

[44] M.I.A. Lourakis, *'levmar: Levenberg-Marquardt nonlinear least squares algorithms in C/C++'*, 2004. See: `http://www.ics.forth.gr/~lourakis/levmar/`.

[45] D.B. Madan, P.P. Carr and E.C. Chang, *'The Varance Gamma Process and Option Pricing'*, European Finance Review, 2: 79-105, 1998.

[46] D. Marquardt, *'An Algorithm for Least-Squares Estimation of Nonlinear Parameters'*, J. Soc. Indust. Appl. Math., 11(2): 431-441, 1963.

[47] R.C. Merton, *'Theory of Rational Option Pricing'*, Bell J. Econ. and Management Sci., 4: 141-183, 1973.

[48] S. Mikhailov and U. Nögel, *'Heston's Stochastic Volatility Model Implementation, Calibration and Some Extensions'*, Willmott Magazine, pages 74-94, 2003.

[49] The Numerical Algorithms Group (NAG), Oxford, United Kingdom, *'NAG Library Function Document e04fcc'*, Working paper. See: `http://www.nag.co.uk/numeric/cl/nagdoc_cl25/pdf/e04/e04fcc.pdf`.

[50] The Numerical Algorithms Group (NAG), Oxford, United Kingdom, *'NAG Library Function Document e04gbc'*, Working paper. See: `http://www.nag.co.uk/numeric/cl/nagdoc_cl25/pdf/e04/e04gbc.pdf`.

[51] The Numerical Algorithms Group (NAG), Oxford, United Kingdom, *'NAG Library Function Document e04unc'*, Working paper. See: `http://www.nag.co.uk/numeric/cl/nagdoc_cl25/pdf/e04/e04unc.pdf`.

[52] The Numerical Algorithms Group (NAG), Oxford, United Kingdom, *'NAG Library Function Document e04wdc'*, Working paper. See: `http://www.nag.co.uk/numeric/cl/nagdoc_cl25/pdf/e04/e04wdc.pdf`.

[53] The Numerical Algorithms Group (NAG), Oxford, United Kingdom, *'NAG Library Function Document s30nac'*, Working paper. See: `http://www.nag.co.uk/numeric/cl/nagdoc_cl25/pdf/s/s30nac.pdf`.

[54] The Numerical Algorithms Group (NAG), Oxford, United Kingdom, *'NAG Library Function Document s30ncc'*, Working paper. See: http://www.nag.co.uk/numeric/cl/nagdoc_cl25/pdf/s/s30ncc.pdf.

[55] U. Naumann, *'The Art of Differentiating Computer Programs. An Introduction to Algorithmic Differentiation.'*, SIAM, 2012.

[56] U. Naumann and J. du Toit, *'Adjoint Algorithmic Differentiation Tool Support for Typical Numerical Patterns in Computational Finance'*, 2014. See: https://www.nag.co.uk/doc/techrep/pdf/tr3_14.pdf.

[57] J. Nocedal and S.J. Wright, *'Numerical Optimization'*, Springer, 2006.

[58] V. Piterbarg, *'Time to Smile'*, Risk, 19(5): 71-75, 2006.

[59] A. Nag and J. McGeever, *'Foreign exchange, the world's biggest market, is shrinking'*, February 11, 2016. See: http://www.reuters.com/article/us-global-fx-peaktrading-idUSKCN0VK1UD.

[60] K. Scheinberg, *'Derivative free optimization method'*, Technical Report, IBM T.J. Watson Research Center, 2000.

[61] K. Schittkowski, *'NLPLSQ: A Fortran Implementation of an SQP-Gauss-Newton Algorithm for Least-Squares Optimization - User's Guide'*, 2015.

[62] R.U. Seydel, *'Tools for Computational Finance'*, Springer, 2006.

[63] D.F. Shanno, *'Conditioning of Quasi-Newton Methods for Function Minimization'*, Math. Comput., 24(111): 647-656, 1970.

[64] P. Spreij, E. Veerman and P. Vlaar, *'Multivariate Feller conditions in term structure models: Why do(n't) we care?'*, 2008. See: http://arxiv.org/abs/0804.1039v1.

[65] A. Wächter, L.T. Biegler, *'On the Implementation of a Primal-Dual Interior Point Filter Line Search Algorithm for Large-Scale Nonlinear Programming'*, Math. Prog., 106(1): 25-57, 2006.

[66] P. Wolfe, *'Convergence Conditions for Ascent Methods'*, SIAM Rev., 11(2): 226-235, 1969.

[67] P. Wolfe, *'Convergence Conditions for Ascent Methods. II: Some Corrections'*, SIAM Rev., 13(2): 185-188, 1971.

[68] H. Zhang, A.R. Conn and K. Scheinberg, *'A derivative-free algorithm for least-squares minimization'*, Comp. Opt. and Appl., 51(2): 481-507, 2010.

Appendix A

Derivations and formulae

A.1 Deriving the Heston pricing PDE

With help of the multi-dimensional Ito lemma and the martingale pricing approach indeed, the Heston pricing PDE is derived.
First of all, in order to apply Ito's lemma, the two Brownian motions need to be uncorrelated. Two independent BM's $\tilde{W}_t^1$ and $\tilde{W}_t^2$ are created such that the correlation structure is preserved between W_t^1 and W_t^2. This method is called the Cholesky decomposition:

$$\begin{aligned}
\begin{bmatrix} 1 & 0 \\ \rho & \sqrt{1-\rho^2} \end{bmatrix} \begin{bmatrix} d\tilde{W}_t^1 \\ d\tilde{W}_t^2 \end{bmatrix} &= \begin{bmatrix} d\tilde{W}_t^1 \\ \rho d\tilde{W}_t^1 + \sqrt{1-\rho^2} d\tilde{W}_t^2 \end{bmatrix} =: \begin{bmatrix} dW_t^1 \\ dW_t^2 \end{bmatrix}, \\
\Rightarrow dW_t^1 dW_t^2 &= d\tilde{W}_t^1 (\rho d\tilde{W}_t^1 + \sqrt{1-\rho^2} d\tilde{W}_t^2), \\
&= \rho \underbrace{\left(d\tilde{W}_t^1\right)^2}_{dt} + \sqrt{1-\rho^2} \underbrace{d\tilde{W}_t^1 d\tilde{W}_t^2}_{0}, \\
&= \rho \cdot dt.
\end{aligned}$$

Using this result it is possible to rewrite the Heston dynamics (3.4) in matrix-vector notation:

$$\begin{aligned}
\begin{bmatrix} dS_t \\ dv_t \end{bmatrix} &= \begin{bmatrix} \mu S_t \\ \lambda(1-v_t) \end{bmatrix} dt + \begin{bmatrix} \sigma\sqrt{v_t} S_t & 0 \\ 0 & \alpha\sqrt{v_t} \end{bmatrix} \begin{bmatrix} dW_t^1 \\ dW_t^2 \end{bmatrix}, \\
&= \begin{bmatrix} \mu S_t \\ \lambda(1-v_t) \end{bmatrix} dt + \begin{bmatrix} \sigma\sqrt{v_t} S_t & 0 \\ 0 & \alpha\sqrt{v_t} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ \rho & \sqrt{1-\rho^2} \end{bmatrix} \begin{bmatrix} d\tilde{W}_t^1 \\ d\tilde{W}_t^2 \end{bmatrix}, \\
&= \begin{bmatrix} \mu S_t \\ \lambda(1-v_t) \end{bmatrix} dt + \begin{bmatrix} \sigma\sqrt{v_t} S_t & 0 \\ \rho\alpha\sqrt{v_t} & \alpha\sqrt{v_t}\sqrt{1-\rho^2} \end{bmatrix} \begin{bmatrix} d\tilde{W}_t^1 \\ d\tilde{W}_t^2 \end{bmatrix}, \\
&= \underline{\mu} dt + \underline{\sigma} d\tilde{\mathbf{W}}_t. \qquad \text{(A.1)}
\end{aligned}$$

After having rewritten the model, apply the martingale pricing approach, which was initiated by Cox and Ross [19] and Harrison and Kreps [36]. It is known that for option price C_t and bank account $dB_t = \mu B_t dt$:

$$\mathbb{E}^{\mathbb{Q}}\left[\frac{C_T}{B_T} \middle| \mathcal{F}_t \right] = \frac{C_t}{B_t} =: \Pi_t.$$

The goal is to find the dynamics of portfolio Π_t:

$$d\Pi_t = d\left(\frac{C_t}{B_t}\right) = \frac{1}{B_t} dC_t - \mu \frac{C_t}{B_t} dt.$$

The only unknown here is dC_t so use the multi-dimensional Ito lemma with $\mathbf{X}_t = [S_t,\ v_t]^T$:

$$\begin{aligned}
dC_t &= \left(\frac{\partial C_t}{\partial t} + \sum_{i=1}^{n} \underline{\mu}_i \frac{\partial C_t}{\partial X_i} + \frac{1}{2}\sum_{i,j,k=1}^{n} \underline{\sigma}_{i,k}\underline{\sigma}_{j,k}\frac{\partial^2 C_t}{\partial X_i \partial X_j}\right) dt + \sum_{i,j=1}^{n} \underline{\sigma}_{i,j}\frac{\partial C_t}{\partial X_i} d\tilde{W}_t^j, \\
&= \left(\frac{\partial C_t}{\partial t} + \mu S_t \frac{\partial C_t}{\partial S_t} + \lambda(1-v_t)\frac{\partial C_t}{\partial v_t} + \frac{1}{2}\sigma^2 v_t S_t^2 \frac{\partial^2 C_t}{\partial S_t^2} + \sigma\rho\alpha v_t S_t \frac{\partial^2 C_t}{\partial S_t \partial v_t} + \frac{1}{2}\alpha^2 v_t \frac{\partial^2 C_t}{\partial v_t^2}\right) dt \\
&\quad + \sigma\sqrt{v_t} S_t \frac{\partial C_t}{\partial S_t} d\tilde{W}_t^1 + \rho\alpha\sqrt{v_t}\frac{\partial C_t}{\partial v_t} d\tilde{W}_t^1 + \alpha\sqrt{v_t}\sqrt{1-\rho^2}\frac{\partial C_t}{\partial v_t} d\tilde{W}_t^2. \\
&= \left(\frac{\partial C_t}{\partial t} + \mu S_t \frac{\partial C_t}{\partial S_t} + \lambda(1-v_t)\frac{\partial C_t}{\partial v_t} + \frac{1}{2}\sigma^2 v_t S_t^2 \frac{\partial^2 C_t}{\partial S_t^2} + \sigma\rho\alpha v_t S_t \frac{\partial^2 C_t}{\partial S_t \partial v_t} + \frac{1}{2}\alpha^2 v_t \frac{\partial^2 C_t}{\partial v_t^2}\right) dt \\
&\quad + \sigma\sqrt{v_t} S_t \frac{\partial C_t}{\partial S_t} dW_t^1 + \alpha\sqrt{v_t}\frac{\partial C_t}{\partial v_t} dW_t^2. \qquad \text{(A.2)}
\end{aligned}$$

$d\Pi_t$ should be free of dt-terms because the drift term must be eliminated from the portfolio dynamics in order to obtain martingality. Applying this results in the Heston pricing PDE in terms of S_t:

$$\begin{aligned}
&\frac{1}{B_t}\left(\frac{\partial C_t}{\partial t} + \mu S_t \frac{\partial C_t}{\partial S_t} + \lambda(1-v_t)\frac{\partial C_t}{\partial v_t} + \frac{1}{2}\sigma^2 v_t S_t^2 \frac{\partial^2 C_t}{\partial S_t^2}\right. \\
&\qquad \left. + \sigma\rho\alpha v_t S_t \frac{\partial^2 C_t}{\partial S_t \partial v_t} + \frac{1}{2}\alpha^2 v_t \frac{\partial^2 C_t}{\partial v_t^2}\right) - \mu\frac{C_t}{B_t} = 0, \\
&\Rightarrow \frac{\partial C_t}{\partial t} + \mu S_t \frac{\partial C_t}{\partial S_t} + \lambda(1-v_t)\frac{\partial C_t}{\partial v_t} + \frac{1}{2}\sigma^2 v_t S_t^2 \frac{\partial^2 C_t}{\partial S_t^2} \\
&\qquad + \sigma\rho\alpha v_t S_t \frac{\partial^2 C_t}{\partial S_t \partial v_t} + \frac{1}{2}\alpha^2 v_t \frac{\partial^2 C_t}{\partial v_t^2} - \mu C_t = 0. \qquad \text{(A.3)}
\end{aligned}$$

It is possible to simplify the PDE (A.3) further by doing the log-transformation on asset spot-price: $x_t = \log(S_t)$. In order to do so, rewrite the derivatives of option price C_t with respect to S_t in terms of C_t derivatives with respect to x_t:

$$\begin{aligned}
\frac{\partial C_t}{\partial S_t} &= \frac{1}{S_t}\frac{\partial C_t}{\partial x_t}, \\
\frac{\partial^2 C_t}{\partial S_t^2} &= \frac{1}{S_t^2}\left[\frac{\partial^2 C_t}{\partial x_t^2} - \frac{\partial C_t}{\partial x_t}\right], \\
\frac{\partial^2 C_t}{\partial S_t \partial v_t} &= \frac{1}{S_t}\frac{\partial^2 C_t}{\partial x_t \partial v_t}.
\end{aligned}$$

Applying the above described transformation and the transformation $\tau = T - t$ results in the Heston pricing PDE in terms of x_t, so corresponding to the Heston formulation in (3.6):

$$\begin{aligned}
&-\frac{\partial C_t}{\partial \tau} + \left(\mu - \frac{1}{2}\sigma^2 v_t\right)\frac{\partial C_t}{\partial x_t} + \lambda(1-v_t)\frac{\partial C_t}{\partial v_t} + \frac{1}{2}\sigma^2 v_t \frac{\partial^2 C_t}{\partial x_t^2} \\
&\qquad + \sigma\rho\alpha v_t \frac{\partial^2 C_t}{\partial x_t \partial v_t} + \frac{1}{2}\alpha^2 v_t \frac{\partial^2 C_t}{\partial v_t^2} - \mu C_t = 0. \qquad \text{(A.4)}
\end{aligned}$$

A.2 Distribution characteristics

1. Gamma distribution:

- Parameters $\alpha, \beta > 0$.
- $x \in (0, \infty)$.
- Probability density function:

$$f(x) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x},$$

  where

$$\Gamma(z) = \int_0^\infty t^{z-1} e^{-t}\, dt.$$

- Characteristic function:

$$\phi(x) = \left(1 - \frac{ix}{\beta}\right)^{-\alpha}.$$

- Cumulants:

$$\kappa_1 = \frac{\alpha}{\beta},\ \kappa_2 = \frac{\alpha}{\beta^2},\ \kappa_3 = \frac{2\alpha}{\beta^3},\ \kappa_4 = \frac{6\alpha}{\beta^4}.$$

- Cumulative distribution function:

$$F(x) = \frac{1}{\Gamma(\alpha)} \gamma(\alpha, \beta x),$$

  where

$$\gamma(s, x) = \int_0^x t^{s-1} e^{-t}\, dt.$$

2. Normal-inverse Gaussian distribution:
   - Parameters $\mu, \alpha, \beta, \delta \in \mathbb{R},\ \gamma = \sqrt{\alpha^2 - \beta^2}$.
   - $x \in \mathbb{R}$.
   - Probability density function:

$$f(x) = \frac{\alpha\delta \cdot K_1\left(\alpha\sqrt{\delta^2 + (x-\mu)^2}\right)}{\pi\sqrt{\delta^2 + (x-\mu)^2}} e^{\delta\gamma + \beta(x-\mu)},$$

     where $K_1(\cdot)$ is the modified Bessel function of the third kind.
   - Characteristic function:

$$\phi(x) = \exp\left\{i\mu x + \delta\left(\gamma - \sqrt{\alpha^2 - (\beta + ix)^2}\right)\right\}.$$

   - Cumulants:

$$\kappa_1 = \mu + \frac{\delta\beta}{\gamma},\ \kappa_2 = \frac{\delta\alpha^2}{\gamma^3},\ \kappa_3 = \frac{3\beta\delta\alpha^2}{\gamma^5},\ \kappa_4 = \frac{3\delta\alpha^4 + 12\delta(\alpha\beta)^2}{\gamma^7}.$$

   - Cumulative distribution function: no analytic expression. In this case an approximation is done using a numerical quadrature rule applied on the pdf.

3. Inverse Gaussian distribution:

   - Parameters $\mu, \lambda > 0$.
   - $x \in (0, \infty)$.
   - Probability density function:

$$f(x) = \sqrt{\frac{\lambda}{2\pi x^3}} \exp\left\{-\frac{\lambda(x-\mu)^2}{2\mu^2 x}\right\}.$$

   - Characteristic function:

$$\phi(x) = \exp\left\{\frac{\lambda}{\mu}\left(1 - \sqrt{1 - \frac{2\mu^2 i x}{\lambda}}\right)\right\}.$$

   - Cumulants:

$$\kappa_1 = \mu,\ \kappa_2 = \frac{\mu^3}{\lambda},\ \kappa_3 = \frac{3\mu^5}{\lambda^2},\ \kappa_4 = \frac{15\mu^7}{\lambda^3}.$$

   - Cumulative distribution function:

$$F(x) = \Phi\left(\sqrt{\frac{\lambda}{x}}\left(\frac{x}{\mu} - 1\right), 0, 1\right) + \exp\left(\frac{2\lambda}{\mu}\right)\Phi\left(-\sqrt{\frac{\lambda}{x}}\left(\frac{x}{\mu} + 1\right), 0, 1\right),$$

     where $\Phi(\,\cdot\,, 0, 1)$ is the standard normal CDF.

4. Exponentially modified Gaussian distribution:

   - Parameters $\mu \in \mathbb{R},\ \sigma^2 > 0,\ \lambda > 0$.
   - $x \in \mathbb{R}$.
   - Probability density function:

$$f(x) = \frac{\lambda}{2} e^{\frac{\lambda}{2}\left(2\mu + \lambda\sigma^2 - 2x\right)} \cdot \operatorname{erfc}\left(\frac{\mu + \lambda\sigma^2 - x}{\sigma\sqrt{2}}\right).$$

   - Characteristic function:

$$\phi(x) = \left(1 - \frac{ix}{\lambda}\right)^{-1} \exp\left\{i\mu x - \frac{1}{2}\sigma^2 x^2\right\}.$$

   - Cumulants:

$$\kappa_1 = \mu + \frac{1}{\lambda},\ \kappa_2 = \sigma^2 + \frac{1}{\lambda^2},\ \kappa_3 = \frac{2\sigma}{\lambda^3},\ \kappa_4 = \frac{3}{\sigma^4\lambda^2}\left(1 + \frac{3}{\sigma^2\lambda^2}\right)\left(\sigma^2 + \frac{1}{\lambda^2}\right).$$

   - Cumulative distribution function:

$$F(x) = \Phi\left(u, 0, v\right) - \exp\left\{-u + \frac{v^2}{2} + \log\left(\Phi\left(u, v^2, v\right)\right)\right\},$$

     where $u = \lambda(x - \mu),\ v = \lambda\sigma$ and $\Phi(\,\cdot\,, \mu, \sigma)$ is Gaussian CDF.

A.3 Truncation range COS method

As stated in Section 4.1.3, the derivatives of the chf's, which are at this point unknown analytically for the Heston model with term structure and are too lengthy when expressed analytically for the single-term Heston model, can be approximated using central finite differences:

$$\begin{aligned}
\left.\frac{d\phi(-it)}{dt}\right|_{t=0} &\approx \frac{1}{2\Delta t}\left[\phi\left(-i\Delta t\right)-\phi\left(i\Delta t\right)\right]+\mathcal{O}(\Delta t^2),\\
\left.\frac{d^2\phi(-it)}{dt^2}\right|_{t=0} &\approx \frac{1}{(\Delta t)^2}\left[\phi\left(-i\Delta t\right)-2+\phi\left(i\Delta t\right)\right]+\mathcal{O}(\Delta t^2),\\
\left.\frac{d^3\phi(-it)}{dt^3}\right|_{t=0} &\approx \frac{1}{2(\Delta t)^3}\left[\phi\left(-i2\Delta t\right)-2\phi\left(-i\Delta t\right)+2\phi\left(i\Delta t\right)-\phi\left(i\Delta t\right)\right]+\mathcal{O}(\Delta t^2),\\
\left.\frac{d^4\phi(-it)}{dt^4}\right|_{t=0} &\approx \frac{1}{(\Delta t)^4}\left[\phi\left(-i2\Delta t\right)-4\phi\left(-i\Delta t\right)+6-4\phi\left(i\Delta t\right)+\phi\left(i2\Delta t\right)\right]+\mathcal{O}(\Delta t^2).
\end{aligned}$$

These cumulant approximations can be used in equations (4.8), (5.10a) and (5.10b) to end up with truncation range $[a,b]$.

A.4 Formula for the number of terms within the COS method

As described in Chapter 5, some improvements have been made to the COS method. One of them is the way to determine the amount of terms N in the Fourier cosine expansion of a density function for a given truncation range $[a,b]$. Remember that the goal is to minimize the series truncation error of the density function on the interval $[a,b]$:

$$\epsilon_2(x) = \left|\sum_{k=N}^{+\infty} F_k\cdot\cos\left(k\pi\frac{x-a}{b-a}\right)\right|.$$

One thing that should be kept in mind regarding the value for N is that a larger value of N will result in a lower value of $\epsilon_2(x)$, but this comes at the cost of slowing down an implementation of the COS method. Therefore it is desirable to have an expression for N that addresses this trade-off between speed and accuracy. $\epsilon_2(x)$ is basically the difference between (4.3) and (4.4) due to the truncation of the summation. On assumption that the terms in the sum are convergent towards zero in the tail, it seems to be sufficient to try and say something about the size of the N-th term in the summation of (4.4). In [26], with help of [8], it is stated that the cosine series expansion exhibits exponential convergence in our case. Combining this property together with the decaying properties of a pdf, it can be said that the assumptions are met and therefore the choice to try and bound the N-th term in the summation of (4.4) is a sensible thing to do. The choice to work with N in stead of $N-1$ is without loss of generality, it is simply more convenient.

To summarize, the objective is to bound the absolute value of the N-th term of the summation in (4.4) under a given accuracy level ε which can be set as desired by a user. In order to do so, consider the following equation where $\Re\{\cdot\}$ and $\Im\{\cdot\}$ denote the real and imaginary part

respectively.

$$\begin{aligned}
&\left|F_N \cdot \cos\left(k\pi\frac{x-a}{b-a}\right)\right| \leq \varepsilon \\
\Leftrightarrow\ &|F_N| \cdot \underbrace{\left|\cos\left(k\pi\frac{x-a}{b-a}\right)\right|}_{\leq 1} \leq \varepsilon \\
\Rightarrow\ &|F_N| \leq \varepsilon \\
\Leftrightarrow\ &\left|\frac{2}{b-a}\Re\left\{\phi\left(\frac{N\pi}{b-a}\right)\cdot\exp\left(\frac{-iaN\pi}{b-a}\right)\right\}\right| \leq \varepsilon \\
\Leftrightarrow\ &\left|\Re\left\{\phi\left(\frac{N\pi}{b-a}\right)\cdot\exp\left(\frac{-iaN\pi}{b-a}\right)\right\}\right| \leq \frac{(b-a)\varepsilon}{2} \\
\Leftrightarrow\ &\left|\Re\left\{\phi\left(\frac{N\pi}{b-a}\right)\right\}\cdot\cos\left(\frac{-aN\pi}{b-a}\right) - \Im\left\{\phi\left(\frac{N\pi}{b-a}\right)\right\}\cdot\sin\left(\frac{-aN\pi}{b-a}\right)\right| \leq \frac{(b-a)\varepsilon}{2}
\end{aligned}$$

For the time being do the following substitution

$$X = \frac{N\pi}{b-a},$$

resulting in the following inequality that should be satisfied:

$$\left|\Re\left\{\phi\left(X\right)\right\}\cdot\cos\left(-aX\right) - \Im\left\{\phi\left(X\right)\right\}\cdot\sin\left(-aX\right)\right| \leq \frac{(b-a)\varepsilon}{2}. \tag{A.5}$$

A.4.1 Single term Heston case

Before considering the case of the Heston model with term structure, which is a complicated case, start the analysis by first taking the original Heston model (so the Heston model with one term in the term structure). The joint chf for this model is defined in equation (3.16), which is stated again below for convenience.

$$\phi_{0T}(X,V|x_0,v_0) = e^{C(X,V)+D_2(X,V)v_0+iXx_0}.$$

In an option pricing environment the joint chf is not required, but the marginal chf is. The latter is simply the joint chf evaluated at $V=0$. From now on the following definition of the chf will be used, where the dependence on time interval $[0,T]$ is dropped , as well as the dependence on V, x_0 and v_0, in order to simplify notation:

$$\begin{aligned}
\phi(X) &= e^{C(X)+D_2(X)v_0+iXx_0}, && \text{(A.6)}\\
C(X) &= i\mu X\tau + \frac{\lambda}{\alpha^2}\left(-2\log\left(\frac{1-\tilde{g}e^{-d\tau}}{1-\tilde{g}}\right)+(\lambda-\rho\sigma\alpha iX-d)\tau\right), && \text{(A.7)}\\
D_2(X) &= \frac{\lambda-\rho\sigma\alpha iX+d}{\alpha^2}\left(\frac{g-\tilde{g}e^{-d\tau}}{1-\tilde{g}e^{-d\tau}}\right), && \text{(A.8)}\\
d(X) &= \sqrt{(\lambda-\rho\sigma\alpha iX)^2+\alpha^2\sigma^2X(i+X)}, && \text{(A.9)}\\
g(X) &= \frac{\lambda-\rho\sigma\alpha iX-d}{\lambda-\rho\sigma\alpha iX+d}, && \text{(A.10)}\\
\tilde{g}(X) &= g(X). && \text{(A.11)}
\end{aligned}$$

Start by rewriting (A.6):

$$\begin{aligned}\phi(X) &= \exp\{C(X) + D_2(X)v_0 + iXx_0\},\\ &= \exp\{\Re\{C(X)\} + i\cdot\Im\{C(X)\} + [\Re\{D_2(X)\} + i\cdot\Im\{D_2(X)\}]v_0 + iXx_0\},\\ &= e^Z\left[\cos(W) + i\cdot\sin(W)\right], \end{aligned} \tag{A.12}$$

where $W = \Im\{C(X)\} + \Im\{D_2(X)\} + Xx_0$ and $Z = \Re\{C(X)\} + \Re\{D_2(X)\}v_0$. Going back and substituting (A.12) into (A.5) gives:

$$\begin{aligned} &\left|e^Z\cos(W)\cos(-aX) - e^Z\sin(W)\sin(-aX)\right| \leq \frac{(b-a)\varepsilon}{2},\\ \Leftrightarrow\ &\left|e^Z\right|\cdot\underbrace{\left|\cos(W-aX)\right|}_{\leq 1} \leq \frac{(b-a)\varepsilon}{2},\\ \Rightarrow\ &e^Z \leq \frac{(b-a)\varepsilon}{2}. \end{aligned} \tag{A.13}$$

The current equation of interest is given by (A.13). The first step is to look into the building blocks of $\Re\{C(X)\}$ and $\Re\{D_2(X)\}$. To simplify calculations, a leading order approximation in terms of X, denoted as $\overset{\text{loa}}{\approx}$, approach will be taken. This means that only the leading term in X will be kept for the next step of the computation and all the other (lower order) terms are discarded. It is possible to apply this technique because typically the value of N is large and therefore the value of X will be large as well. Start with $d(X)$, as given in (A.9).

$$\begin{aligned} d(X) &= \sqrt{(\lambda - \rho\sigma\alpha iX)^2 + \alpha^2\sigma^2X(i+X)},\\ &= \sqrt{\underbrace{\left[\lambda^2 + \sigma^2\alpha^2X^2(1-\rho^2)\right]}_{M} + i\,\underbrace{\left[\sigma\alpha X(\sigma\alpha - 2\lambda\rho)\right]}_{N}},\\ &= \pm\left(\underbrace{\sqrt{\frac{M+\sqrt{M^2+N^2}}{2}}}_{\gamma} + i\cdot\underbrace{\operatorname{sgn}(N)\sqrt{\frac{-M+\sqrt{M^2+N^2}}{2}}}_{\delta}\right), \end{aligned}$$

where

$$\operatorname{sgn}(x) := \begin{cases} -1 & \text{if } x<0,\\ 0 & \text{if } x=0,\\ 1 & \text{if } x>0. \end{cases}$$

Consider γ separately:

$$\begin{aligned} \gamma &= \sqrt{\frac{M+\sqrt{M^2+N^2}}{2}},\\ &= \sqrt{\frac{1}{2}\left\{\lambda^2 + \sigma^2\alpha^2(1-\rho^2)X^2 + \sqrt{\sigma^4\alpha^4(1-\rho^2)^2X^4 + \mathcal{O}(X^2)}\right\}},\\ &\overset{\text{loa}}{\approx} \sqrt{\frac{1}{2}\left\{\lambda^2 + \sigma^2\alpha^2(1-\rho^2)X^2 + \sqrt{\sigma^4\alpha^4(1-\rho^2)^2X^4}\right\}},\\ &\overset{\text{loa}}{\approx} \sigma\alpha\sqrt{1-\rho^2}X. \end{aligned} \tag{A.14}$$

In a similar fashion an expression for δ is obtained:

$$\begin{aligned}
\delta &= \operatorname{sgn}(N)\sqrt{\frac{-M+\sqrt{M^2+N^2}}{2}}, \\
&\overset{\text{loa}}{\approx} \operatorname{sgn}(N)\sqrt{\frac{1}{2}\left\{-(\lambda^2+\sigma^2\alpha^2(1-\rho^2)X^2)+\sqrt{\sigma^4\alpha^4(1-\rho^2)^2X^4}\right\}}, \\
&= \operatorname{sgn}(N)\frac{\lambda}{\sqrt{2}}i. \qquad \text{(A.15)}
\end{aligned}$$

Combining (A.14) and (A.15) leads to the following leading order approximation of $d(x)$:

$$\begin{aligned}
d(X) &= \pm(\gamma + i\cdot\delta), \\
&\overset{\text{loa}}{\approx} \pm\left(\sigma\alpha\sqrt{1-\rho^2}X + \operatorname{sgn}(N)\frac{\lambda}{\sqrt{2}}i\right), \\
&\overset{\text{loa}}{\approx} \sigma\alpha\sqrt{1-\rho^2}X, \qquad \text{(A.16)}
\end{aligned}$$

where in (A.16) the positive branch of the complex square root has been picked. Note that $d(X) \geq 0$. By definition $\alpha > 0$, $\sigma > 0$, $-1 < \rho < 1$, $N > 0$ and $b > a$. This means that $X > 0$ and thus $d(X) > 0$. Next consider $g(X)$ as defined in (A.10) and perform a leading order approximation:

$$\begin{aligned}
g(X) &= \frac{\lambda - \rho\sigma\alpha iX - d}{\lambda - \rho\sigma\alpha iX + d}, \\
&\overset{\text{loa}}{\approx} \frac{\lambda - \rho\sigma\alpha iX - \sigma\alpha\sqrt{1-\rho^2}X}{\lambda - \rho\sigma\alpha iX + \sigma\alpha\sqrt{1-\rho^2}X}, \\
&= \frac{\left[\lambda - \sigma\alpha\sqrt{1-\rho^2}X\right] + i\cdot[-\rho\sigma\alpha X]}{\left[\lambda + \sigma\alpha\sqrt{1-\rho^2}X\right] + i\cdot[-\rho\sigma\alpha X]}, \\
&= \frac{\left[\lambda - \sigma\alpha\sqrt{1-\rho^2}X\right]\cdot\left[\lambda + \sigma\alpha\sqrt{1-\rho^2}X\right] + \left[-2\rho\sqrt{1-\rho^2}\right]\cdot\left[-2\rho\sqrt{1-\rho^2}\right]}{\left[\lambda + \sigma\alpha\sqrt{1-\rho^2}X\right]^2 + [-\rho\sigma\alpha X]^2} \\
&\quad + i\cdot\frac{\left[\lambda + \sigma\alpha\sqrt{1-\rho^2}X\right]\cdot\left[-2\rho\sqrt{1-\rho^2}\right] - \left[\lambda - \sigma\alpha\sqrt{1-\rho^2}X\right]\cdot\left[-2\rho\sqrt{1-\rho^2}\right]}{\left[\lambda + \sigma\alpha\sqrt{1-\rho^2}X\right]^2 + [-\rho\sigma\alpha X]^2} \\
&\overset{\text{loa}}{\approx} \left[2\rho^2 - 1\right] + i\left[-2\rho\sqrt{1-\rho^2}\right]. \qquad \text{(A.17)}
\end{aligned}$$

Now a leading order approximation is present of all the building blocks of $\Re\{C(X)\}$ and $\Re\{D_2(X)\}$. Due to the fact that $d(X) > 0$ and $\tau > 0$, it is clear that $e^{-d(X)\tau} \to 0$ when N (and therefore X) becomes large. Using the previous result and the results from (A.16) and

(A.17) it is possible to do a leading order approximation of $\Re\{C(X)\}$.

$$\begin{aligned}
\Re\{C(X)\} &= \Re\left\{i\mu X\tau + \frac{\lambda}{\alpha^2}\left[-2\log\left(\frac{1-\tilde{g}e^{-d\tau}}{1-\tilde{g}}\right) + (\lambda - \rho\sigma\alpha iX - d)\tau\right]\right\}, \\
&\overset{\text{loa}}{\approx} -\frac{2\lambda}{\alpha^2}\Re\left\{\log\left(\frac{1-\left(\left[2\rho^2-1\right] + i\left[-2\rho\sqrt{1-\rho^2}\right]\right)e^{-\tau(\sigma\alpha\sqrt{1-\rho^2}X)}}{1-\left(\left[2\rho^2-1\right] + i\left[-2\rho\sqrt{1-\rho^2}\right]\right)}\right)\right\} \\
&\quad + \frac{\lambda}{\alpha^2}(\lambda - \sigma\alpha\sqrt{1-\rho^2}X)\tau, \\
&\overset{\text{loa}}{\approx} -\frac{2\lambda}{\alpha^2}\Re\left\{\log\left(\frac{1}{1-\left(\left[2\rho^2-1\right] + i\left[-2\rho\sqrt{1-\rho^2}\right]\right)}\right)\right\} + \frac{\lambda}{\alpha^2}(\lambda - \sigma\alpha\sqrt{1-\rho^2}X)\tau, \\
&\overset{\text{loa}}{\approx} \frac{\lambda}{\alpha^2}\log\left(4\left(1-\rho^2\right)\right) + \frac{\lambda}{\alpha^2}(\lambda - \sigma\alpha\sqrt{1-\rho^2}X)\tau, \\
&\overset{\text{loa}}{\approx} -\frac{\lambda\tau\sigma}{\alpha}\sqrt{1-\rho^2}X. \qquad \text{(A.18)}
\end{aligned}$$

In a similar fashion, the following leading order approximation of $\Re(D_2(X))$ is computed:

$$\begin{aligned}
\Re\{D_2(X)\} &= \Re\left\{\frac{\lambda - \rho\sigma\alpha iX + d}{\alpha^2}\left(\frac{g-\tilde{g}e^{-d\tau}}{1-\tilde{g}e^{-d\tau}}\right)\right\}, \\
&= \Re\left\{\frac{\lambda - \rho\sigma\alpha iX - d}{\alpha^2}\left(\frac{1-e^{-d\tau}}{1-\tilde{g}e^{-d\tau}}\right)\right\}, \\
&\overset{\text{loa}}{\approx} \frac{\lambda}{\alpha^2} - \frac{1}{\alpha^2}\sigma\alpha\sqrt{1-\rho^2}X, \\
&\overset{\text{loa}}{\approx} -\frac{\sigma}{\alpha}\sqrt{1-\rho^2}X. \qquad \text{(A.19)}
\end{aligned}$$

Using the leading order approximations from (A.18) and (A.19), it is possible to perform the final steps of getting a leading order approximation solution for inequality(A.13):

$$\begin{aligned}
&\exp\left\{\Re\{C(X)\} + \Re\{D_2(X)\}v_0\right\} \leq \frac{(b-a)\varepsilon}{2}, \\
\overset{\text{loa}}{\Leftrightarrow}\ &\exp\left\{-\frac{\lambda\tau\sigma}{\alpha}\sqrt{1-\rho^2}X + -\frac{\sigma}{\alpha}\sqrt{1-\rho^2}Xv_0\right\} \leq \frac{(b-a)\varepsilon}{2}, \\
\Leftrightarrow\ &X \geq \log\left(\frac{(b-a)\varepsilon}{2}\right)\cdot\left[-\frac{\sigma\sqrt{1-\rho^2}}{\alpha}(\lambda\tau + v_0)\right]^{-1}, \\
\Leftrightarrow\ &N \geq \frac{b-a}{\pi}\cdot\log\left(\frac{(b-a)\varepsilon}{2}\right)\cdot\left[-\frac{\sigma\sqrt{1-\rho^2}}{\alpha}(\lambda\tau + v_0)\right]^{-1}. \qquad \text{(A.20)}
\end{aligned}$$

This means that given a truncation range $[a,b]$ and a value for ε (e.g. $\varepsilon = 10^{-15}$) one has an expression for the amount of terms N in the Fourier cosine expansion.

A.4.2 Multi term Heston case

Up next is the formula for the Heston model with term structure. Recall from Section 3.2 that the joint chf for this model is given by:

$$\begin{aligned}
\phi_{0v}(X,V|x_0,v_0) &= \exp(C_{0v}(X,V) + D_{0v,2}(X,V)v_0 + iXx_0),\\
C_{0v}(X,V) &= C_{uv}(X,V) + C_{0u}\left(X, \frac{1}{i}D_{uv,2}(X,V)\right),\\
D_{0v,2}(X,V) &= D_{0u,2}\left(X, \frac{1}{i}D_{uv,2}(X,V)\right),\\
C_{uv}(X,V) &= i\mu_{uv}X\tau_{uv} + \frac{\lambda_{uv}}{\alpha_{uv}^2}\left(-2\log\left(\frac{1-\tilde{g}_{uv}e^{-d_{uv}\tau_{uv}}}{1-\tilde{g}_{uv}}\right) + (\lambda_{uv} - \rho_{uv}\sigma_{uv}\alpha_{uv}iX - d_{uv})\tau_{uv}\right),\\
D_{uv,2}(X,V) &= \frac{\lambda_{uv} - \rho_{uv}\sigma_{uv}\alpha_{uv}iX + d}{\alpha_{uv}^2}\left(\frac{g_{uv} - \tilde{g}_{uv}e^{-d_{uv}\tau_{uv}}}{1 - g_{uv}^{\sim}e^{-d_{uv}\tau_{uv}}}\right),\\
\tilde{g}_{uv} &= \frac{\lambda_{uv} - \rho_{uv}\sigma_{uv}\alpha_{uv}iX - d_{uv} - iV\alpha_{uv}^2}{\lambda_{uv} - \rho_{uv}\sigma_{uv}\alpha_{uv}iX + d_{uv} - iV\alpha_{uv}^2},\\
g_{uv}(X,V) &= \frac{\lambda_{uv} - \rho_{uv}\sigma_{uv}\alpha_{uv}iX - d_{uv}}{\lambda_{uv} - \rho_{uv}\sigma_{uv}\alpha_{uv}iX + d_{uv}},\\
d_{uv}(X,V) &= \sqrt{(\lambda_{uv} - \rho_{uv}\sigma_{uv}\alpha_{uv}iX)^2 + \alpha_{uv}^2\sigma_{uv}^2X(i+X)}.
\end{aligned}$$

Once again the marginal chf is obtained by evaluating the joint chf above at $V = 0$.
First of all consider a term structure of two terms, s.t.

$$[0,T] = \underbrace{[t_0,t_u]}_{\tau_{0u}} \cup \underbrace{[t_u,t_v]}_{\tau_{uv}}.$$

To simplify notation, denote $\tau_{0u} = \tau_1$ and $\tau_{uv} = \tau_2$. Do this for all the other parameters, e.g. $\lambda_{0u} = \lambda_1$. The notation of $C_{0v}(X,V)$ where v does not denote the end of the first term will remain unchanged. However, $C_{uv}(X,V)$ will from now on be denoted as $C_2(X,V)$. Similarly $C_{0u}(X,V) = C_1(X,V)$ and $D_{uv,2}(X,V) = D_{2,2}(X,V)$.
Similar to (A.13), the following equation is of interest:

$$\exp\left\{\Re\left\{C_{0v}(X,0)\right\} + \Re\left\{D_{0v,2}(X,0)\right\}v_0\right\} \leq \frac{(b-a)\varepsilon}{2}. \tag{A.21}$$

Note that the dependence on V is not neglected yet because of the recursive definitions of $C_{0v}(X,V)$ and $D_{0v,2}(X,V)$ where the second argument of both functions might still play a role. First of all consider the first term in the exponent in equation (A.21):

$$\Re\left\{C_{0v}(X,0)\right\} = \Re\left\{C_2(X,0)\right\} + \underbrace{\Re\left\{C_1\left(X, \frac{1}{i}D_{2,2}(X,0)\right)\right\}}_{\Omega}. \tag{A.22}$$

Because $d_i(X,V)$ and $g_i(X,V)$ are in fact only functions of X, i.e. they do not depend on the second variable V, the leading order approximations for the single-term case can be extended to the multi-term case as follows:

$$d_i(X,V) \overset{\text{loa}}{\approx} \sigma_i\alpha_i\sqrt{1-\rho_i^2}X, \tag{A.23}$$

$$g_i(X,V) \overset{\text{loa}}{\approx} \left[2\rho_i^2 - 1\right] + i\left[-2\rho_i\sqrt{1-\rho_i^2}\right]. \tag{A.24}$$

The first part of (A.22) is still similar to the result for the Heston model with only one term since $g_i(X,V) = \tilde{g}_i(X,V)$ holds for $V = 0$ and there is no further dependence of $C_2(X,V)$ on the second argument. Therefore:

$$\Re\left\{C_2(X,0)\right\} \overset{\text{loa}}{\approx} -\frac{\lambda_2\tau_2\sigma_2}{\alpha_2}\sqrt{1-\rho_2^2}X. \tag{A.25}$$

On the other hand, the second part of (A.22) does depend on the second argument, so the results for the single-term case cannot be extended to the multi-term case as easily as for the first part of (A.22). Note that in this case $g_i(X,V) \neq \tilde{g}_i(X,V)$ does not hold any more ($V \neq 0$ in this situation) so it is time to look into that equation for this specific case:

$$\tilde{g}_1\left(X, \frac{1}{i}D_{2,2}(X,0)\right) = \frac{\lambda_1 - \rho_1\sigma_1\alpha_1 iX - d_1 - D_{2,2}(X,0)\alpha_1^2}{\lambda_1 - \rho_1\sigma_1\alpha_1 iX + d_1 - D_{2,2}(X,0)\alpha_1^2}. \tag{A.26}$$

Note that here we need $D_{2,2}(X,0)$ of which we already know the real part from calculations for the single term case. Calculate the imaginary part of $D_{2,2}(X,0)$ before continuing leading order approximation of (A.26).

$$\Re\left\{D_{2,2}(X,0)\right\} \overset{\text{loa}}{\approx} -\frac{\sigma_2}{\alpha_2}\sqrt{1-\rho_2^2}X, \tag{A.27}$$

$$\begin{aligned}
\Im\left\{D_{2,2}(X,0)\right\} &= \Im\left\{\frac{\lambda_2 - \rho_2\sigma_2\alpha_2 iX - d_2}{\alpha_2^2}\left(\frac{1-e^{-d_2\tau_2}}{1-\tilde{g}_2 e^{-d_2\tau_2}}\right)\right\}, \\
&\overset{\text{loa}}{\approx} \frac{1}{\alpha_2}\Im\left\{\left[\lambda_2 - \rho_2\sigma_2\alpha_2 iX - \sigma_2\alpha_2\sqrt{1-\rho_2^2}X - \frac{\lambda_2}{\sqrt{2}}i\right]\cdot\left[1 - \frac{\rho_2}{\sqrt{1-\rho_2^2}}i\right]\right\}, \\
&= -\frac{\lambda_2}{\alpha_2}\left[\frac{1}{\sqrt{2}} + \frac{\rho_2}{\sqrt{1-\rho_2^2}}\right], \\
&\overset{\text{loa}}{\approx} 0.
\end{aligned} \tag{A.28}$$

Combining the results from (A.23) and (A.26) till (A.28), continue as follows:

$$\begin{aligned}
\tilde{g}_1\left(X, \frac{1}{i}D_{2,2}(X,0)\right) &\overset{\text{loa}}{\approx} \frac{\lambda_1 - \rho_1\sigma_1\alpha_1 iX - \sigma_1\alpha_1\sqrt{1-\rho_1^2}X + \frac{\alpha_1^2\sigma_2}{\alpha_2}\sqrt{1-\rho_2^2}X}{\lambda_1 - \rho_1\sigma_1\alpha_1 iX + \sigma_1\alpha_1\sqrt{1-\rho_1^2}X + \frac{\alpha_1^2\sigma_2}{\alpha_2}\sqrt{1-\rho_2^2}X}, \\
&= \frac{\overbrace{\left[\lambda_1 - \sigma_1\alpha_1\sqrt{1-\rho_1^2}X + \frac{\alpha_1^2\sigma_2}{\alpha_2}\sqrt{1-\rho_2^2}X\right]}^{\Delta} + \overbrace{\left[-\rho_1\sigma_1\alpha_1 X\right]}^{\Gamma} i}{\underbrace{\left[\lambda_1 + \sigma_1\alpha_1\sqrt{1-\rho_1^2}X + \frac{\alpha_1^2\sigma_2}{\alpha_2}\sqrt{1-\rho_2^2}X\right]}_{\Sigma} + \underbrace{\left[-\rho_1\sigma_1\alpha_1 X\right]}_{\Pi} i}, \\
&= \frac{\Delta\Sigma + \Gamma\Pi}{\Sigma^2 + \Pi^2} + \frac{\Gamma\Sigma - \Delta\Pi}{\Sigma^2 + \Pi^2}i, \\
&\overset{\text{loa}}{\approx} \frac{\mathcal{O}(X^2)}{\mathcal{O}(X^2)} + \frac{\mathcal{O}(X^2)}{\mathcal{O}(X^2)}i, \\
&\overset{\text{loa}}{\approx} \eta_1 + \chi_1 i,
\end{aligned} \tag{A.29}$$

where $\eta_1, \chi_1 \in \mathbb{R}$ are both constant. This means that $\tilde{g}_1\left(X, \frac{1}{i}D_{2,2}(X,0)\right)$ does not depend on X after doing a leading order approximation. This proves to be a useful result for the next step.

Go back to (A.22) and try to find a leading order approximation for the second part, Ω, where results from (A.23), (A.24) and (A.29) are used as well as the fact that $e^{-d_1\tau_1} \to 0$ as N grows large.

$$\begin{aligned}
\Omega &= \Re\left\{C_1\left(X, \frac{1}{i}D_{2,2}(X,0)\right)\right\}, \\
&= \frac{\lambda_1}{\alpha_1^2}\left(-2\cdot\Re\left\{\log\left(\frac{1-\tilde{g}_1 e^{-d_1\tau_1}}{1-\tilde{g}_1}\right)\right\} + (\lambda_1 - d_1)\tau_1\right), \\
&\overset{\text{loa}}{\approx} \underbrace{\frac{-2\lambda_1}{\alpha_1^2}\cdot\Re\left\{\log\left(\frac{1}{(1-\eta_1)-\chi_1 i)}\right)\right\} + \frac{\lambda_1^2\tau_1}{\alpha_1^2}}_{\mathcal{O}(1)} - \frac{\lambda_1\tau_1\sigma_1}{\alpha_1}\sqrt{1-\rho_1^2}X, \\
&\overset{\text{loa}}{\approx} -\frac{\lambda_1\tau_1\sigma_1}{\alpha_1}\sqrt{1-\rho_1^2}X. \qquad \text{(A.30)}
\end{aligned}$$

Approximate the second part of (A.21) using the same intermediate results and property as above:

$$\begin{aligned}
\Re\left\{D_{0v,2}(X,0)\right\} &= \Re\left\{D_{1,2}\left(X, \frac{1}{i}D_{2,2}(X,0)\right)\right\}, \\
&\overset{\text{loa}}{\approx} \Re\left\{D_{1,2}\left(X, -\frac{1}{i}\frac{\sigma_2}{\alpha_2}\sqrt{1-\rho_2^2}X\right)\right\}, \\
&= \Re\left\{\frac{\lambda_1 - \rho_1\sigma_1\alpha_1 iX + d_1}{\alpha_1^2}\left(\frac{g_1 - \tilde{g}_1 e^{-d_1\tau_1}}{1-\tilde{g}_1 e^{-d_1\tau_1}}\right)\right\}, \\
&\overset{\text{loa}}{\approx} \Re\left\{\frac{\lambda_1 - \rho_1\sigma_1\alpha_1 iX + d_1}{\alpha_1^2}\left(\frac{g_1 - (\eta_1+\chi_1 i)e^{-d_1\tau_1}}{1-(\eta_1+\chi_1 i)e^{-d_1\tau_1}}\right)\right\}, \\
&\overset{\text{loa}}{\approx} \Re\left\{\frac{\lambda_1 - \rho_1\sigma_1\alpha_1 iX + d_1}{\alpha_1^2}g_1\right\}, \\
&= \Re\left\{\frac{\lambda_1 - \rho_1\sigma_1\alpha_1 iX + d_1}{\alpha_1^2}\frac{\lambda_1 - \rho_1\sigma_1\alpha_1 iX - d_1}{\lambda_1 - \rho_1\sigma_1\alpha_1 iX + d_1}\right\}, \\
&\overset{\text{loa}}{\approx} \Re\left\{\frac{\lambda_1 - \rho_1\sigma_1\alpha_1 iX - \sigma_1\alpha_1\sqrt{1-\rho_1^2}X}{\alpha_1^2}\right\}, \\
&\overset{\text{loa}}{\approx} -\frac{\sigma_1}{\alpha_1}\sqrt{1-\rho_1^2}X. \qquad \text{(A.31)}
\end{aligned}$$

Combining (A.25), (A.30) and (A.31) one can get the leading order approximation of the left hand side of (A.21) and then solve the inequality for X and after that for N:

$$\begin{aligned}
&\exp\left\{\Re\left\{C_{0v}(X,0)\right\} + \Re\left\{D_{0v,2}(X,0)\right\}v_0\right\} \leq \frac{(b-a)\varepsilon}{2}, \\
\overset{\text{loa}}{\Leftrightarrow}\ &\exp\left\{-\frac{\lambda_2\tau_2\sigma_2}{\alpha_2}\sqrt{1-\rho_2^2}X - \frac{\lambda_1\tau_1\sigma_1}{\alpha_1}\sqrt{1-\rho_1^2}X - \frac{\sigma_1}{\alpha_1}\sqrt{1-\rho_1^2}Xv_0\right\} \leq \frac{(b-a)\varepsilon}{2}, \\
\Leftrightarrow\ &\exp\left\{X\left[-\sum_{i=1}^{2}\frac{\lambda_i\tau_i\sigma_i}{\alpha_i}\sqrt{1-\rho_i^2} - \frac{\sigma_1}{\alpha_1}\sqrt{1-\rho_1^2}v_0\right]\right\} \leq \frac{(b-a)\varepsilon}{2}, \\
\Leftrightarrow\ &X \geq \log\left(\frac{(b-a)\varepsilon}{2}\right)\left[-\sum_{i=1}^{2}\frac{\lambda_i\tau_i\sigma_i}{\alpha_i}\sqrt{1-\rho_i^2} - \frac{\sigma_1}{\alpha_1}\sqrt{1-\rho_1^2}v_0\right]^{-1}, \\
\Leftrightarrow\ &N \geq \frac{b-a}{\pi}\log\left(\frac{(b-a)\varepsilon}{2}\right)\left[-\sum_{i=1}^{2}\frac{\lambda_i\tau_i\sigma_i}{\alpha_i}\sqrt{1-\rho_i^2} - \frac{\sigma_1}{\alpha_1}\sqrt{1-\rho_1^2}v_0\right]^{-1}. \qquad \text{(A.32)}
\end{aligned}$$

Now consider the general n-term case, s.t.

$$[0,T] = \underbrace{[t_0,t_1]}_{\tau_1} \cup \underbrace{[t_1,t_2]}_{\tau_2} \cup \ldots \cup \underbrace{[t_{n-1},t_n]}_{\tau_n}.$$

In a similar fashion as before, the following equation is of interest:

$$\exp\left\{\Re\left\{C_{0n}(X,0)\right\} + \Re\left\{D_{0n,2}(X,0)\right\} v_0\right\} \quad \leq \quad \frac{(b-a)\varepsilon}{2}. \tag{A.33}$$

Rewrite the recursive definition of $D_{0n,2}(X,V)$ as follows:

$$\begin{aligned}
D_{0n,2}(X,V) &= D_{0(n-1),2}\left(X, \frac{1}{i} D_{n,2}(X,V)\right), \\
&= D_{0(n-2),2}\left(X, \frac{1}{i} D_{n-1,2}\left[X, \frac{1}{i} D_{n,2}(X,V)\right]\right), \\
&= D_{0(n-3),2}\left(X, \frac{1}{i} D_{n-2,2}\left[X, \frac{1}{i} D_{n-1,2}\left(X, \frac{1}{i} D_{n,2}(X,V)\right)\right]\right), \\
&= \ldots = D_{1,2}\left(X, \frac{1}{i} D_{2,2}\left[X, \frac{1}{i} D_{3,2}\left(X, \ldots \frac{1}{i} D_{n,2}(X,V)\right)\right]\right).
\end{aligned}$$

From (A.33) it is clear that this expression is evaluated at $V = 0$, thus:

$$D_{0n,2}(X,0) = D_{1,2}\left(X, \frac{1}{i} D_{2,2}\left[X, \frac{1}{i} D_{3,2}\left(X, \ldots \frac{1}{i} D_{n,2}(X,0)\right)\right]\right). \tag{A.34}$$

Similar to (A.27) and (A.28) the following leading order approximation is obtained:

$$D_{n,2}(X,0) \overset{\text{loa}}{\approx} -\frac{\sigma_n}{\alpha_n}\sqrt{1-\rho_n^2} X.$$

This implies that:

$$\Rightarrow D_{n-1,2}\left(X, \frac{1}{i} D_{n,2}(X,0)\right) \overset{\text{loa}}{\approx} D_{n-1,2}\left(X, -\frac{1}{i}\frac{\sigma_n}{\alpha_n}\sqrt{1-\rho_n^2} X\right).$$

Similar to (A.29):

$$\tilde{g}_{n-1}\left(X, -\frac{1}{i}\frac{\sigma_n}{\alpha_n}\sqrt{1-\rho_n^2} X\right) \overset{\text{loa}}{\approx} \eta_{n-1} + \chi_{n-1} i,$$

where $\eta_{n-1},\ \chi_{n-1} \in \mathbb{R}$ are again constant.
In similar fashion as before the following results are obtained:

$$\begin{aligned}
D_{n-1,2}\left(X, \frac{1}{i} D_{n,2}(X,0)\right) &\overset{\text{loa}}{\approx} D_{n-1,2}\left(X, -\frac{1}{i}\frac{\sigma_n}{\alpha_n}\sqrt{1-\rho_n^2} X\right), \\
&\overset{\text{loa}}{\approx} -\frac{\sigma_{n-1}}{\alpha_{n-1}}\sqrt{1-\rho_{n-1}^2} X, \\
D_{1,2}\left(X, \frac{1}{i} D_{2,2}(X,0)\right) &\overset{\text{loa}}{\approx} D_{2,2}\left(X, -\frac{1}{i}\frac{\sigma_2}{\alpha_2}\sqrt{1-\rho_2^2} X\right), \\
&\overset{\text{loa}}{\approx} -\frac{\sigma_1}{\alpha_1}\sqrt{1-\rho_1^2} X.
\end{aligned}$$

Going back to (A.34):

$$\begin{aligned} D_{0n,2}(X,0) &= D_{1,2}\left(X, \frac{1}{i}D_{2,2}\left[X, \frac{1}{i}D_{3,2}\left(X, \dots \frac{1}{i}D_{n,2}(X,0)\right)\right]\right), \\ &\overset{\text{loa}}{\approx} \dots \overset{\text{loa}}{\approx} \\ &\overset{\text{loa}}{\approx} D_{1,2}\left(X, \frac{1}{i}D_{2,2}(X,0)\right), \\ &\overset{\text{loa}}{\approx} -\frac{\sigma_1}{\alpha_1}\sqrt{1-\rho_1^2}X. \end{aligned} \tag{A.35}$$

For $C_{0n}(X,V)$ also use its recursive definition to rewrite as below where $V=0$ has already been substituted:

$$\begin{aligned} C_{0n}(X,0) &= C_n(X,0) + C_{0(n-1)}\left(X, \frac{1}{i}D_{n,2}(X,0)\right), \\ &= C_n(X,0) + C_{n-1}\left(X, \frac{1}{i}D_{n,2}(X,0)\right) + C_{0(n-2)}\left(X, \frac{1}{i}D_{n-1,2}\left[X, \frac{1}{i}D_{n,2}(X,0)\right]\right), \\ &= C_n(X,0) + C_{n-1}\left(X, \frac{1}{i}D_{n,2}(X,0)\right) + C_{n-2}\left(X, \frac{1}{i}D_{n-1,2}\left[X, \frac{1}{i}D_{n,2}(X,0)\right]\right) \\ &\quad + C_{0(n-3)}\left(X, \frac{1}{i}D_{n-2,2}\left[X, \frac{1}{i}D_{n-1,2}\left(X, \frac{1}{i}D_{n,2}(X,0)\right)\right]\right), \\ &= \dots = \\ &= C_n(X,0) + C_{n-1}\left(X, \frac{1}{i}D_{n,2}(X,0)\right) + \dots + C_1\left(X, \frac{1}{i}D_{2,2}\left(X, \dots \frac{1}{i}D_n(X,0)\right)\right). \end{aligned}$$

Earlier on one might have noticed that the leading order approximation of $C_i(X,V)$ did not depend on the value of V. This is due to the fact that $\tilde{g}_i(X,V)$ is the only expression in $C_i(X,V)$ that depends on V. Consider the following general case where argument V will be written as a function of X, i.e. $\tilde{g}_i(X,V(X))$. Because typically the function $V(X)$ will be determined by $D_{j,2}(X,0)$ (note here the evaluation at $V=0$ which always takes place in order to get the marginal chf) which is always of order X as seen before in (A.35), where the value of n is arbitrary. Using similar techniques as in (A.29) one obtains:

$$\begin{aligned} \tilde{g}_i(X,V(X)) &= \frac{\lambda_i - \rho_i\sigma_i\alpha_i iX - d_i - iV(X)\alpha_i^2}{\lambda_i - \rho_i\sigma_i\alpha_i iX + d_i - iV(X)\alpha_i^2}, \\ &\overset{\text{loa}}{\approx} \frac{\lambda_i - \rho_i\sigma_i\alpha_i iX - d_i - i\mathcal{O}(X)\alpha_i^2}{\lambda_i - \rho_i\sigma_i\alpha_{uv} iX + d_i - i\mathcal{O}(X)\alpha_i^2}, \\ &\overset{\text{loa}}{\approx} \dots \overset{\text{loa}}{\approx} \eta_i + \chi_i i. \end{aligned}$$

Since this is constant and independent of X, it is clear that the leading order approximation of $C_i(X,V)$ has no dependence on the second argument. Therefore it is possible to write the following:

$$\Re\left\{C_{0n}(X,0)\right\} \overset{\text{loa}}{\approx} -\sum_{i=1}^{n}\frac{\lambda_i\tau_i\sigma_i}{\alpha_i}\sqrt{1-\rho_i^2}X. \tag{A.36}$$

Using the results of (A.35) and (A.36) it is possible to rewrite (A.33) in the following fashion:

$$\begin{aligned} & \exp\left\{\Re\left\{C_{0n}(X,0)\right\}+\Re\left\{D_{0n,2}(X,0)\right\}v_0\right\} \quad \leq \quad \frac{(b-a)\varepsilon}{2}, \\ \underset{\Leftrightarrow}{\log} \quad & \exp\left\{-\sum_{i=1}^{n}\frac{\lambda_i\tau_i\sigma_i}{\alpha_i}\sqrt{1-\rho_i^2}X+-\frac{\sigma_1}{\alpha_1}\sqrt{1-\rho_1^2}Xv_0\right\} \quad \leq \quad \frac{(b-a)\varepsilon}{2}, \\ \Leftrightarrow \quad & N \quad \geq \quad \frac{b-a}{\pi}\log\left(\frac{(b-a)\varepsilon}{2}\right)\left[-\sum_{i=1}^{n}\frac{\lambda_i\tau_i\sigma_i}{\alpha_i}\sqrt{1-\rho_i^2}-\frac{\sigma_1}{\alpha_1}\sqrt{1-\rho_1^2}v_0\right]^{-1}. \end{aligned} \tag{A.37}$$

This is the general n-period formula for the amount of terms N in the Fourier cosine expansion for the Heston model with term structure.

A.4.3 Asymptotic results

Now that a lower bound has been given for the value of N used in the COS method, it is crucial to know what the formula looks like asymptotically. The goal of this is to see what will happen with our formulae (A.20) and (A.37) when the Heston parameters approach certain values or bounds. The cases that will be considered are:

1. $\sigma \to 0$,
2. $\alpha \to 0$,
3. $\tau \to \infty$,
4. $|\rho| \to 1$.

For now only consider the case of a single-term Heston model. At the end of this section similar results for the multi-term Heston model will be given.

The first case, where $\sigma \to 0$, implies that the asset dynamics of the Heston model reduce to the following:

$$dS_t \quad = \quad \mu S_t dt.$$

It is obvious that all randomness is gone from the dynamics and all that is left is a deterministic process with only drift and no diffusion, in other words all dependence on the volatility process has been eliminated. Since the Heston model, a stochastic volatility model, has been chosen for a reason, this is an unrealistic situation to consider because when $\sigma \to 0$ the volatility component completely disappears.
In case a situation might arise where this behaviour of σ becoming small is still required, note that from the functional form of (A.37) it is obvious that $N \to \infty$ as $\sigma \to 0$. Some simple numeric testing with different sets of realistic parameter combinations have shown that if $\sigma > 0.01$ then no drastic increase can be found for N. So the use of (A.37) with $\sigma > 0.01$ will be safe in most of the situations. However, one must always double check this and ask the question whether such small values of σ are realistic in the context one is working in and if the Heston model is still an appropriate choice.

In the second case, where $\alpha \to 0$, the dynamics of the model will reduce to:

$$\begin{cases} dS_t &= \mu S_t dt + \sigma\sqrt{v_t}S_t dW_t^1, \\ dv_t &= \lambda(1-v_t)dt. \end{cases}$$

It is clear that the diffusion term has disappeared from the variance process, so only the drift term remains. As a consequence, the model is not a stochastic volatility model any more, but it has reduces to the Black-Scholes model with a special form of time dependent volatility, namely a volatility with drift. Another consequence is that the correlation between the Brownian motions is gone as well, since there is only one Brownian motion left. Thus, once again this asymptotic situation is not worth considering since the Heston model reduces to another model.
Once more, if for some reason small values of α are expected, note that $N \to 0$ as $\alpha \to 0$. This is clear when looking at (A.37), and confirmed by some numerical tests.

The third case, in which $\tau \to \infty$, the dynamics of the process are unchanged. This means that it is useful to look at the consequences for the chf and its leading order approximation and finally at the formula for N. Going back to (A.6)-(A.11) and the results that follow, it is clear that the leading order approximations of the functions $d(X),\ g(X)$ and $\tilde{g}(X)$ do not change as a result of letting $\tau \to \infty$. A consequence of this, combined with the fact that $\tau \to \infty$ is that $e^{-d(X)\tau}$ will approach zero at an even faster rate in the case when N (and therefore X) is large. Because of this the results for $\Re\{D_2(X)\}$ are also unchanged. The same holds for $\Re\{C(X)\}$, but note the dependence here on τ:

$$\Re\{C(X)\} \overset{\text{loa}}{\approx} -\frac{\lambda\tau\sigma}{\alpha}\sqrt{1-\rho^2}X.$$

This implies that if τ will grow larger and larger, so will $\Re\{C(X)\}$ in the negative sense (due to the minus sign). Since all the results still hold, the formula for N will still hold as well:

$$N \geq \frac{b-a}{\pi} \cdot \log\left(\frac{(b-a)\varepsilon}{2}\right) \cdot \left[-\frac{\sigma\sqrt{1-\rho^2}}{\alpha}(\lambda\tau + v_0)\right]^{-1}.$$

$$\begin{aligned} \tau \to \infty \quad &\Rightarrow \quad \lambda\tau \to \infty, \\ &\Rightarrow \quad -\frac{\sigma\sqrt{1-\rho^2}}{\alpha}(\lambda\tau + v_0) \to -\infty, \\ &\Rightarrow \quad \left[-\frac{\sigma\sqrt{1-\rho^2}}{\alpha}(\lambda\tau + v_0)\right]^{-1} \to 0 \ \text{ from below.} \end{aligned}$$

If one notes that ε is typically a very small positive value, then it is obvious that the logarithmic term in the formula will be negative. Therefore it can be concluded that $N \to 0$ if $\tau \to \infty$. Due to the similarity with the multi-term case this result will also hold in that situation. Numerical tests confirm that this decay is indeed the case and that the decay is moderate.
Note that even though the case of $\tau \to \infty$ is considered quite thoroughly, from a practical point of view it does not make a lot of sense to consider. This is due to the fact that options are not quoted with maturities that approach infinity. On the other hand, there are in fact options with large maturities such as 10 or more years, but those are very illiquid and thus not expected to be seen when one typically uses the Heston model with the COS method.

In the fourth and final case, where $|\rho| \to 1$, the analysis is done for each of the two separate cases $\rho \to 1$ and $\rho \to -1$. In this case the dynamics remain unchanged and the first step is to reconsider all the intermediate results from the analysis of the previous sections. First of all, consider the function $d(X)$ as defined in (A.9) and its leading order approximation in (A.16). As $\rho \to 1$, $M \to \lambda^2$ and $N \to \sigma\alpha X(\sigma\alpha - 2\lambda)$. This means that the leading order approximation in (A.16) is no longer valid as some terms drop out and as $\rho \to 1$ the imaginary part δ starts to

play a bigger and bigger role and cannot be neglected. From some numerical tests with various parameter combinations it is clear that this imaginary part cannot be ignored for $\rho > 0.99$. In the end, after redoing the leading order analysis, the following approximation of $d(X)$ is obtained:

$$\lim_{\rho \to 1} d(X) \overset{\text{loa}}{\approx} \sqrt{\frac{\lambda^2 + \sigma\alpha X(\sigma\alpha - 2\lambda)}{2}} + i \cdot \text{sgn}(\sigma\alpha - 2\lambda)\sqrt{\frac{-\lambda^2 + \sigma\alpha X(\sigma\alpha - 2\lambda)}{2}} \quad \text{(A.38)}$$

which holds only in the limit. From now on the assumption will be made that $\rho \leq 0.99$ and that therefore the leading order approximation from (A.16) is still valid for a sufficiently large value of X. Numerical tests with $\rho \leq 0.99$ for $g(X)$ from (A.17), $\Re\{C(X)\}$ from (A.18) and $\Re\{D_2(X)\}$ from (A.19) all indicate that the leading approximations all hold for $\rho \leq 0.99$.
Now considering the other case in which $\rho \to -1$, the same conclusions can be drawn when $\rho \geq -0.99$, apart from the fact that (A.38) is not applicable any more. The following holds in this situation:

$$\lim_{\rho \to -1} d(X) \overset{\text{loa}}{\approx} \sqrt{\frac{\lambda^2 + \sigma\alpha X(\sigma\alpha + 2\lambda)}{2}} + i \cdot \text{sgn}(\sigma\alpha + 2\lambda)\sqrt{\frac{-\lambda^2 + \sigma\alpha X(\sigma\alpha + 2\lambda)}{2}} \quad \text{(A.39)}$$

All in all, in the situation where $|\rho| = 1$, the value of N will blow up to infinity as a division by 0 will take place due to the factor $1 - \rho^2$ that is present. As long as $|rho| \leq 0.99$ holds, the formula in (A.37) will still give a useful and appropriate value of N.

Appendix B

Monte Carlo simulation using Andersen QE method

While implementing the COS-method, some suspicions arose about the correctness of codes. Therefore it was decided to make an implementation of a Monte Carlo method as a reference value. The method could be of use when pricing exotic options instead of standard European options, because in that case some of the analytic results might not hold any more.

There exist quite a few different Monte Carlo methods for solving the problem at hand, but by far the most simple and popular one uses an Euler discretization. For the ordinary Black-Scholes equation this all works out well, but using this method for the Heston model is not advised since the variance process could go below zero with non-zero probability, which results in the square root of a negative number. This will cause the scheme for stepping forward in time to fail. Some alternatives have been proposed such as the Kahl-Jackel [40] scheme and the Broadie-Kaya [9] scheme, but in this thesis Andersen's Quadratic-Exponential (QE) scheme [2] is used. In his paper, Andersen gets his results for the Heston formulation (3.1) without the drift term. However, this method should be applied to the Heston formulation with term structure (3.19). Therefore, the most important results for the implementation are derived again in this section.

First of all, integrate the SDE of the variance process in (3.19):

$$\begin{aligned} v_{t+\Delta t} &= v_t + \int_t^{t+\Delta t} \lambda_u(1-v_u)\, du + \int_t^{t+\Delta t} \alpha_u \sqrt{v_u}\, dW_u^2, \\ \Leftrightarrow \int_t^{t+\Delta t} \sqrt{v_u}\, dW_u^2 &= \frac{1}{\alpha_t}\left(v_{t+\Delta t} - v_t - \lambda_t \Delta t + \lambda_t \int_t^{t+\Delta t} v_u\, du \right). \end{aligned} \tag{B.1}$$

Doing a Cholesky decomposition similar to the one described in Appendix A.1, where $d\tilde{W}_t^1 = dW_t^2$ and $d\tilde{W}_t^2 = d\tilde{W}_t$ is chosen independent of that. This results in:

$$\begin{bmatrix} \sigma_t\sqrt{v_t} & 0 \\ 0 & \alpha_t\sqrt{v_t} \end{bmatrix} \cdot \begin{bmatrix} dW_t^1 \\ dW_t^2 \end{bmatrix} = \begin{bmatrix} \sigma_t\rho_t\sqrt{v_t}dW_t^2 + \sigma_t\sqrt{1-\rho_t^2}\sqrt{v_t}d\tilde{W}_t \\ \alpha_t\sqrt{v_t}dW_t^2 \end{bmatrix}. \tag{B.2}$$

Using the Cholesky decomposition (B.2):

$$dx_t = (\mu_t - \frac{1}{2}\sigma_t^2 v_t)dt + \sigma_t\rho_t\sqrt{v_t}dW_t^2 + \sigma_t\sqrt{1-\rho_t^2}\sqrt{v_t}d\tilde{W}_t. \tag{B.3}$$

Now rewrite (B.3) in integral form where (B.1) is subsituted:

$$\begin{aligned}
x_{t+\Delta t} &= x_t + \int_t^{t+\Delta t} (\mu_u - \frac{1}{2}\sigma_u^2 v_u)\, du + \int_t^{t+\Delta t} \sigma_u \rho_u \sqrt{v_u}\, dW_u^2 \\
&\quad + \int_t^{t+\Delta t} \sigma_u \sqrt{1-\rho_u^2}\sqrt{v_u}\, d\tilde{W}_u, \\
&= x_t + \mu_t \Delta t - \frac{1}{2}\sigma_t^2 \int_t^{t+\Delta t} v_u\, du + \frac{\sigma_t \rho_t}{\alpha_t}\left(v_{t+\Delta t} - v_t - \lambda_t \Delta t + \lambda_t \int_t^{t+\Delta t} v_u\, du\right) \\
&\quad + \sigma_t \sqrt{1-\rho_t^2} \int_t^{t+\Delta t} \sqrt{v_u}\, d\tilde{W}_u, \\
\Leftrightarrow x_{t+\Delta t} &= x_t + \mu_t \Delta t + \frac{\sigma_t \rho_t}{\alpha_t}\left(v_{t+\Delta t} - v_t - \lambda_t \Delta t\right) + \left(\frac{\lambda_t \sigma_t \rho_t}{\alpha_t} - \frac{1}{2}\sigma_t^2\right) \int_t^{t+\Delta t} v_u\, du \qquad \text{(B.4)} \\
&\quad + \sigma_t \sqrt{1-\rho_t^2} \int_t^{t+\Delta t} \sqrt{v_u}\, d\tilde{W}_u.
\end{aligned}$$

Approximate the first integral of (B.4) as follows:

$$\int_t^{t+\Delta t} v_u\, du = \Delta t \left[\gamma_t v_t + \gamma_2 v_{t+\Delta t}\right],$$

where $\gamma_1 = \gamma_2 = 0$ for the Euler discretization, $\gamma_1 = \gamma_2 = \frac{1}{2}$ for the central discritezation. Here the central discretization is used.
Since $\tilde{W}_t$ is independent of v_t, conditional on v_t and $\int_t^{t+\Delta t} v_u\, du$, the second integral of (B.4), the Ito integral, is Gaussian with mean zero and variance $\int_t^{t+\Delta t} v_u\, du$.
Using the above, equation (B.4) is rewritten into the following discretization scheme $\hat{x}_{t+\Delta t}$ which is an approximation to $x_{t+\Delta t}$.

$$\begin{aligned}
\hat{x}_{t+\Delta t} &= \hat{x}_t + \mu_t \Delta t + \frac{\sigma_t \rho_t}{\alpha_t}\left(\hat{v}_{t+\Delta t} - \hat{v}_t - \lambda_t \Delta t\right) + \Delta t \left(\frac{\lambda_t \sigma_t \rho_t}{\alpha_t} - \frac{1}{2}\sigma_t^2\right)\left[\gamma_t \hat{v}_t + \gamma_2 \hat{v}_{t+\Delta t}\right] \\
&\quad + \sigma_t \sqrt{\Delta t}\sqrt{1-\rho_t^2}\sqrt{\gamma_1 \hat{v}_t + \gamma_2 \hat{v}_{t+\Delta t}} \cdot Z, \\
\Leftrightarrow \hat{x}_{t+\Delta t} &= \hat{x}_t + K_0 + K_1 \hat{v}_t + K_2 \hat{v}_{t+\Delta t} + \sqrt{K_3 \hat{v}_t + K_4 \hat{v}_{t+\Delta t}} \cdot Z, \qquad \text{(B.5)} \\
K_0 &= \left(\mu_t - \frac{\lambda_t \sigma_t \rho_t}{\alpha_t}\right)\Delta t, \\
K_1 &= \gamma_1 \Delta t \left(\frac{\lambda_t \sigma_t \rho_t}{\alpha_t} - \frac{1}{2}\sigma_t^2\right) - \frac{\sigma_t \rho_t}{\alpha_t}, \\
K_2 &= \gamma_2 \Delta t \left(\frac{\lambda_t \sigma_t \rho_t}{\alpha_t} - \frac{1}{2}\sigma_t^2\right) + \frac{\sigma_t \rho_t}{\alpha_t}, \\
K_3 &= \gamma_1 \sigma_t^2 (1-\rho_t^2)\Delta t, \\
K_4 &= \gamma_2 \sigma_t^2 (1-\rho_t^2)\Delta t.
\end{aligned}$$

Here Z is a standard Gaussian random variable independent of $\hat{v}_t$.

Note that $\hat{x}_{t+\Delta t}$ depends on $\hat{x}_t$ as well as $\hat{v}_{t+\Delta t}$ and $\hat{v}_t$. Therefore it is time to look into the process of dv_t as defined in the second SDE of (3.4). First of all, let $\Delta t > 0$ and note that conditional on v_t, $v_{t+\Delta t}$ has the following first two moments:

$$\mathbb{E}[v_{t+\Delta t}|v_t] = 1 + (v_t - 1)e^{-\lambda_t \Delta t}, \qquad \text{(B.6)}$$

$$Var(v_{t+\Delta t}|v_t) = \frac{v_t \alpha_t^2 e^{-\lambda_t \Delta t}}{\lambda_t}\left(1 - e^{-\lambda_t \Delta t}\right) + \frac{\alpha_t^2}{2\lambda_t}\left(1 - e^{-\lambda_t \Delta t}\right)^2. \qquad \text{(B.7)}$$

The discretization of the variance process is based on sampling from a moment-matched Gaussian density. Thus the goal is to match both $\mathbb{E}[\hat{v}_{t+\Delta t}]$ and $Var(\hat{v}_{t+\Delta t})$ to the exact values $m = \mathbb{E}[v_{t+\Delta t}|v_t = \hat{v}_t]$ and $s^2 = Var(v_{t+\Delta t}|v_t = \hat{v}_t)$ which can be computed using equations (B.6), (B.7). Using (B.6) and (B.7) write:

$$\psi \quad = \quad \frac{s^2}{m^2} \quad > \quad 0. \tag{B.8}$$

Choose $\psi_c = 1.5$ as indicated in the paper. Note that by definition $\psi_c \in [1, 2]$, but according to Andersen the choice of ψ_c has a relatively small effect on the performance of the scheme. Therefore a choice of $\psi_c = 1.5$ is proposed for numerical tests. Now the Andersen QE algorithm is as follows:

1. Given $\hat{v}_t$, calculate (B.6) and (B.7).
2. Compute ψ from (B.8).
3. Draw a uniform random number U_v.
4. If $\psi \leq \psi_c$:
   (a) Compute a and b as follows:
   $$\begin{aligned} b &= \sqrt{\frac{2}{\psi} - 1 + \sqrt{\frac{2}{\psi}\left(\frac{2}{\psi} - 1\right)}}, \\ a &= \frac{m}{1 + b^2}. \end{aligned}$$
   (b) Compute $Z_v = N^{-1}(U_v)$.
   (c) Set $\hat{v}_{t+\Delta t} = a(b + Z_v)^2$.
5. Otherwise, if $\psi > \psi_c$:
   (a) Compute β and p as follows:
   $$\begin{aligned} p &= \frac{\psi - 1}{\psi + 1}, \\ \beta &= \frac{1 - p}{m}. \end{aligned}$$
   (b) Set $\hat{v}_{t+\Delta t} = \Psi^{-1}(U_v; p, \beta)$, where $\Psi^{-1}(u; p, \beta)$ is given as:
   $$\Psi^{-1}(u; p, \beta) \quad = \quad \begin{cases} 0, & 0 \leq u \leq p, \\ \frac{1}{\beta} \log\left(\frac{1-p}{1-u}\right), & p < u \leq 1. \end{cases}$$
6. Draw a standard normal random number Z independent of all random numbers used for $\hat{v}_{t+\Delta t}$.
7. Given $\hat{x}_t$, $\hat{v}_t$ and $\hat{v}_{t+\Delta t}$, compute $\hat{x}_{t+\Delta t}$ from equation (B.5).

This concludes a description of the method and option contracts can now be priced using the algorithm described above.

Furthermore, a correction is proposed by Andersen [2] to modify the Andersen QE scheme such that the following discrete-time martingale condition holds:

$$\mathbb{E}\left[e^{-\mu\Delta t}\hat{S}_{t+\Delta t}|\hat{S}_t,\hat{v}_t\right] = \hat{S}_t \tag{B.9}$$

The reason this is done is that this property holds for the process for S_t in continuous time, so naturally this should hold for a good discretization. Furthermore, martingality is sometimes required for no-arbitrage purposes. Therefore, the proposed martingale correction from the paper will be derived for the new formulation of the Heston model (3.19).
For the current scheme in (B.5) the condition (B.9) does not hold. Using the relationship $\hat{S}_t = e^{\hat{x}_t}$ it is possible to use (B.5) and some properties conditional expectations to calculate the conditional expectation in (B.9) and derive a value K_0^* that ensures the required martingality:

$$\begin{aligned}
\mathbb{E}\left[e^{-\mu\Delta t}\hat{S}_{t+\Delta t}|\hat{S}_t,\hat{v}_t\right] &= \mathbb{E}\left[\mathbb{E}\left[e^{-\mu\Delta t}\hat{S}_{t+\Delta t}|\hat{S}_t,\hat{v}_t,\hat{v}_{t+\Delta t}\right]|\hat{S}_t,\hat{v}_t\right] \\
&= \mathbb{E}\left[\mathbb{E}\left[e^{-\mu\Delta t}\hat{S}_t e^{K_0^*+K_1\hat{v}_t+K_2\hat{v}_{t+\Delta t}+\sqrt{K_3\hat{v}_t+K_4\hat{v}_{t+\Delta t}}\cdot Z}|\hat{S}_t,\hat{v}_t,\hat{v}_{t+\Delta t}\right]|\hat{S}_t,\hat{v}_t\right] \\
&= \hat{S}_t e^{-\mu\Delta t+K_0^*+K_1\hat{v}_t}\mathbb{E}\left[e^{K_2\hat{v}_{t+\Delta t}}\mathbb{E}\left[e^{\sqrt{K_3\hat{v}_t+K_4\hat{v}_{t+\Delta t}}\cdot Z}|\hat{S}_t,\hat{v}_t,\hat{v}_{t+\Delta t}\right]|\hat{S}_t,\hat{v}_t\right] \\
&= \hat{S}_t e^{-\mu\Delta t+K_0^*+K_1\hat{v}_t}\mathbb{E}\left[e^{K_2\hat{v}_{t+\Delta t}}e^{\frac{1}{2}(K_3\hat{v}_t+K_4\hat{v}_{t+\Delta t})}|\hat{S}_t,\hat{v}_t\right] \\
&= \hat{S}_t e^{-\mu\Delta t+K_0^*+\left(K_1+\frac{1}{2}K_3\right)\hat{v}_t}\mathbb{E}\left[e^{\left(K_2+\frac{1}{2}K_4\right)\hat{v}_{t+\Delta t}}|\hat{S}_t,\hat{v}_t\right] \\
&= \hat{S}_t \\
\Rightarrow 1 &= e^{-\mu\Delta t+K_0^*+\left(K_1+\frac{1}{2}K_3\right)\hat{v}_t}\mathbb{E}\left[e^{\left(K_2+\frac{1}{2}K_4\right)\hat{v}_{t+\Delta t}}|\hat{S}_t,\hat{v}_t\right] \\
\Leftrightarrow K_0^* &= \mu\Delta t-\left(K_1+\frac{1}{2}K_3\right)\hat{v}_t-\log\left(\mathbb{E}\left[e^{\left(K_2+\frac{1}{2}K_4\right)\hat{v}_{t+\Delta t}}|\hat{S}_t,\hat{v}_t\right]\right)
\end{aligned} \tag{B.10}$$

In order for the expression in (B.10) to be meaningful at all, the expectation in the equation must be finite. In the paper, Andersen derives some regularity conditions that ensure the existence of the expectation in a certain closed form. For the Heston model with term structure this comes down to the following expression of K_0^*:

$$\begin{aligned}
A &:= K^2+\frac{1}{2}K_4, \\
K_0^* &= \begin{cases} \mu\Delta t-\left(K_1+\frac{1}{2}K_3\right)\hat{v}_t-\frac{Ab^2a}{1-2Aa}+\frac{1}{2}\log(1-2Aa), & \psi\le\psi_c, \\ \mu\Delta t-\left(K_1+\frac{1}{2}K_3\right)\hat{v}_t-\log\left(p+\frac{\beta(1-p)}{\beta-A}\right), & \psi>\psi_c. \end{cases}
\end{aligned} \tag{B.11}$$

Now the Andersen QE scheme with the enforced martingale condition (B.9) is given by the scheme (B.5) where K_0 is replaced by K_0^* as given in (B.11).

Appendix C

Parameter sets

In this chapter a full description of all the parameter sets used for the testing of the various option pricing methods in Section 5.4 can be found.

C.1 Parameters used for Table 5.9

$$\begin{aligned}
[\alpha_1, \alpha_2, \alpha_3] &= [4.5, 6, 7], \\
[\rho_1, \rho_2, \rho_3] &= [-0.3, -0.25, -0.4], \\
[\lambda_1, \lambda_2, \lambda_3] &= [2.5, 2.5, 2.5], \\
[\sigma_1, \sigma_2, \sigma_3] &= [0.07, 0.09, 0.10], \\
[\tau_1, \tau_2, \tau_3] &= [0.25, 0.5, 1], \\
[t_0, T] &= [0, 1.75], \\
S_0 &= 100, \\
K &= 100, \\
v_0 &= 1, \\
r_f,\ r_d &= 0, \\
\alpha &= 1.
\end{aligned}$$

C.2 Parameters used for Table 5.10

$$\begin{aligned}
[\alpha_1, \alpha_2, \alpha_3] &= [15, 12, 18], \\
[\rho_1, \rho_2, \rho_3] &= [-0.05, 0.1, 0.1], \\
[\lambda_1, \lambda_2, \lambda_3] &= [2.5, 2.5, 2.5], \\
[\sigma_1, \sigma_2, \sigma_3] &= [0.05, 0.06, 0.08], \\
[\tau_1, \tau_2, \tau_3] &= [0.2, 0.5, 0.6], \\
[t_0, T] &= [0, 1.3], \\
S_0 &= 100, \\
K &= 100, \\
v_0 &= 1, \\
r_f,\ r_d &= 0, \\
\alpha &= 1.
\end{aligned}$$

C.3 Parameters used for Table 5.11

$$
\begin{aligned}
[\alpha_1, \alpha_2] &= [2, 1.5]\,, \\
[\rho_1, \rho_2] &= [-0.3, -0.4]\,, \\
[\lambda_1, \lambda_2] &= [2.5, 2.5]\,, \\
[\sigma_1, \sigma_2] &= [0.05, 0.08]\,, \\
[\tau_1, \tau_2] &= [0.5, 1]\,, \\
[t_0, T] &= [0, 1.5]\,, \\
S_0 &= 100, \\
K &= 100, \\
v_0 &= 1, \\
r_f,\ r_d &= 0, \\
\alpha &= 1.
\end{aligned}
$$

C.4 Parameters used for Table 5.12

$$
\begin{aligned}
[\alpha_1, \alpha_2, \alpha_3] &= [15, 12, 13]\,, \\
[\rho_1, \rho_2, \rho_3] &= [-0.3, -0.5, -0.4]\,, \\
[\lambda_1, \lambda_2, \lambda_3] &= [2.5, 2.5, 2.5]\,, \\
[\sigma_1, \sigma_2, \sigma_3] &= [0.7, 0.8, 1.65]\,, \\
[\tau_1, \tau_2, \tau_3] &= [0.5, 1.0, 0.8]\,, \\
[t_0, T] &= [0, 2.3]\,, \\
S_0 &= 100, \\
K &= 100, \\
v_0 &= 1, \\
r_f,\ r_d &= 0, \\
\alpha &= 1.
\end{aligned}
$$

Appendix D

Additional information on numerical optimization

As an introduction to numerical optimization the book by Nocedal and Wright [57] is used. Note that throughout this section all the material is composed of excerpts from various sources and summarized to give a clear overview of all the relevant different optimization techniques.

D.1 First and second order optimality conditions

This section contains a summary of material from Nocedal and Wright [57].
Before the formal definitions of the optimality conditions can be stated, some definitions must be given to have a basic understanding of what will follow afterwards. From this point onwards only minimization problems are considered. Remember that the maximization of a function g is the same as minimizing the function $-g$.

- $\mathbf{x}^*$ is a *local solution* of (6.13) if $\mathbf{x}^* \in \Omega$ and there exists a neighbourhood $\mathcal{N}$ of $\mathbf{x}^*$ such that $f(\mathbf{x}) \geq f(\mathbf{x}^*) \;\; \forall \mathbf{x} \in \mathcal{N} \cap \Omega$.
- $\mathbf{x}^*$ is a *strict local solution* of (6.13) if $\mathbf{x}^* \in \Omega$ and there exists a neighbourhood $\mathcal{N}$ of $\mathbf{x}^*$ such that $f(\mathbf{x}) > f(\mathbf{x}^*) \;\; \forall \mathbf{x} \in \mathcal{N} \cap \Omega \;\; (\mathbf{x} \neq \mathbf{x}^*)$.
- $\mathbf{x}^*$ is a *isolated local solution* of (6.13) if $\mathbf{x}^* \in \Omega$ and there exists a neighbourhood $\mathcal{N}$ of $\mathbf{x}^*$ such that $\mathbf{x}^*$ is the only local solution in $\mathcal{N} \cap \Omega$.
- The *active set* $\mathcal{A}(\mathbf{x})$ at any feasible point $\mathbf{x}$ consists of the equality constraint indices from $\mathcal{E}$ together with the inequality constraint indices i for which holds that $c_i(\mathbf{x}) = 0$, i.e.

$$\mathcal{A}(\mathbf{x}) \;\; = \;\; \mathcal{E} \;\cup\; \{i \in \mathcal{I} \mid c_i(\mathbf{x}) = 0\}.$$

- At a feasible point $\mathbf{x}$, the inequality constraint $i \in \mathcal{I}$ is said to be *active* if $c_i(\mathbf{x}) = 0$ and *inactive* if the strict inequality $c_i(\mathbf{x}) > 0$ is satisfied.
- Let $A(\mathbf{x}^*)$ represent the matrix whose rows are the active constraint gradients at the optimal point, i.e.

$$A(\mathbf{x}^*)^T \;\; = \;\; [\nabla c_i(\mathbf{x}^*)]_{i \in \mathcal{A}(\mathbf{x}^*)}\,.$$

- Given feasible point $\mathbf{x}$ and its active set $\mathcal{A}(\mathbf{x})$, the *linear independence constraint qualification (LICQ)* holds if the set of active constraint gradients $\{\nabla c_i(\mathbf{x}),\; i \in \mathcal{A}(\mathbf{x})\}$ is linearly independent. In general, if LICQ holds, none of the active constraint gradients can be zero.

- The *Lagrangian function* for problem (6.13) is defined as:

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) \quad = \quad f(\mathbf{x}) - \sum_{i \in \mathcal{E} \cup \mathcal{I}} \lambda_i c_i(\mathbf{x})$$

  Here the scalar quantity λ_i is called a *Lagrange multiplier* for the constraint c_i.

Using the definitions above it is possible to state the first-order necessary conditions for $\mathbf{x}^*$ to be a local minimizer. The following conditions are called the first-order necessary conditions because they involve properties of the gradients, which are first-order derivative vectors of the objective and constraints.
Suppose that $\mathbf{x}^*$ is a local solution of (6.13), f and c_i are continuously differentiable and also suppose that the LICQ hold at that point. Then there exists a Lagrange multiplier vector $\boldsymbol{\lambda}^* = \left[\lambda_1^*, \ldots, \lambda_{|\mathcal{E} \cup \mathcal{I}|}^*\right]^T$ with components $\lambda_i^*,\ i \in \mathcal{E} \cup \mathcal{I}$ such that the following conditions are satisfied:

$$\begin{aligned}
\nabla_x \mathcal{L}(\mathbf{x}^*, \boldsymbol{\lambda}^*) &= 0, && \text{(D.1a)} \\
c_i(\mathbf{x}^*) &= 0, \quad \forall i \in \mathcal{E}, && \text{(D.1b)} \\
c_i(\mathbf{x}^*) &\geq 0, \quad \forall i \in \mathcal{I}, && \text{(D.1c)} \\
\lambda_i^* &\geq 0, \quad \forall i \in \mathcal{I}, && \text{(D.1d)} \\
\lambda_i^* c_i^*(\mathbf{x}^*) &= 0, \quad \forall i \in \mathcal{E} \cup \mathcal{I}. && \text{(D.1e)}
\end{aligned}$$

These conditions are known as the *Karush-Kuhn-Tucker (KKT) conditions.*
The conditions described in (D.1e) are called *complementary conditions*, they imply that either constraint i is active or that $\lambda_i^* = 0$, or possibly both. *Strict complementarity* holds if exactly one of λ_i^* and $c_i(\mathbf{x}^*)$ is zero for each index $i \in \mathcal{I}$. In other words, $\lambda_i^* > 0 \ \ \forall i \in \mathcal{I} \cap \mathcal{A}(\mathbf{x}^*)$.
Note that for problem (6.13) and a solution point $\mathbf{x}^*$ there may be multiple Lagrange multiplier vectors satisfying the KKT conditions. However, when the LICQ holds, one knows for certain that the optimal $\boldsymbol{\lambda}^*$ is unique.

Moving on to the second-order necessary conditions, which as one expects involves properties of the second-order derivative vectors (Hessians) of the objective and constraints, some new definitions are required:

- $\mathcal{F}(\mathbf{x})$ is the set of first-order feasible directions at $\mathbf{x}$. A more formal definition would be: given feasible point $\mathbf{x}$ and active constraint set $\mathcal{A}(\mathbf{x})$, the set of *linearised feasible directions* $\mathcal{F}(\mathbf{x})$ is:

$$\mathcal{F}(\mathbf{x}) \quad = \quad \{\mathbf{d} \mid \mathbf{d}^T \nabla c_i(\mathbf{x}) = 0,\ \forall i \in \mathcal{E};\ \mathbf{d}^T c_i(\mathbf{x}) \geq 0,\ \forall i \in \mathcal{A}(\mathbf{x}) \cap \mathcal{I}\}.$$

- Given $\mathcal{F}(\mathbf{x}^*)$ and a Lagrange multiplier vector $\boldsymbol{\lambda}^*$ satisfying the KKT conditions, define the *critical cone* as:

$$C(\mathbf{x}^*, \boldsymbol{\lambda}^*) \quad = \quad \{\mathbf{w} \in \mathcal{F}(\mathbf{x}^*) \mid \nabla c_i(\mathbf{x}^*)^T \mathbf{w} = 0,\ \forall i \in \mathcal{A}(\mathbf{x}^*) \cap \mathcal{I} \text{ with } \lambda_i^* > 0\} \quad \text{(D.2)}$$

  The critical cone contains those directions $\mathbf{w}$ that would tend to adhere to the active inequality constraints even when we were to make small changes to the objective, as well as to the equality constraints. The critical cone contains directions from $\mathcal{F}(\mathbf{x}^*)$ for which it's not clear from the first derivatives alone whether f will increase or decrease.

Now all definitions needed for the second-order necessary conditions are available. Suppose that $\mathbf{x}^*$ is a local solution of (6.13) and that the LICQ hold at that point. Let $\boldsymbol{\lambda}^*$ be the Lagrange multiplier vector for which the KKT conditions are satisfied. Then:

$$\mathbf{w}^T \nabla^2_{xx} \mathcal{L}(\mathbf{x}^*, \boldsymbol{\lambda}^*) \mathbf{w} \quad \geq \quad 0, \qquad \forall \mathbf{w} \in C(\mathbf{x}^*, \boldsymbol{\lambda}^*). \tag{D.3}$$

In other words, if $\mathbf{x}^*$ is a local solution, then the Hessian of the Lagrangian has non-negative curvature along the critical directions, i.e. the directions in $C(\mathbf{x}^*, \boldsymbol{\lambda}^*)$.

Of course one should be interested in the second-order sufficient conditions as well. Suppose for some feasible point $\mathbf{x}^* \in \mathbb{R}^n$ there exists a Lagrange multiplier vector $\boldsymbol{\lambda}^*$ such that the KKT conditions are satisfied. Suppose also that

$$\mathbf{w}^T \nabla^2_{xx} \mathcal{L}(\mathbf{x}^*, \boldsymbol{\lambda}^*) \mathbf{w} \quad > \quad 0, \qquad \forall \mathbf{w} \in C(\mathbf{x}^*, \boldsymbol{\lambda}^*), \ \mathbf{w} \neq \mathbf{0}. \tag{D.4}$$

Then $\mathbf{x}^*$ is a strict local solution of (6.13).

Essentially, the second order conditions concern the curvature of the Lagrangian function in the 'undecided' directions (the directions $w \in \mathcal{F}(\mathbf{x}^*)$ for which $\mathbf{w}^T \nabla f(\mathbf{x}^*) = 0$).
The second order conditions are sometimes in a form that is slightly weaker but easier to verify than (D.3) and (D.4). This form uses a two-sided projection of $\nabla^2_{xx} \mathcal{L}(\mathbf{x}^*, \boldsymbol{\lambda}^*)$ onto subspaces related to $C(\mathbf{x}^*, \boldsymbol{\lambda}^*)$. The simplest case is when $\boldsymbol{\lambda}^*$ satisfies the KKT conditions, is unique and strict complementarity holds. In this case, the definition (D.2) of $C(\mathbf{x}^*, \boldsymbol{\lambda}^*)$ reduces to:

$$\begin{aligned} C(\mathbf{x}^*, \boldsymbol{\lambda}^*) &= \text{Null} \left\{ \left[\nabla c_i(\mathbf{x}^*)^T \right]_{i \in \mathcal{A}(\mathbf{x}^*)} \right\}, \\ &= \text{Null} \left\{ A(\mathbf{x}^*) \right\}. \end{aligned}$$

In other words, the critical cone is the null-space of the matrix whose rows are the active constraint gradients at $\mathbf{x}^*$. Now define Z to be the matrix with full column rank whose columns span the space $C(\mathbf{x}^*, \boldsymbol{\lambda}^*)$, i.e.

$$C(\mathbf{x}^*, \boldsymbol{\lambda}^*) \quad = \quad \left\{ Z\mathbf{u} \ \middle| \ \mathbf{u} \in \mathbb{R}^{|\mathcal{A}(\mathbf{x}^*)|} \right\}.$$

Hence, reformulate (D.3) as

$$\mathbf{u}^T Z^T \nabla^2_{xx} \mathcal{L}(\mathbf{x}^*, \boldsymbol{\lambda}^*) Z\mathbf{u} \quad \geq \quad 0, \qquad \forall \mathbf{u},$$

or $Z^T \nabla^2_{xx} \mathcal{L}(\mathbf{x}^*, \boldsymbol{\lambda}^*) Z$ is positive semi-definite.
Similarly, reformulate (D.4) as

$$\mathbf{u}^T Z^T \nabla^2_{xx} \mathcal{L}(\mathbf{x}^*, \boldsymbol{\lambda}^*) Z\mathbf{u} \quad > \quad 0, \qquad \forall \mathbf{u},$$

or $Z^T \nabla^2_{xx} \mathcal{L}(\mathbf{x}^*, \boldsymbol{\lambda}^*) Z$ is positive definite.
The matrix Z can be computed numerically such that these conditions can be checked by forming the matrices and finding the corresponding eigenvalues (positive eigenvalues are needed for positive definiteness).

D.2 Levenberg-Marquardt method

This section contains a summary of material from Nocedal and Wright [57], and the articles by Levenberg and Marquardt [42, 46].

The Levenberg-Marquardt method is very much alike the Gauss-Newton method. In the Levenberg-Marquardt method the same Hessian approximation is used, but instead of using a line search this method uses a trust region. This different approach guarantees that one of the pitfalls of the Gauss-Newton method is circumvented, namely the behaviour of the method when the Jacobian is (nearly) rank-deficient. Due to the same use of Hessian approximation, the local convergence properties of both methods are similar.

For a trust region that's a sphere, the sub-problem one has to solve at every iteration is:

$$\min_{\mathbf{p}} \frac{1}{2}||J_k\mathbf{p} + \mathbf{r}_k||_2^2, \qquad \text{subject to } ||\mathbf{p}||_2 \leq \Delta_k. \tag{D.5}$$

Here, $\Delta_k > 0$ denotes the trust region radius. Using some results on trust regions, one can say the following about the solution of (D.5): when the solution $\mathbf{p}_k^{GN}$ of the Gauss-Newton system of equations (6.20) lies strictly within the trust region (i.e. $||\mathbf{p}_k^{GN}||_2 < \Delta_k$), then this step solves subproblem (D.5) as well. If not, there exists a $\lambda_k > 0$ so that the solution $\mathbf{p}_k^{LM}$ of subproblem (D.5) satisfies $||\mathbf{p}_k^{LM}||_2 = \Delta_k$ and

$$(J_k^T J_k + \lambda_k I)\mathbf{p}_k^{LM} \quad = \quad -J_k^T \mathbf{r}_k.$$

The goal now is to find a value of λ_k that approximately matches the given Δ_k. This can be done using a root-finding algorithm.
In some situations a least-squares problem can be poorly scaled, meaning that the variable numbers could lie within totally different ranges, possibly providing difficulties in the numerics, or giving bad solutions. A way to solve this matter is to not use a spherical trust region but an ellipsoidal trust region. This approach involves a diagonal scaling matrix D_k with positive diagonal entries, such that (D.5) is now subject to $||D_k\mathbf{p}||_2 \leq \Delta_k$.

D.3 Sequential Quadratic Programming methods

This section contains a summary of material from Nocedal and Wright [57].
The Sequential Quadratic Programming (SQP) approach can be used in both line search and trust region frameworks and is appropriate for any problem size. SQP methods are especially effective when the minimization problem includes strong non-linearities in the constraints. There are two types of active-set SQP methods, namely the IQP and the EQP approach. In the first approach, an inequality-constrained program is solved at each iteration. While solving this program, a step must be computed and an estimate of the optimal active set must be generated. In the EQP approach the computations of step and optimal active step are decoupled. The first task within this method is to estimate the optimal active set, after which the second task of solving an equality-constrained quadratic program for finding the step is performed.

Consider an equality-constrained version of problem (6.13) (i.e. $\mathcal{I} = \emptyset$):

$$\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x}) \qquad \text{subject to } \mathbf{c}(\mathbf{x}) = \mathbf{0}, \tag{D.6}$$

with smooth functions $f : \mathbb{R}^n \to \mathbb{R}$ and $\mathbf{c} : \mathbb{R}^n \to \mathbb{R}^m$. The general idea of a SQP method is to model (D.6) at the current iterate $\mathbf{x}_k$ by a quadratic programming sub-problem, then use the minimizer of this sub-problem to define a new iterate $\mathbf{x}_{k+1}$. This means that a quadratic sub-problem needs to be designed such that it produces good new iterates.

One could say that SQP methods are an application of Newton's method to the KKT conditions for the (D.6), which are similar conditions to those stated in (D.1a)-(D.1e). The Lagrangian function of this problem is

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) \quad = \quad f(\mathbf{x}) - \boldsymbol{\lambda}^T \mathbf{c}(\mathbf{x}).$$

Denote the Jacobian matrix of the constraints as $A(\mathbf{x}) = [\nabla c_1(\mathbf{x}), \ldots, \nabla c_m(\mathbf{x})]^T$, where $c_i(\mathbf{x})$ is the i-th component of equality-constraint vector $\mathbf{c}(\mathbf{x})$. The KKT conditions for the equality-constrained problem (D.6) can be written as a system of $n+m$ equations in the $n+m$ unknowns $\mathbf{x}$ and $\boldsymbol{\lambda}$:

$$\mathbf{F}(\mathbf{x}, \boldsymbol{\lambda}) \quad = \quad \begin{bmatrix} \nabla f(\mathbf{x}) - A(\mathbf{x})^T \boldsymbol{\lambda} \\ \mathbf{c}(\mathbf{x}) \end{bmatrix} \quad = \quad \mathbf{0}. \tag{D.7}$$

Any optimal solution $(\mathbf{x}^*, \boldsymbol{\lambda}^*)$ solving problem (D.6) for which $A(\mathbf{x}^*)$ has full rank, satisfies the KKT conditions (D.7). The above mentioned approach is to solve system (D.7) using Newton's method. For this, first consider the Jacobian of $\mathbf{F}(\mathbf{x}, \boldsymbol{\lambda})$ w.r.t. $\mathbf{x}$ and $\boldsymbol{\lambda}$:

$$\mathbf{F}'(x, \lambda) \quad = \quad \begin{bmatrix} \nabla^2_{xx} \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) & -A(\mathbf{x})^T \\ A(\mathbf{x}) & 0 \end{bmatrix}.$$

Now, the Newton step is given by

$$\begin{bmatrix} \mathbf{x}_{k+1} \\ \boldsymbol{\lambda}_{k+1} \end{bmatrix} \quad = \quad \begin{bmatrix} \mathbf{x}_k \\ \boldsymbol{\lambda}_k \end{bmatrix} + \begin{bmatrix} \mathbf{p}_k \\ \mathbf{p}_\lambda \end{bmatrix} \tag{D.8}$$

where $\mathbf{p}_k$ and $\mathbf{p}_\lambda$ solve the following system:

$$\begin{bmatrix} \nabla^2_{xx} \mathcal{L}_k & -A_k^T \\ A_k & 0 \end{bmatrix} \begin{bmatrix} \mathbf{p}_k \\ \mathbf{p}_\lambda \end{bmatrix} \quad = \quad \begin{bmatrix} -\nabla f_k + A_k^T \boldsymbol{\lambda}_k \\ -\mathbf{c}_k \end{bmatrix}. \tag{D.9}$$

This iteration is well-defined when the r.h.s. matrix in (D.9) is non-singular. This non-singularity holds if the following assumptions hold:

(i). Matrix $A(\mathbf{x})$ has full rank.

(ii). The matrix $\nabla^2_{xx} \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda})$ is positive definite on the tangent space of the constraints, i.e. $\mathbf{d}^T \nabla^2_{xx} \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) \mathbf{d} > 0 \;\; \forall \mathbf{d} \neq \mathbf{0} \;\; : \;\; A(\mathbf{x})\mathbf{d} = \mathbf{0}$.

Assumption (i) is simply the LICQ assumption from Appendix D.1, assumption (ii) holds whenever one is close to the optimal solution and the second-order sufficient condition (D.4) is satisfied at the solution. Under these assumptions, the Newton iteration (D.8) is quadratically convergent.

An alternative way of viewing the Newton iteration from equations (D.8) and (D.9) is to model the problem (D.6) using the following quadratic program:

$$\min_{\mathbf{p}} \qquad f_k + \nabla f_k^T \mathbf{p} + \frac{1}{2} \mathbf{p}^T \nabla^2_{xx} \mathcal{L}_k \mathbf{p}, \tag{D.10a}$$

$$\text{subject to} \qquad A_k \mathbf{p} + \mathbf{c}_k = \mathbf{0}. \tag{D.10b}$$

Under assumptions (i) and (ii), the problem defined in (D.10a) and (D.10b) has a unique solution (p_k, l_k) satisfying

$$\begin{aligned} \nabla^2_{xx} \mathcal{L}_k p_k + \nabla f_k - A_k^T l_k &= 0, \\ A_k p + c_k &= 0. \end{aligned}$$

The solution vectors $\mathbf{p}_k$ and $\mathbf{l}_k$ can be identified with the solution of the Newton equations (D.9) by subtracting $A_k^T \boldsymbol{\lambda}_k$ from both sides of the first equation, which yields the following system:

$$\begin{bmatrix} \nabla^2_{xx}\mathcal{L}_k & -A_k^T \\ A_k & 0 \end{bmatrix} \begin{bmatrix} \mathbf{p}_k \\ \boldsymbol{\lambda}_{k+1} \end{bmatrix} = \begin{bmatrix} -\nabla f_k \\ -\mathbf{c}_k \end{bmatrix}. \tag{D.12}$$

Since the coefficient matrix is non-singular, one has that $\boldsymbol{\lambda}_{k+1} = \mathbf{l}_k$ and that $\mathbf{p}_k$ solves (D.10a)-(D.10b) and (D.9). Now both the solution of the quadratic program (D.10a)-(D.10b) or the solution generated by Newton's method (D.8), (D.9) applied to the optimality conditions of the problem can define the new iterate $(\mathbf{x}_{k+1}, \boldsymbol{\lambda}_{k+1})$. The Newton approach provides the analysis, and on the other hand the SQP approach enables the derivation of algorithms and the extension in the inequality-constrained case.

In short, the SQP method for solving (D.6) can be summarized as follows:

1. Choose an initial guess $(\mathbf{x}_0, \boldsymbol{\lambda}_0)$.
2. Evaluate $f_k,\ \nabla f_k,\ \nabla^2_{xx}\mathcal{L}_k,\ \mathbf{c}_k$, and A_k.
3. Solve (D.10a)-(D.10b) and obtain solution vectors $\mathbf{p}_k,\ \mathbf{l}_k$.
4. Set $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{p}_k$ and $\boldsymbol{\lambda}_{k+1} = \mathbf{l}_k$, or apply line search in the direction.
5. If a convergence test is satisfied, then stop. If not, go back to step (2) and repeat the process.

So far only problems with only equality-constraints have been considered. It is easy to extend the framework to include inequality constraints into the problem. Therefore, consider the general non-linear problem defined in (6.13). Now linearize both the equality and inequality constraints to get the following:

$$\min_{\mathbf{p}} \quad f_k + \nabla f_k^T \mathbf{p} + \frac{1}{2}\mathbf{p}^T \nabla^2_{xx}\mathcal{L}_k \mathbf{p}, \tag{D.13a}$$

$$\text{subject to} \quad \nabla \mathbf{c}_k(\mathbf{x}_k)^T \mathbf{p} + c_i(\mathbf{x}_k) = 0, \quad i \in \mathcal{E}, \tag{D.13b}$$

$$\nabla \mathbf{c}_k(\mathbf{x}_k)^T \mathbf{p} + c_i(\mathbf{x}_k) \geq 0, \quad i \in \mathcal{I}. \tag{D.13c}$$

After solving this quadratic programming problem, the new iterate is given by $(\mathbf{x}_k + \mathbf{p}_k, \boldsymbol{\lambda}_{k+1})$, where $\mathbf{p}_k$ and $\boldsymbol{\lambda}_{k+1}$ are respectively the solution and the corresponding Lagrange multiplier of the quadratic programming problem (D.13a)-(D.13c). This means that a local SQP algorithm for solving problem (6.13) is basically given by the algorithm described above, with the difference that the step is computed from (D.13a)-(D.13c) instead of (D.10a)-(D.10b). This approach is categorized as an IQP approach. Here, the active constraints $\mathcal{A}_k$ at the solution of (D.13a)-(D.13c) provides an educated guess of the active stet at the solution of the non-linear program (6.13). If the SQP method manages to provide an accurate guess of the optimal active set, the convergence will be rapid and the method will act like a Newton method for equality-constrained optimization. The main drawback of this IQP approach is the possible high costs of solving the quadratic program (D.13a)-(D.13c) when the problem is large.

The EQP on the other hand selects a subset of constraints at each iteration to be the current working set and consecutively solves only equality-constrained subproblems of the form (D.10a)-(D.10b) where the constraints in the working sets are imposed as equalities and all other constraints are neglected for the time being. This approach has the advantage that the equality-constrained quadratic subproblems are cheaper to solve than (D.13a)-(D.13c) in case of a large-scale problem.

D.4 IPOPT

This section contains a summary of material from Nocedal and Wright [57], and the article by Wächter and Biegler [65].
Consider a problem formulation that is as follows:

$$\min_{\mathbf{x}\in\mathbb{R}^n} \quad f(\mathbf{x}), \tag{D.14a}$$

$$\text{s.t.} \quad \mathbf{c}(\mathbf{x}) = \mathbf{0}, \tag{D.14b}$$

$$x_i \geq 0, \quad \forall i \in \{1, 2, \ldots, n\}. \tag{D.14c}$$

Note that this problem formulation can be transformed into the more general case of any (non-)linear constraints and box-bounds. As a barrier method, the proposed algorithm approximates solutions for a sequence of barrier problems

$$\min_{\mathbf{x}\in\mathbb{R}^n} \quad \varphi_\mu(\mathbf{x}) := f(\mathbf{x}) - \mu \sum_{i=1}^{n} \ln(x_i) \tag{D.15a}$$

$$\text{s.t.} \quad \mathbf{c}(\mathbf{x}) = \mathbf{0} \tag{D.15b}$$

for a decreasing sequence of barrier parameters μ converging to zero. This can be seen as applying a homotopy method (see chapter 11.3 from Nocedal & Wright [57]) to the primal-dual equations (see chapter 14 from Nocedal & Wright [57]):

$$\nabla f(\mathbf{x}) + \nabla \mathbf{c}(\mathbf{x})\boldsymbol{\lambda} - \mathbf{z} = \mathbf{0}, \tag{D.16a}$$

$$\mathbf{c}(\mathbf{x}) = \mathbf{0}, \tag{D.16b}$$

$$XZ\mathbf{e} - \mu\mathbf{e} = \mathbf{0}, \tag{D.16c}$$

where $X := \text{diag}(\mathbf{x})$, $Z := \text{diag}(\mathbf{z})$, $\mathbf{e}$ is the vector of ones using the correct dimension and μ the homotopy parameter converging to zero. Furthermore, $\boldsymbol{\lambda} \in \mathbb{R}^m$ and $\mathbf{z} \in \mathbb{R}^n$ correspond to the Lagrangian multipliers for respectively the constraints (D.14b) and (D.14c). If one was to consider (D.16a)-(D.16c) with $\mu = 0$ and $x_i, z_i \geq 0\ \forall i$, the equations have transformed into the KKT conditions for the original problem (D.14a)-(D.14c). The method implemented in IPOPT approximates the solution to the interior-point problem (D.15a)-(D.15b) for a constant value of μ_j, decreases the value of μ_j afterwards and continues the solution of the next barrier problem (with barrier parameter μ_{j+1}) from the approximated solution of the previous barrier problem. For solving barrier problem (D.15a)-(D.15b), given a fixed value of μ_j, a damped Newton's method is applied to the primal-dual equations (D.16a)-(D.16c).
Filter methods offer an alternative to merit functions, being used to guarantee global convergence in non-linear programming algorithms. The general idea is that trial points are accepted if they result to an improvement in objective function or constraint violation. Merit functions on the other hand use a combination of these two measures. When considering the barrier problem (D.15a)-(D.15b) for a fixed value of μ_j, the idea is to interpret the barrier problem as a bi-objective optimization problem with two separate goals: minimizing the objective function and minimizing the constraint violation.
It is common for non-linear optimization methods to have second-order corrections. These corrections improve a proposed step once a trial point has been rejected. The second-order corrections, for a given step d_k, try to reduce infeasibility by applying additional Newton-type steps for the constraints at the next point $x_k + d_k$, making use of the Jacobian at the current position x_k.
All the above can be summarized into the following filter line search algorithm for solving the barrier problem (D.15a)-(D.15b):

1. Initialize.
2. Check convergence for the overall problem.
   - If convergence took place, then stop.
   - If not, continue with the next step.
3. Check convergence for the barrier problem.
4. Compute the search direction.
5. Backtracking line search:
   (5.1) Initialize line search.
   (5.2) Compute the new trial point.
   (5.3) Check acceptability to the filter.
      - If trial step is accepted, continue at step (5.4).
      - If trial step is rejected, go to step (5.5).
   (5.4) Check sufficient decrease w.r.t. the current iterate.
      - If trial step is accepted, go to step (6).
      - If trial step is rejected, continue at step (5.5).
   (5.5) Initialize the second-order correction.
   (5.6) Compute the second-order correction.
   (5.7) Check acceptability to the filter (in second-order conditions).
      - If trial step is accepted, continue at step (5.8).
      - If trial step is rejected, go to step (5.10).
   (5.8) Check sufficient decrease w.r.t. the current iterate (in second-order conditions).
      - If trial step is accepted, go to step (6).
      - If trial step is rejected, continue at step (5.9).
   (5.9) Next second-order correction.
      - If certain conditions satisfied, abort the second-order conditions and continue at step (5.10).
      - Otherwise, go back to step (5.6).
   (5.10) Choose the new trial step size.
      - If the trial step size becomes to small, go to step (9).
      - Otherwise, go back to step (5.2).
6. Accept the trial point.
7. Augmenting filter if necessary.
8. Continue at step (2).
9. Feasibility restoration phase.
   After this phase is completed, contine with regular iteration in step (8).

D.5 List of derivative-based solvers and their properties

This section contains a list of the derivative-based solvers and the relevant properties for calibration.

- **Ceres**
  The Ceres solver [3] by Google is an open source C++ library for both non-linear least-squares optimization problems with box-constraints as well as general unconstrained optimization problems. Ceres contains various optimization algorithms, providing good choices of algorithms for many types of optimization problems. Within Ceres the following methods can be found within the two categories of trust region methods and line search methods:

  - Trust region solvers:
    - Levenberg-Marquardt,
    - Powell's dogleg,
    - Subspace dogleg methods.
  - Line search solvers, including a number of variants of:
    - Non-linear conjugate gradients,
    - BFGS,
    - LBFGS.

  The user has the freedom of providing the first-order derivatives or deciding that the solver should do the work by letting the solver use either finite differences or AD.
  The method chosen for the Heston term structure calibration is the Levenberg-Marquardt method because of all the possibilities within the Ceres solver this method performed the best.

- **CMinpack**
  CMinpack [20] is an open-source library which can be used to solve non-linear systems of equations or for a least-squares minimization of the residual of a set of (non-)linear equations. CMinpack implements the Levenberg-Marquardt algorithm for unconstrained optimization using automatic scaling of variables.
  It is possible to include box-constraints by using the parameter conversions (6.11) and (6.12) that have been discussed in Section 6.4.1.
  In case of non-linear least-squares optimization, the user has the freedom to provide the first-order derivatives him/herself (solver name lmder) or decide to let the solver use finite difference approximation for the first-order derivatives (solver name lmdif).

- **IPOPT**
  IPOPT [65], or more complete Interior Point OPTimizer, is an open-source software package for non-linear optimization. This software is designed to handle box-bounds, as well as (non-)linear constraints. IPOPT implements a primal-dual interior point algorithm with a filter line search method. Appendix D.4 can be consulted if more information on the algorithm behind IPOPT is desired by the reader.
  Note that this solver does not recognize whether the objective function has a least-squares structure as in (6.14). This means that less of the available information is being used by the solver, so it would be expected that this solver will perform worse than least-squares solvers.
  The IPOPT solver is engineered in such a way that gradient and Hessian information are exploited when the user provides them. In case no Hessians are provided, the solver will approximate them using a BFGS-update quasi-Newton method.

- **Levmar**
  Levmar [44] is an open-source implementation of the Levenberg-Marquardt algorithm allowing for both constrained (box-bounds, linear equations and inequality constraints) as well as unconstrained optimization problems. Because it uses the Levenberg-Marquardt algorithm, the solver knows about the least-squares structure of the objective function.
  Levmar allows for both user-provided Jacobians (solver names dlevmar_der (unconstrained) and dlevmar_bc_der (box-bounds)) as well as derivative approximations within the solver (solver names dlevmar_dif (unconstrained) and dlevmar_bc_dif (box-bounds)). *"Notice that using finite differences to approximate the Jacobian results in repetitive evaluations of the function to be fitted. Aiming to reduce the total number of these evaluations, the _dif functions implement secant approximations to the Jacobian using Broyden's rank one updates."*

- **NAG e04fcc**
  This NAG optimizer [49] is *"is a comprehensive algorithm for finding an unconstrained minimum of a sum of squares of m non-linear functions in n variables ($m \geq n$)."* In short, this solver uses a combined Gauss-Newton method and modified Newton algorithm. One can use the earlier mentioned parameter conversions (6.11) and (6.12) to include box-constraints.
  No derivatives need to be supplied to this solver, all the necessary derivatives will be approximated internally using finite differences.

- **NAG e04gbc**
  This NAG optimizer [50] is in essence the same as NAG's e04fcc, apart from the fact that the user now has to supply the first order derivatives and no internal approximation is being done.

- **NAG e04unc** This NAG optimizer [51] is *"designed to minimize an arbitrary smooth sum of squares function subject to constraints (which may include simple bounds on the variables, linear constrains and smooth non-linear constraints) using a SQP method."* This solver combines a SQP method with the discussed least-squares properties for cheap computation of the Hessian of the Lagrangian.
  The Lagrangian for problem (6.13) is defined in (D.17) and consider its second derivative w.r.t $\mathbf{x}$ in (D.18):

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) = f(\mathbf{x}) - \sum_{i \in \mathcal{E} \cup \mathcal{I}} \lambda_i c_i(\mathbf{x}), \tag{D.17}$$

$$\nabla^2_{xx} \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) = \nabla^2 f(\mathbf{x}) - \sum_{i \in \mathcal{E} \cup \mathcal{I}} \lambda_i \nabla^2 c_i(\mathbf{x}). \tag{D.18}$$

  For more information on the Lagrangian and its relation to the optimality conditions, see Appendix D.1.
  In the current situation, the functions $c_i(\mathbf{x})$ can represent box-bounds, linear constraints and smooth non-linear constraints. The way the least-squares form of the objective function is used is by approximating the Hessian $\nabla^2 f(\mathbf{x})$ by its first derivatives as explained in Section 6.4.2. Generally speaking, any non-linear least-squares SQP method uses these results. The difference between the various SQP methods is mainly how the second term of the Hessian of the Lagrangian is dealt with. Note that in the case of no non-linear constraints but only linear constraints or box-constraints, the second term in the Hessian of the Lagrangian disappears and one is left with a method similar to the Gauss-Newton method.

The distinguishing feature of e04unc compared to other SQP solvers in dealing with the non-linear constraints is a positive definite quasi-Newton approximation to the Hessian of the Lagrangian using the BFGS updating strategy.
The first-order derivatives must be supplied to the solver by the user. This can be done using analytic derivatives, finite differences or by using AD. If no gradients are provided, the solver approximates them itself using finite differences.

- **NAG e04wdc**
  This NAG optimizer [52] is very similar to NAG's e04unc, but is based on NPOPT, which is a part of the SNOPT-package discussed in [32]. One of the differences between e04wdc and e04unc is that the first does not recognize any additional information provided by a least-squares formulation of the objective function.
  First-order derivatives can be supplied by the user or approximated internally by the solver itself using finite differences.

- **Schittkoswki: NLPLSQ**
  The NLPLSQ solver by Schittkowski [61] is an implementation of a SQP Gauss-Newton algorithm for non-linear least-squares optimization with (non-) linear constraints and box-bounds. By the introduction of slack-variables, *"the problem is transformed to a general smooth non-linear program subsequently solved by the SQP code NLPQLP. ... The additionally introduced variables are eliminated in the quadratic programming subproblem, so that calculation time is not increased significantly."*
  First-order derivatives have to be provided by the user in the form of an analytic derivative, a finite difference approximation or using AD.

D.6 Derivative-free optimization

This section contains a summary of material from Nocedal and Wright [57], Conn et al[15], Nocedal and Wright [57], Scheinberg [60] and Zhang et al [68].
This section discusses only model-based derivative-free optimization (see option three in Section 6.4.4). The underlying principle is the same as in the derivative-based optimization: the solution is found in an iterative process where a model of the objective function is built at each iteration point $\mathbf{x}_k$. The model then determines the new search direction or directly the new iterate $\mathbf{x}_{k+1}$. Whereas in the derivative-based optimization the model is assembled using derivative information at the current iterate $\mathbf{x}_k$, the model-based derivative-free optimization relies on a model which interpolates the objective function at chosen sample (interpolation) points across the domain. The points are selected in such a way that the local geometry of the objective is captured but at the same time not too close to each other, otherwise the model would be exposed to the same problems as finite differences (for example, the noise of the function evaluations). After the new point $\mathbf{x}_{k+1}$ is selected, the new point extends the set of interpolation points (or replaces one of them) and a new model is built. The process repeats till certain convergence criteria are satisfied. Note that in that way the most of the interpolation points are reused from iteration to iteration leading to significant savings of function evaluations. In contrast, a solver using FD approximations would evaluate the objective function multiple times at each iterate to obtain the gradient approximation but all these evaluations would be discarded after the new iterate is accepted. The rest of the chapter focuses on the model of the objective built by interpolations.

Consider the function $f(\mathbf{x})$, $\mathbf{x} \in \mathbb{R}^n$, which is to be interpolated by a (quadratic) polynomial $Q(\mathbf{x})$ at the set of interpolation points $Y = \{\mathbf{y}^j\}_{j=0}^{p} \subset \mathbb{R}^n$. This means that one needs to find

the function $Q(\mathbf{x})$ s.t.

$$Q(\mathbf{y}^j) \quad = \quad f(\mathbf{y}^j) \quad \forall\ j = 0, \ldots, p. \tag{D.19}$$

The set Y is called *poised* w.r.t. a given polynomial subspace if it can be interpolated by polynomials from this subspace. In other words, there exists polynomials from this subspace s.t. for any function f the condition (D.19) holds. The set Y is called *well-poised* if it remains poised (w.r.t a given polynomial subspace) under small perturbations. This definition is not yet formulated in a mathematically correct way, but for now this intuitive definition will suffice. For the optimization algorithm, the interest is only in sets of interpolation points that are well-poised.
Let $\mathcal{P}_n^d$ be the space of polynomials of degree less than or equal to d in $\mathbb{R}^n$. The dimension of this space is denoted by $q_1 = q + 1$. A basis $\phi = \{\phi_0(\mathbf{x}), \phi_1(\mathbf{x}), \ldots, \phi_q(\mathbf{x})\}$ of $\mathcal{P}_n^d$ is a set of q_1 polynomials of degree less than or equal to d that span $\mathcal{P}_n^d$.
The concept of poisedness relates to the non-singularity of a matrix Φ that will be introduced now. Say that $\{\phi_i(\cdot)\}_{i=0}^q$ spans the space $\mathcal{P}_n^d$. This means that any polynomial $Q(\mathbf{x})$ can be written as $Q(\mathbf{x}) = \sum_{i=0}^q \alpha_i \phi_i(\mathbf{x})$ for some coefficient vector $\boldsymbol{\alpha} = [\alpha_0, \ldots, \alpha_q]$. This allows one to rewrite the interpolation condition from (D.19) as a system of linear equations

$$\sum_{i=0}^{q} \alpha_i \phi_i(\mathbf{y}^j) \quad = \quad f(\mathbf{y}^j) \quad \forall\ j = 0, \ldots, p, \tag{D.20}$$

with coefficient matrix Φ, which is defined as

$$\Phi(\phi, Y) \quad = \quad \begin{pmatrix} \phi_0(\mathbf{y}^0) & \cdots & \phi_q(\mathbf{y}^0) \\ \vdots & & \vdots \\ \phi_0(\mathbf{y}^p) & \cdots & \phi_q(\mathbf{y}^p) \end{pmatrix}. \tag{D.21}$$

Now for a given set Y and function f, an interpolation polynomial Q exists and is unique if and only if the matrix $\Phi(\phi, Y)$ is square and non-singular. Assuming that the matrix is square from now on (i.e. $q = p$) and that the cardinality of Y and that of the basis $\{\phi_i(\cdot)\}$ are both equal to p. Now it can be stated that with respect to a given space of polynomials $\mathcal{P}_n^d$, the set Y is *poised* if $\Phi(\phi, Y)$ is non-singular. Note that the definition of poisedness here is independent of the polynomial basis that is chosen, as long as the basis spans the same space.

At this point, theoretically speaking, a poised set Y guarantees that the linear system (D.20) can be solved and that the interpolation polynomial Q can be found. On the other hand, numerically speaking, the matrix $\Phi(\phi, Y)$ from (D.21) can be ill-conditioned in some cases, even when non-singularity holds.
Recall that the condition number of a non-singular matrix A is defined as

$$\text{cond}_s(A) \quad = \quad ||A||_s \cdot ||A^{-1}||_s.$$

Here, s denotes the possible choice of different norms $||\cdot||_1$, $||\cdot||_2$ and $||\cdot||_\infty$. By default the Euclidian norm is chosen. Now, a problem is called well-conditioned when its solution is not affected greatly by small perturbations to the data that define the problem. If this is not the case, the problem is said to be ill-conditioned. In case the problem can be formulated as a linear system of equations, the condition number of the coefficient matrix can be used to quantify the conditioning.
This conditioning of the matrix depends on the choice of polynomial basis $\{\phi_i(\cdot)\}$. This dependence on choice of polynomial basis implies that in general, the condition number of $\Phi(\phi, Y)$

cannot be considered as an indicator for well-poisedness. However, for a specific choice of polynomial basis, namely the natural basis $\overline{\phi}$, and for a scaled version $\hat{Y}$ of Y, the condition number of $\Phi(\overline{\phi}, \hat{Y})$ provides a useful measure of well-poisedness. Note that the natural basis $\overline{\phi}$ is the basis of polynomials as they appear in the Taylor expansion.

As an example, a basis can be chosen such that $\Phi(\phi, Y)$ is the identity matrix. This basis is a commonly used one, and is called the the Lagrange fundamental polynomial basis. Given a set of interpolation points $Y = \{\mathbf{y}^j\}_{j=0}^p$, a basis of polynomials $\ell_i(\mathbf{x})$ ($i \in \{0, \ldots, p\}$ is called a basis of Lagrange polynomials if

$$\ell_i(\mathbf{y}^j) \;=\; \delta_{ij} \;=\; \begin{cases} 1, & i = j, \\ 0, & i \neq j. \end{cases}$$

When the set Y is poised, the basis of Lagrange polynomials exists and is uniquely defined. This existence implies poisedness of the sampling set. If the basis of Lagrange polynomials exists, There is a matrix A_ϕ with columns begin coefficients of these polynomials in the basis ϕ, s.t. $\Phi(\phi, Y)A_\phi = I$. As a result, the matrix $\Phi(\phi, Y)$ is non-singular.

At this moment still no measure of well-poisedness has been defined. This measure of poisedness should indicate how well an interpolation set Y spans the region of interest. In the non-linear case, one should consider how well the vectors $\phi(\mathbf{y}^j)$ for $j \in \{0, \ldots, p\}$ span the set of all $\phi(\mathbf{x})$ in the region of interest. It is obvious that this measure is going to depend on both the region of interest as well as the interpolation set Y. The (well-)poisedness of set Y has to depend on the polynomial space from which an interpolant is chosen as well.

From now on, consider the following equivalent definitions of well-poised sets:
Let $\Lambda > 0$ and a set $B \in \mathbb{R}^n$ be given. Let $\phi = \{\phi_j\}_{j=0}^p$ be a basis in $\mathcal{P}_n^d$. A poised set $Y = \{\mathbf{y}^j\}_{j=0}^p$ is said to be Λ-*poised in* B if and only if one of the following equivalent statements hold:

(i). For the basis of Lagrange polynomials associated with Y, the following holds:

$$\Lambda \;\geq\; \max_{0 \leq i \leq p} \max_{\mathbf{x} \in B} |\ell_i(\mathbf{x})|.$$

(ii). For any $\mathbf{x} \in B$ there exists $\boldsymbol{\lambda}(\mathbf{x}) \in \mathbb{R}^{p_1}$ s.t.

$$\sum_{j=0}^{p} \lambda_j(\mathbf{x})\phi(\mathbf{y}^j) \;=\; \phi(\mathbf{x}) \quad \text{with} \quad ||\boldsymbol{\lambda}(\mathbf{x})||_\infty \leq \Lambda.$$

(iii). Replacing any point in the set Y by any $\mathbf{x} \in B$ can increase the volume of the set $\{\phi(\mathbf{y}^j)\}_{j=0}^p$ at most by a factor Λ.

A thing to remark is that this definition does not require Y to be contained in B, in which the absolute values of the Lagrange polynomials are maximized. In fact, the set Y can be arbitrary far away from set B as long as all the Lagrange polynomials are bounded in absolute value on B. Note however that in order to guarantee the validity of algorithms that construct Λ-poised sets, points in B have to be generated.
The second definition shows that Λ is a measure of how well $\phi(Y)$ spans $\phi(B)$. If $\phi(Y)$ spans $\phi(B)$ well, then any vector in $\phi(B)$ can be expressed as a linear combinations of the vectors in $\phi(Y)$ with reasonably small coefficients. The third definition has a similar statement. If $\phi(Y)$

spans $\phi(B)$ well, then it is not possible to substantially increase the volume of $\phi(Y)$ by replacing one point in Y by any point in B.
In all these definitions, the constant Λ can be seen as a measure of distance to some non-poised set. It can be shown that $1/\Lambda$ bounds the distance to linear dependency of the vectors $\overline{\phi}(\mathbf{y}^j)$, $j = 0, \ldots, p$. The system represented by these vectors becomes increasingly linearly dependent as the value of Λ increases. Note however that the actual distance to singularity depends on how ϕ is chosen.
Now regard the following properties of Λ-poised sets:

(i). If B contains a point in Y and Y is Λ-poised in B, then $\Lambda \geq 1$.

(ii). If Y is Λ-poised in a given set B, then it is Λ-poised (with same constant Λ) in any subset of B.

(iii). If Y is Λ-poised in B for a given Λ-value, then it is $\tilde{\Lambda}$-poised for any $\tilde{\Lambda} \geq \Lambda$.

(iv). For any $\mathbf{x} \in \mathbb{R}^n$, if $\boldsymbol{\lambda}(\mathbf{x})$ is the solution of $\sum_{j=0}^{p} \lambda_j(\mathbf{x})\phi(\mathbf{y}^j) = \phi(\mathbf{x})$, then

$$\sum_{j=0}^{p} \lambda_j(\mathbf{x}) \quad = \quad 1.$$

(v). The poisedness constant Λ does not depend on the scaling of the sample set Y. For example, if Y is Λ-poised in B, then for any $\Delta > 0$, $\hat{Y} = Y/\Delta$ is Λ-poised in $\hat{B} = B/\Delta$.

(vi). Λ-poisedness is invariant with respect to a shift of the coordinates in the interpolation set Y.

Recall the earlier mentioned statement that for a specific choice of polynomial basis ϕ and interpolation set Y, the condition number of the coefficient matrix provides a useful measure of well-poisedness. In order to achieve this feature, choose the natural basis $\overline{\phi}$ as a polynomial basis, and for a scaled version $\hat{Y}$ of Y s.t.

$$\begin{aligned} \hat{Y} &= \{\mathbf{0}, \hat{\mathbf{y}}^1, \ldots, \hat{\mathbf{y}}^p\} \subset B(\mathbf{0}; 1), \\ \max_{1 \leq j \leq p} ||\hat{\mathbf{y}}^i - \hat{\mathbf{y}}^0|| &= \max_{1 \leq j \leq p} ||\hat{\mathbf{y}}^i|| \quad = \quad 1. \end{aligned}$$

Here $B(\mathbf{0}; 1)$ is the unit ball around the origin. Note that the choice of Y containing the origin is without loss of generality since Λ-poisedness is invariant with respect to a shift of the coordinates in the interpolation set Y. In derivative-free optimization based on interpolation, the best iterate so far is typically the center of the interpolation and it can always be seen as the origin. The condition number of the matrix $\Phi(\overline{\phi}, \hat{Y})$ depends on the scaling of $\hat{Y}$, but is independent of any region B. Λ-poisedness on the other hand is independent of scaling but does depend on the region B. Now fix the region to be the smallest ball centered at the origin that contains Y, noted by $B = B(\mathbf{0}; \Delta(Y))$. Now scale B and Y s.t. the radius of B equals 1. This results in the unit ball $B(\mathbf{0}; 1)$ and sample set $\hat{Y}$ which is contained in this unit ball centred at the origin and contains at least one point on the boundary of the ball.

Now consider the matrix $\hat{\Phi} = \Phi(\overline{\phi}, \hat{Y})$. It can be shown that in order to bound the condition number $\text{cond}_2(\hat{\Phi})$ in terms of Λ, it is sufficient to bound $||\hat{\Phi}^{-1}||_2$. Conversely, in order to bound Λ in terms of $\text{cond}_2(\hat{\Phi})$, it is sufficient to bound it in terms of $||\hat{\Phi}^{-1}||_2$. Furthermore it can be shown that if $\hat{\Phi}$ is non-singular and $||\hat{\Phi}^{-1}||_2 \leq \Lambda$, the set $\hat{Y}$ is $\sqrt{p_1}\Lambda$-poised in the unit ball

$B(\mathbf{0};1)$ centred at the origin. Conversely, if the set $\hat{Y}$ is Λ-poised in the unit ball $B(\mathbf{0};1)$ centred at the origin, then

$$||\hat{\Phi}^{-1}||_2 \quad \leq \quad \theta\sqrt{p_1}\Lambda. \tag{D.22}$$

Here, $\theta > 0$ is dependent on both n and d but independent of $\hat{Y}$ and Λ.

Now that it is known how the condition number of coefficient matrix $\hat{\Phi}$ relates to the measure of poisedness, the interpolation conditions and the matrix $\hat{\Phi}$ can be used directly in a derivative-free algorithm. This all can be done without computing any Lagrange polynomials or equivalents, and it is still possible to maintain sufficient poisedness of the interpolation sets which guarantees a reasonable bound on the interpolation error.
From now on, consider only the case of quadratic polynomials, i.e. $d = 2$. Assume that $Y = \{\mathbf{y}^j\}_{j=0}^p \subset \mathbb{R}^n$ with $p_1 = p + 1 = \frac{(n+1)(n+2)}{2}$ is a poised set of sample points contained in the ball $B(\mathbf{y}^0;\Delta(Y))$ of radius $\Delta = \Delta(Y)$. Furthermore assume that the objective function f is twice continuously differentiable in an open domain Ω containing $B(\mathbf{y}^0;\Delta)$ and $\nabla^2 f$ is Lipschitz continuous in Ω with constant $\nu_2 > 0$. Now it is possible to build the quadratic interpolation model

$$\begin{aligned} Q(\mathbf{y}) \quad &= \quad c + \mathbf{g}^T\mathbf{y} + \frac{1}{2}\mathbf{y}^T H\mathbf{y}, \\ &= \quad c + \sum_{1\leq k\leq n} g_k y_k + \frac{1}{2}\sum_{1\leq k,\ell\leq n} h_{k\ell} y_k y_\ell. \end{aligned} \tag{D.23}$$

Here, H is a symmetric $n \times n$ matrix. The unknown coefficients c, $g_1, \ldots, g_n$ and h_{kl} for $1 \leq \ell \leq k \leq n$ are uniquely defined by the interpolation conditions (D.20) because the sample set is poised. Now assume without loss of generality, as argued before, that $\mathbf{y}^0 = \mathbf{0}$. Consider a point $\mathbf{y} \in B(\mathbf{0};\Delta)$ for which an error in the function value

$$Q(\mathbf{y}) \quad = \quad f(\mathbf{y}) + e^f(\mathbf{y}),$$

gradient

$$\nabla Q(\mathbf{y}) \quad = \quad \nabla f(\mathbf{y}) + e^g(\mathbf{y}),$$

and Hessian

$$H \quad = \quad \nabla^2 f(\mathbf{y}) + e^H(\mathbf{y})$$

will be estimated. If the above described assumptions hold, for all points $\mathbf{y} \in B(\mathbf{0};\Delta)$, the following statements hold for some constants $\kappa_{eH}, \kappa_{eg}, \kappa_{ef} > 0$ which can be found in [15]:

- the error between the Hessians of the quadratic interpolation model and the objective function satisfies

$$||\nabla^2 f(\mathbf{y}) - \nabla^2 Q(\mathbf{y})||_2 \quad \leq \quad \kappa_{eH}\Delta,$$

- the error between the gradients of the quadratic interpolation model and the objective function satisfies

$$||\nabla f(\mathbf{y}) - \nabla Q(\mathbf{y})||_2 \quad \leq \quad \kappa_{eg}\Delta^2,$$

- the error between the quadratic interpolation model and the objective function satisfies

$$|f(\mathbf{y}) - Q(\mathbf{y})| \quad \leq \quad \kappa_{ef}\Delta^3.$$

Note that the magnitude of the error constants $\kappa_{eH}, \kappa_{eg}, \kappa_{ef}$ can be expressed in terms of $||\hat{\Phi}||_2$, thus one is able to express the error bounds in terms of the poisedness constant Λ.