BERGISCHE UNIVERSITÄT WUPPERTAL

Hierarchical Approaches in Modeling and Numerical Simulation for Mathematical Finance

Dissertation

zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

der

Fakultät 4 - Mathematik und Naturwissenschaften

Anna Lisa Clevenhaus

supervised by

Prof. Dr. Matthias Ehrhardt

Prof. Dr. Michael Günther

2025-09-25

Acknowledgement

An dieser Stelle möchte ich meinen besonderen Dank nachstehenden Personen aussprechen, ohne deren Mithilfe die Anfertigung dieser Promotionsschrift niemals zustande gekommen wäre:

Mein besonderer Dank gilt Prof. Dr. Matthias Ehrhardt und Prof. Dr. Michael Günther für ihre herausragende Unterstützung bei der Durchführung der gesamten Arbeit, angefangen mit der Freiheit im Findungsprozess bis hin zum Weben des roten Fadens durch die unterschiedlichen Themen, sowie abschließend den damit verbundenen Möglichkeiten des Wachstums auf persönlicher und fachlicher Ebene. Ich habe unsere Dialoge stets als Motivation und wegweisend empfunden.

Außerdem möchte ich Prof. Dr. Claudia Totzeck und Prof. Dr. Daniel Sevcovic meinen ausdrücklich herzlichen Dank für die erfolgreiche Zusammenarbeit, ihr Engagement und ihre Unterstützung insbesondere bei Schwierigkeiten aussprechen.

Mein Dank gilt auch den aktuellen und ehemaligen ACM Mitgliedern für ihr offenes Ohr und die vielen produktiven Gespräche innerhalb und außerhalb der Universität. Hier möchte ich Hanna, Michelle, Tatiana, Friedemann, Lorenc sowie Kevin, Daniel und Konrad hervorheben.

Ebenfalls danke ich dem Dekanat, vor allem Kerstin und Isabella für den motivierende Zuspruch.

Zudem danke ich meiner Familie, besonders Annette und Wolfgang, und meinen Freunden für die unermüdliche Unterstützung.

Zuletzt bedanke ich mich noch bei meinem Ehemann Matthias für seine Geduld, seine Ermutigung, sein Engagement und Verständnis für den Prozess der Dissertation, sowie meiner Tochter Helena für unzählig motivierende Momente durch ihre Fähigkeit zur Begeisterung.

Foreword

This thesis is build on the following publication during my Phd from December 2018 to May 2025.

- A. Clevenhaus, M. Ehrhardt, M. Günther, and D. Ševčovič. Pricing American Options with a Non-Constant Penalty Parameter. *J. Risk Financial Manag.* 13(6) (2020), 124. DOI: 10.3390/jrfm13060124

  As the American early exercise results in a free boundary problem, in this article we add a penalty term to obtain a partial differential equation, and we also focus on an improved definition of the penalty term for American options. We replace the constant penalty parameter with a time-dependent function. The novelty and advantage of our approach consists in introducing a bounded, time-dependent penalty function, enabling us to construct an efficient, stable, and adaptive numerical approximation scheme, while in contrast, the existing standard approach to the penalization of the American put option-free boundary problem involves a constant penalty parameter. To gain insight into the accuracy of our proposed extension, we compare the solution of the extension to standard reference solutions from the literature. This illustrates the improvement of using a penalty function instead of a penalizing constant.

  Used in Chapter 2, 4, 6

- A. Clevenhaus, M. Ehrhardt, and M. Günther. An ADI Sparse Grid method for pricing efficiently American Options under the Heston model. *Adv. Appl. Math. Mech.* 13 (2021), 1384-1397. DOI: 10.4208/aamm.OA-2020-0317

  One goal of financial research is to determine fair prices on the financial market. As financial models and the data sets on which they are based are becoming ever larger and thus more complex, financial instruments must be further developed to adapt to the new complexity, with short runtimes and efficient use of memory space. Here we show the effects of combining known strategies and incorporating new ideas to further improve numerical techniques in computational finance. In this paper we combine an alternating direction implicit (ADI) scheme for the temporal discretization with a sparse grid approach and the combination technique. The later approach considerably reduces the number of 'spatial' grid points. The presented standard financial problem for the valuation of American options using the Heston model is chosen to illustrate the advantages of our approach, since it can easily be adapted to other more complex models.

Used in Chapter 2, 3, 6

- A. Clevenhaus, M. Ehrhardt, and M. Günther. The Parareal Algorithm and the Sparse Grid Combination Technique in the Application of the Heston Model. In: M. Ehrhardt and M. Günther (eds.), *Progress in Industrial Mathematics at ECMI 2021*, pages 477–483, Cham, 2022. Springer International Publishing, 2023. DOI: 10.1007/978-3-031-11818-0_62

  The sparse grid combination technique is an efficient method to reduce the curse of dimensionality for high-dimensional problems, since it uses only selected grids for spatial discretization. To further reduce the computational complexity in the temporal dimension, we choose the Parareal algorithm, a parallel-in-time algorithm. For the coarse and fine solvers in time, we use an efficient implementation of the Alternating Direction Implicit (ADI) method, which is an unusual choice due to the larger computational cost compared to the usual choice of one-step or Runge-Kutta methods. In this paper we combine both approaches and therefore obtain a even more efficient computational method for parallelism. The application problem is to determine a fair price of a Put option using the Heston model with correlation. We analyze this model as an example to illustrate this advantageous combination of the sparse grid with the Parareal algorithm. Finally, we present further ideas to improve this advantageous combination of methods.

  Used in Chapter 2, 3, 6

- A. Clevenhaus, C. Totzeck, and M. Ehrhardt. A numerical study of the effects of different boundary conditions on the variance of the Heston model. In: K. Burnecki, J. Szwabiński, and M. Teuerle (eds.), *Progress in Industrial Mathematics at ECMI 2023*. Springer, 2025.

  The well-posedness analysis of a parabolic partial differential equation (PDE), such as the Heston PDE, requires the proper definition of an initial condition and boundary conditions. In contrast to the asset boundary conditions, the variance boundary conditions cannot be directly derived. In the literature different approaches to the variance boundary conditions are discussed, for example they consider the challenge of singularities when the variance approaches zero. This work focuses on the sensitivity of numerical approximations of the solution with respect to the variance boundary conditions.

Used in Chapter 3, 4, 6

- A. Clevenhaus, C. Totzeck, and M. Ehrhardt. A gradient-based calibration method for the Heston model. *Int. J. Comput. Math.*, 101(9-10) (2024), 1094–1112. DOI: 10.1080/00207160.2024.2353189

  The Heston model is a well-known two-dimensional financial model. Because the Heston model contains implicit parameters that cannot be determined directly from real market data, calibrating the parameters to real market data is challenging. In addition, some of the parameters in the model are non-linear, which makes it difficult to find the global minimum of the optimization problem within the calibration. In this paper, we present a first step towards a novel space mapping approach for parameter calibration of the Heston model. Since the space mapping approach requires an optimization algorithm, we focus on deriving a gradient descent algorithm. To this end, we determine the formal adjoint of the Heston PDE, which is then used to update the Heston parameters. Since the methods are similar, we consider a variation of constant and time-dependent parameter sets. Numerical results show that our calibration of the Heston PDE works well for the various challenges in the calibration process and meets the requirements for later incorporation into the space mapping approach. Since the model and the algorithm are well known, this work is formulated as a proof of concept. Used in Chapter 4, 5, 6

- A. Clevenhaus, C. Totzeck, and M. Ehrhardt. A Space Mapping approach for the calibration of financial models with the application to the Heston model. *arXiv preprint* arXiv:2501.14521 (January 2025), submitted to J. Comput. Finance. DOI: 10.48550/arXiv.2501.14521

  Used in Chapter 4, 5, 6

- A. Clevenhaus, M. Ehrhardt and M. Günther, A parallel Sparse Grid Combination Technique using the Parareal Algorithm, Preprint 21/18, June 2021. `https://www.imacm.uni-wuppertal.de/fileadmin/imacm/preprints/2021/imacm_21_18.pdf`

  Used in Chapter 2, 3, 6

List of Symbols

Mathematical symbols

exp	Exponential
$\frac{\partial u}{\partial t}$	Partial derivative of u w.r.t. t
$\int$	Integral
log	Natural logarithm
$\mathcal{C}^{\alpha,\beta}$	Denotes the smoothness/differentiability of a function
d	Differential
∇	Gradient
∂	Boundary
sinh	Hyperbolic sine
$\surd$	Square Root
$\sum$	Sum
$^\top$	Transposed Matrix
$^{-1}$	Inverse

Sets

$\mathbb{N}$	Set of natural numbers without zero
$\mathbb{N}_0$	Set of natural numbers with zero
$\mathbb{R}$	Set of real numbers

$\mathbb{R}_{>0}$	Set of real numbers larger zero.
$\mathbb{X}$	Set of all possible calibration parameters $\boldsymbol{\xi}$

Vectors and Matrices

$\boldsymbol{\Delta}$	Temporal spacing for the Parareal
$\boldsymbol{\eta}$	d-dimensional vector with intermediate results for the grid transformation
$\boldsymbol{\gamma}$	d-dimensional vector of the transformation coefficient to the unit interval
$\boldsymbol{\lambda}$	d-dimensional Lagrangian multiplier for the ADI-IT schemes
$\boldsymbol{\mu}(\mathbf{X}, t)$	Drift term of a stochastic process
$\boldsymbol{\sigma}(\mathbf{X}, t)$	Diffusion term of a stochastic process
$\boldsymbol{\tau}$	Grid point within the fine Parareal grid
$\boldsymbol{\varrho}$	$d-$dimensional variable
$\boldsymbol{\zeta}$	d-dimensional vector with intermediate results for the grid transformation
$\boldsymbol{h}$	Gradient direction resulting from an calibration step
$\boldsymbol{X}$	Monte Carlo estimate
$\mathbf{A}$	$d \times d-$dimensional matrix within the divergent form
$\mathbf{b}$	$d-$dimensional vector within the divergent form
$\mathbf{e}$	$d-$dimensional unit vector
$\mathbf{h}$	d-dimensional spatial spacing vector
$\mathbf{l} = (\mathrm{l}_1, \ldots, \mathrm{l}_d) \in \mathbb{N}^d$	d-dimensional vector for the sparse grid
$\mathbf{M} = (\mathrm{M}_1, \ldots, \mathrm{M}_d) \in \mathbb{N}^d$	d-dimensional vector for the number of spatial slices
$\mathbf{m} = (\mathrm{m}_1, \ldots, \mathrm{m}_d) \in \mathbb{N}^d$	d-dimensional index vector for the spatial directions
$\mathbf{N}$	Multi-indice for the Parareal grid
$\mathbf{W} = (\mathrm{W}^1, \mathrm{W}^2, \ldots, \mathrm{W}^d)$	d-dimensional vector of independent Brownian motions
$\mathbf{x} = (\mathrm{x}_1, \ldots, \mathrm{x}_d)$	d-dimensional vector of the spatial dimensions

$\mathbf{X} = (\mathrm{X}_1, \mathrm{X}_2, \dots, \mathrm{X}_d)$	d-dimensional vector of the spatial dimensions within a stochastic context
$\mathbf{y} = (\mathrm{y}_1, \dots, \mathrm{y}_d)$	d-dimensional arbitrary vector on Ω_d
$\mathbf{Z} = (\mathrm{Z}_1, \dots, \mathrm{Z}_d)$	$d-$dimensional vector of independent random variables
$\vec{\mathbf{n}}$	Normal vector
h	Arbitrary direction

Greek symbols

α	Constant
β	Constant
$\boldsymbol{\xi}$	Vector of the calibration parameter for the space mapping
Δ	Temporal spacing
δ	Finite difference stencil
$\delta(\tau)$	Affine function
$\delta(\tau)$	Time dependent affine function within the penalty term
$\epsilon > 0$	Constant within the projected Armijo rule
Γ	Spatial boundary for a specific spatial direction
γ	Transformation coefficient
ι	Iteration parameter within the projected Armijo rule
$\kappa > 0$	Risk neutral long term mean of the variance
$\mu > 0$	Risk neutral mean reversion rate
ν	Parameter for the exponential boundary fit
Ω	Spatial domain
ω	Solution of the stochastic financial problem
Φ	Probability function of the standard normal distribution

ϕ	Payoff-function
ψ	Lagrangian multiplier within the space mapping
$\Psi(\mathbf{X})$	General Payoff-function
$\rho \in [-1, 1]$	Correlation between the asset and the variance
$\sigma > 0$	Volatility
$\tau \in [0, T]$	Reversed time and coarse temporal grid
θ	Variable of the implicitness of the scheme
$\Upsilon \in \mathbb{R}$	Stochastic variable for the variance of the asset
$\upsilon \in \mathbb{R}$	Deterministic variable for the variance of the asset
$\varepsilon > 0$	Constant
φ	Lagrangian multiplier function within the space mapping
ϑ	Parameter for the exponential boundary fit

Latin Symbols

$\boldsymbol{r}$	Misalignment function within the space mapping
$\boldsymbol{s}$	Space mapping function
$\mathcal{A}$	Spatial operator for the second order derivatives
$\mathcal{E}$	Operator for the Lagrangian constraint
$\mathcal{F}$	Temporal operator for the fine solver within the Parareal
$\mathcal{G}$	Temporal operator for the coarse solver within the Parareal
$\mathcal{J}$	Cost functional operator for the space mapping
$\mathcal{L}$	Spatial operator of a parabolic PDE
$\mathcal{N}$	Normal distributed
$\mathcal{O}$	Order
$\mathcal{P}$	Projection operator within the projected Armijo rule

$\mathcal{Q}$	Risk-neutral measure
$\mathrm{A}(\mathbf{x}, t)$	Matrix generated from $\mathcal{A}$
M	Number of slice for a spatial direction
m	Index variable for a grid node
N	Constant for the number of slices for the coarse temporal discretization
n	Index variable for the fine temporal discretization
s	Integral index
U	Fully discrete solution for u
v	Integral index
z	Integral index
a	Boundary condition case for $\nu_{\min}$
b	Boundary condition case for $\nu_{\min}$
c	Boundary condition case for $\nu_{\max}$
d	Boundary condition case for $\nu_{\max}$
e	Boundary condition case for $\nu_{\max}$
f	Boundary condition case for $\nu_{\max}$
p	Index variable for a specific path
$\mathsf{d}_1, \mathsf{d}_2$	Intermediate results for the semi-analytical solution of the BS
$\mathtt{V}$	Hierarchical surplus within the sparse grids
a	Constant within the affine function in the time dependent penalty term
$a(\mathbf{x}, t)$	Function for the second derivative
$A(S, T)$	Averaging function within Asian options
B	Constant for the Binary option
b	Constant within the affine function in the transformed time-dependent penalty term

$b(\mathbf{x},t)$	Function for the first derivative
C	Computational time for the computation of the solution
c	Computation time for one time step
$c(\mathbf{x},t)$	Coefficient function for the spatial operator
C_1, C_2	Constant within the affine function in the transformed time-dependent penalty term
D	Continuous dividend rate
d	Number of dimensions
$E[X_t]$	Expectation value
F	Fine temporal operator reusing the intermediate coarse result within the Parareal
$f(\mathbf{x},t)$	Right hand side function of the linear parabolic PDE
G	Coarse temporal operator reusing the intermediate fine result within the Parareal
$g\Big(u(x_a,\tau)\Big)$	Penalty term
I	Intermediate representations for the constraint within the Lagrangian
J	Cost functional for the space mapping approach
K	Strike, exercise price
L	Lagrangian for the constrained parameter calibration problem
$m \in \mathbb{N}_0$	Surplus limit in the sparse grid
$N \in \mathbb{N}$	Constant Number for a specific parameter
n	Index variable for the (coarse) temporal discretization
P	Number of path in the Monte Carlo simulation
p	Order of accuracy for the sparse grid
$p(S,\tau)$	Penalty term for the BS model
q	Index variable for the surplus of the sparse grids within the combination technique

$r > 0$	risk-free interest rate
$S > 0$	Underlying asset in the stochastic setting
$s > 0$	Underlying asset in the deterministic setting
$t \in [0, T]$	Time
$T > 0$	Maturity date
U	Semi-discrete solution for u
$u(\boldsymbol{\xi})$	Fair price of an option for the model parameter $\boldsymbol{\xi}$
$u(\mathbf{x}, t), u(\mathbf{x}, \tau)$	Solution/fair price of a(n option pricing) problem
$v(\hat{x}, \hat{\tau})$	Transformation of $u(x, \tau)$
$w(h)$	Bounded error function for the spacings h_i at the i-th spatial direction
$X \in \mathbb{R}$	$\log -$transformed underlying, asset in the stochastic setting
$x \in \mathbb{R}$	$\log -$transformed underlying, asset in the deterministic setting
$Y, \tilde{Y}, \hat{Y}$	Intermediate results of the ADI schemes
$Z \in \mathbb{R}$	$\log -$ normalized underlying, asset in the stochastic setting
$z \in \mathbb{R}$	$\log -$ normalised underlying, asset in the deterministic setting
B1-B8	Boundary cases
C0-C7	Parameter constellation cases for the gradient descent algorithm.
D1-D5	Parameter sets for discretization
Guess	Guess sets for initial parameters.
K1-K5,K5a	Parameter sets generated from real market data
P1,P2	Parameter sets for American put option pricing
T1-T5	Test cases for the gradient descent algorithm

Symbols as Subscripts

0	Initial condition for the stochastic variable

$\mathbf{l}$	Parameter w.r.t. the sparse grid given by $\mathbf{l}$
$\mathbf{M}$	Parameter w.r.t. the tensor grid given by $\mathbf{M}$
$\mathbf{m}$	Index vector for the grid point position
$\mathbf{M}_\mathbf{l}$	Parameter w.r.t. the sparse grid given by $\mathbf{M}$ and $\mathbf{l}$
$\mathcal{F}$	Parameter w.r.t. the fine solver within the Parareal
$\mathcal{G}$	Parameter w.r.t. the coarse solver within the Parareal
A	Parameter w.r.t. the Adjoint
BS	Parameter w.r.t. the Black-Scholes model
data	Parameter w.r.t. (reference) data
f	Parameter w.r.t. the fine solver within the space mapping
H	Parameter w.r.t. the Heston model
k	Index variable
Parareal	Parameter w.r.t. the Parareal
Serial	Parameter w.r.t. the serial computation
ν	Parameter w.r.t. ν
a	Spatial boundary
b	Spatial boundary
c	Spatial boundary
d	Spatial boundary
τ	Parameter w.r.t. τ
a	Spatial direction w.r.t. the asset
d	Number of dimensions
f	Free boundary value
$i, j = 1, \ldots, d$	Index variable for the dimension

K	Grid point matching K
n	Parameter at the $n-$th time step
N_p	Number of Paths
S	Parameter w.r.t. S
s	Parameter w.r.t. the sparse grid
t	Parameter w.r.t. t
x	Parameter w.r.t. x
$\mathbf{M}, \mathbf{m}$	Grid point defined by the multi-indice
M_i, m_i	m_i-th grid point in the $i-$th direction

Symbols as Upperscripts

$*$	argmin of the corresponding calibration problem
$+$	Finite difference forward stencil
$-$	Finite difference backward stencil
0	Finite difference central stencil
$\bar{\ }$	Denotes transformation
$\boldsymbol{-}$	Antithetic variant of the parameter
$\hat{\ }$	Denotes transformation
ι	Parameter within the $\iota-$th loop within the Armijo rule
AC	Parameter w.r.t. American call options
AP	Parameter w.r.t. American put options
As	Asian Option
A	Parameter w.r.t. American options
Ba	Basket Option
Bi	Binary Option

cal	Calibrated value of the parameter
comm	Parameter w.r.t. communication
EC	Parameter w.r.t. European call options
EP	Parameter w.r.t. European put options
E	Parameter w.r.t. European options
init	Initial value/guess of the parameter
opt	Optimal value of the parameter
Ra	Rainbow Option
ref	Reference value of the parameter
sparse	Parameter w.r.t. the sparse grid
spot	Spot value of the variable
max	Maximal value of the parameter
min	Minimal value of the parameter
$\mp$	Function including the antithetic
$\pm$	Finite difference combined stencil
$\tilde{}$	Denotes transformation
Υ	Parameter w.r.t. Υ
d	d-dimensional
k	Parameter within the $k-$th iteration
p	$p-$th path
S	Parameter w.r.t. S
X	Parameter w.r.t. X

List of Figures

List of Tables

List of Algorithms

Contents

Chapter 1

Introduction

The financial market reflects the changes in the world. Therefore, politics, economics and even nature have a direct impact on the prices and stability of the financial market. These interdependencies lead to many uncertainties in the market, which in the worst case can cause rapid price changes. A historical example is the tulip mania starting in 1634, when tulips were traded at extraordinary high prices. When this speculative bubble collapsed in 1637, it was the first recorded asset bubble in history. To this day, the term "tulip mania" is used as a metaphor for any major economic bubble in which asset prices deviate from their intrinsic value. Since then, the world has faced several financial crises, such as the Wall Street Crash of 1929, which started the Great Depression (1929-1939), as well as the global financial crisis of 2007-2009, which led to the international banking crisis [105]. Today, we are facing the global economic consequences of the COVID-19 pandemic. Thus, financial mathematics has a direct impact on our daily lives.

1.1 Options

In finance, derivatives can be understood either as a kind of insurance or as a tool for profiting from risk speculations. A derivative is a right whose price depends on the market price of an underlying asset. Generally, an underlying is also a financial instrument, so it can be a derivative itself. Other common types of underlyings are indices, currencies or securities. We focus on options, a special type of derivatives. Other derivatives are futures or swaps. An option is a contract between a writer and a holder. The writer sets the terms of the contract and sells the option. The holder buys the option from the writer for the market value, also known as the premium. The option contract gives the buyer the right, but not the obligation, to exercise the exchange trade by the predetermined expiration date $T > 0$. Thus, we are dealing with a time t with $0 \leq t \leq T$. The contract specifies whether the buyer wants to buy (Call) or sell (Put) a predefined amount of an underlying asset S at a predefined exercise price K. The exercise price is also called the strike price. Since the price of the underlying S varies over time, we write $S(t)$ to denote the price at time t. We will focus on European and American plain vanilla options, as well as Asian options, and include other exotic options to give an overall perspective. We will start with European plain vanilla options.

1.1.1 European Options

European plain vanilla options describe a contract between a holder and a writer, [106].

Definition 1 (European call/put option)**.**
A European call/put is a contract between the writer (the party that sells the option) and the holder (the party that buys the option). The contract gives the buyer the right, but not the obligation, to buy (Call option) or sell (Put option) an underlying asset S at an agreed fixed strike price $K > 0$ on a specific date $T > 0$.

Since the holder has no obligation to exercise the option, his biggest loss is the premium, the cost of the option itself. On the other hand, the writer must commit to exercising the option if the holder wishes to exercise it. Therefore, the risk analysis and the corresponding mathematical methods are different from the point of view. Since there is an asymmetry between writing and owning an option, there are different points of view. In this thesis, we focus on the long position, the holder's point of view.

Whether the holder exercises the option depends on the difference between the underlying and the strike. A Call option is exercised if $S > K$ holds because the spot price S is higher than the strike price. The profit is measured by $S - K$. If the spot price is lower than the market price, the holder buys the underlying at the market price and does not exercise the option. For Put options, the exercise range is $K - S$, because if $S > K$, the holder will sell to the market at the higher spot price. If $K = S$, both options are worthless because the market price is equal to the strike.

Definition 2 (Payoff-function European call/put option)**.**
The payoff-function ϕ of the European call/put option is given by

$$\phi(S(T)) = \begin{cases} \max\Big(S(T) - K, 0\Big), & \text{for } S(T) \geq 0 \quad \text{(Call)}, \\ \max\Big(K - S(T), 0\Big), & \text{for } S(T) \geq 0 \quad \text{(Put)}. \end{cases} \tag{1.1}$$

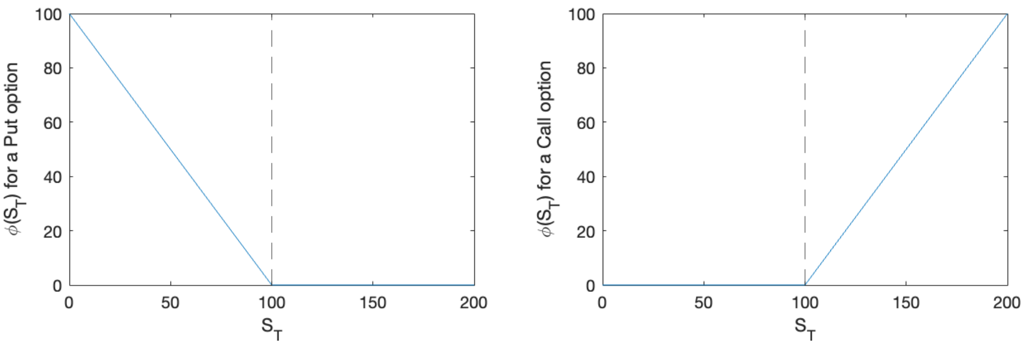


Figure 1.1: European payoff-function for $S \in [0, 250]$ with $K = 100$. On the left side is the payoff-function for the Put option and on the right for the Call option. The dashed line marks the strike.

where $S(T) \in [0, \infty[$.

Since the strike price separates the At-The-Money (ATM), In-The-Money (ITM), and Out-The-Money (OTM) options, we have a region of interest around the strike price K. The ATM option is given when $K = S$, regardless of whether it's a Call or Put option type. An option is ITM when it makes a profit and OTM when it is worthless. The in-the-money and out-the-money regions are swapped for Put and Call options, since for a Call option the region where $S > K$ is in-the-money and for Put options out-the-money, since the value of the Put option is zero if $S > K$. For the region $S < K$, a Put option is in-the-money and a Call option is out-the-money. The most interesting part is the region around the strike price, because that is where the region swap is. So one tries to get a lot of grid points around the strike price or since the spot price is often close to the strike price.

Let ω be the value of an option. The value depends at least on the underlying and the time, e.g. for the one dimensional case where the underlying is given by S, we obtain $\omega(S(t), t)\colon \mathbb{R}_{\geq 0} \times [0, T] \to \mathbb{R}_{\geq 0}$. Later we will consider more complex models, so ω will depend on other parameters as well. Our goal is to determine the fair price of an option value. When we refer to a specific option, we specify the option value. For a European call option, we denote the value by ω^{EC}, and for a European put option, by ω^{EP}.

A market is considered arbitrage free if there is no strategy to make a certain profit without risk. Since we are assuming an arbitrage-free market, these assumptions about the market follow immediately:

- No arbitrage possible.
- No dividends are paid on the underlying.

- The risk-free interest rate for cash investments and loans is the same and is $r > 0$ for continuous interest.
- The market is liquid and trading is possible at all times.

By assuming a continuous interest rate, we can determine the price we have to invest now to get back the amount K at time T. Therefore, we discount K with $\exp(-rT)$.

Proposition 1. *Put-Call-Parity*
Under the principle of non-arbitrage and the resulting market conditions

$$S(t) + \omega^{\mathrm{EP}}(S(t), t) - \omega^{\mathrm{EC}}(S(t), t) = K \exp\Big(- r(T-t)\Big) \tag{1.2}$$

holds for $0 < t < T$.

From the Put-Call-Parity, we can derive bounds for the option values for European options.

Proposition 2. *European options are bounded by*

- $\max\Big(S(t) - K\exp\Big(-r(T-t)\Big), 0\Big) \leq \omega^{\mathrm{EC}}(S(t), t) \leq S(t)$
- $\max\Big(K\exp\Big(-r(T-t)\Big) - S(t)t, 0\Big) \leq \omega^{\mathrm{EP}}(S(t), t) \leq K\exp\Big(-r(T-t)\Big)$

1.1.2 American Options

In addition to European options, we will also discuss American options. However, both options can be exercised in Europe and America. The difference between European and American options is the ability to exercise the option. While European options can only be exercised at expiration T, American options can be exercised at any time before $t < T$ and at expiration $t = T$. Thus, for the value of American options ω^{A} and European options ω^{E}, the relation

$$\omega^{\mathrm{A}}(S(t), t) \geq \omega^{\mathrm{E}}(S(t), t) \tag{1.3}$$

must hold, and the bounds for American options are derived.

Proposition 3. *American options are bounded by*

- $\omega^{\mathrm{AC}}\Big(S(t), t\Big) \geq \omega^{\mathrm{EC}}\Big(S(t), t\Big)$, *equality holds if no dividends are paid*

- $K \exp\Big(-r(T-t)\Big) \leq S(t) + \omega^{\mathrm{AP}}\Big(S(t),t\Big) - \omega^{\mathrm{AC}}\Big(S(t),t\Big) \leq K$
- $\Big(K \exp\Big(-r(T-t)\Big) - S(t), 0\Big) \leq \omega^{\mathrm{AP}}\Big(S(t),t\Big) \leq K$

From the non-arbitrage principle, the consistency of the option values and the European payoff-function, we conclude that

$$\omega^{\mathrm{AP}}\Big(S(t),t\Big) \geq \max\Big(K - S(t), 0\Big) \quad \text{and} \quad \omega^{\mathrm{AC}}\Big(S(t),t\Big) \geq \max\Big(S(t) - K, 0\Big),$$

$$\omega^{\mathrm{AP}}\Big(S(t),t\Big) \geq \phi\Big(S(T)\Big) \quad \text{and} \quad \omega^{\mathrm{AC}}\Big(S(t),t\Big) \geq \phi\Big(S(T)\Big).$$

Thus, there exists an intersection S_f of the option value with the payoff. Due to the convexity of the option value, the existence and uniqueness of the intersection point is guaranteed. The contact point divides the function into a continuation region and a stopping region. Within the continuation region, exercising the option would result in an immediate loss, so the holder does not exercise the option and the option contract continues. Within the stop region, the exercise of the option results in a profit, so the holder exercises the option and terminates the contract.

1.1.3 Asian and Exotic Options

Other options are called exotic options. Exotic options are divided into path-dependent and path-independent options. Options that depend on the underlying for a certain period of time are called path-dependent, e.g. Asian options.

Asian options are path-dependent exotic options. For Asian options, the payoff depends on the average of the asset value S over the time interval T, where $A(S)$ refers to the averaging function. The payoff ϕ^{AS} is given by

$$\phi^{\mathrm{As}}\Big(A(S)\Big) = \begin{cases} \max\Big(A(S) - K, 0\Big), & \text{for call,} \\ \max\Big(K - A(S), 0\Big), & \text{for put.} \end{cases}, \tag{1.4}$$

There are different types of Asian options that are characterized by the definition of $A(S)$, e.g. arithmetic or geometric. In the following, we will use the arithmetic-average Asian option with

$$A(S) = \frac{1}{T}\int_0^T S(t)\,\mathrm{d}t$$

and $S(t) > 0$. Note that there are several different ways to define the payoff as well as the averaging function.

Binary options are path-independent options. The execution of the option depends only on a predefined bound K. The value of the option is given by the asset-independent constant B. The payoff-function for a European binary put option is given by

$$\phi^{\mathrm{Bi}}(S(T)) = \begin{cases} B, & \text{for } S(T) < K, \\ 0, & \text{for } S(T) \geq K \end{cases},$$

analogously for call options. Other exotic options can be of European or American type, depending on the possibility of exercising the option. The exotic options listed above are single-asset options because they depend on only one asset value. However, there are several types of options that depend on several different assets, called multi-asset options, e.g. Rainbow or Basket options. Note that there are many other option types that can be used for pricing, for more see [35, 91].

1.2 Financial Models and Stochastic Calculus

The asset is influenced by real circumstances, so one approach is to model the asset behavior by a stochastic process. We discuss the general idea of stochastic processes and the derivation of the corresponding parabolic partial differential equations, e.g. convection-diffusion equations. We start with the definition of a Brownian motion.

Definition 3 (Brownian motion [34])**.** A standard one-dimensional Brownian motion on $[0, T]$ is a stochastic process $\{\mathrm{W}(t), 0 \leq t \leq T\}$ with the following properties.

- $\mathrm{W}(0) = 0$;
- the mapping $t \mapsto \mathrm{W}(t)$ is, with probability 1, a continuous function on $[0, T]$;
- the increments $\{\mathrm{W}(t_1) - \mathrm{W}(t_0), \mathrm{W}(t_2) - \mathrm{W}(t_1), \ldots, W(t_\mathrm{k}) - W(t_{\mathrm{k}-1})\}$ are independent for any k and any $0 \leq t_0 < t_1 < \cdots < t_\mathrm{k} \leq T$;
- $\mathrm{W}(t) - \mathrm{W}(\mathrm{s}) \sim \mathcal{N}(0, t - \mathrm{s})$ for any $0 \leq \mathrm{s} < t \leq T$.

In a d-dimensional setting we get $\mathbf{X} = (\mathrm{X}^1, \mathrm{X}^2, \ldots, \mathrm{X}^d)$, where $\mathbf{X}$ contains the financial model variables. Let X^a denote the variable describing the asset, e.g. in a one-dimensional setting $\mathbf{X} = \mathrm{X}^a = S$. Usually $\mathrm{X}^1 = \mathrm{X}^a$ holds. In general, we can define a *stochastic differential equation* (SDE) for modeling the asset as in Definition 4.

Definition 4 (General SDE Model)**.** Let $\mathbf{X} = (\mathrm{X}^1, \mathrm{X}^2, \ldots, \mathrm{X}^d)$ be a d-dimensional variable and $\mathbf{W} = (\mathrm{W}^1, \mathrm{W}^2, \ldots, \mathrm{W}^d)$ a d-dimensional Brownian motion with given deterministic functions $\boldsymbol{\mu}(\mathbf{X}, t)$, as the drift term $\boldsymbol{\mu}\colon \mathbb{R}^d \times [0,T] \to \mathbb{R}^d$, and $\boldsymbol{\sigma}(\mathbf{X}, t)$ as the diffusion coefficient $\boldsymbol{\sigma}\colon \mathbb{R}^{d\times d} \times [0,T] \to \mathbb{R}^{d\times d}$. Then $\mathbf{X}$ at time t is represented by a general d-dimensional time-homogeneous *stochastic differential equation* (SDE), more precisely an Itô process

$$\mathrm{d}\mathbf{X}_t = \boldsymbol{\mu}(\mathbf{X}_t, t)\,\mathrm{d}t + \boldsymbol{\sigma}(\mathbf{X}_t, t)\,\mathrm{d}\mathbf{W}_t, \tag{1.5}$$

with $\mathbf{X}_0 = \mathbf{X}^{\text{init}} \in \mathbb{R}^d$ given or component-wise

$$\mathrm{d}\mathrm{X}^i_t = \boldsymbol{\mu}_i(\mathbf{X}_t, t)\,\mathrm{d}t + \sum_{j=1}^{d} \boldsymbol{\sigma}_{ij}(\mathbf{X}_t, t)\,\mathrm{d}W^j_t, \quad 1 \le i \le d. \tag{1.6}$$

The drift term $\boldsymbol{\mu}$ and the diffusion term $\boldsymbol{\sigma}$ are given by the underlying model for the asset. To determine the fair price of the option at the current time $t = 0$, we compute

$$\omega(\mathbf{X}, 0) = \exp(-rT)\, E\Big(\Psi(\mathbf{X})\Big), \tag{1.7}$$

where $\mathbf{X}$ is the solution for (1.5) and Ψ is the payoff-function, e.g. (1.1). To derive a partial differential equation (PDE) formulation of the pricing problem, we can either use Itô's Lemma in combination with standard no-arbitrage arguments, or use the Feynmac-Kac formulation [86]. Let $u(\mathbf{x}, t)\colon \mathbb{R}^d \times [0,T] \to \mathbb{R}$ and a $\mathcal{C}^{2,1}$-smooth function with derivatives in $\mathbf{x}$ be bounded, $f\colon \mathbb{R}^d \times [0,T] \to \mathbb{R}$ is bounded and $\Psi(\mathbf{X})$ twice differentiable, as well as $c\colon \mathbb{R}^d \times [0,T] \to \mathbb{R}$ bounded from below [86, 40]. Then u admits the Feynman-Kac representation over the expectation value E

$$\begin{aligned} u(\mathbf{x}, t) = E\bigg[&\int_t^T \exp\Big(-\int_t^{\mathrm{s}} c(\mathbf{X}^{t,\mathbf{x}}_{\mathrm{v}}, \mathrm{v})\mathrm{dv}\Big)\, f(\mathbf{X}^{t,\mathbf{x}}_{\mathrm{s}}, \mathrm{s})\,\mathrm{ds} \\ &+ \exp\bigg(-\int_t^T c(\mathbf{X}^{t,\mathbf{x}}_{\mathrm{v}}, \mathrm{v})\,\mathrm{dv}\bigg)\ \Psi(\mathbf{X}^{t,\mathbf{x}}_T)\bigg] \end{aligned} \tag{1.8}$$

for all $(\mathbf{x}, t) \in \mathbb{R}^d \times [0,T]$ and $\mathbf{X}^{t,\mathbf{x}}_s$ the solution to the SDE. Note that $\mathbf{X}$ is the SDE equivalent representation of $\mathbf{x}$. Then, equation (1.8) represents the SDE (1.5) and is a solution to a linear parabolic PDE, defined in Definition 5.

Definition 5 (Parabolic PDE)**.** A linear PDE

$$\begin{aligned} \frac{\partial}{\partial t} u(\mathbf{x},t) - \mathcal{L}\Big[u\Big](\mathbf{x},t) &= f(\mathbf{x},t) \\ u(\mathbf{x},T) &= \Psi(\mathbf{x}) \end{aligned} \tag{1.9}$$

with the spatial operator $\mathcal{L}\colon \mathcal{C}^2(\Omega) \to \mathcal{C}^0(\Omega)$

$$\mathcal{L}[u](\mathbf{x},t) = \mathcal{A}[u](\mathbf{x},t) - \sum_{i=1}^{d} b_i(\mathbf{x},t) \frac{\partial}{\partial \mathrm{x}_i} u(\mathbf{x},t) + c(\mathbf{x},t) \cdot u(\mathbf{x},t) \tag{1.10}$$

where the spatial operator $\mathcal{A}\colon \mathcal{C}^2(\Omega) \to \mathcal{C}^0(\Omega)$ is defined by

$$\mathcal{A}[u](\mathbf{x},t) = -\frac{1}{2} \sum_{i=1}^{d} \sum_{j=1}^{d} a_{ij}(\mathbf{x},t) \frac{\partial^2}{\partial \mathrm{x}_i \partial \mathrm{x}_j} u(\mathbf{x},t) \tag{1.11}$$

and

$$\begin{aligned} a_{i,j}(\mathbf{x},t) &= \sum_{\mathrm{k}=1}^{d} \boldsymbol{\sigma}_{i\mathrm{k}}(\mathbf{x},t) \boldsymbol{\sigma}_{\mathrm{k}j}(\mathbf{x},t) \\ b_i(\mathbf{x},t) &= \boldsymbol{\mu}_i(\mathbf{x},t) \end{aligned} \tag{1.12}$$

is (uniform) parabolic, if $\mathcal{A}$ is (uniform) elliptic. Given the matrix $\mathrm{A} = (a_{i,j})$ from $\mathcal{A}$, it is called

- *elliptic*, if A is positive definite for each $\mathbf{x} \in \Omega_d$, i.e., it holds

$$\boldsymbol{\varrho}^\top \mathrm{A}(\mathbf{x},t) \boldsymbol{\varrho} > 0 \text{ for all } \boldsymbol{\varrho} \in \mathbb{R}^d. \tag{1.13}$$

- *uniform elliptic*, in $\Omega \subset \mathbb{R}d$, if a constant $\varepsilon > 0$ exists such that

$$\boldsymbol{\varrho}^\top \mathrm{A}(\mathbf{x},t) \boldsymbol{\varrho} \geq \varepsilon ||\boldsymbol{\varrho}||_2^2 \quad \text{for all } \boldsymbol{\varrho} \in \mathbb{R}^d \quad \text{and all } \mathbf{x} \in \Omega. \tag{1.14}$$

If $\mathrm{u}(\mathbf{x},t)$ is a bounded solution to equation (1.9), then $\mathrm{u}(\mathbf{x},t) = u(\mathbf{x},t)$. Thus it has an unique solution [86, 40].

The parabolic PDE can be rewritten into the *divergent form* given by

$$\frac{\partial}{\partial t} u(\mathbf{x},t) - \nabla \cdot \mathbf{A}(\mathbf{x},t) \nabla u(\mathbf{x},t) + \mathbf{b}(\mathbf{x},t) \nabla u(\mathbf{x},t) + c(\mathbf{x},t) u(\mathbf{x},t) = f(\mathbf{x},t) \tag{1.15}$$

For option pricing problems, the payoff is generally non-smooth and not twice differentiable, so smoothing techniques must be used to make it applicable [40]. Under the risk-neutral measure $\mathbb{Q}$, for European options we get $f(\mathbf{x},t) = 0$ and in finance we usually get $c(\mathbf{x},t) = r$. Note that most of the time we are interested in the fair price of a particular point, namely the spot point denoted by $\mathbf{x}^{\text{spot}}$.

Since we obtain a terminal condition at $t = T$, we reverse the time $\tau = T - t \in [0,T]$ to obtain an initial condition. For the payoff-function, we assume without loss of generality, that $\mathrm{x}_\mathrm{a} = s$. Since if the condition is not fulfilled, we can either transform x_a to s or the payoff condition itself, see Section 4.1. As mathematical problem a *Initial Boundary Value Problem (IBVP)* arises and we obtain a definition for IBVPs for a European plain vanilla option problem, see Definition 6.

Definition 6 (IBVP for European plain vanilla options)**.** The d-dimensional option pricing IBVP for European plain vanilla options with $\mathbf{x} = (\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_d)$ is given by

$$\begin{aligned} \frac{\partial}{\partial \tau} u(\mathbf{x},\tau) &= \mathcal{L}\Big[u\Big](\mathbf{x},\tau) \\ u(\mathbf{x},0) &= \phi(s). \end{aligned} \tag{1.16}$$

The ‘spatial’ boundary conditions as well as $\mathcal{L}$ depend on the underlying model.

American options are more expensive than European options because American options give the holder the right to exercise the option before the expiration date T, see Section 1.1.2. To price an American put option u, we look for $\Big(u(\mathbf{x},\tau), \mathbf{x}_f(\tau)\Big)$ such that the Definition 1.18 is satisfied. Since $\mathbf{x}_f$ is a priori unknown, it must also be determined. Let x_a be the spatial variable for the asset, e.g. $\mathrm{x}_\mathrm{a} = S$. To guarantee consistency for $\mathbf{x}_f(\tau)$, we get the condition for x_a

$$\frac{\partial u}{\partial \mathrm{x}_\mathrm{a}}\Big(\mathrm{x}_{\mathrm{a}f}(\tau)\Big) = -1 \tag{1.17}$$

because $u(\mathbf{x},\tau)$ tangentially touches the payoff-function. This is also called a high-contact condition or smooth pasting. The additional ’spatial’ boundary conditions are determined by the underlying model and are the same for American and European options. The contact point $\mathbf{x}_f$ exists for each time step and $\mathbf{x}_f(\tau)$ gives the early exercise curve, separating the holding and stopping regions. The curve is continuously differentiable. For a Put option it is non-decreasing and asymptotically bounded [102, 103]. Mathematically speaking, the contact point is a free boundary value.

Definition 7 (Linear Complementary Problem for American options)**.** The d-dimensional option pricing *Linear Complementary Problem (LCP)* for American options with $\mathbf{x} =$

$(\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_d)$ is given by

$$\begin{aligned}\frac{\partial}{\partial \tau}u(\mathbf{x},\tau) - \mathcal{L}\big[u\big](\mathbf{x},\tau) &\geq 0, \qquad 0 \leq \tau \leq T,\\ u(\mathbf{x},\tau) &\geq \phi(s)\\ \left(\frac{\partial}{\partial \tau}u(\mathbf{x},\tau) - \mathcal{L}\big[u\big](\mathbf{x},\tau)\right)\Big(u(\mathbf{x},\tau) - \phi(s)\Big) &= 0\\ u(\mathbf{x},0) &= \phi(s)\end{aligned} \tag{1.18}$$

where $\phi(s)$ is the initial condition given from the payoff-function of the corresponding European option. The 'spatial' boundary conditions as well as $\mathcal{L}$ depend on the underlying model defining $\mathbf{x}$ as well.

For the one-dimensional case with x_a describing the asset, the penalty term approach is an alternative. In this approach, the inequality is rewritten as an equality due to an additional term, the penalty term. The resulting problems are nonlinear IBVPs. The penalty term forces the free boundary condition to be satisfied asymptotically and should be zero for the continuation region and positive in the stopping region where it penalizes the problem by a factor [93, 108, 67].

Definition 8 (Penalty Term Approach)**.** With the introduction of a penalty term $g\big(u(\mathrm{x}_\mathrm{a},\tau)\big)$, the American option pricing problem is given by

$$\frac{\partial}{\partial \tau}u(\mathrm{x}_\mathrm{a},\tau) - \mathcal{L}\big[u\big](\mathrm{x}_\mathrm{a},\tau) = g\big(u(\mathrm{x}_\mathrm{a},\tau)\big). \tag{1.19}$$

Note that there are other problem representations that refer to other numerical solvers, such as numerical integration [8, 28].

In addition to the determination of the option price itself, the calibration of parameters to real market data is also an area of research. As financial models aim to approximate the real market behavior of various underlyings. Since most model parameters are implicit in the real market data, calibrating the model parameters to the real market data is challenging [20, 48, 71, 79, 81, 96]. Furthermore, the challenge increases with the complexity of the model [97]. A general calibration problem is given in Definition 9.

Definition 9 (Calibration problem)**.** Let $\boldsymbol{\xi}$ denote the vector of the parameters which should be calibrated and $\mathbb{X}$ be the subset of all possible parameter sets, s.t. $\boldsymbol{\xi} \in \mathbb{X}$. The solution of the underlying model with respect to the parameters in $\boldsymbol{\xi}$ is given by $u(\boldsymbol{\xi})$ and

u_{data} represents the given data. The calibration problem is

$$\min_{\xi \in \mathbb{X}} \mathcal{J}(u(\xi), u_{\text{data}}), \tag{1.20}$$

where $\mathcal{J}$ denotes an operator of a cost functional.

1.3 Numerical Methods

In general, analytical solutions are not available for the financial models and therefore numerical methods are needed. Since the financial methods can be rewritten in the generalized form of either the SDE or the PDE representation, see Definitions 4 and 5, the standard numerical methods can be applied to many problems in finance and other research areas. Unfortunately, the methods usually can't be applied directly to financial problems, e.g. the initial conditions for European and Barrier options are only continuous. Therefore, adjustments are made by using grid transformations or smoothing techniques, see Section 4.1. In addition, the boundary conditions are sometimes also adapted to the criteria of the numerical technique, as discussed in Section 4.4. Since we can overcome these shortcomings of the original model, we begin by stating standard numerical methods for solving the mathematical problems of computing a fair price.

Monte Carlo methods are standard numerical solvers for SDEs [34, 35, 91, 96]. These methods are based on the central limit theorem for independent and identically distributed random numbers. Within the simulation, several different random walks are computed, and then the mean of the random walks provides the input for the payoff-function. An advantage is that the simulation is almost independent of the number of dimensions, but it suffers from a low convergence rate when considering one-step discretizations as Euler-Maruyama schemes [34, 35, 91]. Multistep methods, e.g. the Milstein method or even Runge-Kutta stochastic schemes are also available, but more complex to implement, especially for a high number of dimensions [34, 35, 91]. Regardless of whether the mathematical problem is an IBVP or an LCP, the PDEs arising from financial models are convection-diffusion equations [40]. The first approach for temporal discretization is the θ-method [35]. Unfortunately, it results in a discretization matrix with a large bandwidth. To overcome this, we can split the spatial operator, e.g. by using Locally One-Dimensional (LOD) methods, fractional step methods, component-wise splitting schemes as well as alternating direction implicit (ADI) schemes. For a general overview and analysis of such methods we refer to [53, 74]. The ADI schemes are widely used in financial research [38, 57, 61]. In 2004, Ikonen and Toivanen adapted the ADI techniques for LCPs and introduced the

ADI-IT schemes [54, 55, 39]. An alternative approach for solving one-dimensional American option pricing problems is the penalty approach [83, 35, 104, 93, 108, 67]. Due to the restriction to one dimension, the θ-discretization is a feasible discretization for this type of problem [35]. Since the American option pricing problem results in a free boundary value problem, the contact point is a priori unknown and therefore interpreted as "free". These problems have to be solved numerically [19, 83]. Among the methods mentioned above, these schemes are proposed to solve the problem, e.g. the projected SOR scheme [19], the binomial method, front-fixing schemes [83] as well as Monte Carlo simulation techniques [86]. These schemes implicitly compute the free boundary value. Other researchers have focused on an explicit representation of the free boundary value [93, 108, 67].

For the spatial discretization, finite differences and finite element methods are standard [38, 39, 56, 109]. Finally, after temporal and spatial discretization, a system of equations is derived. These systems can be solved by standard numerical methods, e.g. Gauß-algorithm, LU-decomposition, also iterative methods, e.g. the PSOR, Gauß-Seidel-algorithm, can be considered [35]. In the spatial dimension we observe an exponential growth of grid points with increasing dimension, since a complete tensor-based grid contains $\mathcal{O}(N^d)$ grid nodes ('curse of dimensionality'). This shows that the number of equations is directly related to the number of dimensions, and the complexity of the solver is directly related to the bandwidth of the matrices, which depends on the temporal solver and the number of dimensions. Therefore, these approaches are computationally expensive when applied to high-dimensional problems.

In computational finance, we look for new ways to reduce memory and run time as models become more complex, and we strive for higher stability and accuracy as run time and memory increase with the number and dimension of models. We then present advanced and combined methods. For the spatial discretization, high-order schemes can be considered, there are also approaches to reduce the bandwidth of the high-order schemes by introducing high-order compact schemes [22, 23, 24, 41, 45]. Another approach is to combine different types of standard techniques with a hierarchy. A spatial example is the Richardson extrapolation [73, 88]. In our case, we focus on the same hierarchy approach for the spatial and temporal discretizations, namely the multi-grid approach. Within the multi-grid approach, different sets of grids are defined and combined to obtain the desired results. The algorithm chosen for the temporal hierarchical approach is the Parareal developed by Lions, Mayday and Turinici [70]. The Parareal works on a fine and a coarse grid and consists of a parallel computation followed by a serial correction step. Therefore, it can be considered as a multiple shooting method as well. To reduce the number of grid points in space, other grid structures have been developed, such as multigrid methods, within which sparse grids are a special case [5, 10, 31, 84]. We focus on the sparse grid approach using the combination technique to reduce the effects of increasing the dimension. Sparse grids were developed by Smolyak [92] for numerical integration purposes. Later the approach was extended in [5, 89, 90, 107] and Hendricks et al. [41] used the approach for financial applications. Within the sparse grid approach, several sparse grids are computed individually and the solution is given by the combination technique for the sparse grids. By combining the sparse grid combination technique and the parareal, we can apply improvement strategies [12, 13]. One strategy is to also compute the sparse grids in parallel in the serial part of the algorithm. The other idea is to reduce the computational cost by reusing intermediate results from either the fine or coarse solver for the other solver.

Since our goal today is to compute the fair price of an option, and the parameters of the model are directly influenced by the environment, it is necessary to fit the parameters to the real market data. This calibration process is challenging since most of the model parameters are only implicitly contained in the real market data and there are various calibration techniques for financial models available in the literature [20, 48, 71, 79, 81, 96]. In our case we focus on a specific two-dimensional financial model, the Heston model, see Section 4.3. For the Heston model in the setting of constant parameters and for very specific use cases, there are approaches based on the closed-form valuation formula [81, 20]. These are fast and provide information about the global minimum. For more general cases, the stochastic nature of the Heston model allows Monte Carlo optimization methods [96], which can also be used to calibrate the stock price and variance. The Monte Carlo theory is well established, but the approaches are computationally expensive and do not provide information about global or local minima. Recently, calibration approaches using neural networks, deep learning strategies, and parallel GPU implementation of the Heston model have been proposed [48, 71, 72, 68, 30]. The networks must be trained individually for each model, and training requires appropriate data. Again, there is no information about global or local minima. An advantage is that once the neural networks are trained, they can be evaluated quickly. Our calibration algorithm for the Heston model is independent of a specific characteristic function and easily extendable to time-dependent parameters [16]. The core of the algorithm is based on space mapping [2], a new approach in the context of financial research that uses an iterative procedure that minimizes the residuum of a fine and a coarse model. In fact, to calibrate the parameters of the fine model, the coarse model is optimized and the fine model is only evaluated. For the optimization we use techniques from [47, 100] and derive a gradient descent algorithm for the Heston model [14]. The gradient descent algorithms have previously been used only implicitly in the context of neural network approximations for the Heston model [72]. As a calibration problem, we choose an Asian short maturity option problem as the fine model, and since the European option problem is an approximation to it, it provides the coarse model. To our knowledge, there is only one paper dealing with the calibration of Asian options under the Heston model, Khalife et al. [63]. Overall, in this thesis we discuss three different hierarchical approaches, a spatial and a temporal one, both using multigrid representations, as well as a hierarchical calibration approach between two financial problems.

1.4 Outline

The next Chapter 2 deals with the temporal discretization. We introduce the Monte Carlo approach for SDEs and apply the Euler-Maruyama discretization. We then focus on the temporal discretization of IBVPs and LCPs and present the θ-method as a standard introduction to temporal differentiation schemes for PDEs. This is followed by a first standard improvement technique, the splitting of the spatial operator, which leads to the ADI schemes for IBVPs and ADI-IT schemes for LCPs. However, for high-dimensional problems, even for the split schemes, serial computation is slower than parallel computation. Therefore, we introduce our first hierarchical multi-grid approach, the Parareal. The improvement within Parareal is the possibility of partial parallelism of the computation.

Chapter 3 focuses on the spatial discretization for the PDEs. First, two spatial grids are introduced and we present the well-known standard finite difference method for the derivative approximation. In addition to the standard central second-order differences, we introduce the upwind stencil for the diffusion term, since the diffusion term of the Heston model undergoes a sign change. We then introduce the course of dimensionality resulting from standard methods, e.g. finite differences, and a possible curse, namely the sparse grid approach. The sparse grid approach is a multi-grid approach and combines different sparse grids to derive the solution. The combination technique is derived within the approach as well as a discussion of the required error splitting structure. Within this approach, we introduce a grid transformation and add a short discussion about different transformations. Furthermore, we present an additional approach that combines the sparse grid technique and the Parareal to further reduce the computational time.

Before introducing the explicit financial model in Chapter 4, we present techniques that allow the application of the numerical methods introduced earlier. We begin with smoothing techniques, followed by a discussion of grid transformations. The Black-Scholes model, the Heston model, and transformations for the Heston model are then presented. The Black-Scholes model is a one-dimensional SDE model that considers Brownian motion for the asset, and the Heston model is a two-dimensional model that considers volatility as an additional stochastic process. Both models are well known in financial research. The corresponding mathematical problems arising from different option pricing strategies are also presented. For the Black-Scholes model, we introduce a time-dependent penalty term for option pricing. For the Heston model, we discuss European, American and Asian option pricing.

Finally Chapter 5 introduces the last hierarchical approach, the space mapping. As a control problem, we define the calibration problem of the Asian put option as the fine model, while the coarse model is given by the corresponding European put option problem. For both option pricing problems, we consider the Heston model as the underlying financial model for the asset simulation, but in the SDE formulation for the fine model and in the PDE representation for the coarse model. We introduce the general idea of space mapping in the application of the control problem. Since the space mapping approach requires coarse model optimization, in our case a calibration method for the Heston PDE, we derive a gradient descent algorithm. Finally, the chapter 5 ends with the overall space mapping algorithm for the control problem.

Now that all the theory is covered, we can present the numerical results for our various advanced hierarchical methods in the Chapter 6. The results are organized according to the underlying financial model. Thus, the Black-Scholes model is discussed first, followed by the hierarchical approaches with the Heston model. Finally, the thesis ends with a conclusion and an outlook, see Chapter 7. In this chapter a conclusion about the different hierarchical approaches is presented as well as ideas for future topics within these research areas.

Chapter 2

Temporal Discretization

In Chapter 1, we introduced standard problems used in financial modeling, namely SDEs, PDEs, and LCPs. In this section, we will introduce a time discretization and a solver for each type of mathematical problem. We solve the SDEs forward in time using t and the PDEs and LCPs backward in time using $\tau = T - t$. Thus we have the same time interval $t, \tau \in [0, T]$. Furthermore, we use a uniform discretization for both time representations, which is described in the corresponding sections.

2.1 The Monte Carlo Method

We are interested in a fair price for an option whose payoff $\Psi(\mathbf{X})$ depends on the discount factor $\exp(-rT)$ and the expectation value $E\Big(\Psi(\mathbf{X})\Big)$. Therefore, we need to compute the expectation value based on the solution $\mathbf{X}$ of the underlying financial SDE modeled with the corresponding stochastic variables $\mathbf{X}_t = (\mathrm{X}_t^1, \mathrm{X}_t^2 \ldots, \mathrm{X}_t^d)$. We use the Monte Carlo method to derive the expectation value. Within the Monte Carlo method, the expectation value is given by

$$E\Big(\Psi(\mathbf{X})\Big) \approx \Psi(\boldsymbol{X}_P), \tag{2.1}$$

with the Monte Carlo estimate

$$\Psi(\boldsymbol{X}_P) = \frac{\sum_{\mathsf{p}=1}^{P} \Psi(\mathbf{X}^{\mathsf{p}})}{P}, \tag{2.2}$$

where $\mathbf{X}^{\mathsf{p}}$ is a simulated random path of the SDE for $\mathbf{X}$ and P is the number of paths generated. Note, that each path is independent from the other paths. This technique is based on the law of large numbers [34], as it ensures that the estimate converges to the correct value as the number of paths P increases

$$\boldsymbol{X}_P \to E(\mathbf{X}) \text{ with probability 1 as } P \to \infty. \tag{2.3}$$

From the central limit theorem an information about the likely magnitude of the error in the estimate after a finite number of paths is provided. Each path is created using random numbers, where a sequence of $\mathrm{Z}_1, \mathrm{Z}_2, \dots$ is created with the criteria

- for normalization purposes Z_i are uniformly distributed between 0 and 1;
- the Z_i are independent of each other.

The uniformly distributed random variables can be transformed to any other distribution needed, which is usually easier than generating random numbers within this distribution directly. Since the computation of the random number is done by an algorithm, and thus after generating enough random numbers, one can guess the Z_{i+1} from $\mathrm{Z}_1, \dots, \mathrm{Z}_i$ [34]. Thus, the independence of the random variables is directly related to the underlying algorithm used to generate the sequence of numbers. A good algorithm aims for a large i up to which the sequence $\mathrm{Z}_1, \dots, \mathrm{Z}_i$ resembles an independent random number. Therefore, the algorithms for generating random numbers are called *pseudo-random* number generators. Thus, it is possible to use the uniformly distributed random variables, since for this distribution advanced pseudo-random number generators are available. These numbers are used to simulate the Brownian motion. From the definition of the Brownian motion in Definition (3), it follows

$$\mathrm{W}(t) \sim \mathcal{N}(0, t). \tag{2.4}$$

Therefore we have to adjust the random numbers by multiplying $\sqrt{\mathrm{d}t}$ to simulate the desired distribution for the Brownian motion.

For the temporal discretization, we use the well-known Euler-Maruyama scheme [35]. Let $\mathbf{Z}_t = \mathrm{Z}_1, \mathrm{Z}_2 \dots, \mathrm{Z}_d$ be a d-dimensional vector of random numbers generated for the time t, we get

$$\mathrm{d}\mathbf{W}_t \approx \sqrt{\Delta_t}\mathbf{Z}_t \tag{2.5}$$

component wise. Now we discretize the time uniformly with $t_n = n\,\Delta_t$ with $\Delta_t = \frac{T}{N_t}$ with $N_t \in \mathbb{N}$ and $n = 0, 1, \dots, N_t$. The Euler-Maruyama discretization scheme is

$$\mathbf{X}_{n+1} = \mathbf{X}_n + \boldsymbol{\mu}\Big(\mathbf{X}_n, t_n\Big)\,\Delta_t + \boldsymbol{\sigma}\Big(\mathbf{X}_n, t_n\Big)\sqrt{\Delta_t}\mathbf{Z}_n \tag{2.6}$$

where $\mathbf{X}_n$ is the solution of $\mathbf{X}$ at t_n. The Euler scheme has a strong order of convergence of 0.5 and a weak order of convergence of 1. To improve accuracy, one can either increase the number of paths P, but for a reduction of one decimal point one needs ten times the number of paths used before [34, 35]. This behavior is costly in terms of computation. Another approach is to reduce the variance either by separating the main part or by using antithetic variables [34, 35]. We apply variance reduction by introducing a so-called antithetic path $\mathbf{X}^-$

$$\mathbf{X}_{n+1}^- = \mathbf{X}_n^- + \boldsymbol{\mu}\Big(\mathbf{X}_n^-, t_n\Big)\,\Delta_t - \boldsymbol{\sigma}\Big(\mathbf{X}_n^-, t_n\Big)\sqrt{\Delta_t}\mathbf{Z}_n. \tag{2.7}$$

This approach leads to

$$\begin{aligned} \Psi(\boldsymbol{X}_P) &= \frac{\sum_{\mathsf{p}=1}^{P}\Psi(\mathbf{X}^{\mathsf{p}})}{P}, \quad \text{where } \mathbf{X}^{\mathsf{p}} \text{ is simulated by (2.6) and} \\ \Psi(\boldsymbol{X}_P^-) &= \frac{\sum_{\mathsf{p}=1}^{P}\Psi(\mathbf{X}^{\mathsf{p}-})}{P}, \quad \text{where } \mathbf{X}^{\mathsf{p}-} \text{ is simulated by (2.7).} \end{aligned} \tag{2.8}$$

Then the payoff-function is then given using the antithetic variable

$$\Psi^{\mp}(\boldsymbol{X}_P, \boldsymbol{X}_P^-) = \frac{1}{2}\Big(\Psi(\mathbf{X}_P) + \Psi(\mathbf{X}_P^-)\Big), \tag{2.9}$$

and the fair price of the option is given by

$$\omega^{\mp}\Big(\boldsymbol{X}_P, \boldsymbol{X}_P^-, 0) = \exp(-rT)\cdot\Psi^{\mp}(\boldsymbol{X}_P, \boldsymbol{X}_P^-). \tag{2.10}$$

Note that there are higher order schemes, such as the Milstein scheme [34] or stochastic Runge-Kutta schemes [35].

2.2 Temporal Discretization for Mathematical Problems with PDEs

Under the SDE representation, we learned about IBVPs and LCPs, both of which contain a parabolic PDE. Within the PDE we use the reversed time τ and also the uniform discretization with $\tau_n = n\,\Delta_\tau$ with $\Delta_\tau = \frac{T}{N_\tau}$ with $N_\tau \in \mathbb{N}$ and $n = 0, \ldots, N_\tau$. Note that for some numerical treatment $N_\tau = N_t$ holds. We define $u(\mathbf{x}, \tau_n)$ as the solution at time step τ_n.

2.2.1 Methods for IBVPs

For IBVPs resulting from the penalty approach, we obtain the general form

$$\frac{\partial}{\partial\tau}u(\mathrm{x_a},\tau_n) - \mathcal{L}\Big[u\Big](\mathrm{x_a},\tau_n) = g\Big(u(\mathrm{x_a},\tau)\Big) \quad \tau > 0, \tag{2.11}$$

supplied with appropriate initial and boundary data. The next step is to choose an appropriate time discretization method, we start with the θ-discretization. It is given by

$$u(\mathrm{x_a},\tau_{n+1}) = u(\mathrm{x_a},\tau_n)+(1-\theta)\Delta_\tau\mathcal{L}[u](\mathrm{x_a},\tau_n)+\theta\Delta_\tau\mathcal{L}[u](\mathrm{x_a},\tau_{n+1})+\Delta_\tau g\Big(u(\mathrm{x_a},\tau_n)\Big), \tag{2.12}$$

where $\theta > 0$ denotes the implicitness of the scheme. Depending on the choice of θ, different standard schemes can be derived. For $\theta = 0$, we get the explicit Euler scheme and for $\theta = 1$ the implicit Euler scheme. Both Euler schemes exhibit first order accuracy in time. The second order accuracy scheme, the Crank-Nicolson scheme, is given by $\theta = 0.5$. For the European option cases, the θ-method can also be used; to adapt the method, the right-hand term has to be set to zero. Unfortunately, for high-dimensional problems, the computational cost increases significantly due to the large bandwidth of the corresponding discretization matrices. To reduce the bandwidth, we apply a splitting technique for the spatial operator, namely the *Alternating Direction Implicit (ADI)* schemes. As those schemes are only applied to IBVPS arising from European option pricing where the right hand side is zero, the general $d-$dimensional IVBP reduces to

$$\frac{\partial}{\partial\tau}u(\mathbf{x},\tau_n) = \mathcal{L}\Big[u\Big](\mathbf{x},\tau_n) \quad \tau_n > 0, \tag{2.13}$$

with the splitting of the operator $\mathcal{L}$ given by

$$\mathcal{L}\Big[u\Big](\mathbf{x},\tau_n) = \mathcal{L}_0\Big[u\Big](\mathbf{x},\tau_n) + \mathcal{L}_1\Big[u\Big](\mathbf{x},\tau_n) + \ldots + \mathcal{L}_d\Big[u\Big](\mathbf{x},\tau_n), \tag{2.14}$$

with

$$\begin{aligned} \mathcal{L}_0[u](\mathbf{x},\tau_n) &= \frac{1}{2}\sum_{i=1}^{d}\sum_{\substack{j=1\\ j\neq i}}^{d} a_{ij}(\mathbf{x},\tau_n)\frac{\partial^2}{\partial \mathrm{x}_i\partial \mathrm{x}_j}u(\mathbf{x},\tau_n) \\ \mathcal{L}_i[u](\mathbf{x},\tau_n) &= \frac{1}{2}a_{ii}(\mathbf{x},\tau_n,)\frac{\partial^2}{\partial \mathrm{x}_i^2}u(\mathbf{x},\tau_n) + b_i(\mathbf{x},\tau_n)\frac{\partial}{\partial \mathrm{x}_i}u(\mathbf{x},\tau_n) + \frac{1}{d}c(\mathbf{x},\tau_n)u(\mathbf{x},\tau_n), \end{aligned} \tag{2.15}$$

for $i = 1,\ldots,d$. The four well-known ADI schemes are the Douglas (DO) scheme (2.16), the Craig-Sneyd (CS) scheme (2.17), the modified Craig-Sneyed (mCS) scheme (2.18), and the Hundsdorfer-Verwer (HV) scheme (2.19).

Douglas (DO) scheme

$$\left\{\begin{array}{rclccl} Y_0 &=& u(\mathbf{x},\tau_n) &+& & \Delta_\tau\,\mathcal{L}\Big[u\Big](\mathbf{x},\tau_n),\\ Y_i &=& Y_{i-1} &+& \theta & \Delta_\tau\Big(\mathcal{L}_i\Big[Y_i\Big]-\mathcal{L}_i\Big[u\Big](\mathbf{x},\tau_n)\Big), \quad \text{for } i=1,\dots,d\\ u(\mathbf{x},\tau_{n+1}) &=& Y_d. &&& \end{array}\right. \tag{2.16}$$

Craig-Sneyed (CS) scheme

$$\left\{\begin{array}{rclccl} Y_0 &=& u(\mathbf{x},\tau_n) &+& & \Delta_\tau\,\mathcal{L}\Big[u\Big](\mathbf{x},\tau_n),\\ Y_i &=& Y_{i-1} &+& \theta & \Delta_\tau\Big(\mathcal{L}_i\Big[Y_i\Big]-\mathcal{L}_i\Big[u\Big](\mathbf{x},\tau_n)\Big), \quad \text{for } i=1,\dots,d,\\ \tilde{Y}_0 &=& Y_0 &+& \frac{1}{2} & \Delta_\tau\Big(\mathcal{L}_0\Big[Y_d\Big]-\mathcal{L}_0\Big[u\Big](\mathbf{x},\tau_n)\Big),\\ \tilde{Y}_i &=& \tilde{Y}_{i-1} &+& \theta & \Delta_\tau\Big(\mathcal{L}_i\Big[\tilde{Y}_i\Big]-\mathcal{L}_i\Big[u\Big](\mathbf{x},\tau_n)\Big), \quad \text{for } i=1,\dots,d,\\ u(\mathbf{x},\tau_{n+1}) &=& \tilde{Y}_d. &&& \end{array}\right. \tag{2.17}$$

modified Craig-Sneyed (mCS) scheme

$$\left\{\begin{array}{rclccl} Y_0 &=& u(\mathbf{x},\tau_n) &+& & \Delta_\tau\,\mathcal{L}\Big[u\Big](\mathbf{x},\tau_n),\\ Y_i &=& Y_{i-1} &+& \theta & \Delta_\tau\Big(\mathcal{L}_i\Big[Y_i\Big]-\mathcal{L}_i\Big[u\Big](\mathbf{x},\tau_n)\Big), \quad \text{for } i=1,\dots,d,\\ \hat{Y}_0 &=& Y_0 &+& \theta & \Delta_\tau\Big(\mathcal{L}_0\Big[Y_d\Big]-\mathcal{L}_0\Big[u\Big](\mathbf{x},\tau_n)\Big),\\ \tilde{Y}_0 &=& \hat{Y}_0 &+& (\frac{1}{2}-\theta) & \Delta_\tau\Big(\mathcal{L}\Big[Y_d\Big]-\mathcal{L}\Big[u\Big](\mathbf{x},\tau_n)\Big),\\ \tilde{Y}_i &=& \tilde{Y}_{i-1} &+& \theta & \Delta_\tau\Big(\mathcal{L}_i\Big[\tilde{Y}_i\Big]-\mathcal{L}_i\Big[u\Big](\mathbf{x},\tau_n)\Big), \quad \text{for } i=1,\dots,d,\\ u(\mathbf{x},\tau_{n+1}) &=& \tilde{Y}_d. &&& \end{array}\right. \tag{2.18}$$

Hundsdorfer-Verwer (HV) scheme

$$\left\{\begin{array}{rclccl} Y_0 &=& u(\mathbf{x},\tau_n) &+& & \Delta_\tau\,\mathcal{L}\Big[u\Big](\mathbf{x},\tau_n),\\ Y_i &=& Y_{i-1} &+& \theta & \Delta_\tau\Big(\mathcal{L}_i\Big[Y_i\Big]-\mathcal{L}_i\Big[u\Big](\mathbf{x},\tau_n)\Big), \quad \text{for } i=1,\dots,d,\\ \tilde{Y}_0 &=& Y_0 &+& \frac{1}{2} & \Delta_\tau\Big(\mathcal{L}_0\Big[Y_d\Big]-\mathcal{L}_0\Big[u\Big](\mathbf{x},\tau_n)\Big),\\ \tilde{Y}_i &=& \tilde{Y}_{i-1} &+& \theta & \Delta_\tau\Big(\mathcal{L}_i\Big[\tilde{Y}_i\Big]-\mathcal{L}_i\Big[Y_d\Big]\Big), \qquad \text{for } i=1,\dots,d,\\ u(\mathbf{x},\tau_{n+1}) &=& \tilde{Y}_d. &&& \end{array}\right. \tag{2.19}$$

In the DO schemes, a forward Euler predictor step is followed by d implicit but unidirectional corrector steps that serve to stabilize the predictor step. Since the first step is explicit, the bandwidth of the matrix is secondary to the dimension of the problem itself, with respect to (w.r.t.) the complexity. Due to the unidirectional structure, the bandwidth is reduced for an implicit step and thus the complexity of the system of equations is reduced. Since the first two lines of all ADI models are equivalent, one can consider the DO scheme as the basic scheme and the CS, mCS, and HV as extensions to it. These extensions have the similarity of a second predictor step followed by d unidirectional correction steps. If no mixed derivatives are considered, e.g. because the spatial directions are uncorrelated, the CS scheme is reduced to the DO scheme. The advanced ADI schemes differ in the choice of the second predictor and the correction step. More precisely, the CS scheme uses the same structure as the DO scheme, but only twice. In the mCS scheme, the second predictor step actually contains two successive correction steps. For $\theta = 0.5$ these steps are combined into the second correction step for the CS scheme and thus in this and only in this case the mCS and CS schemes are the same. Finally, the HV scheme considers the second unidirectional steps w.r.t. an intermediate result instead of the solution of the last step. Since the mixed derivatives have a higher bandwidth, they are always considered explicitly.

The order of consistency differs between the schemes. While the DO scheme is only consistent of order one for any θ, the mCS and HV schemes are consistent of order two, as is the CS scheme if and only if $\theta = 0.5$. The low order consistency for the DO scheme results from the explicit and singular treatment of the mixed derivative term. The stability of the ADI schemes has been treated in [57, 61, 62, 58] in the von Neumann sense. The underlying problems were multidimensional convection-diffusion problems with mixed derivative terms. The results show a stability that is independent of the time step Δ_τ but dependent on the choice of θ. Depending on the number of spatial dimensions and the spatial dimensions themselves, a lower bound for the unconditional stability can be derived. Lower bounds for θ for the unconditional stability of the different ADI schemes based on the theoretical results and numerical experiments in [38, 37, 56] are given in Table 2.1.

Historically, McKee and Mitchell generalized the original ADI scheme for two-dimensional diffusion equations in [21, 85] first to diffusion and later to convection-diffusion equations with mixed derivatives and derived the DO scheme [77, 78]. To obtain a stable second-order ADI scheme with mixed derivatives, Craig and Sneyd developed the CS scheme [18]. In't Hout and Welfert [57] constructed the second order mCS scheme to obtain more freedom in the choice of θ compared to the CS scheme. The HV scheme was designed

Scheme	$d = 2$	$d = 3$
DO	0.5	$\frac{2}{3}$
CS	0.5	0.5
mCS	$\frac{1}{3}$	$\max\left\{\frac{1}{3}, \frac{2}{13}(2\max\{\rho\}+1)\right\}$
HV	$\frac{1}{2}+\frac{1}{6}\sqrt{3}$	$\frac{1}{2}+\frac{1}{6}\sqrt{3}$

Table 2.1: Lower stability bounds for θ, with $\max\{\rho\}$ as the maximum of the correlation values between the spatial dimensions.

by Hundsdorfer [52, 53] and Verwer et al. [101] for the numerical solution of convection-diffusion-reaction equations without mixed derivative terms. The integration of mixed derivative terms was initiated by [57, 59]. Note that the intelligent implementation of the ADI schemes [95] already reduces the runtime further.

2.2.2 Temporal Discretization for LCPs

For $0 < \tau_n < T$, the solution $u(\mathbf{x}, \tau_n)$ of the semi-discrete *Partial Differential Complementary Problem (PDCP)* resulting from the American option pricing problem gives an approximation for $u(\mathbf{x}, \tau_n)$:

$$\begin{cases} \frac{\partial}{\partial\tau}u(\mathbf{x},\tau_n) - \mathcal{L}\big[u\big](\mathbf{x},\tau_n) & \geq 0, \\ u(\mathbf{x},\tau) & \geq \phi(s), \\ \Big(\frac{\partial}{\partial\tau}u(\mathbf{x},\tau_n) - \mathcal{L}\big[u\big](\mathbf{x},\tau_n)\Big)\Big(u(\mathbf{x},\tau_n) - \phi(s)\Big) & = 0, \\ u(\mathbf{x},0) & = \phi(s), \end{cases} \tag{2.20}$$

for $n = 0, \dots, N_\tau$. We use the Ikonen-Toivanen (IT) splitting technique to approach the problem considering a two-step system [54, 55, 60]. In the first step we solve a system of linear equations and in the second step a variable update is performed.

In the first step we solve the ODE equation with the additional Lagrangian multiplier $\boldsymbol{\lambda} \in \mathbb{R}^d$ using the alternating direction implicit schemes (2.16)–(2.19) for the time discretization [56]. Depending on the chosen time discretization, one obtains to solve several linear systems. In our case, one of the ADI schemes presented above, where the first line is extended with an additional term for the Lagrangian multiplier. Since the first equation of the ADI schemes is similar, the following equation holds for all schemes

$$Y_0 = u(\mathbf{x},\tau_n) + \Delta_\tau\mathcal{L}\big[u\big](\mathbf{x},\tau_n)\,\boxed{+\Delta_\tau\boldsymbol{\lambda}_n}, \tag{2.21}$$

and the rest of the schemes remains the same. The second fractional step updates the $\boldsymbol{\lambda}$ and $u(\mathbf{x},\tau_n)$ so that they satisfy the constraints. The variable update can be done component-wise by using

$$\begin{cases} u(\mathbf{x},\tau_{n+1}) & = \max\Big(u(\mathbf{x},0) \quad , \tilde{u}(\mathbf{x},\tau_{n+1}) - \Delta_\tau \boldsymbol{\lambda}_n\Big), \\ \boldsymbol{\lambda}_{n+1} & = \max\Big(0 \quad , \boldsymbol{\lambda}_n + \frac{u(\mathbf{x},0)-\tilde{u}(\mathbf{x},\tau_{n+1})}{\Delta_\tau}\Big), \end{cases} \tag{2.22}$$

where $\tilde{u}(\mathbf{x},\tau_{n+1})$ is an intermediate solution; the solution of the ADI scheme with equation (2.21). If an initial condition is given, $\boldsymbol{\lambda}_0$ is set as the zero vector. Finally, we can combine both steps and obtain the overall scheme ADI-IT schemes. The DO-IT scheme is given in (2.23), the CS-IT scheme in (2.24), and the mCS-IT scheme in (2.25), and finally the HV-IT scheme is given by (2.26).

DO-IT scheme

$$\begin{cases} \begin{cases} Y_0 & = u(\mathbf{x},\tau_n) + \Delta_\tau \mathcal{L}\Big[u\Big](\mathbf{x},\tau_n) \boxed{+\Delta_\tau \boldsymbol{\lambda}_n}, \\ Y_i & = Y_{i-1} + \theta\, \Delta_\tau \Big(\mathcal{L}_i\Big[Y_i\Big] - \mathcal{L}_i\Big[u\Big](\mathbf{x},\tau_n)\Big), \quad \text{for } i = 1,\dots,d \\ \tilde{u}(\mathbf{x},\tau_{n+1}) & = Y_d \end{cases} \\ \begin{cases} u(\mathbf{x},\tau_{n+1}) & = \max\Big(u(\mathbf{x},0) \quad , \tilde{u}(\mathbf{x},\tau_{n+1}) - \Delta_\tau \boldsymbol{\lambda}_n\Big), \\ \boldsymbol{\lambda}_{n+1} & = \max\Big(0 \quad , \boldsymbol{\lambda}_n + \frac{u(\mathbf{x},0)-\tilde{u}(\mathbf{x},\tau_{n+1})}{\Delta_\tau}\Big), \end{cases} \end{cases} \tag{2.23}$$

CS-IT scheme

$$\begin{cases} \begin{cases} Y_0 & = u(\mathbf{x},\tau_n) + \Delta_\tau \mathcal{L}\Big[u\Big](\mathbf{x},\tau_n) \boxed{+\Delta_\tau \boldsymbol{\lambda}_n}, \\ Y_i & = Y_{i-1} + \theta\, \Delta_\tau \Big(\mathcal{L}_i\Big[Y_i\Big] - \mathcal{L}_i\Big[u\Big](\mathbf{x},\tau_n)\Big), \quad \text{for } i = 1,\dots,d, \\ \tilde{Y}_0 & = Y_0 + \frac{1}{2}\, \Delta_\tau \Big(\mathcal{L}_0\Big[Y_d\Big] - \mathcal{L}_0\Big[u\Big](\mathbf{x},\tau_n)\Big), \\ \tilde{Y}_i & = \tilde{Y}_{i-1} + \theta\, \Delta_\tau \Big(\mathcal{L}_i\Big[\tilde{Y}_i\Big]) - \mathcal{L}_i\Big[u\Big](\mathbf{x},\tau_n)\Big), \quad \text{for } i = 1,\dots,d, \\ \tilde{u}(\mathbf{x},\tau_{n+1}) & = \tilde{Y}_d \end{cases} \\ \begin{cases} u(\mathbf{x},\tau_{n+1}) & = \max\Big(u(\mathbf{x},0) \quad , \tilde{u}(\mathbf{x},\tau_{n+1}) - \Delta_\tau \boldsymbol{\lambda}_n\Big), \\ \boldsymbol{\lambda}_{n+1} & = \max\Big(0 \quad , \boldsymbol{\lambda}_n + \frac{u(\mathbf{x},0)-\tilde{u}(\mathbf{x},\tau_{n+1})}{\Delta_\tau}\Big), \end{cases} \end{cases} \tag{2.24}$$

mCS-IT scheme

$$
\left\{
\begin{array}{l}
\left\{
\begin{array}{lllll}
Y_0 &= u(\mathbf{x},\tau_n) &+ & \Delta_\tau \mathcal{L}\Big[u\Big](\mathbf{x},\tau_n)\,\boxed{+\Delta_\tau \boldsymbol{\lambda}_n}, & \\
Y_i &= Y_{i-1} &+ \ \theta & \Delta_\tau \Big(\mathcal{L}_i\Big[Y_i\Big] - \mathcal{L}_i\Big[u\Big](\mathbf{x},\tau_n)\Big), & \text{for } i = 1,\dots,d, \\
\hat{Y}_0 &= Y_0 &+ \ \theta & \Delta_\tau \Big(\mathcal{L}_0\Big[Y_d\Big] - \mathcal{L}_0\Big[u\Big](\mathbf{x},\tau_n)\Big), & \\
\tilde{Y}_0 &= \hat{Y}_0 &+ \ (\frac{1}{2}-\theta) & \Delta_\tau \Big(\mathcal{L}\Big[Y_d\Big] - \mathcal{L}\Big[u\Big](\mathbf{x},\tau_n)\Big), & \\
\tilde{Y}_i &= \tilde{Y}_{i-1} &+ \ \theta & \Delta_\tau \Big(\mathcal{L}_i\Big[\tilde{Y}_i\Big] - \mathcal{L}_i\Big[u\Big](\mathbf{x},\tau_n)\Big), & \text{for } i = 1,\dots,d, \\
\tilde{u}(\mathbf{x},\tau_{n+1}) &= \tilde{Y}_d & & &
\end{array}
\right. \\
\left\{
\begin{array}{llll}
u(\mathbf{x},\tau_{n+1}) &= \max & \Big(u(\mathbf{x},0) & , \tilde{u}(\mathbf{x},\tau_{n+1}) - \Delta_\tau \boldsymbol{\lambda}_n\Big), \\
\boldsymbol{\lambda}_{n+1} &= \max & \Big(0 & , \boldsymbol{\lambda}_n + \frac{u(\mathbf{x},0)-\tilde{u}(\mathbf{x},\tau_{n+1})}{\Delta_\tau}\Big),
\end{array}
\right.
\end{array}
\right.
\tag{2.25}
$$

HV-IT scheme

$$
\left\{
\begin{array}{l}
\left\{
\begin{array}{lllll}
Y_0 &= u(\mathbf{x},\tau_n) &+ & \Delta_\tau \mathcal{L}\Big[u\Big](\mathbf{x},\tau_n)\,\boxed{+\Delta_\tau \boldsymbol{\lambda}_n}, & \\
Y_i &= Y_{i-1} &+ \ \theta & \Delta_\tau \Big(\mathcal{L}_i\Big[Y_i\Big] - \mathcal{L}_i\Big[u\Big](\mathbf{x},\tau_n)\Big], & \text{for } i = 1,\dots,d, \\
\tilde{Y}_0 &= Y_0 &+ \ \frac{1}{2} & \Delta_\tau \Big(\mathcal{L}_0\Big[Y_d\Big] - \mathcal{L}_0\Big[u\Big](\mathbf{x},\tau_n)\Big), & \\
\tilde{Y}_i &= \tilde{Y}_{i-1} &+ \ \theta & \Delta_\tau \Big(\mathcal{L}_i\Big[\tilde{Y}_i\Big] - \mathcal{L}_i\Big[Y_d\Big]\Big), & \text{for } i = 1,\dots,d, \\
\tilde{u}(\mathbf{x},\tau_{n+1}) &= \tilde{Y}_d & & &
\end{array}
\right. \\
\left\{
\begin{array}{llll}
u(\mathbf{x},\tau_{n+1}) &= \max & \Big(u(\mathbf{x},0) & , \tilde{u}(\mathbf{x},\tau_{n+1}) - \Delta_\tau \boldsymbol{\lambda}_n\Big), \\
\boldsymbol{\lambda}_{n+1} &= \max & \Big(0 & , \boldsymbol{\lambda}_n + \frac{u(\mathbf{x},0)-\tilde{u}(\mathbf{x},\tau_{n+1})}{\Delta_\tau}\Big),
\end{array}
\right.
\end{array}
\right.
\tag{2.26}
$$

The computational cost of the ADI and ADI-IT schemes are almost similar, since the additional part in the ADI-IT schemes can be done in an explicit and even parallel way [60].

2.3 The Parareal Approach

The Parareal was developed by Lions, Maday, and Turinici and can be viewed as either a multigrid or a multiple shooting method [70]. It is an iterative parallel-in-time algorithm using two temporal operators, where the operator $\mathcal{F}$ represents a fine solver running in parallel over several time slices and the operator $\mathcal{G}$ is used to represent a coarse solver performing a serial update and correction step. Both solvers are assumed to be convergent and stable for the chosen step size. Within the parareal, $\mathcal{G}$ is the bottleneck for speedup and convergence rate, since this solver is of lower order than $\mathcal{F}$.

Below the fine and coarse solvers, this advanced technique works on two temporal grids, as in the Definition 10.

Definition 10 (Parareal grid)**.** We consider the time domain $[0, T]$ in a continuous setting, $(\tau, \boldsymbol{\tau}) \in [0, T]$, where τ gives the point w.r.t. the coarse time grid and $\boldsymbol{\tau}$ for the fine time grid. Using the multi-indices

$$\begin{aligned} \mathbf{N} &= (N_\tau, \mathrm{N}_\tau) \in \mathbb{N}^2, \\ \mathbf{n} &= (n, \mathrm{n}) \in \mathbb{N}^2, \end{aligned} \tag{2.27}$$

with $n = 0, \dots, N_\tau$ and $\mathrm{n} = 0, \dots, \mathrm{N}_\tau$. We can define a grid on $[0, T]$ with

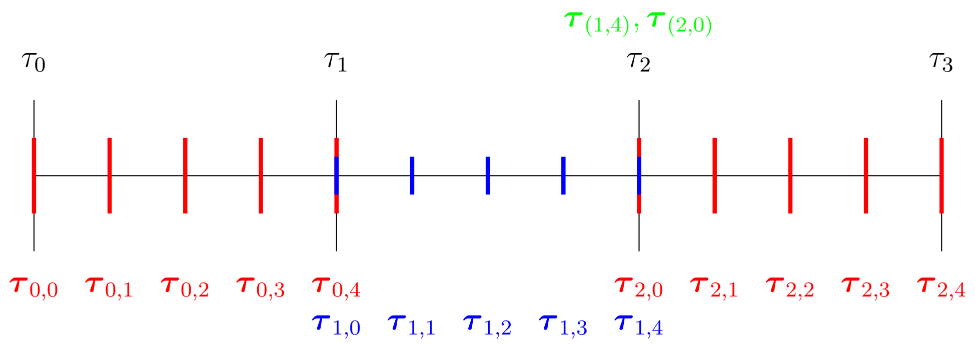
$$\boldsymbol{\Delta} = \left(\Delta_\tau, \frac{\Delta_\tau}{\mathrm{N}_\tau} \right). \tag{2.28}$$

With N_τ and n we divide the interval into N_τ equal slices, defining the coarse grid. Within each slice $[\tau_n, \tau_{n+1}]$ we define a fine grid with N_τ, n. We obtain

$$\boldsymbol{\tau}_{\mathbf{n}} = \tau_n + \mathrm{n} \cdot \frac{\Delta_\tau}{\mathrm{N}_\tau} = \Delta_\tau (n + \frac{\mathrm{n}}{\mathrm{N}_\tau}) \tag{2.29}$$

for the fine grid. So n denotes the grid point within the coarse grid from which the interval starts and n gives the grid node within the interval.

Example: With $\mathbf{N} = (3, 4)$ we observe three coarse intervals each with five fine grid points, where the first $\boldsymbol{\tau}_{(0,0)}$ and $\boldsymbol{\tau}_{(0,4)}$ overlap with τ_0 and τ_1, the interval boundary. There are also points defined twice, e.g. $\boldsymbol{\tau}_{(0,4)}$ and $\boldsymbol{\tau}_{(1,0)}$. This example is shown in Figure 2.1. We will see later that the value can be different for both values.



$\tau_{(1,4)}, \tau_{(2,0)}$
τ_0
τ_1
τ_2
τ_3
$\tau_{0,0}$ $\tau_{0,1}$ $\tau_{0,2}$ $\tau_{0,3}$ $\tau_{0,4}$
$\tau_{2,0}$ $\tau_{2,1}$ $\tau_{2,2}$ $\tau_{2,3}$ $\tau_{2,4}$
$\tau_{1,0}$ $\tau_{1,1}$ $\tau_{1,2}$ $\tau_{1,3}$ $\tau_{1,4}$

Figure 2.1: The discrete temporal grid for the Parareal with $\mathbf{N} = (3, 4)$.

The number of time steps in each slice is defined by the corresponding temporal solver, $N_{\mathcal{G}}$ and $N_{\mathcal{F}}$ respectively. Let U_n^k denote the semi-discrete solution at the k-th iteration and the $n-$th time step for $u(\mathbf{x}, \tau_n)$. Since we have an initial condition, we set $U_0^k = u(\mathbf{x}, 0)$ and compute initial values for each time slice using the coarse solver, s.t. $U_{n+1}^0 = \mathcal{G}(U_n^0, \tau_n, \tau_{n+1})$. Now the iterative procedure starts with a parallel computation of the solution of each time slice with the fine solver $\tilde{U}_{n+1}^k = \mathcal{F}(U_n^k, \tau_n, \tau_{n+1})$. The parallel computation is followed by a serial correction update step with the coarse solver $U_n^{k+1} = \mathcal{G}(U_n^{k+1}, \tau_n, \tau_{n+1}) + \tilde{U}_{n+1}^k - \hat{U}_{n+1}^k$, where $\hat{U}_{n+1}^k = \mathcal{G}(U_n^k, \tau_n, \tau_{n+1})$ is given from the last iteration. The short formulation of the iterative procedure is given by

$$\begin{aligned} U_{n+1}^{k+1} &= \mathcal{G}(U_n^{k+1}, \tau_n, \tau_{n+1}) + \mathcal{F}(U_n^k, \tau_n, \tau_{n+1}) - \mathcal{G}(U_n^k, \tau_n, \tau_{n+1}) \\ &= \underbrace{\hat{U}_n^{k+1}}_{\text{serial update}} + \underbrace{\tilde{U}_n^k}_{\text{computed in parallel}} + \underbrace{\hat{U}_n^k}_{\text{given from the last iteration}} . \end{aligned} \tag{2.30}$$

Algorithm 1 visualizes the Parareal. Note that in the k-th iteration, the k-th interval is solved similar to a serial calculation. Thus at least after N_τ iterations the same accuracy of the serial computation is reached. Therefore $k^{\max} \leq N_\tau$ holds and $k^{\max} \ll N_\tau$ should be assumed. This allows us to restrict the iterative procedure to work only on the intervals $k \leq n \leq N_\tau - 1$. Besides a termination by the iteration count, one can also consider implementing an accuracy restriction.

Now we look at the speedup. In the analysis we neglect the initialization time and compare the results with the theoretical results about the speedup of Parareal without communication costs [33]. Following the approach of Minion [82]. Let $c_{\mathcal{F}}$ and $c_{\mathcal{G}}$ be the computation time for one time step for the fine and coarse solvers. The number of time steps for each interval $[\tau_n, \tau_{n+1}]$ for the different solvers is denoted by $N_{\mathcal{F}}$ and $N_{\mathcal{G}}$ and $N_{\text{Serial}} = N_\tau \cdot N_{\mathcal{F}}$ represents the number of serial time steps needed for a serial computation

Algorithm 1: The Parareal.

```
*Initialize the first time value by the initial condition for each iteration*
for $k = 0 : k^{\max}$ do
    $U_0^k = u(\mathbf{x}, 0)$
end
*Compute initial values for each time interval*
for $n = 1 : N_\tau - 1$ do
    $U_{n+1}^0 = \mathcal{G}(U_n^0, \tau_n, \tau_{n+1})$
end
*Iterative procedure of the Parareal*
k=0
while $k < k^{\max}$ do
    *Parallel Approximation*
    for $n = k : N_\tau - 1$ do
        $\tilde{U}_{n+1}^k = \mathcal{F}(U_n^k, \tau_n, \tau_{n+1})$
    end
    *Serial Update*
    for $n = k : N_\tau - 1$ do
        $\hat{U}_{n+1}^{k+1} = \mathcal{G}(U_n^k, \tau_n, \tau_{n+1})$
        $U_{n+1}^{k+1} = \hat{U}_{n+1}^{k+1} + \tilde{U}_{n+1}^k - \hat{U}_{n+1}^k$
    end
    $k = k + 1$
end
```

with the same accuracy. We get the runtime for the serial computation of

$$\mathrm{C}_{\text{Serial}} = N_{\text{Serial}} \cdot \mathrm{c}_{\mathcal{F}}. \tag{2.31}$$

Now we determine the computation time of the Parareal. In a first step, the intermediate approximations of τ_n are computed by the coarse solver, resulting in a computation time of $N_{\mathcal{G}} \cdot \mathrm{c}_{\mathcal{G}}$ for each of the N_τ time slices, then the iteration starts. One iteration of the Parareal consists of one run of the fine solver in parallel with a serial time of $N_\tau \cdot N_{\mathcal{F}} \cdot \mathrm{c}_{\mathcal{F}}$ distributed over $N_P \in \mathbb{N}$ processors and a serial run of the coarse solver. We obtain the following pattern for the computation time

$$\begin{aligned}
k = 0:&\quad N_\tau \cdot N_{\mathcal{G}}\mathrm{c}_{\mathcal{G}}\\
k = 1:&\quad N_\tau \cdot N_{\mathcal{G}}\mathrm{c}_{\mathcal{G}} + k \cdot \frac{N_\tau N_{\mathcal{F}}}{N_P}\mathrm{c}_{\mathcal{F}}\\
k = 2:&\quad (N_\tau - 1) \cdot N_{\mathcal{G}}\mathrm{c}_{\mathcal{G}} + k \cdot \frac{(N_\tau - 1)N_{\mathcal{F}}}{N_P}\mathrm{c}_{\mathcal{F}}\\
k = 3:&(N_\tau - 2) \cdot N_{\mathcal{G}}\mathrm{c}_{\mathcal{G}} + k \cdot \frac{(N_\tau - 2)N_{\mathcal{F}}}{N_P}\mathrm{c}_{\mathcal{F}}
\end{aligned} \tag{2.32}$$

Assuming we have k iterations, we can compute the computation time of the Parareal

$$\begin{aligned}
\mathrm{C}_{\text{Parareal}} &= (k+1) \cdot N_\tau \cdot N_{\mathcal{G}}\mathrm{c}_{\mathcal{G}} + k \cdot \frac{N_\tau N_{\mathcal{F}}}{N_P}\mathrm{c}_{\mathcal{F}} - \sum_{i=1}^{k}(k-1)\Big(N_{\mathcal{G}}\mathrm{c}_{\mathcal{G}} + \frac{N_{\mathcal{F}}}{N_P}\mathrm{c}_{\mathcal{F}}\Big)\\
&= (k+1) \cdot N_\tau \cdot N_{\mathcal{G}}\mathrm{c}_{\mathcal{G}} + k \cdot \frac{N_{\text{Serial}}}{N_P}\mathrm{c}_{\mathcal{F}} - \frac{k(k-1)}{2}\Big(N_{\mathcal{G}}\mathrm{c}_{\mathcal{G}} + \frac{N_{\mathcal{F}}}{N_P}\mathrm{c}_{\mathcal{F}}\Big)\\
&= \Big((k+1)N_\tau - \frac{k(k-1)}{2}\Big) \cdot N_{\mathcal{G}}\mathrm{c}_{\mathcal{G}} + \Big(kN_\tau - \frac{k(k-1)}{2}\Big)\frac{N_{\mathcal{F}}}{N_P}\mathrm{c}_{\mathcal{F}}
\end{aligned}$$

From the serial and parallel computation, we can derive a lower and an upper bound for $k \geq 1$ the speedup. The lower bound is given by

$$\begin{aligned}
\frac{\mathrm{C}_{\text{Serial}}}{\mathrm{C}_{\text{Parareal}}} &= \frac{N_{\text{Serial}} \cdot \mathrm{c}_{\mathcal{F}}}{\Big((k+1)N_\tau - \frac{k(k-1)}{2}\Big) \cdot N_{\mathcal{G}}\mathrm{c}_{\mathcal{G}} + \Big(kN_\tau - \frac{k(k-1)}{2}\Big)\frac{N_{\mathcal{F}}}{N_P}\mathrm{c}_{\mathcal{F}}}\\
&= \frac{N_{\text{Serial}} \cdot \mathrm{c}_{\mathcal{F}}}{(k+1) \cdot \underbrace{N_\tau \cdot N_{\mathcal{G}}}_{\leq N_{\text{Serial}}} \cdot \mathrm{c}_{\mathcal{G}} + k \cdot \frac{N_{\text{Serial}}}{N_P} \cdot \mathrm{c}_{\mathcal{F}}}\\
&\geq \frac{\mathrm{c}_{\mathcal{F}}}{(k+1) \cdot \underbrace{\mathrm{c}_{\mathcal{G}}}_{\leq \mathrm{c}_{\mathcal{F}}} + \frac{k}{N_P} \cdot \mathrm{c}_{\mathcal{F}} + 1}\\
&\geq \frac{1}{(k+1) + \frac{k}{N_P}} = \frac{N_P}{kN_P + 1k + N_P}.
\end{aligned} \tag{2.33}$$

We observe that the lower bound shows that if the computation time of the coarse and fine solvers is the same, we don't get any speedup. On the contrary, we slow down the algorithm with each iteration. The upper bound can be derived analogously

$$\begin{aligned}
\frac{\mathrm{C}_{\text{Serial}}}{\mathrm{C}_{\text{Parareal}}} &= \frac{N_{\text{Serial}} \cdot \mathrm{c}_{\mathcal{F}}}{\Big((k+1)N_\tau - \frac{k(k-1)}{2}\Big) \cdot N_{\mathcal{G}}\mathrm{c}_{\mathcal{G}} + \Big(kN_\tau - \frac{k(k-1)}{2}\Big) \cdot \underbrace{\frac{N_{\mathcal{F}}}{N_P}}_{\geq \frac{N_{\mathcal{G}}}{N_P}} \cdot \underbrace{\mathrm{c}_{\mathcal{F}}}_{\mathrm{c}_{\mathcal{G}}}} \\
&\leq \frac{N_{\text{Serial}} \cdot \mathrm{c}_{\mathcal{F}}}{\Big((k+1)N_\tau - \frac{k(k-1)}{2}\Big) \cdot N_{\mathcal{G}} \cdot \mathrm{c}_{\mathcal{G}} + \Big(kN_\tau - \frac{k(k-1)}{2}\Big) \cdot \frac{N_{\mathcal{G}}}{N_P} \cdot \mathrm{c}_{\mathcal{G}}} \\
&= \frac{N_{\text{Serial}} \cdot \mathrm{c}_{\mathcal{F}}}{\Big((k+1+\frac{k}{N_P})N_\tau - \frac{k(k-1)}{2} - \frac{k(k-1)}{2N_P}\Big) \cdot N_{\mathcal{G}} \cdot \mathrm{c}_{\mathcal{G}}} \\
&= \frac{N_\tau}{\Big((k+1+\frac{k}{N_P})N_\tau - \underbrace{\frac{k(k-1)}{2}}_{\leq \frac{1}{2}k^2} - \underbrace{\frac{k(k-1)}{2N_P}}_{\leq \frac{1}{2}k^2}\Big)} \frac{N_{\mathcal{F}} \cdot \mathrm{c}_{\mathcal{F}}}{N_{\mathcal{G}} \cdot \mathrm{c}_{\mathcal{G}}} \\
&= \frac{N_\tau}{(k+1+\frac{k}{N_P})N_\tau - k^2} \frac{N_{\mathcal{F}} \cdot \mathrm{c}_{\mathcal{F}}}{N_{\mathcal{G}} \cdot \mathrm{c}_{\mathcal{G}}}
\end{aligned} \tag{2.34}$$

It shows that the speedup is directly related to the difference in computational time for the fine and coarse solvers. As $N_{\mathcal{F}} > N_{\mathcal{G}}$ and $\mathrm{c}_{\mathcal{F}} > \mathrm{c}_{\mathcal{G}}$ is assumed, the second part is always larger one. Now we focus on the first term

$$\begin{aligned}
\frac{N_\tau}{(k+1+\frac{k}{N_P})N_\tau - k^2} &\geq 1 \\
N_\tau &\geq \Big(k+1+\frac{k}{N_P}\Big)N_\tau - k^2 \\
\Big(1-k-1-\frac{k}{N_P}\Big)N_\tau &\geq -k^2 \\
-k\Big(1+\frac{1}{N_P}\Big)N_\tau &\geq -k^2 \\
N_\tau &\leq \frac{k}{1+\frac{1}{N_P}} = \frac{N_P k}{N_P+1}
\end{aligned} \tag{2.35}$$

Therefore if the condition above is violated, the speedup is reduced, as

$$\lim_{N_P \to \infty} \frac{N_P k}{N_P + 1} = k \tag{2.36}$$

is given and $k^{\max} \ll N_\tau$ is assumed previously, the number of processors and time slices has to be chosen carefully to obtain a minimal slowdown from this part.

Chapter 3

Spatial Discretization

In the introduction in Chapter 1, we presented the general notation for mathematical models that arise in financial modeling. We observe that in both European and American option pricing problems, a parabolic PDE operator $\mathcal{L}$ is given by the underlying model. Thus, we present the spatial grid for $u(\mathbf{x}, \tau_n)$ with $\mathbf{x} = (\mathrm{x}_1, \mathrm{x}_2, \dots, \mathrm{x}_d)$ and the discretization as well as the sparse grid idea using a general d-dimensional parabolic operator formulation. On a rectangular domain $\Omega_d \times [0, T]$ with suitable initial and boundary conditions, the spatial operator is given by

$$\mathcal{L}\Big[u\Big](\mathbf{x}, \tau_n) = \sum_{i,j=1}^{d} a_{ij}(\mathbf{x}, \tau_n) \frac{\partial^2}{\partial \mathrm{x}_i \partial \mathrm{x}_j} u(\mathbf{x}, \tau_n) + \sum_{i=1}^{d} b_i(\mathbf{x}, \tau_n) \frac{\partial}{\partial \mathrm{x}_i} u(\mathbf{x}, \tau_n) + c(\mathbf{x}, \tau_n) \cdot u(\mathbf{x}, \tau_n). \tag{3.1}$$

In this chapter, we assume that the models satisfy these requirements without loss of generality. A general grid is presented in the first section. In the second section, we discuss the derivation of the local approximation of the derivatives. We present the standard finite difference method, considering second-order central and upwind schemes. Then, the combination technique for sparse grids is introduced. With this method, the number of grid points and thus the computational effort can be reduced.

3.1 Spatial Grids

For all spatial grids we consider a uniform grid spacing as given in Definition 11. Note that other non-uniform grid spacings are also used [44, 57]. Often, transformations between non-uniform and uniform grids are used to obtain the desired numerical structure [44, 57]. These transformations are presented and applied to financial models in Chapter 4.

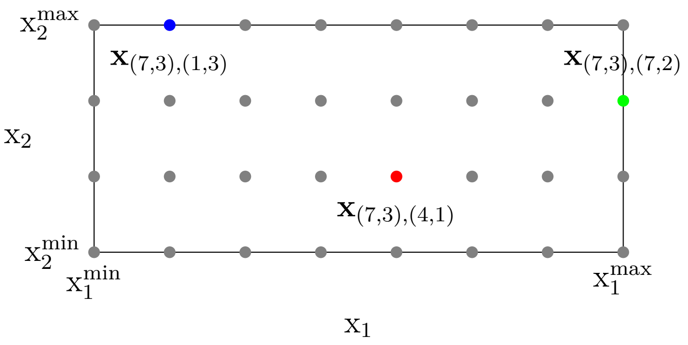


Figure 3.1: The discrete grid $\Omega_{(7,3)}$ with the example points $\mathbf{x}_{(7,3),(4,1)}$ in red, $\mathbf{x}_{(7,3),(1,3)}$ in blue and $\mathbf{x}_{(7,3),(7,2)}$ in green.

Definition 11 (Uniform grid)**.** We consider a d-dimensional domain Ω_d in a continuous setting, $\mathbf{x} \in \Omega_d$ is given by $\mathbf{x} = (\mathrm{x}_1, \mathrm{x}_2, \dots, \mathrm{x}_d)$, where x_i is the point with respect to the i-th coordinate direction for $i = 1, \dots, d$. Using the multi-indices

$$\mathbf{M} = (\mathrm{M}_1, \mathrm{M}_2, \dots, \mathrm{M}_d) \in \mathbb{N}^d, \tag{3.2}$$

$$\mathbf{m} = (\mathrm{m}_1, \mathrm{m}_2, \dots, \mathrm{m}_d), \in \mathbb{N}_0^d \tag{3.3}$$

with $\mathrm{m}_i = 0, \dots, \mathrm{M}_i$ for $i = 1, \dots, d$, we define a tensor based grid $\Omega_{\mathbf{M}}$. The grid nodes are given by

$$\mathbf{x}_{\mathbf{M},\mathbf{m}} = (\mathrm{x}_{\mathrm{M}_1,\mathrm{m}_1}, \mathrm{x}_{\mathrm{M}_2,\mathrm{m}_2}, \dots, \mathrm{x}_{\mathrm{M}_d,\mathrm{m}_d}), \tag{3.4}$$

where $\mathrm{m}_i = 0, 1, \dots, \mathrm{M}_i$ corresponds to the m_i-th node in the i-th coordinate. For a truncated domain $\Omega_{\mathbf{M}} = [\mathrm{x}_1^{\min}, \mathrm{x}_1^{\max}] \times [\mathrm{x}_2^{\min}, \mathrm{x}_2^{\max}] \times \dots \times [\mathrm{x}_d^{\min}, x_d^{\max}]$, we get the spacing

$$\mathbf{h} = (\mathrm{h}_1, \mathrm{h}_2, \dots, \mathrm{h}_d) = \left(\frac{\mathrm{x}_1^{\max} - \mathrm{x}_1^{\min}}{\mathrm{M}_1}, \frac{\mathrm{x}_2^{\max} - \mathrm{x}_2^{\min}}{\mathrm{M}_2}, \dots, \frac{\mathrm{x}_d^{\max} - \mathrm{x}_d^{\min}}{\mathrm{M}_d} \right). \tag{3.5}$$

Example of a general uniform grid: Let $\Omega_{(7,3)}$ be given on the truncated rectangular domain $[\mathrm{x}_1^{\min}, \mathrm{x}_1^{\max}] \times [\mathrm{x}_2^{\min}, \mathrm{x}_2^{\max}]$. We observe $\mathbf{M} = (7, 3)$ and thus $\mathrm{m}_1 = 0, 1, \dots, 7$ and $\mathrm{m}_2 = 0, 1, \dots, 3$. We will now present certain grid points on this grid, namely $\mathbf{x}_{(7,3),(1,3)}$, $\mathbf{x}_{(7,3),(4,1)}$ and $\mathbf{x}_{(7,3),(7,2)}$. Figure 3.1 visualizes the grid $\Omega_{\mathbf{M}} = \Omega_{(7,3)}$ with the example points.

Definition 12 (Sparse grid)**.** We consider the grid given by Definition 11 and add an additional indice

$$\mathbf{l} = (\mathrm{l}_1, \mathrm{l}_2, \dots, \mathrm{l}_d) \in \mathbb{N}_0^2, \tag{3.6}$$

such that

$$\mathbf{M}_{\mathbf{l}} = (\mathrm{M}_{\mathrm{l}_1}, \mathrm{M}_{\mathrm{l}_2}, \ldots, \mathrm{M}_{\mathrm{l}_d}) = (2^{\mathrm{l}_1}, 2^{\mathrm{l}_2}, \ldots, 2^{\mathrm{l}_d}). \tag{3.7}$$

We define a similar tensor-based grid $\Omega_{\mathbf{M}_{\mathbf{l}}}$ with the grid nodes

$$\mathbf{x}_{\mathbf{M}_{\mathbf{l}},\mathbf{m}} = (\mathrm{x}_{\mathrm{M}_1,\mathrm{m}_1}, \mathrm{x}_{\mathrm{M}_2,\mathrm{m}_2}, \ldots, \mathrm{x}_{\mathrm{M}_d,\mathrm{m}_d}), \tag{3.8}$$

where $\mathrm{m}_{\mathrm{l}_i} = 0, 1, \ldots, \mathrm{M}_{\mathrm{l}_i} = 2^{\mathrm{l}_i}$ corresponds to the m_i-th node in the i-th coordinate. With $\Omega_{\mathbf{M}_{\mathbf{l}}} = [0,1]^d$, we gain

$$\mathrm{h}_{\mathbf{l}} = (\mathrm{h}_1, \mathrm{h}_2, \ldots, \mathrm{h}_d) = (2^{-\mathrm{l}_1}, 2^{-\mathrm{l}_2}, \ldots, 2^{-\mathrm{l}_d}). \tag{3.9}$$

Thus, any sparse grid is a uniform grid, but not vice versa. Furthermore, the d-dimensional sparse grid is restricted to the domain $[0,1]^d$ with the number of grid points in the $i-$th direction is given by $\mathrm{M}_{\mathrm{l}_i} + 1$, where $\mathrm{M}_{\mathrm{l}_i}$ is a power of 2.

Example for a sparse grid: Let $\Omega_{(4,8)}$ be on the rectangular domain $[0,1] \times [0,1]$. We observe $\mathbf{l} = (\mathrm{l}_1, \mathrm{l}_2) = (2,3)$. For the first grid coordinate we get $\mathrm{M}_{\mathrm{l}_1} = 2^{\mathrm{l}_1} = 2^2 = 4$. Thus, the five grid points are represented by $\mathrm{m}_{\mathrm{l}_1} = 0, 1, \ldots, 4$ and the grid spacing on the unit domain results in $\mathrm{h}_1 = 2^{-\mathrm{l}_1} = 0.25$. Similarly for the second dimension with $\mathrm{M}_{\mathrm{l}_2} = 2^{\mathrm{l}_2} = 2^3 = 8$, the nine grid points are represented by $\mathrm{m}_{\mathrm{l}_2} = 0, 1, \ldots, 8$ with a grid spacing of $\mathrm{h}_2 = 2^{-\mathrm{l}_2} = 2^{-3} = 0.125$. Now we have defined the tensor grid $\Omega_{(4,8)}$ and we can show a certain grid point on this grid. The point $\mathbf{x}_{(4,8),(1,4)}$, refers to the grid point $(\mathrm{m}_1 \cdot 2^{\mathrm{l}_1}, m_2 \cdot 2^{\mathrm{l}_2}) = (1 \cdot 2^{-2}, 4 \cdot 2^{-3}) = (0.25, 0.5)$. And the grid point $(0.75, 0.125) = (3 \cdot 2^{-2}, 1 \cdot 2^{-3})$ is represented by $\mathbf{x}_{(4,8),(3,1)}$ and $\mathbf{x}_{(4,8),(2,8)} = (0.5, 1)$. Figure 3.2 visualizes the grid $\Omega_{\mathbf{M}_{\mathbf{l}}} = \Omega_{(2,3)}$ with the example points.

3.2 Finite Difference Methods

The spatial derivatives can be approximated in different ways, e.g. with finite differences [38, 39, 56], finite-element-finite-volume [109], multigrid [10, 11] or spectral methods [44]. We will focus on finite differences. The finite difference stencils are derived using a Taylor expansion under the assumption that u is sufficiently smooth. As a first step, we present the second-order central difference stencils for the first, second, and mixed derivatives. The stencil for the first and second derivatives in the spatial direction i with the corresponding

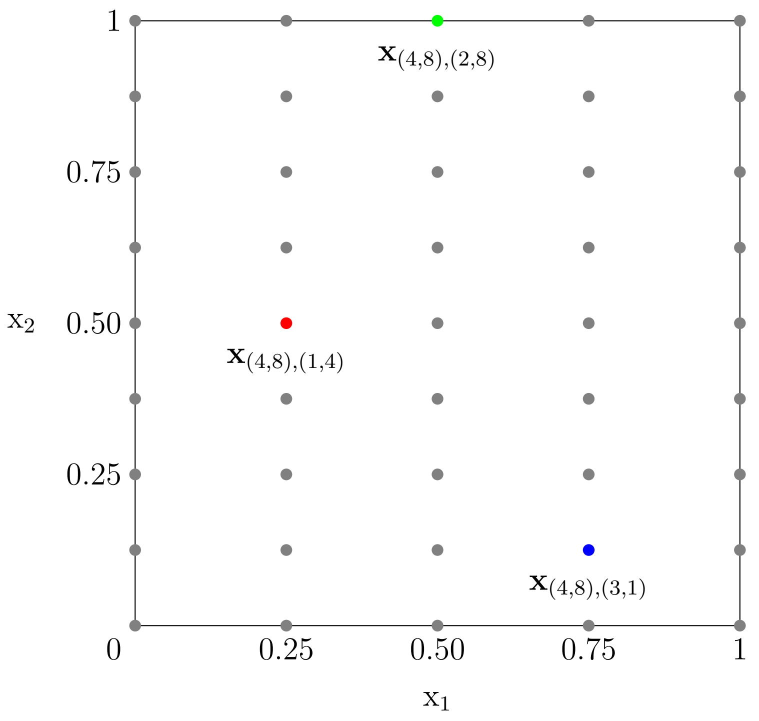

$\mathbf{x}_{(4,8),(2,8)}$
$\mathbf{x}_{(4,8),(1,4)}$
$\mathbf{x}_{(4,8),(3,1)}$
1
0.75
0.50
0.25
0
x_2
0.25
0.50
0.75
1
x_1

Figure 3.2: The discrete sparse grid $\Omega_{(4,8)}$ with the example points $\mathbf{x}_{(4,8),(1,4)}$ in red, $\mathbf{x}_{(4,8),(2,8)}$ in green and $\mathbf{x}_{(4,8),(3,1)}$ in blue.

unit vector $\mathbf{e}_i$ at the grid point $(\mathbf{x}_{\mathbf{M},\mathbf{m}}, \tau_n)$ is given by

$$\begin{aligned}
\delta_i^0 u(\mathbf{x}_{\mathbf{M},\mathbf{m}}, \tau_n) &= \frac{1}{2\mathrm{h}_i}\Big(u(\mathbf{x}_{\mathbf{M},\mathbf{m}} + \mathrm{h}_i\mathbf{e}_i, \tau_n) - u(\mathbf{x}_{\mathbf{M},\mathbf{m}} - \mathrm{h}_i\mathbf{e}_i, \tau_n)\Big) \\
&= \frac{\partial}{\partial \mathrm{x}_i} u(\mathbf{x}_{\mathbf{M},\mathbf{m}}, \tau_n) + \mathcal{O}(\mathrm{h}_i^2) \\
\delta_i^2 u(\mathbf{x}_{\mathbf{M},\mathbf{m}}, \tau_n) &= \frac{1}{\mathrm{h}_i^2}\Big(u(\mathbf{x}_{\mathbf{M},\mathbf{m}} + \mathrm{h}_i\mathbf{e}_i, \tau_n) - 2u(\mathbf{x}_{\mathbf{M},\mathbf{m}}, \tau_n) + u(\mathbf{x}_{\mathbf{M},\mathbf{m}} - \mathrm{h}_i\mathbf{e}_i, \tau_n)\Big) \\
&= \frac{\partial^2}{\partial \mathrm{x}_i^2} u(\mathbf{x}_{\mathbf{M},\mathbf{m}}, \tau_n) + \mathcal{O}(\mathrm{h}_i^2)
\end{aligned} \tag{3.10}$$

The second-order mixed derivative stencil is derived by using the first derivative stencil in two different directions $i, j = 1, 2, \ldots, d$ with $i \neq j$. It reads

$$\begin{aligned}
\delta_i^0\delta_j^0 u(\mathbf{x}_{\mathbf{M},\mathbf{m}}, \tau_n) &= \frac{1}{4\mathrm{h}_i\mathrm{h}_j}\Big(u(\mathbf{x}_{\mathbf{M},\mathbf{m}} + \mathrm{h}_i\mathbf{e}_i + \mathrm{h}_j\mathbf{e}_j, \tau_n) - u(\mathbf{x}_{\mathbf{M},\mathbf{m}} - \mathrm{h}_i\mathbf{e}_i + \mathrm{h}_j\mathbf{e}_j, \tau_n) \\
&\qquad - u(\mathbf{x}_{\mathbf{M},\mathbf{m}} + \mathrm{h}_i\mathbf{e}_i - \mathrm{h}_j\mathbf{e}_j, \tau_n) + u(\mathbf{x}_{\mathbf{M},\mathbf{m}} - \mathrm{h}_i\mathbf{e}_i - \mathrm{h}_j\mathbf{e}_j, \tau_n)\Big) \\
&= \frac{\partial^2}{\partial \mathrm{x}_i \partial \mathrm{x}_j} u(\mathbf{x}_{\mathbf{M},\mathbf{m}}, \tau_n) + \mathcal{O}(\mathrm{h}_i^2) + \mathcal{O}(\mathrm{h}_j^2) + \mathcal{O}(\mathrm{h}_i^2\mathrm{h}_j^2)
\end{aligned} \tag{3.11}$$

Unfortunately, some financial models are fully or only for a restricted parameter range convection-dominated [94, 57, 9]. The implementation of schemes capable of handling convection-dominated PDEs in option pricing is a topic of current research [9]. The most common strategy for a parameter restricted convection-domination is a grid transformation [57, 94]. In our case we use the upwind scheme presented by LeVeque [69]. Let $\mathrm{ad}(\mathrm{x}_i)$ be the advection term in the $i-$th spatial direction. In the case $\mathrm{ad}(\mathrm{x}_i) > 0$, the second order backward stencil

$$\begin{aligned}\delta_i^- u(\mathbf{x}_{\mathbf{M},\mathbf{m}},\tau_n) &= \frac{1}{2\mathrm{h}_i}\Big(3u(\mathbf{x}_{\mathbf{M},\mathbf{m}},\tau_n) - 4u(\mathbf{x}_{\mathbf{M},\mathbf{m}} - \mathrm{h}_i\mathbf{e}_i,\tau_n) + u(\mathbf{x}_{\mathbf{M},\mathbf{m}} - 2\mathrm{h}_i\mathbf{e}_i,\tau_n)\Big)\\ &= \frac{\partial}{\partial \mathrm{x}_i}u(\mathbf{x}_{\mathbf{M},\mathbf{m}},\tau_n) + \mathcal{O}(\mathrm{h}_i^2)\end{aligned} \tag{3.12}$$

is used, and if $\mathrm{ad}(\mathrm{x}_i) < 0$ the second order forward stencil

$$\begin{aligned}\delta_i^+ u(\mathbf{x}_{\mathbf{M},\mathbf{m}},\tau_n) &= \frac{1}{2\mathrm{h}_i}\Big(-3u(\mathbf{x}_{\mathbf{M},\mathbf{m}},\tau_n) + 4u(\mathbf{x}_{\mathbf{M},\mathbf{m}} + \mathrm{h}_i\mathbf{e}_i,\tau_n) - u(\mathbf{x}_{\mathbf{M},\mathbf{m}} + \mathrm{h}_i\mathbf{e}_i,\tau_n)\Big)\\ &= \frac{\partial}{\partial \mathrm{x}_i}u(\mathbf{x}_{\mathbf{M},\mathbf{m}},\tau_n) + \mathcal{O}(\mathrm{h}_i^2).\end{aligned} \tag{3.13}$$

To avoid the explicit computation of the sign change, one can use a *combined stencil*

$$\begin{aligned}\delta_i^{\pm} u(\mathbf{x}_{\mathbf{M},\mathbf{m}},\tau_n) &= \frac{\mathrm{ad}(\mathrm{x}_i)}{2\mathrm{h}_i}\Big(\frac{1}{2}u(\mathbf{x}_{\mathbf{M},\mathbf{m}} - 2\mathrm{h}_i\mathbf{e}_i,\tau_n) - 2u(\mathbf{x}_{\mathbf{M},\mathbf{m}} - \mathrm{h}_i\mathbf{e}_i,\tau_n) + 3u(\mathbf{x}_{\mathbf{M},\mathbf{m}},\tau_n)\\ &\qquad - 2u(\mathbf{x}_{\mathbf{M},\mathbf{m}} + \mathrm{h}_i\mathbf{e}_i,\tau_n) + \frac{1}{2}u(\mathbf{x}_{\mathbf{M},\mathbf{m}} + 2\mathrm{h}_i\mathbf{e}_i,\tau_n)\Big)\\ &\quad + \frac{|\mathrm{ad}(\mathrm{x}_i)|}{2\mathrm{h}_i}\Big(\frac{1}{2}u(\mathbf{x}_{\mathbf{M},\mathbf{m}} - 2\mathrm{h}_i\mathbf{e}_i,\tau_n) - 2u(\mathbf{x}_{\mathbf{M},\mathbf{m}} - \mathrm{h}_i\mathbf{e}_i,\tau_n)\\ &\qquad + 2u(\mathbf{x}_{\mathbf{M},\mathbf{m}} + \mathrm{h}_i\mathbf{e}_i,\tau_n) - \frac{1}{2}u(\mathbf{x}_{\mathbf{M},\mathbf{m}} + 2\mathrm{h}_i\mathbf{e}_i,\tau_n)\Big)\\ &= \frac{\partial}{\partial \mathrm{x}_i}u(\mathbf{x}_{\mathbf{M},\mathbf{m}},\tau_n) + \mathcal{O}(\mathrm{h}_i^2).\end{aligned} \tag{3.14}$$

for linear systems. Besides the presented second-order stencils, there are higher-order stencils with more grid points, as well as compact high-accuracy stencils with fewer grid points than the corresponding normal stencils of the same order [41, 42].

A uniform tensor grid discretization with $M \in \mathbb{N}$ grid points in each direction corresponds to $\mathcal{O}(M^d)$ grid points in total for d dimensions. This increase for higher dimensions leads to excessive memory requirements and is therefore computationally expensive. This growth is referred to as the 'curse of dimensionality' as for higher dimensions, we on the one hand gain a better model accuracy and on the other hand enhance the computational complexity significantly. To reduce the number of grid points, sparse grids and the combination technique are used and combined. This is the topic of the next section.

3.3 Sparse Grids

To introduce the sparse grid combination technique, we follow the approach of Reisinger [89] and Hendricks [40]. Therefore, we discuss the approach in detail for the two-dimensional case and then generalize it to d dimensions. For the two-dimensional case, the grid domain is fixed to $\Omega_2 = [0,1]^2$ with the grids $\Omega_{\mathbf{M}_\mathrm{l}}$ being the tensor-based grids introduced in Definition 12. The solution $u_\mathbf{l}$ is defined on $\Omega_{\mathbf{M}_\mathrm{l}}$ with $\mathbf{l} = (\mathrm{l}_1, \mathrm{l}_2) \in \mathbb{N}_0^2$ and $\mathbf{M} = (2^{\mathrm{l}_1}, 2^{\mathrm{l}_2})$ with mesh width $\mathbf{h} = (\mathrm{h}_1, \mathrm{h}_2) = (2^{-\mathrm{l}_1}, 2^{-\mathrm{l}_2})$.

To use the combination technique, we consider the error splitting

$$u - u_\mathbf{l} = \mathrm{h}_1^2 w_1(\mathrm{h}_1) + h_2^2 w_2(\mathrm{h}_2) + \mathrm{h}_1^2\mathrm{h}_2^2 w_{1,2}(\mathrm{h}_1, \mathrm{h}_2), \tag{3.15}$$

where w_1 depends only on h_1 similarly w_2 depends only on h_2. Furthermore, h_1 and h_2 are independent of each other. Each of w_1, w_2, $w_{1,2}$ is bounded. Since the error-splitting structure is the key assumption within the combination technique, we need to check whether the second-order finite differences satisfy this requirement. The splitting of finite differences of a linear scheme has been analyzed by [6, 89, 43]. Bungartz et al. [6] showed that the second-order central difference scheme satisfies the splitting structure for the two-dimensional Laplace equation. Reisinger [89] extended this framework for a wider class of PDEs and proved the desired splitting structure for the poison equation with the second-order central difference scheme with homogeneous Dirichlet boundary conditions. He also derived conditions under which the sparse grid approach is advantageous. The error splitting structure of the fourth-order high-compact scheme was analyzed by Hendricks et al. [43]. The discrete solutions of the finite difference schemes of the sparse grids are combined by interpolation. Since the error splitting structure must be preserved over the entire domain to achieve the desired accuracy, the interpolation must also preserve the splitting structure. For second-order finite differences, a multi-linear interpolation preserves the desired error structure. Hendricks et al. [43] showed that multi-linear interpolation is insufficient for fourth-order schemes because it reduces the convergence at the interpolated points to second order. Only the points contained in all grids converge to the desired order. Unfortunately, this is only true for the center point with $(0.5, 0.5)$. Therefore, univariate cubic spline interpolations were used.

In summary, the derivation of the desired error splitting scheme depends on the form of the truncation error for the discretization as well as on the interpolation and the smoothness of the solution, so that the higher derivatives are bounded for the discrete solutions. Hendricks [40] states the key property formulation of Reisinger [89] for general finite difference methods with an order of accuracy p. For a two-dimensional linear PDE with the discrete operator $\mathcal{L}_{\mathbf{l}}$ with $\mathbf{x}_{\mathbf{M}_{\mathbf{l}},\mathbf{m}}$ the domain $\Omega_{\mathbf{M}_{\mathbf{l}}}$, the requirements are reduced to

1. The scheme has a point-wise truncation error of the form
$$(\mathcal{L} - \mathcal{L}_{\mathbf{l}})u(\mathbf{x}_{\mathbf{M}_{\mathbf{l}},\mathbf{m}}) = \sum_{k_1=1}^{2} \sum_{k_2=1}^{2} w_{m_{k_1},m_{k_2}}(\mathbf{x}_{\mathbf{M}_{\mathbf{l}},\mathbf{m}}; \mathrm{h}_{m_{k_1}}, \mathrm{h}_{m_{k_2}})\mathrm{h}_{m_{k_1}}^{p} \cdot \mathrm{h}_{m_{k_2}}^{p} \tag{3.16}$$

2. Stability of the discretization scheme.

3. Sufficiently smooth initial data and compatible boundary data, such that the mixed derivatives of required order are bounded.

Note that in the case of $p = 2$ the error splitting structure in (3.16) reduces to (3.15). In our case, the error splitting structure is given [89, 43], as well as the stability of the scheme. As mentioned before, the smoothness of the solution for financial problems is not given for all problems. Because for many options the payoff-function as an initial condition is not smooth, e.g. for the European plain vanilla options. Thus, we must apply smoothing techniques to recover the desired bounds, see Section 4.1.

Assuming that the conditions are met and the required error splitting structure is given, the next step in deriving the combination technique is to define a *hierarchical surplus*

$$\mathtt{V}(u_{\mathbf{l}}) = u_{\mathbf{l}} - u_{\mathbf{l}-\mathbf{e}_1} - u_{\mathbf{l}-\mathbf{e}_2} + u_{\mathbf{l}-\mathbf{e}_1-\mathbf{e}_2}, \quad \mathbf{e}_1 = (1,0), \mathbf{e}_2 = (0,1). \tag{3.17}$$

Inserting it into the error splitting, with

$$|\mathbf{l}|_1 = \sum_{i=1}^{d} \mathrm{l}_i \tag{3.18}$$

we obtain

$$\begin{aligned}\mathtt{V}(u-u_l) &= \mathrm{h}_1^2 w_1(\mathrm{h}_1) + \mathrm{h}_2^2 w_2(\mathrm{h}_2) + \mathrm{h}_1^2\mathrm{h}_2^2 w_{1,2}(\mathrm{h}_1,\mathrm{h}_2) - 4\mathrm{h}_1^2 w_1(2\mathrm{h}_1)\\ &- \mathrm{h}_2^2 w_2(\mathrm{h}_2) - 4\mathrm{h}_1^2\mathrm{h}_2^2 w_{1,2}(2\mathrm{h}_1,\mathrm{h}_2) - \mathrm{h}_1^2 w_1(\mathrm{h}_1) - 4\mathrm{h}_2^2 w_2(2\mathrm{h}_2)\\ &- 4\mathrm{h}_1^1\mathrm{h}_2^2 w_{1,2}(\mathrm{h}_1,2\mathrm{h}_2) + 4\mathrm{h}_1^2 w_1(2\mathrm{h}_1) + 4\mathrm{h}_2^2 w_2(2\mathrm{h}_2)\\ &+ 16\mathrm{h}_1^2\mathrm{h}_2^2 w_{1,2}(2\mathrm{h}_1,2\mathrm{h}_2)\\ &= \mathrm{h}_1^2\mathrm{h}_2^2 w_{1,2}(\mathrm{h}_1,\mathrm{h}_2) - 4\mathrm{h}_1^2\mathrm{h}_2^2 w_{1,2}(2\mathrm{h}_1,\mathrm{h}_2) - 4\mathrm{h}_1^2\mathrm{h}_2^2 w_{1,2}(\mathrm{h}_1,2\mathrm{h}_2)\\ &+ 16\mathrm{h}_1^2\mathrm{h}_2^2 w_{1,2}(2\mathrm{h}_1,2\mathrm{h}_2)\\ &= \mathcal{O}(\mathrm{h}_1^2\mathrm{h}_2^2) = \mathcal{O}(2^{-2\mathrm{l}_1}\cdot 2^{-2\mathrm{l}_2}) = \mathcal{O}(2^{-2(\mathrm{l}_1+\mathrm{l}_2)}) = \mathcal{O}(2^{-2|\mathbf{l}|_1}).\end{aligned} \tag{3.19}$$

The surplus can be interpreted as the information gain of the solution $u_{\mathbf{l}}$ on the sparse grid $\Omega_{\mathbf{M}_{\mathbf{l}}}$ with $\mathbf{M}_{\mathbf{l}} = (2^{\mathrm{l}_1},\ldots,2^{\mathrm{l}_d})$. The solutions $u_{\mathbf{l}}$ where $|\mathbf{l}|_1$ are the same, have the same number of grid nodes as well as the same surplus and therefore the same information gain. The combination technique is motivated by getting the highest information gain from the subsolutions. Therefore subsolutions with a high information gain corresponding to a high surplus are used. The combined sparse grid solution u^s_m is defined as the sum of all surpluses with $|\mathbf{l}|_1 \le m$ for $m \in \mathbb{N}_0$

$$u^s_m = \sum_{|\mathbf{l}|_1 \le m} \mathtt{V}(u_{\mathbf{l}}) = \sum_{|\mathbf{l}|_1 = m} u_{\mathbf{l}} - \sum_{|\mathbf{l}|_1 = m-1} u_{\mathbf{l}} \tag{3.20}$$

The upper error bound can be found by including the surpluses of all subsolutions that are not used to compute u^s_m.

$$||u^s_m - u|| \le \sum_{|\mathbf{l}|_1 > m} ||\mathtt{V}(u_{\mathbf{l}})|| \tag{3.21}$$

as they have a higher surplus $|\mathbf{l}|_1 > m$. Since the number of grids with the same surplus is given by $|\mathbf{l}|_1 + 1$, and since these grids have the same order of accuracy $\mathcal{O}(2^{-2|\mathbf{l}|_1})$, see equation (3.19), we derive the upper bound

$$\begin{aligned}||u^s_m - u|| &\le \sum_{|\mathbf{l}|_1 > m} ||\mathtt{V}(u_{\mathbf{l}})|| = \sum_{|\mathbf{l}|_1 > m} \mathcal{O}(2^{-2|\mathbf{l}|_1})\\ &= \sum_{q>m} \mathcal{O}((q+1)2^{-2q}) = \mathcal{O}(m2^{-2m}).\end{aligned} \tag{3.22}$$

A closer look at the subsolutions within the combined sparse grid solution shows that the

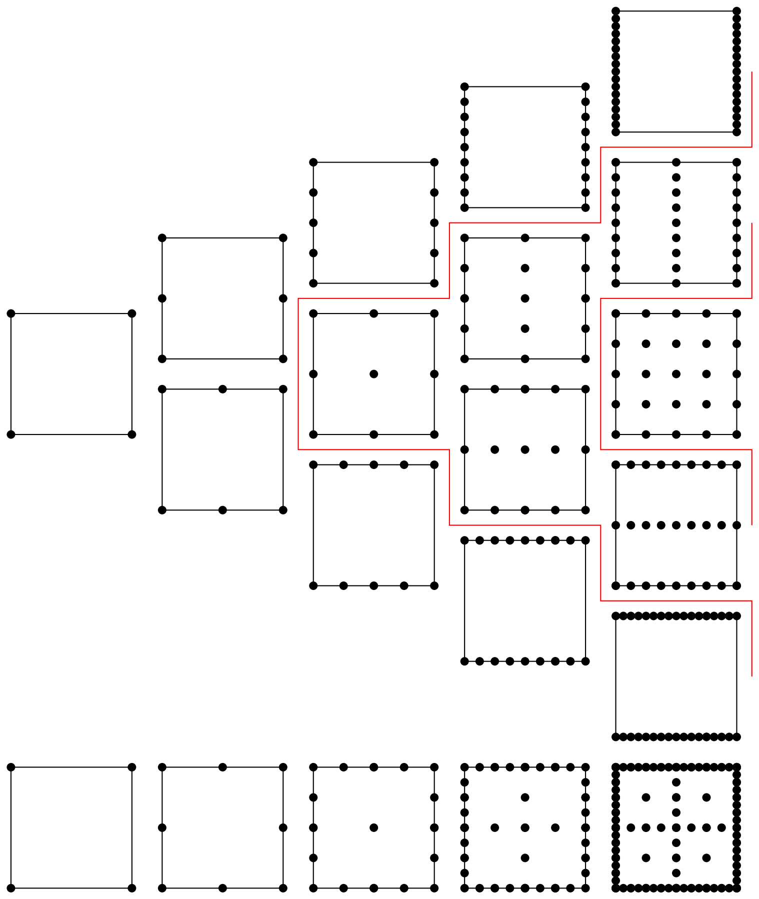


Figure 3.3: Subgrids and corresponding sparse grids for $|\mathbf{l}|_1 = 0, 1, 2, 3, 4$. The separating line between the second and third row contains all grids with $\mathbf{l}^{\min} = 2$ and the other separating line before the last row contains all subgrids with $\mathbf{l}^{\min} = 3$.

sparse grid combination formula at the level m is given by

$$\begin{aligned} u_m^s = \sum_{|\mathbf{l}|_1 \le m} \mathtt{V}(u_\mathbf{l}) &= \sum_{q=0}^{m} \Bigg(\sum_{|\mathbf{l}|_1 = q} u_\mathbf{l} - 2 \sum_{\substack{|\mathbf{l}|_1 = q-1 \\ q-1 \ge 0}} u_\mathbf{l} + \sum_{\substack{|\mathbf{l}|_1 = q-2 \\ q-2 \ge 0}} u_\mathbf{l} \Bigg) \\ &= \sum_{|\mathbf{l}|_1 = m} u_\mathbf{l} - \sum_{|\mathbf{l}|_1 = m-1} u_\mathbf{l} \end{aligned} \tag{3.23}$$

We observe that all subsolutions with $|\mathbf{l}|_1 < m - 1$ cancel out. Figure 3.3 shows that the sparse grid solution also contains highly disordered grids. To avoid numerical instability due to sensitivity to this, we set a minimum mesh width in our numerical experiments with $\mathrm{l}_i \ge \mathbf{l}^{\min}$ for $i = 1, 2, \ldots, d$. To have at least 9 grid points in each dimension, we set $\mathrm{l}^{\min} = 3$.

Additionally for some applications, e.g. the Heston model in Section 4.3, it is sufficient to have more grid point in one spatial direction than in another. Therefore we introduce the concept of *limitation*. Through the limitation, we set individually constraints between l_i and l_j for $i, j = 1, \ldots, d$ and $i \neq j$, e.g. $\mathrm{l}_1 > \mathrm{l}_2$.

Now we generalize the approach for a d-dimensional setting on the domain $\Omega_d = [0,1]^d$. With the family of Cartesian grids $\Omega_{\mathbf{M}_\mathbf{l}}$ defined by the sparse grid in Definition 12. The hierarchical surplus has been extended from the two-dimensional case in equation (3.17) to

$$\mathtt{V}(u_\mathbf{l}) = \mathtt{V}_1\mathtt{V}_2 \ldots \mathtt{V}_d u_\mathbf{l}, \tag{3.24}$$

with $\mathbf{e}_i$ denoting the unit vector in the $i-$th spatial direction

$$\mathtt{V}_i u_\mathbf{l} = \begin{cases} u_\mathbf{l} - u_{\mathbf{l}-\mathbf{e}_i}, & \mathrm{l}_i > 0 \\ u_\mathbf{l}, & \mathrm{l}_i = 0. \end{cases} \tag{3.25}$$

This corresponds to an error splitting of the form

$$u - u_\mathbf{l} = \sum_{k=1}^{d} \sum_{\substack{\{j_1,\ldots,j_k\} \\ \subseteq\{1,\ldots,d\}}} w_{j_1,\ldots,j_k}(\cdot; \mathrm{h}_{j_1}, \ldots, \mathrm{h}_{j_k})\mathrm{h}_{j_1}^p, \ldots, \mathrm{h}_{j_k}^p \tag{3.26}$$

it holds

$$\begin{aligned} \mathtt{V}(u_\mathbf{l}) &= \mathcal{O}(\mathrm{h}_1^p \cdot \mathrm{h}_2^p \cdot \cdots \cdot \mathrm{h}_d^p) \\ &= \mathcal{O}(2^{-\mathrm{l}_1 \cdot p} \cdot 2^{-\mathrm{l}_2 \cdot p} \cdot \cdots \cdot 2^{-\mathrm{l}_d \cdot p}) \\ &= \mathcal{O}(2^{-p|\mathbf{l}|_1}). \end{aligned} \tag{3.27}$$

Therefore the equation for the first criteria for the error splitting changes from (3.16) to

$$(\mathcal{L} - \mathcal{L}_\mathbf{l})u(\mathbf{x}_{\mathbf{M}_\mathbf{l},\mathbf{m}}) = \sum_{k=1}^{d} \sum_{\substack{\{j_1,\ldots,j_k\} \\ \subseteq\{1,\ldots,d\}}} w_{j_1,\ldots,j_k}(\mathbf{x}_{\mathbf{M}_\mathbf{l},\mathbf{m}}; \mathrm{h}_{j_1}, \ldots, \mathrm{h}_{j_k})\mathrm{h}_{j_1}^p, \ldots, \mathrm{h}_{j_k}^p, \tag{3.28}$$

see also [89, 40]. The other criteria stay the same and have to hold as well.

For the derivation of the upper bound, we follow the same steps as in the two-dimensional case. We have the estimate

$$||u - u_m^s|| \leq \sum_{|\mathbf{l}|_1 > m} ||\mathtt{V}(u_\mathbf{l})||, \tag{3.29}$$

and determine the number of d-dimensional grids with $|\mathbf{l}|_1 \in \mathbb{N}_0$

$$\begin{pmatrix} |\mathbf{l}|_1 + d - 1 \\ d - 1 \end{pmatrix} = \mathcal{O}(|\mathbf{l}|_1^{d-1}). \tag{3.30}$$

We derive the upper error bound of

$$\begin{aligned} ||u - u_m^s|| &\leq \sum_{|\mathbf{l}|_1 > m} ||\mathtt{V}(u_{\mathbf{l}})|| \\ &= \sum_{q > m} \mathcal{O}(\underset{(3.30)}{q^{d-1}} \cdot \underset{(3.27)}{2^{-p \cdot q}}) = \mathcal{O}(m^{d-1} 2^{-p \cdot m}). \end{aligned} \tag{3.31}$$

From the cancellation of the subgrids, the sparse grid is given by

$$u_m^s = \sum_{|\mathbf{l}|_1 < m} = \mathtt{V}(u_{\mathbf{l}}) = \sum_{q=0}^{d-1} \begin{pmatrix} d-1 \\ q \end{pmatrix} \sum_{|\mathbf{l}|_1 = m-q} u_{\mathbf{l}} \tag{3.32}$$

For a more detailed derivation, see [40]. Now we can compare the number of grid points for the full tensor grid and the sparse grid approach. Within the full tensor grid we get $\mathcal{O}(m^d)$ points. While for the sparse grid approach we get $\mathcal{O}(m^{d-1})$ grids to compute the solution and each grid has $\mathcal{O}(2^m)$ points. This results in a reduced total number of grid points of $\mathcal{O}(m^{d-1}2^m)$. Now we can combine the two hierarchical multigrid approaches.

3.4 Combination of the Sparse Grids and the Parareal

We use the Parareal on a sparse grid. Within this combination, we use the same temporal solver for $\mathcal{F}$ and $\mathcal{G}$ and define the only difference between these solvers over the underlying grid. The solver $\mathcal{F}$ is defined as a temporal integrator for $u^s_{m_\mathcal{F}}(\mathbf{x}, \tau)$, the sparse grid solution of level $m_\mathcal{F}$, and $\mathcal{G}$ for $u^s_{m_\mathcal{G}}(\mathbf{x}, \tau)$, the solution on $m_\mathcal{G}$. We assume $m_\mathcal{F} \geq m_\mathcal{G}$. Besides the direct combination of the Parareal with the sparse grids, more advanced combinations can be considered by exploiting the special properties of the combination technique. We start with the naive combination of the Parareal algorithm with the sparse grids combination technique.

Let c_m be the computation time of a sparse grid combination technique for level m for one time step and $N_{\text{Serial}} \in \mathbb{N}$ time steps be denoted by

$$\mathrm{C}^s_{\text{Serial}} = N_{\text{Serial}} \cdot \mathrm{c}(m), \text{ with } \mathrm{c}(m) = (m+1) \cdot c_m + m \cdot c_{m-1},$$

as the two-dimensional combination technique combines all solutions on the sparse grid level m and $m-1$, which are $m+1$ and m sparse grids, respectively. For the computation time of the Parareal we get

$$\mathrm{C}^s_{\text{Parareal}} = \Big((k+1)\cdot N_\tau - \frac{k(k-1)}{2})\Big) \cdot N_{\mathcal{G}}\mathrm{c}(m_{\mathcal{G}}) + \Big(kN_\tau - \frac{k(k-1)}{2}) \cdot \frac{N_{\mathcal{F}}}{N_P}\mathrm{c}(m_{\mathcal{F}}). \quad (3.33)$$

From the speedup bounds of the general Parareal from the equations (2.33) and (2.34), we get the bounds for the combination

$$\frac{N_P}{kN_P + k + N_P} \leq \frac{\mathrm{C}^s_{\text{Serial}}}{\mathrm{C}^s_{\text{Parareal}}} \leq \frac{N_{\mathcal{F}}\mathrm{c}(m_{\mathcal{F}})}{N_{\mathcal{G}}\mathrm{c}(m_{\mathcal{G}})} \frac{N_\tau}{(k+1+\frac{k}{N_P})N_\tau - k^2}. \quad (3.34)$$

A further inspection of the speedup of the sparse grid computation on different grids gives $\mathrm{c}(m_{\mathcal{F}})$ and $\mathrm{c}(m_{\mathcal{G}})$

$$\frac{\mathrm{c}(m_{\mathcal{F}})}{\mathrm{c}(m_{\mathcal{G}})} = \frac{(m_{\mathcal{F}}+1)\cdot c_{m_{\mathcal{F}}} + m_{\mathcal{F}} \cdot c_{m_{\mathcal{F}}-1}}{m_{\mathcal{G}} \cdot c_{m_{\mathcal{G}}} + (m_{\mathcal{G}}-1)\cdot c_{m_{\mathcal{G}}}}, \quad (3.35)$$

and thus we get the bounds

$$1 \leq \frac{2m_{\mathcal{F}}+1}{2m_{\mathcal{G}}-1}\frac{c_{\mathcal{F}}}{c_{\mathcal{G}}} \leq \frac{\mathrm{c}(m_{\mathcal{F}})}{\mathrm{c}(m_{\mathcal{G}}-1)} \leq \frac{2m_{\mathcal{F}}+1}{2m_{\mathcal{G}}-1}\frac{c_{m_{\mathcal{F}}}}{c_{m_{\mathcal{G}}-1}}. \quad (3.36)$$

Now we want to reduce the computational cost and focus on the combination technique. For the fine solver with $m_{\mathcal{F}}$ we have

$$u^s_{m_{\mathcal{F}}} = \sum_{|\mathbf{l}|_1 = m_{\mathcal{F}}} u_{\mathbf{l}} - \sum_{|\mathbf{l}|_1 = m_{\mathcal{F}}-1} u_{\mathbf{l}}. \quad (3.37)$$

Our first idea is to take advantage of the two surpluses which are incorporated within the sparse grid solution. So we set $m_{\mathcal{F}} = m$ as the sparse grid level for $\mathcal{F}$ and $m_{\mathcal{G}} = m-1$ for $\mathcal{G}$. Therefore the fine solver has the higher information gain and from the combination technique, we get for $\mathcal{F}$

$$u^s_m = \sum_{|\mathbf{l}|_1=m} u_{\mathbf{l}} - \boxed{\sum_{|\mathbf{l}|_1=m-1} u_{\mathbf{l}}}$$

and for $\mathcal{G}$

$$u^s_{m_{\mathcal{G}}} = \sum_{|\mathbf{l}|_1=m_{\mathcal{G}}} u_{\mathbf{l}} - \sum_{|\mathbf{l}|_1=m_{\mathcal{G}}-1} u_{\mathbf{l}} = \boxed{\sum_{|\mathbf{l}|_1=m-1} u_{\mathbf{l}}} - \sum_{|\mathbf{l}|_1=m-2} u_{\mathbf{l}}.$$

The boxed sums are identical up to the used solver. The speedup in (3.35) changes to

$$\frac{\mathrm{c}(m_{\mathcal{F}})}{\mathrm{c}(m_{\mathcal{G}})} = \frac{\mathrm{c}(m)}{\mathrm{c}(m-1)} = \frac{(m+1)\cdot c_m + m\cdot c_{m-1}}{(m-1)\cdot c_{m-1} + (m-1)\cdot c_{m-2}}, \quad (3.38)$$

	c(m)	c$(m-1)$
Original	$(m+1)\cdot c_m + m\cdot c_{m-1}$	$m\cdot c_{m-1} + (m-1)\cdot c_{m-2}$
Reuse of $\hat{U}_{m-1}$	$(m+1)\cdot c_m + m\cdot c_{m-1}$	$(m-1)\cdot c_{m-2}$
Reuse of $\tilde{U}_{m-1}$	$(m+1)\cdot c_m$	$m\cdot c_{m-1} + (m-1)\cdot c_{m-2}$

Table 3.1: Changes in the computation time for the sparse grid in the Parareal and the improved algorithms.

with

$$1 \leq \frac{2m+1}{2m-1} \leq \frac{\mathrm{c}(m)}{\mathrm{c}(m-1)} \leq \frac{2m+1}{2m-1}\frac{c_m}{c_{m-2}}. \tag{3.39}$$

This change worsens the speedup in contrast to where $m_{\mathcal{F}} \gg m_{\mathcal{G}}$. But from the combination technique, we observe that both solvers compute the solution on the same grids, as the surplus for $m-1$. This indicates that we can reuse the computed solution for either the fine or the coarse solver within the algorithm. Let U_m denote the sum of all subsolutions with surplus m, so $U^k_{m,n}$ denotes the sum of these at time step τ_n within the k-iteration. Therefore we obtain $\hat{U}^k_{m-1,n}$ from the coarse computation used in $\hat{\mathcal{F}}(U^k_n, \hat{U}^k_{m-1,n}, \tau_n, \tau_{n+1})$ and $\tilde{U}^k_{m-1,n}$ from the fine computation used in $\tilde{\mathcal{G}}(U^k_n, \tilde{U}^k_{m-1,n}, \tau_n, \tau_{n+1})$. Now the computation time for the level m is given by

$$\hat{\mathrm{c}}(m) = (m+1)\cdot c_m. \tag{3.40}$$

if reusing $\hat{U}_{m-1}$ and analogously for the reuse of $\tilde{U}_{m-1}$, we get

$$\tilde{\mathrm{c}}(m-1) = (m-1)\cdot c_{m-2}. \tag{3.41}$$

We start by reusing the coarse computation within the fine solver, see Algorithm 2. This approach is only feasible if both solvers are of the same order of accuracy. Since we have assumed that the computation for the time step is identical for each sparse grid, we have to set $N_{\mathcal{F}} = N_{\mathcal{G}}$ to simulate the difference.

The algorithm using the fine subsolutions is given in Algorithm 3. Since the serial computation is the bottleneck of the speedup, this kind of incorporation is more feasible than the other way around. As it speeds up the serial computation without reducing the accuracy. Furthermore, $N_{\mathcal{F}} \gg N_{\mathcal{G}}$ can still hold, which increases the speedup instead of the condition $N_{\mathcal{F}} = N_{\mathcal{G}}$. Both computation times in (3.40) and (3.41) are smaller, than the considered computation time in (3.33) and thus gain a better speedup, this is visualized in the Table 3.1.

Finally, we come to the last improvement achieved by the combination technique. As

Algorithm 2: Iterative procedure of the Parareal reusing the computation of $\hat{U}_{m-1}$ from the coarse solver.

```
$k = 0$
while $k < k^{\max}$ do
    Parallel Approximation
    for $n = k : N_\tau - 1$ do
        $\tilde{U}^k_{n+1} = \hat{\mathcal{F}}(U^k_n, \boxed{\hat{U}^k_{m-1,n}}, \tau_n, \tau_{n+1})$
    end
    Serial Update
    for $n = k : N_\tau - 1$ do
        $\hat{U}^{k+1}_{n+1}, \boxed{\hat{U}^{k+1}_{m-1,n+1}} = \mathcal{G}(U^{k+1}_n, \tau_n, \tau_{n+1})$
        $U^{k+1}_{n+1} = \hat{U}^{k+1}_{n+1} + \tilde{U}^k_{n+1} - \hat{U}^k_{n+1}$
    end
    $k = k + 1$
end
```

Algorithm 3: Iterative procedure of the Parareal reusing the computation of $\tilde{U}_{m-1}$ from the fine solver.

```
$k = 0$
while $k < k^{\max}$ do
    Parallel Approximation
    for $n = k : N_\tau - 1$ do
        $\tilde{U}^k_{n+1}, \boxed{\tilde{U}^k_{m-1,n+1}} = \mathcal{F}(U^k_n, \tau_n, \tau_{n+1})$
    end
    Serial Update
    for $n = k : N_\tau - 1$ do
        $\hat{U}^{k+1}_{n+1} = \tilde{\mathcal{G}}(U^{k+1}_n, \boxed{\tilde{U}^k_{m-1,n+1}}, \tau_n, \tau_{n+1})$
        $U^{k+1}_{n+1} = \hat{U}^{k+1}_{n+1} + \tilde{U}^k_{n+1} - \hat{U}^k_{n+1}$
    end
    $k = k + 1$
end
```

mentioned before, serial computation is the bottleneck of the speedup. Fortunately, we can reduce the computation time even further because the sparse grid solution is the result of a sum of independent subsolutions. Therefore, we can parallelize the computation of those subsolutions. For simplicity reasons, we use the same number of processors N_P as before. It would also be feasible to use another number of processors. The optimized computation time is

$$\begin{aligned}\tilde{\mathrm{C}}^s_{\text{Parareal}} &= \frac{2(k+1)N_\tau - k(k-1)}{2N_P} N_{\mathcal{G}}\tilde{\mathrm{c}}(m-1) + \frac{2kN_\tau - k(k-1)}{2N_P} N_{\mathcal{F}}\mathrm{c}(m) \\ &= \frac{1}{2N_P}\Big((2(k+1)N_\tau - k(k-1))N_{\mathcal{G}}\tilde{\mathrm{c}}(m-1) + (2kN_\tau - k^2 - k)N_{\mathcal{F}}\mathrm{c}(m)\Big)\end{aligned} \tag{3.42}$$

We analyze the speedup again in detail

$$\begin{aligned}\frac{\mathrm{C}^s_{\text{Serial}}}{\tilde{\mathrm{C}}^s_{\text{Parareal}}} &= \frac{N_\tau N_{\mathcal{F}}\mathrm{c}(m)}{\frac{2(k+1)N_\tau - k(k-1)}{2P} N_{\mathcal{G}}\tilde{\mathrm{c}}(m-1) + \frac{2kN_\tau - k(k-1)}{2P} N_{\mathcal{F}}\mathrm{c}(m)} \\ &= \frac{2N_P N_{\mathcal{F}}\mathrm{c}(m)}{2(k+1)N_\tau - k(k-1))N_{\mathcal{G}}\tilde{\mathrm{c}}(m-1) + (2kN_\tau - k(k-1))N_{\mathcal{F}}\mathrm{c}(m)}.\end{aligned} \tag{3.43}$$

A lower bound is given by

$$\begin{aligned}\frac{\mathrm{C}^s_{\text{Serial}}}{\tilde{\mathrm{C}}^s_{\text{Parareal}}} &= \frac{2N_P N_\tau N_{\mathcal{F}}\mathrm{c}(m)}{2(k+1)N_\tau - k(k-1))N_{\mathcal{G}}\tilde{\mathrm{c}}(m-1) + (2kN_\tau - k(k-1))N_{\mathcal{F}}\mathrm{c}(m)} \\ &\geq \frac{N_P N_{\mathcal{F}}\mathrm{c}(m)}{(k+1)\cdot \underbrace{N_{\mathcal{G}}}_{\leq N_{\mathcal{F}}} \cdot\tilde{\mathrm{c}}(m-1) + k\cdot N_{\mathcal{F}}\cdot \mathrm{c}(m)} \\ &\geq \frac{N_P \mathrm{c}(m)}{(k+1)\cdot \underbrace{\tilde{\mathrm{c}}(m-1)}_{<<\mathrm{c}(m)} + k\cdot \mathrm{c}(m)} \\ &\geq \frac{N_P}{2k+1}.\end{aligned} \tag{3.44}$$

This lower bound depends only on the properties of the Parareal, so one can directly optimize the choice of N_P and $k^{\max}$. Since $N_P \gg 2k+1$ should hold to get a profitable speedup. The upper bound is

$$\begin{aligned}
\frac{\mathrm{C}^s_{\text{Serial}}}{\tilde{\mathrm{C}}^s_{\text{Parareal}}} &= \frac{2N_P N_\tau N_\mathcal{F}\mathrm{c}(m)}{2(k+1)N_\tau - k(k-1))N_\mathcal{G}\tilde{\mathrm{c}}(m-1) + (2kN_\tau - k(k-1))\underbrace{N_\mathcal{F}}_{\geq N_\mathcal{G}}\underbrace{\mathrm{c}(m)}_{\geq \tilde{\mathrm{c}}(m-1)}} \\
&\leq \frac{2N_P N_\tau N_\mathcal{F}\mathrm{c}(m)}{((4k+2)N_\tau - 2k(k-1))N_\mathcal{G}\tilde{\mathrm{c}}(m-1)} \\
&= \frac{N_P N_\tau N_\mathcal{F}\mathrm{c}(m)}{((2k+1)N_\tau - k(k-1))N_\mathcal{G}\tilde{\mathrm{c}}(m-1)} \\
&\leq \frac{N_P N_\tau}{(2k+1)N_\tau - k^2}\frac{N_\mathcal{F}\mathrm{c}(m)}{N_\mathcal{G}\tilde{\mathrm{c}}(m-1)}
\end{aligned} \tag{3.45}$$

We see that the upper bound is also larger than before, and it's directly scalable with the right choice of N_P and $k^{\max}$

To determine the correct choice of N_P, we must also consider the communication time. So far, we have neglected communication time, but since parallel computations involve additional expensive communication, communication is the bottleneck of parallelism. Therefore, we now focus on speedup while taking communication time into account. Since the communication time depends on the length of the communicated message, it can be described by

$$\mathrm{c}^{\text{com}}(l) = \alpha^{\text{com}} + \beta^{\text{com}} \cdot l,$$

where α^{com} is the initialization time for parallelism, β^{com} is the communication cost per length of message transmitted, and l is the length of bytes in the message. Both α^{com} and β^{com} are predefined by the computer architecture and are constant. Since each grid in the sparse grid level m consists of 2^m grid points stored as 32 floating point numbers, we can specify $l = 2^m \cdot 32 = 2^{m+5}$. We assume that the communication time is minimized because we only communicate the sum of the subgrids instead of each subgrid solution, and thus we only need to communicate one subsolution. Including the communication cost in the Parareal increases the computation time per iteration by $2N_P\mathrm{c}^{\text{com}}(m)$, since we communicate with each processor twice. Once when we initiate the process and once when the processor returns the result. In each iteration, we have N_τ communications from the parallelism in the serial computation for each time slice, as well as one communication from the parallel update. We get a computation time with communication time given by

$$\begin{aligned}
\mathrm{C}^{\text{com}}_{\text{Parareal}} &= \tilde{\mathrm{C}}^s_{\text{Parareal}} + (k + N_\tau(k+1)) \cdot 2N_P \cdot \mathrm{c}^{\text{com}}(m) \\
&= \frac{\mathrm{C}_1}{2N_P} + \mathrm{C}_2 \cdot 2N_P.
\end{aligned} \tag{3.46}$$

Let all parameters expect N_P be fixed, we search for N_P processors that minimize the computation time for the Parareal with communication

$$N_P^* = \operatorname{argmin}_{N_P \in \mathbb{N}} \mathrm{C}_P^{\mathrm{com}}$$

The result must be a unique positive integer solution. The continuous optimal solution is given by

$$N_P^* = \sqrt{\frac{\mathrm{C}_1}{\mathrm{C}_2}}, \tag{3.47}$$

the optimal integer solution is either $N_P = \lfloor N_P^* \rfloor$ or $N_P = \lceil N_P^* \rceil$. Using the continuous solution a lower bound for the computation time can be derived

$$\mathrm{C}_{\mathrm{Parareal}}^{\mathrm{com}} \geq 2\sqrt{\mathrm{C}_1 \cdot \mathrm{C}_2}$$

and thus an upper bound for the speedup

$$\begin{aligned}
\frac{\mathrm{C}_{\mathrm{Serial}}^{s}}{\mathrm{C}_{\mathrm{Parareal}}^{\mathrm{com}}} &\leq \frac{N_{\mathrm{Serial}} \cdot \mathrm{c}(m)}{2\sqrt{\mathrm{C}_1 \cdot \mathrm{C}_2}} \\
&= \frac{1}{2} \cdot \sqrt{\frac{N_{\mathrm{Serial}}\mathrm{c}(m)}{\mathrm{C}_1 \cdot \mathrm{C}_2}} \\
&= \frac{1}{2} \cdot \sqrt{\frac{N_{\mathrm{Serial}}\mathrm{c}(m)}{\mathrm{C}_1}} \frac{1}{\sqrt{\mathrm{C}_2}} \\
&= \frac{1}{2} \cdot \sqrt{\frac{\mathrm{C}_{\mathrm{Serial}}^{s}}{2N_P\tilde{\mathrm{C}}_{\mathrm{Parareal}}^{s}}} \frac{1}{\sqrt{\mathrm{C}_2}} \\
&= \frac{1}{2\sqrt{N_P\mathrm{C}_2}} \sqrt{\frac{\mathrm{C}_{\mathrm{Serial}}^{s}}{2\tilde{\mathrm{C}}_{\mathrm{Parareal}}^{s}}}
\end{aligned} \tag{3.48}$$

As expected, taking into account the computation time significantly reduces the speedup and therefore has to be considered when choosing N_P.

This approach can easily be extended to multidimensional sparse grids. Then the fine solver would contain only U_m and the coarse solver only the grids at the level U_{m-d-1}. Since more terms cancel out, the speedup would also improve. With the $d-$dimensional combination technique (3.32), the speedup from $\frac{\mathrm{c}(m)}{\tilde{\mathrm{c}}(m-1)}$ enhances to

$$\frac{\mathrm{c}_d(m)}{\tilde{\mathrm{c}}_d(m-d-1)} \gg 1, \tag{3.49}$$

with

$$\begin{aligned} \mathrm{c}_d(m) &= \sum_{q=0}^{d-1} \binom{d-1}{q} \underbrace{\binom{m-q+d-1}{d-1}}_{(3.30)} c(m-q) \\ \tilde{\mathrm{c}}_d(m-1) &= \binom{m-d-2}{d-1} c(m-d-2). \end{aligned} \tag{3.50}$$

Chapter 4

Financial models

The option price depends on the underlying asset model. The dynamics of the asset price can be modeled by a SDE or the corresponding PDE. In the following, we will focus on Put option pricing problems; the derivation for Call options is analogous.

The next question we focus on is how to model the asset, and therefore we present two well-known financial models. The Black-Scholes-(Merton) model is a one-dimensional model and was developed in 1973 [3, 80]; Black and Merton won the Nobel Prize in Economics in 1997 for its derivation, since Scholes died in 1995. The Heston model is a stochastic volatility model and was developed nearly 20 years later and includes a second stochastic dimension, volatility. In the Black-Scholes(-Merton) model, volatility is assumed to be constant. Another extension of the Black-Scholes(-Merton) model is the use of a local volatility function [26, 103]. Thus, the Heston model can be seen as an extension of the Black-Scholes model. These models are standard models in financial research. Of course, there are several other models that consider other parameters as nonconstant, e.g. the Heston-Hull-White model [49] or the Heston model with stochastic correlation [96]. As well as other stochastic volatility models such as the stochastic alpha, beta, rho (SABR) models [64].

4.1 Adjustment for Financial Models

Since the region of interest is around the strike price, we want to gather many grid points there. A common grid transformation is the log-transformation where $x = \log(S)$ and the log-normalized transformation $z = \log(S/K)$ [35, 91]. The log-transformation is used within the Heston model and the log-normalized transformation with the Black-Scholes model.

For the sparse grid approach, a uniform grid on $[0,1]^d$ is assumed and a nonuniform grid for both the asset and the variance is considered. Therefore, we use a uniform grid on $[0,1]^d$, in our case the sparse grid $\Omega_{\mathbf{M_l}}$ with the spatial variable vector $\mathbf{y}$, and reconstruct the nonuniform grid for the transformed spatial variables $\mathbf{x}$ by a smooth transformation acting component wise for for each dimension. Let the transformation be given by

$$\mathbf{x} = \Theta(\mathbf{y}), \tag{4.1}$$

where $[\mathbf{y}^{\min}, \mathbf{y}^{\max}] = [0,1]^d$ is mapped to the arbitrary interval $[\mathbf{x}^{\min}, \mathbf{x}^{\max}]$ with $\mathbf{x}^{\min}, \mathbf{x}^{\max} \in \mathbb{R}^d$ and $\mathbf{x}^{\min} < \mathbf{x}^0 < \mathbf{x}^{\max}$ by

$$\begin{aligned} \mathbf{x} &= \mathbf{x}^0 + \boldsymbol{\gamma} \cdot \sinh\left(\mathbf{y} \cdot (\boldsymbol{\eta} - \boldsymbol{\zeta}) + \boldsymbol{\zeta}\right), \\ \boldsymbol{\zeta} &= \sinh^{-1}\left(\frac{\mathbf{x}^{\min} - \mathbf{x}^0}{\boldsymbol{\gamma}}\right), \\ \boldsymbol{\eta} &= \sinh^{-1}\left(\frac{\mathbf{x}^{\max} - \mathbf{x}^0}{\boldsymbol{\gamma}}\right). \end{aligned} \tag{4.2}$$

Small values of $\boldsymbol{\gamma}$ lead to a very nonuniform grid structure in $\mathbf{x}$, while large values of $\boldsymbol{\gamma}$ lead to a uniform distribution of grid points [56, 94]. Additionally we need Θ^{-1}, for the transformation of the non-uniform financial variable set to the variables on the unit square. We have

$$\mathbf{y} = \Theta^{-1}(\mathbf{x}) = \frac{\boldsymbol{\zeta} + \sinh^{-1}\left(\frac{\mathbf{x}^0 - \mathbf{x}}{\boldsymbol{\gamma}}\right)}{\boldsymbol{\zeta} - \boldsymbol{\eta}} \tag{4.3}$$

These grids are tensor grids. A comparison of the grid transformations w.r.t. S is shown in Figure 4.1. Other transformations that can be considered are spectral methods [44, 7], in some applications a Fast Fourier Transformation is also used [20].

As we have seen, numerical methods, e.g. sparse grids, rely on the smoothness of the solution and hence of the initial data. In financial option pricing, the payoff-function often has discontinuities at the strike price, e.g., European options or Binary options. These discontinuities lead to large errors in the region of interest. Pooley et al. [87] state and compare three different methods for the one-dimensional case of second-order accuracy: averaging the initial conditions, shifting the mesh, and projecting the initial conditions. The supposedly simplest method is mesh shifting. In this method, the strike price is at the midpoint of the neighboring grid points. Then the discontinuity is not visible for the grid and the option price at the strike is recovered by interpolation [94]. Unfortunately, increasing the accuracy by using more grid points isn't as easy as when

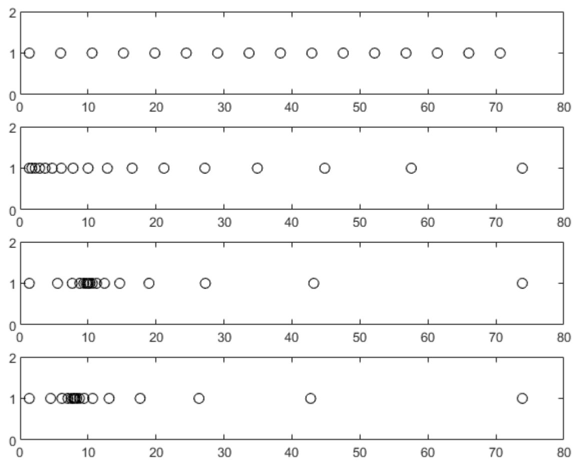


Figure 4.1: Different transformations for $S \in [K \cdot \exp(-2), K \cdot \exp(2)]$ with 17 discrete points, where $K = 10$. The first distribution is a uniform discretization on S, the second shows $S = \exp(x)$ with $x^{\min} = -2$ and $x^{\max} = 2$. The third and fourth distributions show the transformation with the equations (4.1) and $\boldsymbol{\gamma} = 0.25$, where in the third one $\mathbf{y}^0 = K$ and in the last one $\mathbf{y}^0 = \tilde{S} = 8$.

the strike price has to be from the grid even for finer grids. Another simple approach is to average the initial data, where the grid point of the strike price is replaced by an average [65, 98]. Without loss of generality, we denote x_K as the grid point in the asset direction matching the strike price and the corresponding spacing h_K

$$\begin{aligned}\Psi(\mathrm{x}_K, t) &= \frac{1}{(\mathrm{x}_K + 0.5\mathrm{h}_k) - (\mathrm{x}_K - 0.5\mathrm{h}_K)} \int_{\mathrm{x}_K - 0.5\mathrm{h}_K}^{\mathrm{x}_K + 0.5\mathrm{h}_\mathrm{k}} \Psi(\mathrm{v}, t)\,\mathrm{dv} \\ &= \frac{1}{\mathrm{h}_K} \int_{\mathrm{x}_K - 0.5\mathrm{h}_K}^{\mathrm{x}_K + 0.5\mathrm{h}_\mathrm{k}} \Psi(\mathrm{v}, t)\,\mathrm{dv}.\end{aligned} \tag{4.4}$$

An advanced method is the projection of the initial condition [88, 73], where a L_2 projection onto a set of basis functions is performed for the initial payoff profile. Adapting projection and mesh shifting approach for the d-dimensional case is nontrivial. For the grid shift, it's difficult to guarantee the desired structure in higher dimensions. The projection faces difficulties when the discontinuity is not on a grid point. The averaging function has been successfully applied in the two- and three-dimensional case [23, 42] and can be easily extended to arbitrary dimensions via a tensor product of one-dimensional convolutions [40]. Since the problems presented in the next sections have only up to two dimensions in space and the payoff-functions considered are independent of the second spatial dimension, we restrict ourselves to the approach of Kreiss et al. and use (4.4) with adjustments given corresponding to the chosen boundary conditions, see Section 4.4.

4.2 The Black-Scholes Model

To derive their model, Black and Scholes required the following market conditions

- There are no arbitrage opportunities.
- The market is frictionless.
- The asset price follows a geometric Brownian motion.
- The risk-free interest rate $r > 0$ and the volatility $\sigma_s > 0$ of the asset are constant for $0 \leq t \leq T$. No dividends $D > 0$ are paid during this period.
- The option is European.

The Black-Scholes model is based on a geometric Brownian motion for the dynamics of the price

$$\mathrm{d}S_t = rS_t\,dt + \sigma_s S_t\,\mathrm{dW}_t^S, \quad S_0 > 0. \tag{4.5}$$

Therefore, in the Black-Scholes model $\mathbf{x} = S$. In the case of a continuous dividend D, we consider the Black-Scholes-Merton model [80] with the drift term $(r - D)S_t$. Using the reversed time $\tau = T - t$, the Black-Scholes PDE formulation for the fair price $u(\mathbf{x}, \tau) = u(s, \tau)$ of a European put option is

$$\frac{\partial u}{\partial \tau}(s,\tau) = \mathcal{L}_{\mathrm{BS}}\Big[u\Big](s,\tau) \quad 0 < \tau \leq T, \tag{4.6}$$

with the spatial operator

$$\mathcal{L}_{\mathrm{BS}}\Big[u\Big] = \frac{1}{2}\sigma_s s^2 \frac{\partial^2 u}{\partial s^2} + rs\frac{\partial u}{\partial s} - ru, \quad s \geq 0, \tag{4.7}$$

and the initial condition at $\tau = 0$

$$u(s,0) = \max(K - s, 0), \quad s \geq 0. \tag{4.8}$$

At $s = 0$ and $s \to \infty$ Dirichlet boundary conditions are considered

$$s = 0: \quad u(0,\tau) = K\exp(-r\tau), \qquad s \to \infty: \quad u(s,\tau) = 0. \tag{4.9}$$

The Black-Scholes PDE has a semi-analytical solution because it can be transformed into the heat equation for which the semi-analytic solution exists. The semi-analytical solution for a Put option with initial condition in (4.8) and the boundary conditions in equation

(4.9) is given by

$$u(s, T-\tau) = K\exp(-r\tau)\Phi(-\mathtt{d}_2) - s\Phi(-\mathtt{d}_1), \quad s > 0,\ 0 \le \tau < T, \tag{4.10}$$

where Φ is a probability function of the standard normal distribution

$$\Phi(\mathtt{x}) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\mathtt{x}} \exp(-\mathrm{v}^2/2)\,\mathrm{dv}, \quad \mathtt{x} \in \mathbb{R}, \tag{4.11}$$

and $\mathtt{d}_1$ and $\mathtt{d}_2$ are given by

$$\begin{aligned} \mathtt{d}_1 &= \frac{\log(s/K) + (r + \frac{1}{2}\sigma_s^2)\tau}{\sigma_s\sqrt{\tau}}, \\ \mathtt{d}_2 &= \frac{\log(s/K) + (r - \frac{1}{2}\sigma_s^2)\tau}{\sigma_s\sqrt{\tau}}. \end{aligned}$$

The American put option problem can be divided into two regions. In the one-dimensional Black-Scholes model, the exercise region is determined by $0 \le s \le s_f(\tau)$ and the holding region is $s > s_f(\tau)$. In the holding region, the American put option problem with the Black-Scholes equation is equivalent to the European put option problem (4.6). For the other region, $u(s,\tau) = \max(K-s, 0)$ must hold. Substituting this condition into (4.6) we get

$$\frac{\partial}{\partial\tau}\max(K-s,0) - \mathcal{L}_{\mathrm{BS}}[\max(K-s,0)] = rs + r(K-s) = rK > 0, \tag{4.12}$$

that satisfies the condition from Definition 1.18. We get a case system

$$\frac{\partial}{\partial\tau}u(s,\tau) - \mathcal{L}_{\mathrm{BS}}[u](s,\tau) = g(s,\tau) = \begin{cases} rK, & 0 < s \le s_f(\tau), \\ 0, & s > s_f(\tau). \end{cases} \tag{4.13}$$

To simplify the computation, we transform the case system (4.13) in such a way that we obtain the heat equation on the left side. This transformation is known from the European options, since the heat equation has a semi-analytical solution. Since our right-hand side is unequal to zero in contrast to the European option case, we sketch the transformation in more detail. For the transformation of the case system (4.13), we use the well-known transformations

$$z = \log\left(\frac{s}{K}\right), \quad \hat{\tau} = \frac{\sigma_s^2}{2}\tau, \quad v(z,\hat{\tau}) = \frac{u(s,\tau)}{K}, \quad \hat{T} = \frac{\sigma_s^2}{2}T, \tag{4.14}$$

and obtain by using the chain rule

$$-\frac{\sigma_s^2}{2}K\frac{\partial v}{\partial \hat{\tau}}+\frac{\sigma_s^2}{2}s^2\frac{K}{s^2}\left(\frac{\partial^2 v}{\partial z^2}-\frac{\partial v}{\partial z}\right)+rs\frac{K}{s}\frac{\partial v}{\partial z}-rKv=\begin{cases} rK, & z\le z_f(\hat{\tau}),\\ 0, & z>z_f(\hat{\tau}),\end{cases}$$

where $z_f(\hat{\tau})=\log(s_f(\hat{\tau})/K)$ is the free boundary value for z. We simplify this equation and divide by $-\sigma_s/2$

$$\frac{\partial v}{\partial \hat{\tau}}-\frac{\partial^2 v}{\partial z^2}+\left(1-\frac{2r}{\sigma_s^2}\right)\frac{\partial v}{\partial z}+\frac{2r}{\sigma_s^2}v=\begin{cases} -\frac{2r}{\sigma_s^2}, & z\le z_f(\hat{\tau}),\\ 0, & z>z_f(\hat{\tau}).\end{cases} \tag{4.15}$$

In the last transformation step we use the transformation $w(z,\hat{\tau})=\exp(\alpha z+\beta\hat{\tau})v(z,\hat{\tau})$, where

$$\alpha=-\frac{1}{2}\left(\frac{2r}{\sigma_s^2}-1\right)\text{ and }\beta=-\frac{1}{4}\left(\frac{2r}{\sigma_s^2}+1\right)^2. \tag{4.16}$$

Finally, we get the transformed case system

$$\frac{\partial w}{\partial \hat{\tau}}-\frac{\partial^2 w}{\partial z^2}=\hat{g}(z,\hat{\tau})=\exp(-\alpha z-\beta\hat{\tau})\begin{cases} -\frac{2r}{\sigma_s^2} & z\le z_f(\hat{\tau}),\\ 0, & z>z_f(\hat{\tau}).\end{cases} \tag{4.17}$$

The transformed initial condition reads

$$\hat{\phi}(z,\hat{\tau})=\exp\Big(-\alpha z-\beta\hat{\tau}\Big)\ \max\Big(1-\exp(z),0\Big). \tag{4.18}$$

The obtained transformed American put option price problem in a case system formulation is

$$w_{\hat{\tau}}-w_{zz}=\hat{g}(z,\hat{\tau}), \tag{4.19}$$

supplied with the initial and boundary conditions

$$w(z,0)=\hat{\phi}(z,0),\ z\in\mathbb{R},\quad \lim_{z\to\pm\infty}\Big(w(z,\hat{\tau})-\hat{\phi}(z,\hat{\tau})\Big)=0,\quad 0\le\hat{\tau}\le\hat{T}. \tag{4.20}$$

As penalty term, we choose an affine function $\delta(\tau)$. Since at $\tau=0$ the initial condition (4.8) and (4.12) must hold, we obtain

$$\delta(\tau)=a\tau+1, \tag{4.21}$$

so that the penalty function is given by

$$p(s,\tau) = \delta(\tau) \cdot g(s,\tau). \tag{4.22}$$

This choice preserves the novelty of this approach, since the known penalty terms are neither bounded nor independent of the solution itself [83, 35]. The penalized case system for (4.13) is

$$\frac{\partial}{\partial \tau} u(s,\tau) - \mathcal{L}_{\text{BS}}[u](s,\tau) = \delta(\tau) \cdot g(s,\tau). \tag{4.23}$$

For the transformed system we also choose an affine function $\hat{\delta}(\hat{\tau}) = \hat{a}\hat{\tau} + \hat{b}$ as a penalization term for the penalty function $\hat{p} = \hat{\delta}(\hat{\tau})\hat{g}(z,\hat{\tau})$ for the transformed case system (4.17). Since the relation between $w(z,\hat{\tau})$ and $p(s,\tau)$ is given by

$$w(z,\hat{\tau}) = \frac{1}{K} \exp\Big(-\alpha z - \beta\hat{\tau}\Big) u(s,\tau)$$

the same relation must hold for the penalized right hand side. We focus only on the exercise region, since the holding region is always zero. Since we go backward in time with t and forward with τ, we can assume $\frac{1}{K}\exp(-\alpha z - \beta\hat{\tau})p(s,\tau) = -\hat{p}(z,\hat{\tau})$. We get

$$\frac{1}{K}\exp(-\alpha z - \beta\hat{\tau})(a\tau + 1) \cdot (-rK) = -(\hat{a}\tau + \hat{b})\exp(-\alpha z - \beta\hat{\tau})\frac{2r}{\sigma_s^2} \tag{4.24}$$

$$a\tau + 1 = (\hat{a}\hat{\tau} + b)\frac{2}{\sigma_s^2} \tag{4.25}$$

$$a\tau + 1 = (\hat{a}\frac{\sigma_s^2}{2}\tau + b)\frac{2}{\sigma_s^2} \tag{4.26}$$

$$a\tau + 1 = \hat{a}\tau + \frac{2}{\sigma_s^2} b \tag{4.27}$$

Since the equation has to hold at $\tau = 0$ as, well we get

$$\frac{\sigma_s^2}{2} = b. \tag{4.28}$$

The transformed case system with the bounded penalty function is given by

$$\frac{\partial w}{\partial \tau} - \frac{\partial^2 w}{\partial z^2} = \hat{p}(z,\hat{\tau}) = \Big(\hat{a}\hat{\tau} + \frac{\sigma_s^2}{2}\Big) \begin{cases} -\frac{2r}{\sigma_s^2}\exp(-\alpha z - \beta\hat{\tau}), & z \le z_f \\ 0, & z > z_f \end{cases} \tag{4.29}$$

supplied with the initial and boundary conditions from (4.20).

Our approach requires an initial guess for the free boundary value. Since the price of an American option is greater than or equal to the price of the corresponding European option, we can use the intersection between the payoff and the solution of the European put option as the initial guess for the free boundary value z_f, with the result that the bounded penalty term forces the PDE (4.23) to satisfy the conditions from the LCP in Definition 1.18 asymptotically. Since the free boundary approximation and the analysis of the approximation is one of the main research areas in the field of American options, well-known approximation formulas can be found in the literature. Just to mention the approximations of Zhou [108], Starmicar [93] and Evans et al. [27]. Each of these formulations can be considered to obtain the initial values. Since the penalization is based on the initial choice of free boundary, the choice of formula for computing the initial free boundary has a large effect on the accuracy of the method. A detailed analysis of the choice of the initial free boundary computed by different approximation formulas from the literature is part of future research. After calculating the penalty term with the initial free boundary value, we solve the penalized heat equation (4.29). The solution obtained is the solution of the American option problem.

4.3 The Heston Model

The Heston model was developed by Heston in 1993 [46] and describes the dynamics of the underlying asset by a two-dimensional SDE that includes a stochastic process for the underlying asset S and one for the volatility σ_s. This model is an extension of the well-known Black-Scholes-Merton model [3, 80], in that it assumes constant volatility. Instead of the volatility itself, he considered a stochastic process for the variance Υ, since by definition the variance is the square of the volatility of the asset, $\Upsilon = \sigma_s^2$. Heston modeled the variance with a mean reversion process, the Cox-Ingersoll-Ross process. The SDE system of Heston's model under the risk-neutral measure is

$$\begin{cases} \mathrm{d}S_t &= (r-D)S_t\,\mathrm{d}t + \sqrt{\Upsilon_t}S_t\,\mathrm{dW}_t^S, \quad S_0 > 0, \\ \mathrm{d}\Upsilon_t &= \kappa(\mu - \Upsilon_t)\,\mathrm{d}t + \sigma_v\sqrt{\Upsilon_t}\,\mathrm{dW}_t^\Upsilon, \quad \Upsilon_0 > 0, \\ \mathrm{dW}_t^S\mathrm{dW}_t^\Upsilon &= \rho\,\mathrm{d}t \end{cases} \tag{4.30}$$

where $\kappa > 0$ is the mean reversion rate, $\mu > 0$ is the long-term mean, and $\sigma_v > 0$ is the volatility-of-variance. Note that we are working with risk-neutralized pricing probabilities [46]. Since the Brownian motions are correlated by the constant $\rho \in [-1, 1]$, we rewrite the SDE system, such that the Brownian motions are independent. Otherwise, Monte Carlo methods can't be applied directly. From (4.30) we get the *symmetric correlation* *matrix*

$$\mathrm{Corr} = \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix} \tag{4.31}$$

with its Cholesky decomposition $\mathrm{Corr} = \mathrm{Col} \cdot \mathrm{Col}^\top$

$$\mathrm{Col} = \begin{pmatrix} 1 & 0 \\ \rho & \sqrt{1-\rho} \end{pmatrix}. \tag{4.32}$$

This decomposition is used to reformulate the SDE system (4.30) w.r.t. the two independent Brownian motions W_t^1 and W_t^2

$$\begin{cases} \mathrm{d}S_t = (r-D)S_t\,\mathrm{d}t + \sqrt{\Upsilon_t}S_t\,\mathrm{dW}_t^1, & S_0 > 0, \\ \mathrm{d}\Upsilon_t = \kappa(\mu - \Upsilon_t)\,\mathrm{d}t + \rho\sigma_v\sqrt{\Upsilon_t}\,\mathrm{dW}_t^1 + \sqrt{1-\rho}\sigma_v\sqrt{\Upsilon_t}\,\mathrm{dW}_t^2, & \Upsilon_0 > 0, \end{cases}. \tag{4.33}$$

We observe that the family of symmetric instantaneous covariance matrices for $\mathbf{X}_t = (S_t, \Upsilon_t)$ reads as follows

$$\boldsymbol{\sigma}(\mathbf{X}_t)\boldsymbol{\sigma}(\mathbf{X}_t)^\top = \begin{pmatrix} \Upsilon_t S_t^2 & \rho\sigma_v\Upsilon_t S_t \\ \rho\sigma_v\Upsilon_t S_t & \sigma_v^2\Upsilon_t \end{pmatrix}. \tag{4.34}$$

For the variance process to be positive, the Feller condition $2\kappa\mu \geq \sigma_v^2$ must be satisfied. If the condition is violated, problems arise in computing the square root because it is complex. For Asian options, we obtain the fair price by

$$\omega((S,v)),0) = \exp(-rT)E[\phi^{\mathrm{As}}(A(S))]. \tag{4.35}$$

Using the Feynman-Kac formula (1.8) and time reversal, we derive the Heston PDE formulation for the fair price of a European put option $u(\mathbf{x},\tau) = u\Big((v,s),\tau\Big)$ under the risk-neutral measure

$$\frac{\partial}{\partial\tau}u\Big(\mathbf{x},\tau\Big) = \mathcal{L}_{\mathrm{H}}\Big[u\Big]\Big(\mathbf{x},\tau\Big), \tag{4.36}$$

with the spatial operator

$$\mathcal{L}_{\mathrm{H}}[u] = \frac{v}{2}s^2\frac{\partial^2 u}{\partial s^2} + \frac{1}{2}\sigma_v^2 v\frac{\partial^2 u}{\partial v^2} + (r-D)s\frac{\partial u}{\partial s} + \kappa(\mu - v)\frac{\partial u}{\partial v} + \sigma_v v s\rho\frac{\partial^2 u}{\partial s\partial v} - ru. \tag{4.37}$$

and initial condition given by the payoff-function for the European put option (1.1).

Heston proposed boundary conditions for Put options [46].

$$s = 0: \quad u\Big((\upsilon, 0), \tau\Big) = K \exp\Big(-r\tau\Big) \tag{4.38}$$

$$s \to \infty: \quad u\Big((\upsilon, s), \tau\Big) \sim 0 \tag{4.39}$$

$$\upsilon = 0: \quad \frac{\partial}{\partial \tau} u\Big((s, 0), \tau\Big) = (r - D)s \frac{\partial}{\partial s} u\Big((0, s), \tau\Big) + \kappa\mu \frac{\partial}{\partial \upsilon} u\Big((0, s), \tau\Big) - ru\Big((0, s), \tau\Big) \tag{4.40}$$

$$\upsilon \to \infty: \quad u\Big((\upsilon, s), \tau\Big) \sim K \exp\Big(-r\tau\Big). \tag{4.41}$$

For the Heston model, we observe the advection term $\mathrm{ad}(\upsilon) = \kappa(\mu - \upsilon)$ in the coordinate direction υ. This term has a sign change at $\mathrm{ad}(\upsilon) = 0$, which corresponds to the point where $\upsilon = \mu$. Thus, the advection direction changes at $\upsilon = \mu$. Therefore we later use the combined stencil for the differentiation within this coordinate direction.

At $S = 0$ and $\upsilon = 0$, the diffusion terms vanish and we use Fichera theory to determine the necessity of boundary conditions [4, 66]. Therefore we rewrite the Heston PDE (4.36) into divergence form (1.15) and from the underlying operator $\mathcal{L}_{\mathrm{H}}$ we get

$$\mathbf{A}_{\mathrm{H}} = \frac{1}{2}\upsilon \begin{pmatrix} \rho\sigma_\upsilon s & \sigma^2 \\ s^2 & \rho\sigma_\upsilon s \end{pmatrix}, \quad \mathbf{b}_{\mathrm{H}} = -\begin{pmatrix} \kappa(\mu - \upsilon) - \frac{1}{2}\sigma_\upsilon^2 - \frac{1}{2}\rho\sigma_\upsilon \upsilon \\ (r - D)s - \upsilon s - \frac{1}{2}\rho\sigma_\upsilon s \end{pmatrix}. \tag{4.42}$$

At the boundary $s = 0$ the *Fichera condition* is given by

$$\lim_{s \to 0^+} \mathbf{b}_{\mathrm{H}} \cdot \begin{pmatrix} 0 \\ -1 \end{pmatrix} = \Big((r - D)s - \upsilon s - \frac{1}{2}\rho\sigma_\upsilon s\Big) = 0.$$

This corresponds to an (unconditional) outflow boundary, and thus no boundary condition is required from an analytical point of view. Nevertheless, a natural boundary condition arises from the financial context, which is used as a closure condition within the Section 4.4. Considering the boundary $\upsilon = 0$, the *Fichera condition* is given by

$$\lim_{\upsilon \to 0^+} \mathbf{b}_{\mathrm{H}} \cdot \begin{pmatrix} -1 \\ 0 \end{pmatrix} = \lim_{\upsilon \to 0^+} \Big(\kappa(\mu - \upsilon) - \frac{1}{2}\sigma_\upsilon^2 - \frac{1}{2}\rho\sigma_\upsilon \upsilon\Big) = \kappa\mu - \frac{1}{2}\sigma_\upsilon^2. \tag{4.43}$$

Hence, we have the following cases at $\upsilon = 0$:

- outflow boundary: if $\kappa\mu \geq \frac{1}{2}\sigma_\upsilon^2$ we must not supply any analytical boundary condition at $\upsilon = 0$.
- inflow boundary: if $\kappa\mu < \frac{1}{2}\sigma_\upsilon^2$ we have to supply an analytical boundary condition at $\upsilon = 0$.

We obtain an outflow boundary if and only if the Feller condition is satisfied, which is assumed below. However, for the implementation we need to specify a numerical closure condition for $s = 0$ and $\upsilon = 0$. Since at $s = 0$ a closure condition can be derived directly from the financial model, we use the Dirichlet condition proposed by Heston, which is also used in the Black-Scholes model. A discussion of the choice of closure conditions at $\upsilon = 0$ is given in [15].

The American put option problem is treated as an LCP and is therefore solved using the reformulation of Ikonen and Toivanen [54, 55, 60] with an auxiliary variable $\boldsymbol{\lambda}$

$$\begin{cases} \frac{\partial}{\partial \tau} u\Big((\upsilon, s), \tau\Big) - \mathcal{L}_{\mathrm{H}}\Big[u\Big((\upsilon, s), \tau\Big)\Big] = \boldsymbol{\lambda}, \\ \boldsymbol{\lambda} \geq 0, \quad u\Big((\upsilon, s), \tau\Big) - \max(K - s, 0) \geq 0, \\ \Big(u\Big((\upsilon, s), \tau\Big) - \max(K - s, 0)\Big)\boldsymbol{\lambda} = 0, \end{cases} \tag{4.44}$$

for $\Big((\upsilon, s), \tau\Big) \in \Omega_2 \times [0, T]$ with the initial and boundary conditions [55]. The result is a mixed formulation of the LCP problem, where $\boldsymbol{\lambda}$ plays the role of a Lagrange multiplier. The advantage of the LCP formulation of the American option problem is that it avoids explicit computation of the free boundary value $(\upsilon, s)_f$.

4.4 Transformations of the Heston Model

To zoom into the particularly interesting price range near K, where the region changes from ITM to ATM and further to OTM, we use the variable transformation $X = \log(S)$ for the asset. As usually the spot asset is near the strike. We get the log-*transformed Heston SDE*

$$\begin{cases} \mathrm{d}X_t = (r - D - \frac{1}{2}\Upsilon_t)\,\mathrm{d}t + \sqrt{\Upsilon_t}\,\mathrm{dW}_t^X, \\ \mathrm{d}\Upsilon_t = \kappa(\mu - \Upsilon_t)\,\mathrm{d}t + \sigma_\upsilon\sqrt{\Upsilon_t}\,\mathrm{dW}_t^\Upsilon, \\ \mathrm{dW}_t^X\,\mathrm{dW}_t^\Upsilon = \rho\,\mathrm{d}t \end{cases} \tag{4.45}$$

on the semi-unbounded domain $X \in \mathbb{R}$, $\Upsilon \geq 0$, $0 \leq t \leq T$ with the initial condition given by the corresponding payoff-function. Similar to the original system, the log-transformed system can be rewritten in terms of independent Brownian motions

$$\begin{cases} \mathrm{d}X_t = (r - D - \frac{1}{2}\Upsilon_t)\,\mathrm{d}t + \sqrt{\Upsilon_t}\,\mathrm{dW}_t^1, \\ \mathrm{d}\Upsilon_t = \kappa(\mu - \Upsilon_t)\,\mathrm{d}t + \rho\sigma_\upsilon\sqrt{\Upsilon_t}\,\mathrm{dW}_t^1 + \sqrt{1-\rho}\sigma_\upsilon\sqrt{\upsilon_t}\,\mathrm{dW}_t^2. \end{cases} \tag{4.46}$$

and leads to the corresponding log-transformed Heston PDE under the risk-neutral measure for European options with the spatial operator $\mathcal{L}_{\bar{\mathrm{H}}}$

$$\mathcal{L}_{\bar{\mathrm{H}}}[u] = \frac{\upsilon}{2}\frac{\partial^2 u}{\partial x^2} + \frac{1}{2}\sigma_\upsilon^2 \upsilon \frac{\partial^2 u}{\partial \upsilon^2} + \left(r - D - \frac{\upsilon}{2}\right)\frac{\partial u}{\partial x} + \kappa(\mu - \upsilon)\frac{\partial u}{\partial \upsilon} + \sigma_\upsilon \upsilon \rho \frac{\partial^2 u}{\partial x \partial \upsilon} - ru, \tag{4.47}$$

where $\mathbf{x} = (\upsilon, x)$ and $u(\mathbf{x}, \tau)$ is the fair price of the option. The payoff-function as the initial condition must also be transformed, in the case of the European put option we get

$$\bar{\phi}(x) = \max\Big(K - \exp(x), 0\Big). \tag{4.48}$$

We consider the boundary conditions proposed by Heston and apply the log-transformation to obtain

$$x \to -\infty : \quad u\Big((\upsilon, x), \tau\Big) \sim K\exp(-r\tau), \tag{4.49}$$

$$x \to \infty : \quad u\Big((\upsilon, x), \tau\Big) \sim 0, \tag{4.50}$$

$$\upsilon = 0 : \quad \frac{\partial}{\partial \tau}u\Big((0, x), \tau\Big) = (r - D - \frac{\upsilon}{2})\frac{\partial}{\partial x}u\Big((0, x), \tau\Big) + \kappa\mu\frac{\partial}{\partial \upsilon}u\Big((0, x), \tau\Big) - ru\Big((0, x), \tau\Big), \tag{4.51}$$

$$\upsilon \to \infty : \quad u\Big((\upsilon, x), \tau\Big) \sim K\exp(-r\tau). \tag{4.52}$$

Heston also proposed a closed-form solution [46]. In this formulation, the PDE only degenerates to a first-order hyperbolic PDE at $\upsilon = 0$. Therefore, a boundary condition at $x \to -\infty$ is needed and is already provided by Heston. Further, we need to consider the Fichera theory [4, 66] to assess whether it is necessary to provide an analytic boundary condition at $\upsilon = 0$ or not, again. From the divergent form, we get

$$\mathbf{A}_{\bar{\mathrm{H}}} = \frac{1}{2}\upsilon \begin{pmatrix} \sigma_\upsilon^2 & \sigma_\upsilon\rho \\ \sigma_\upsilon\rho & 1 \end{pmatrix}, \qquad \mathbf{b}_{\bar{\mathrm{H}}} = \begin{pmatrix} \kappa(\mu - \upsilon) - \frac{1}{2}\sigma_\upsilon^2 \\ r - D - \frac{\upsilon}{2} - \frac{1}{2}\sigma_\upsilon\rho \end{pmatrix}. \tag{4.53}$$

The Fichera condition at $\upsilon = 0$

$$\lim_{\upsilon \to 0^+} \mathbf{b}_{\bar{\mathrm{H}}} \cdot \begin{pmatrix} -1 \\ 0 \end{pmatrix} = \kappa(\mu - \upsilon) - \frac{1}{2}\sigma_\upsilon^2 \tag{4.54}$$

is the same as before, thus at $\upsilon = 0$ we get an outflow boundary if the Feller condition holds, otherwise we get an inflow boundary. In addition to the Heston PDE with constant parameters for the variance process, we also consider time-dependent parameters $\tilde{\kappa}$, $\tilde{\mu}_\upsilon$,

$\tilde{\sigma}_\upsilon$, and $\tilde{\rho}$, and the corresponding spatial operator $\mathcal{L}_{\tilde{\mathrm{H}}}$ for $u\Big((\upsilon,x),\tau\Big)$ which is

$$\mathcal{L}_{\tilde{\mathrm{H}}}[u] = \frac{\upsilon}{2}\frac{\partial^2 u}{\partial x^2} + \frac{1}{2}\tilde{\sigma}_\upsilon^2 \upsilon \frac{\partial^2 u}{\partial \upsilon^2} + (r - D - \frac{\upsilon}{2})\frac{\partial u}{\partial x} + \tilde{\kappa}(\tilde{\mu} - \upsilon)\frac{\partial u}{\partial \upsilon} + \sigma_\upsilon t \upsilon \tilde{\rho}\frac{\partial^2 u}{\partial x \partial \upsilon} - ru, \quad (4.55)$$

with the initial condition (5.5) and the boundary conditions from equations (4.49), (4.50) and (4.52). From the Fichera theory we gain, that the Feller condition must hold for all τ, such that we get a pure outflow boundary again. Therefore even if for the log-transformed Heston model with and without constant parameters, we don't need an analytical boundary condition at $\upsilon = 0$ as long as the Feller condition holds. However, once we compute the solutions, we need closure conditions when we truncate the domain to $[\upsilon^{\min}, \upsilon^{\max}] \times [x^{\min}, x^{\max}]$. Note that the boundary condition at $x \to \infty$ must be interpolated w.r.t. the variance boundary conditions. For the closure condition at $\upsilon = \upsilon^{\min}$ we consider two different cases. The Dirichlet boundary condition

$$u\Big((\upsilon^{\min}, x), \tau\Big) = \bar{\phi}(\mathbf{x}) \exp(-r\tau) \quad \text{(a)}$$

and, since we have a pure outflow boundary here, an extrapolation via a ghost layer at $\upsilon = \upsilon^{\min} - \mathrm{h}_1$ leads to

$$u\Big((\upsilon^{\min} - \mathrm{h}_1, x), \tau\Big) = u\Big((\upsilon^{\min}, x), \tau\Big). \quad \text{(b)}$$

For the boundary conditions at $\upsilon = \upsilon^{\max}$, we consider four different cases. The first condition was proposed by Heston himself

$$u\Big((\upsilon^{\max}, x), \tau\Big) = K \exp(-r\tau). \quad \text{(c)}$$

This boundary condition causes a jump between $u\Big((\upsilon, x^{\max}), \tau\Big)$ and $u\Big((\upsilon^{\max}, x^{\max}), \tau\Big)$. Therefore, one can use the linear interpolation considered in case (a) or an exponential fit at $x^{\max}$ [76] with parameter $\gamma > 0$ given by

$$\boldsymbol{\vartheta} = 1 + \boldsymbol{\nu}, \qquad \text{and} \qquad \boldsymbol{\nu} = \frac{\Big(\exp(\upsilon - \upsilon^{\max})\Big)^z}{1 - \Big(\exp(\upsilon - \upsilon^{\max})\Big)^\gamma}, \quad (4.56)$$

which leads to the condition

$$u\Big((\upsilon^{\max}, x), \tau\Big) = K \exp(-r\tau), \quad u\Big((\upsilon, x_{\max}), \tau\Big) = \boldsymbol{\vartheta}\Big(\exp(\upsilon - \upsilon^{\max})\Big)^z - \upsilon. \quad \text{(d)}$$

Note that a high value for γ corresponds to a slope where the option price is zero for most variance values, resulting in a better fit to Heston's boundary conditions. In the following, we use the rather large value $\gamma = 20$. Another approach [76] is to include a dependence on x in the Heston condition, for example

$$u\Big((\upsilon^{\max}, x), \tau\Big) = \exp(-r\tau)\left(1 - \frac{\exp(x) - \exp(x^{\min})}{2\Big(\exp(x^{\max}) - \exp(x^{\min})\Big)}\right) \tag{e}$$

combined with a linear interpolation in $x^{\max}$.

The final approximation at $\upsilon = \upsilon^{\max}$, we consider the proposition by Kùtik and Mikula [66] and use artificial homogeneous Neumann boundary conditions. The choice is motivated by the independence from the variance of Heston's original boundary conditions, given a sufficiently large $\upsilon^{\max} = \mathcal{O}(1)$. Thus, we perform an extrapolation over the ghost layer at $\upsilon = \upsilon^{\max} + \mathrm{h}_1$ to obtain

$$u\Big((\upsilon^{\max}, x), \tau\Big) = u\Big((\upsilon^{\max} + \mathrm{h}_1, x), \tau\Big). \tag{f}$$

The second transformation considered in this thesis for the Heston model is the transformation to the unit square (4.2)to apply the sparse grid approach with the Heston model. We consider $\mathbf{x} = (\upsilon, s)$ and transform it to $\mathbf{y} = (\mathrm{y}_1, \mathrm{y}_2)$ with $\mathbf{y} \in [0,1]^2$ For the grid transformation one either uses $\mathbf{x}^0 = (\upsilon^{\text{spot}}, K)$ or the spot asset $\mathbf{x}^0 = (, \upsilon^{\text{spot}}, s^{\text{spot}})$ and generates $\mathbf{x}$ from $\mathbf{y}$ in $[0,1]^d$. Note that when using the first approach, the strike price is definitely a grid point and therefore a smoothing technique is required. Further, for the first choice more smoothing techniques can be applied since the requirement of the strike price matching a grid point is satisfied. The spatial operator for $u(\mathbf{y}, \tau)$ is

$$\begin{aligned}\mathcal{L}_{\hat{\mathrm{H}}}[u] = \frac{1}{2}\upsilon s^2 \left(\frac{\partial \Theta^{-1}(\mathbf{x})}{\partial s}\right)^2 \frac{\partial^2 u}{\partial \mathrm{y}_2{}^2} + \left((r - D)s\frac{\partial \Theta^{-1}(\mathbf{x})}{\partial s} + \frac{1}{2}\upsilon s^2 \frac{\partial^2 \Theta^{-1}(\mathbf{x})}{\partial s^2}\right)\frac{\partial u}{\partial \mathrm{y}_2} \\ + \rho \frac{\partial \Theta^{-1}(\mathbf{x})}{\partial s}\frac{\partial \Theta^{-1}(\mathbf{x})}{\partial \upsilon}\sigma_\upsilon s \upsilon \frac{\partial^2 u}{\partial \mathrm{y}_1 \partial \mathrm{y}_2} + \frac{1}{2}\sigma_\upsilon^2 \upsilon \left(\frac{\partial \Theta^{-1}(\mathbf{x})}{\partial \upsilon}\right)^2 \frac{\partial^2 u}{\partial \mathrm{y}_1{}^2} \\ + \left(\kappa(\mu_\upsilon - \boldsymbol{\nu})\frac{\partial \Theta^{-1}(\mathbf{y})}{\partial \upsilon} + \frac{1}{2}\sigma_\upsilon^2 \upsilon \frac{\partial^2 \Theta^{-1}(\mathbf{x})}{\partial \upsilon^2}\right)\frac{\partial u}{\partial \mathrm{y}_1}\end{aligned} \tag{4.57}$$

with

$$
\begin{aligned}
\frac{\partial \Theta^{-1}(\mathbf{x})}{\partial \mathrm{x}_i} &= -\frac{1}{\gamma_i \sqrt{\frac{(\mathrm{x}_i^0 - \mathrm{x}_i)^2}{\gamma_i^2} + 1}} \\
\frac{\partial^2 \Theta^{-1}(\mathbf{x})}{\partial {\mathrm{x}_i}^2} &= -\frac{\mathrm{x}_i^0 - \mathrm{x}_i}{\gamma_i^3 \left({\frac{(\mathrm{y}_i^0 - \mathrm{x}_i)^2}{\gamma_i}}^2 + 1 \right)^{\frac{3}{2}}}
\end{aligned}
\tag{4.58}
$$

for $i = 1, \ldots, d$. The sparse grid approach requires Dirichlet boundary conditions, so we are limited in the choice of boundary conditions. We use the closure conditions

$$
\begin{aligned}
\mathrm{y}_1 = 0: \quad & u\Big(\mathbf{x}, \tau\Big) = \phi(\Theta^{-1}(\mathrm{y}_2)) \exp(-r\tau) \\
\mathrm{y}_1 = 1 \quad & u\Big(\mathbf{x}, \tau\Big) = K \exp(-r\tau), \\
\mathrm{y}_2 = 0: \quad & u\Big(\mathbf{x}, \tau\Big) = K \exp(-r\tau), \\
\mathrm{y}_2 = 1: \quad & u\Big(\mathbf{x}, \tau\Big) = 0.
\end{aligned}
\tag{4.59}
$$

Chapter 5

Space Mapping

In finance, calibrating model parameters to fit real market data is challenging because most model parameters are implicit in the real market data [20, 48, 71, 79, 81, 96]. The Heston model contains at least four parameters implicit in the market data, namely $\kappa, \mu, \sigma_v, \rho$ and sometimes a fifth parameter, the spot volatility v^{spot}. It has a closed-form valuation formula for this model. Some calibration techniques are based on this formula [81, 20].

The purpose of space mapping is to optimize (or calibrate) an accurate and computationally expensive model (fine) that can be optimized using a surrogate model (coarse) for which efficient optimization algorithms are available. Our control problem is the calibration of the parameters for an Asian option under the Heston model with SDE representation. More specifically, while we want to calibrate the parameters of the SDE for the Asian put option, we will optimize the deterministic PDE model of the European option, which can be solved using techniques from optimization with PDE. In each iteration of the calibration process, we will evaluate the SDE to compute the residuum of the two models. We measure the difference between the fair price given by the numerical solution of our model and the reference data, the subsequent market data, in the cost function. The PDE-constrained optimization problem is solved using a gradient descent method. We formally derive an adjoint-based gradient descent algorithm for the Heston PDE model.

5.1 The Space Mapping Approach

In our case, the accurate model is the Heston SDE for Asian put option pricing and the coarse model is the Heston PDE for European put option pricing. Since we are using the log-transformed Heston model as our representation, we restate the corresponding problems.

Definition 13 (Asian Put Option Problem with the log-transformed Heston Model)**.** For Asian options, the fair price $\omega^{\mp}\big(X, X^{-}, t\big)$ today at $t = 0$ is given by

$$\omega^{\mp}\Big(X, X^{-}, 0\Big) = \exp(-rT)\, E\Big(\frac{1}{2}\phi^{\mathrm{As}}\Big(A(\exp(X))\Big) + \frac{1}{2}\phi^{\mathrm{As}}\Big(A(\exp(X^{-}))\Big)\Big) \tag{5.1}$$

using the discounted expectation value with the transformed payoff

$$\phi^{\mathrm{As}}\Big(A(\exp(X))\Big) = \max\left(K - \frac{1}{T}\int_0^T \exp(X_t)\,\mathrm{d}t, 0\right) \tag{5.2}$$

which holds at the maturity. The stochastic variable X_t is modeled by

$$\begin{cases} \mathrm{d}X_t = (r - D - \frac{1}{2}\Upsilon_t)\,\mathrm{d}t + \sqrt{\Upsilon_t}\,\mathrm{dW}_t^1, \\ \mathrm{d}\Upsilon_t = \kappa(\mu - \Upsilon_t)\,\mathrm{d}t + \rho\sigma_\upsilon\sqrt{\Upsilon_t}\,\mathrm{dW}_t^1 + \sqrt{1-\rho}\sigma_\upsilon\sqrt{\Upsilon_t}\,\mathrm{dW}_t^2. \end{cases} \tag{5.3}$$

Definition 14 (European Put Option Problem with the log-transformed Heston Model)**.** The fair price $u(\mathbf{x}, \tau)$ of the two-dimensional option pricing PDE for European plain vanilla put options with $\mathbf{x} = (\upsilon, x)$ is given by

$$\begin{aligned} \frac{\partial}{\partial \tau}u(\mathbf{x}, \tau) &= \mathcal{L}\Big[u(\mathbf{x}, \tau)\Big], \\ u(\mathbf{x}, 0) &= \bar{\phi}(x), \end{aligned} \tag{5.4}$$

with the transformed payoff-function

$$\bar{\phi}(x) = \max\Big(K - \exp(x), 0\Big). \tag{5.5}$$

Since the underlying model is determined as the log-transformed Heston model, we obtain the following spatial operator in the case of constant parameters

$$\mathcal{L}_{\bar{\mathrm{H}}}[u] = \frac{\upsilon}{2}\frac{\partial^2 u}{\partial x^2} + \frac{1}{2}\sigma_\upsilon^2\upsilon\frac{\partial^2 u}{\partial \upsilon^2} + \sigma_\upsilon\upsilon\rho\frac{\partial^2 u}{\partial x \partial \upsilon} + \left(r - D - \frac{\upsilon}{2}\right)\frac{\partial u}{\partial x} + \kappa(\mu - \upsilon)\frac{\partial u}{\partial \upsilon} - ru, \tag{5.6}$$

and

$$\mathcal{L}_{\tilde{\mathrm{H}}}[u] = \frac{v}{2}\frac{\partial^2 u}{\partial x^2} + \frac{1}{2}\tilde{\sigma}_v^2 v \frac{\partial^2 u}{\partial v^2} + \tilde{\sigma}_v v \tilde{\rho} \frac{\partial^2 u}{\partial x \partial v} + (r - D - \frac{v}{2})\frac{\partial u}{\partial x} + \tilde{\kappa}(\tilde{\mu} - v)\frac{\partial u}{\partial v} - ru, \quad (5.7)$$

in the nonconstant case. We assume that the Feller condition is satisfied and thus for both cases the spatial boundary conditions

$$x \to -\infty : \quad u\Big((v,x),\tau\Big) \sim K\exp(-r\tau), \quad (5.8)$$

$$x \to \infty : \quad u\Big((v,x),\tau\Big) \sim 0, \quad (5.9)$$

$$v \to \infty : \quad u\Big((v,x),\tau\Big) \sim K\exp(-r\tau) \quad (5.10)$$

hold. From the operators we obtain either

$$\mathbf{A}_{\tilde{\mathrm{H}}} = \frac{1}{2}v\begin{pmatrix} \sigma_v^2 & \sigma_v\rho \\ \sigma_v\rho & 1 \end{pmatrix}, \qquad \mathbf{b}_{\tilde{\mathrm{H}}} = \begin{pmatrix} \kappa(\mu - v) - \frac{1}{2}\sigma_v^2 \\ r - D - \frac{v}{2} - \frac{1}{2}\sigma_v\rho \end{pmatrix}, \quad (5.11)$$

or

$$\mathbf{A}_{\tilde{\mathrm{H}}} = \frac{1}{2}v\begin{pmatrix} \sigma_v^2 & \sigma_v\rho \\ \sigma_v\rho & 1 \end{pmatrix}, \qquad \mathbf{b}_{\tilde{\mathrm{H}}} = \begin{pmatrix} \kappa(\mu - v) - \frac{1}{2}\sigma_v^2 \\ r - D - \frac{v}{2} - \frac{1}{2}\sigma_v\rho \end{pmatrix} \quad (5.12)$$

for the divergent form.

In the following, we will refer to the Asian option pricing problem from Definition 13 as the fine model, and the coarse model is given in Definition 14. For both models, we are interested in the single price for the spot variable $\mathbf{x}^{\text{spot}} = (v^{\text{spot}}, x^{\text{spot}})$. The predefined asset is part of the contract, but v^{spot} must be determined implicitly from the market data or, as in our case, predefined by a guess. We want to calibrate $\boldsymbol{\xi} = (\sigma_v, \rho, \kappa, \mu) \in \mathbb{X} \subset \mathbb{R}^4$ with the advantage that both models contain the parameters we want to calibrate. Although we expect the optimal values of the parameters to be different for the two models, the space mapping technique helps us to calibrate the parameters of the fine model while only optimizing the coarse model. In the following, we distinguish between $\boldsymbol{\xi}_{\text{f}} \in \mathbb{X}_{\text{f}}$, the parameter vector for the fine model, and $\boldsymbol{\xi}_{\text{c}} \in \mathbb{X}_{\text{c}}$, the parameter vector for the coarse model. In the subsets of all possible solutions, we denote $\mathbb{X}_{\text{f}}$ and $\mathbb{X}_{\text{c}}$, respectively. Similarly, we denote the option price of the fine model by u_{f} and the option price of the coarse model by u_{c}.

Since we are using real market data u_{data} as ground truth for the calibration, we define the cost function as follows

$$\mathcal{J}(u(\boldsymbol{\xi}); u_{\text{data}}) = \frac{1}{2} \int_0^T \|u - u_{\text{data}}\|^2 \, d\tau, \tag{5.13}$$

where $\boldsymbol{\xi}$ denotes the parameters for calibration. Now we want to approximate the solution of the fine calibration problem

$$\begin{gathered} \min_{\boldsymbol{\xi}_{\text{f}} \in \mathbb{X}_{\text{f}}} \mathcal{J}(u_{\text{f}}(\boldsymbol{\xi}_{\text{f}}); u_{\text{data}}) \\ \text{subject to} \\ u_{\text{f}} \text{ solves the problem from Definition 13} \end{gathered} \tag{5.14}$$

Note that we do not solve the fine optimization problem, but only evaluate the cost functional for given parameter sets $\boldsymbol{\xi}_{\text{f}}$ during the space mapping algorithm. The optimization problem aimed at calibrating the coarse model for given real market data is given by

$$\begin{gathered} \min_{\boldsymbol{\xi}_{\text{c}} \in \mathbb{X}_{\text{C}}} \mathcal{J}(u_{\text{c}}(\boldsymbol{\xi}_{\text{c}}); u_{\text{data}}) \\ \text{subject to} \\ u_{\text{c}} \text{ solves the problem from Definition 14.} \end{gathered} \tag{5.15}$$

In the following we assume that both problems (5.14) and (5.15) admit a unique solution with a unique minimizer

$$\boldsymbol{\xi}_{\text{f}}^* = \operatorname{argmin}_{\boldsymbol{\xi}_{\text{f}} \in \mathbb{X}_{\text{f}}} \mathcal{J}(u_{\text{f}}(\boldsymbol{\xi}_{\text{f}}); u_{\text{data}}) \quad \text{and} \quad \boldsymbol{\xi}_{\text{c}}^* = \operatorname{argmin}_{\boldsymbol{\xi}_{\text{c}} \in \mathbb{X}_{\text{C}}} \mathcal{J}(u_{\text{c}}(\boldsymbol{\xi}_{\text{c}}); u_{\text{data}}). \tag{5.16}$$

The solution of the coarse calibration problem will be our initial guess for the optimal parameter set of the fine model. However, in the space mapping approach we want to iteratively improve the parameter values of the fine model and we exploit the approximation properties of the coarse model. Let's assume that there exists a so-called *space mapping function* $\boldsymbol{s}\colon \mathbb{X}_{\text{f}} \to \mathbb{X}_{\text{c}}$, $\boldsymbol{\xi}_{\text{f}} \mapsto \mathrm{s}(\boldsymbol{\xi}_{\text{f}})$, which satisfies

$$\boldsymbol{s}(\boldsymbol{\xi}_{\text{f}}) := \operatorname{argmin}_{\boldsymbol{\xi}_{\text{c}} \in \mathbb{X}_{\text{c}}} \boldsymbol{r}\Big(u_{\text{c}}(\boldsymbol{\xi}_{\text{c}}), u_{\text{f}}(\boldsymbol{\xi}_{\text{f}})\Big) = \operatorname{argmin}_{\boldsymbol{\xi}_{\text{c}} \in \mathbb{X}_{\text{c}}} \|u_{\text{c}}(\boldsymbol{\xi}_{\text{c}}) - u_{\text{f}}(\boldsymbol{\xi}_{\text{f}})\|, \tag{5.17}$$

for some *misalignment function* $\boldsymbol{r}$.

Assuming that the coarse model is a good approximation of the fine model, and choosing

$\mathcal{J}$ as the misalignment function, we expect the following condition to be satisfied

$$\begin{aligned} \boldsymbol{s}(\boldsymbol{\xi}_{\mathrm{f}}^*) &= \operatorname{argmin}_{\boldsymbol{\xi}_{\mathrm{c}}\in X_{\mathrm{c}}} \mathcal{J}\Big(u_{\mathrm{c}}(\boldsymbol{\xi}_{\mathrm{c}}); u_{\mathrm{f}}(\boldsymbol{\xi}_{\mathrm{f}}^*)\Big) && (5.18)\\ &\approx \operatorname{argmin}_{\boldsymbol{\xi}_{\mathrm{c}}\in X_{\mathrm{c}}} \mathcal{J}\Big(u_{\mathrm{c}}(\boldsymbol{\xi}_{\mathrm{c}}); u_{\mathrm{c}}(\boldsymbol{\xi}_{\mathrm{c}}^*)\Big) && (5.19)\\ &\approx \operatorname{argmin}_{\boldsymbol{\xi}_{\mathrm{c}}\in X_{\mathrm{c}}} \mathcal{J}\Big(u_{\mathrm{c}}(\boldsymbol{\xi}_{\mathrm{c}}); u_{\mathrm{data}}\Big) = \boldsymbol{\xi}_{\mathrm{c}}^*. && (5.20)\end{aligned}$$

The underlying assumption is, that the optimal states u_{f} and u_{c} are both good approximations of the ground truth u_{data}, each for the respective model.

The strategy of the space mapping algorithm is to solve

$$\boldsymbol{s}(\boldsymbol{\xi}_{\mathrm{f}}^*) - \boldsymbol{\xi}_{\mathrm{c}}^* = 0. \tag{5.21}$$

Note that we will not approximate the entire function $\mathbf{s}$, but only evaluate it along the iterations. Fully approximating $\mathbf{s}$ is a much harder and probably an ill-posed task.

For the numerical results we use the *Aggressive Space Mapping* (ASM) algorithm [75]. In fact, we use a simplified version of the ASM algorithm [75], since due to the linearity of the state problems (4.45) and (4.47) we can approximate the Jacobian of the space mapping function by the identity [1]. Since the parameter domain for κ, μ, σ_v and ρ is restricted, as well as the constraint that the Feller condition must be satisfied, we use the *projected Armijo rule* [100]. In the projected Armijo rule, we choose the maximum $\sigma^\iota \in \{1, 1/2, 1/4, \dots\}$ for which

$$\mathcal{J}\Big(u(\mathcal{P}(\boldsymbol{\xi}^\iota)), u_{\mathrm{data}}\Big) - \mathcal{J}\Big(u(\boldsymbol{\xi}^k), u_{\mathrm{data}}\Big) \le -\frac{\epsilon}{\sigma^\iota}\Big\|\mathcal{P}\big(\boldsymbol{\xi}^\iota\big) - \boldsymbol{\xi}^k\Big\|_2^2, \tag{5.22}$$

with

$$\boldsymbol{\xi}^{k+1} = \mathcal{P}(\boldsymbol{\xi}^\iota), \quad \boldsymbol{\xi}^\iota = \boldsymbol{\xi}^k - \sigma^\iota \boldsymbol{h}^k \quad \text{and} \quad \boldsymbol{h}^k = -\Big(\mathbf{s}(\boldsymbol{\xi}_{\mathrm{f}}^k) - \boldsymbol{\xi}_{\mathrm{c}}^*\Big) \tag{5.23}$$

Here $\epsilon \in (0,1)$ is a numerical constant that depends on the problem and is typically chosen to be $\epsilon = 10^{-4}$. We will use this value for the numerical results later. Finally, we get the adapted ASM, see Algorithm 4. The main ideas of the space mapping approach are summarized, for more details we recommend [2, 25, 99, 75].

5.2 Optimal Control of the Coarse Model

In the following, we formally derive the first-order optimality system of the coarse calibration problem that we solve in step (1b) of Algorithm 4. For notational convenience, we

Algorithm 4: The simplified Aggressive Space Mapping (ASM) Algorithm.

```
Result: optimized $\boldsymbol{\xi}_{\mathrm{f}}$
$\boldsymbol{\xi}_{\mathrm{f}}^0 = \boldsymbol{\xi}_{\mathrm{c}}^* = \mathrm{argmin}_{\boldsymbol{\xi}_{\mathrm{c}} \in \mathbb{X}_{\mathrm{c}}} \mathcal{J}(u_{\mathbf{c}}(\boldsymbol{\xi}_{\mathrm{c}}), u_{\mathrm{data}})$;
for $k = 0, 1, \ldots$ do
    1) evaluate space mapping function $\mathbf{s}(\boldsymbol{\xi}_{\mathrm{f}}^k)$ by ;
        (a) evaluate fine model from Definition 13 to obtain $u_{\mathrm{f}}$ with $\boldsymbol{\xi}_{\mathrm{f}}^k$
        (b) Perform a coarse model optimization
          $\mathbf{s}(\boldsymbol{\xi}_{\mathrm{f}}^k) = \mathrm{argmin}_{\boldsymbol{\xi}_{\mathrm{c}} \in X_{\mathrm{c}}} \mathcal{J}\Big(u_{\mathrm{c}}(\boldsymbol{\xi}_{\mathrm{c}}); u_{\mathrm{f}}(\boldsymbol{\xi}_{\mathrm{f}}^k)\Big)$
    2) Compute $\boldsymbol{h}^k = -\Big(\mathbf{s}(\boldsymbol{\xi}_{\mathrm{f}}^k) - \boldsymbol{\xi}_{\mathrm{c}}^*\Big)$
    3) Update control $\boldsymbol{\xi}_{\mathrm{f}}^{k+1} = \boldsymbol{\xi}_{\mathrm{f}}^k + \boldsymbol{h}^k$ using the projected Armijo rule (5.22) to
      restrict $\boldsymbol{\xi}_{\mathrm{f}}^{k+1}$ to the boundary
    while $\|\mathbf{s}(\boldsymbol{\xi}_{\mathrm{f}}^k) - \boldsymbol{\xi}_{\mathrm{c}}^*\| >$ tolerance;
end
```

use the abbreviation $J(u, \boldsymbol{\xi}) =$ for $\mathcal{J}\Big(u(\boldsymbol{\xi}); u_{\mathrm{data}})\Big)$ in the derivation, since the algorithm calibrates to a given data set, regardless of whether that data is derived from the market or some other model. Specifically, we formally derive a gradient-based algorithm using a Lagrangian approach to solve (5.15).

5.2.1 First-order optimality conditions for the Heston model

Let us denote the Lagrange multipliers by $\psi = (\varphi, \varphi^{\mathrm{a}}, \varphi^{\mathrm{b}}, \varphi^{\mathrm{c}}, \varphi^{\mathrm{d}})$, set $\Omega = (0, \infty) \times (-\infty, \infty)$ and split the boundary $\partial\Omega$ into

$$\Gamma_{\mathrm{a}} = \partial\Omega \cap \{x = -\infty\}, \qquad \Gamma_{\mathrm{b}} = \partial\Omega \cap \{x = \infty\}, \tag{5.24}$$

$$\Gamma_{\mathrm{c}} = \partial\Omega \cap \{v = 0\}, \qquad \Gamma_{\mathrm{d}} = \partial\Omega \cap \{v = \infty\}. \tag{5.25}$$

First, we focus on the log-transformed Heston equation with constant parameters (4.47) and write

$$\frac{\partial u}{\partial \tau} - \nabla \cdot \mathbf{A}\nabla u - \mathbf{b} \cdot \nabla u + ru = 0, \tag{5.26}$$

where $\mathbf{A}$ and $\mathbf{b}$ are either given by $\mathbf{A}_{\bar{\mathrm{H}}}, \mathbf{b}_{\bar{\mathrm{H}}}$ or $\mathbf{A}_{\tilde{\mathrm{H}}}, \mathbf{b}_{\tilde{\mathrm{H}}}$ as in Definition 14. Next, we define the operator $\mathcal{E}$ that will represent the constraint in the Lagrangian. Since at Γ_{c} no boundary condition needs to be given, we introduce $\tilde{\Omega} = \Omega \cap \Gamma_{\mathrm{c}}$ and the operator $\mathcal{E}$ is implicitly defined by

$$\begin{aligned}
\Big\langle \mathcal{E}(u,\boldsymbol{\xi}),\psi\Big\rangle = &\int_0^T\int_{\tilde\Omega}\Big[\frac{\partial u}{\partial\tau}-\nabla\cdot\mathbf{A}\nabla u - b\cdot\nabla u + ru\Big]\varphi\,\mathrm{dz}\,\mathrm{d}\tau \\
&+\int_0^T\int_{\Gamma_\mathrm{a}}\Big[u-\exp(-r\tau)\Big]\varphi^\mathrm{a}\,\mathrm{ds}\,\mathrm{d}\tau+\int_0^T\int_{\Gamma_\mathrm{b}}u\varphi^\mathrm{b}\,\mathrm{ds}\,\mathrm{d}\tau \\
&+\int_0^T\int_{\Gamma_\mathrm{d}}\Big[u-\exp(-r\tau)\Big]\varphi^\mathrm{d}\,\mathrm{ds}\,\mathrm{d}\tau \\
=: &\; I_1+I_2+I_3+I_4.
\end{aligned} \tag{5.27}$$

The Lagrangian for the constrained parameter calibration problem is then given by

$$L(u,\boldsymbol{\xi},\psi)=J(u,\boldsymbol{\xi})-\Big\langle\mathcal{E}(u,\boldsymbol{\xi}),\psi\Big\rangle. \tag{5.28}$$

We formally compute the first-order optimality conditions by setting $dL=0$. For details on the method we refer to [47, 100]. Before computing the Gâteaux derivatives of L in arbitrary directions [47], we note that by Green's first identity it holds

$$\int_\Omega(\mathbf{b}\cdot\nabla u)\,\varphi\,\mathrm{dz}=\int_{\partial\Omega}(\mathbf{b}\cdot\vec{\mathbf{n}})\,u\varphi\,\mathrm{ds}-\int_\Omega u\nabla\cdot(\mathbf{b}\varphi)\,\mathrm{dz}. \tag{5.29}$$

Therefore, we can rewrite

$$I_1=\int_0^T\int_{\tilde\Omega}\varphi u_\tau+\mathbf{A}\nabla u\cdot\nabla\varphi-\frac{1}{2}\mathbf{b}\cdot\nabla u\varphi+\frac{1}{2}u\mathbf{b}\cdot\nabla\varphi+\Big(r+\frac{1}{2}\nabla\cdot\mathbf{b}\Big)u\varphi\,\mathrm{dz} \tag{5.30}$$

$$-\int_{\partial\Omega}(\mathbf{A}\nabla u)\cdot\vec{\mathbf{n}}\varphi\,\mathrm{ds}-\frac{1}{2}\int_{\partial\Omega}(\mathbf{b}\cdot\vec{\mathbf{n}})\,u\varphi\,\mathrm{ds}\,\mathrm{d}\tau \tag{5.31}$$

$$=\Bigg[\int_\Omega\varphi u\,\mathrm{dz}\Bigg]_{\tau=0}^{\tau=T}+\int_0^T\int_{\tilde\Omega}u\Big[-\frac{\partial\varphi}{\partial\tau}-\nabla\cdot\mathbf{A}^\top\nabla\varphi+\mathbf{b}\cdot\nabla\varphi+(r+\nabla\cdot\mathrm{b})\varphi\Big]\,\mathrm{dz} \tag{5.32}$$

$$+\int_{\partial\Omega}\Big[(\mathbf{A}^\top\nabla\varphi)\cdot\vec{\mathbf{n}}-(\mathbf{b}\cdot\vec{\mathbf{n}})\,\varphi\Big]u\,\mathrm{ds}-\int_{\partial\Omega}(\mathbf{A}\nabla u)\cdot\vec{\mathbf{n}}\varphi\,\mathrm{ds}\,\mathrm{d}\tau. \tag{5.33}$$

As e is linear in u, the Gâteaux derivative in some arbitrary direction h reads

$$\mathrm{d}_u\Big\langle\mathcal{E}(u,\boldsymbol{\xi}),\psi\Big\rangle[h]=\Big[\int_\Omega\varphi h\,dz\Big]_{\tau=0}^{\tau=T} \tag{5.34}$$

$$+\int_0^T\int_\Omega h\Big[-\frac{\partial\varphi}{\partial\tau}-\nabla\cdot\mathbf{A}^\top\nabla\varphi+\mathbf{b}\cdot\nabla\varphi+(r+\nabla\cdot\mathbf{b})\,\varphi\Big]\,\mathrm{dz} \tag{5.35}$$

$$+\int_{\partial\Omega}\Big[(\mathbf{A}^\top\nabla\varphi)\cdot\vec{\mathbf{n}}-(\mathbf{b}\cdot\vec{\mathbf{n}})\varphi\Big]h\,\mathrm{ds}-\int_{\partial\Omega}(\mathbf{A}\nabla h)\cdot\vec{\mathbf{n}}\varphi\,\mathrm{ds}\,\mathrm{d}\tau \tag{5.36}$$

$$+\int_0^T\int_{\Gamma_\mathrm{a}}h\varphi^\mathrm{a}\,\mathrm{ds}\,\mathrm{d}\tau+\int_0^T\int_{\Gamma_\mathrm{b}}h\varphi^\mathrm{b}\,\mathrm{ds}\,\mathrm{d}\tau+\int_0^T\int_{\Gamma_\mathrm{d}}h\varphi^\mathrm{d}\,\mathrm{ds}\,\mathrm{d}\tau. \tag{5.37}$$

For the cost functional we have

$$\mathrm{d}_u J(u, \boldsymbol{\xi})[h] = \int_0^T \int_\Omega h(u - u_{\text{data}}) \,\mathrm{d}z \,\mathrm{d}\tau. \tag{5.38}$$

To identify the adjoint equation, we consider

$$\begin{aligned} 0 &= \mathrm{d}_u L(u, \boldsymbol{\xi}, \psi)[h] \\ &= \int_0^T h\Big[\int_\Omega (u - u_{\text{data}}) + \frac{\partial \varphi}{\partial \tau} + \nabla \cdot \mathbf{A}^\top \nabla \varphi - \mathbf{b} \cdot \nabla \varphi - (r + \nabla \cdot \mathbf{b})\varphi\Big] \,\mathrm{d}z \,\mathrm{d}\tau. \end{aligned} \tag{5.39}$$

for arbitrary h. Note that we are not allowed to vary u at $u(\mathbf{x}, 0)$ as the initial condition is fixed. Therefore we have $h(\mathbf{x}, 0) \equiv 0$.

For choosing $h \equiv 0$ on $\partial\Omega$ and $h(\mathbf{x}, T) = 0$, we find with the Variational Lemma

$$\frac{\partial \varphi}{\partial \tau} + \nabla \cdot \mathbf{A}^\top \nabla \varphi - \mathbf{b} \cdot \nabla \varphi - (r + \nabla \cdot \mathbf{b})\varphi = -(u - u_{\text{data}}) \quad \text{on } \Omega. \tag{5.40}$$

Now, choosing $h(\mathbf{x}, T) \neq 0$, we then obtain the terminal condition $\varphi(\mathbf{x}) = 0$.

We consider the four boundary conditions separately. At Γ_{c}, also the parabolic adjoint PDE degenerates to a first-order hyperbolic PDE, and thus we have to consider the Fichera theory [4, 66] for the variance again.

The Fichera condition with respect to the variance at $\upsilon = 0$ of the adjoint is the same as before. Therefore, no analytic boundary condition is supplied for this boundary, since we assume that the Feller condition holds.

On Γ_{a} we have

$$0 = \int_{\Gamma_{\text{a}}} \Big[(\mathbf{A}^\top \nabla \varphi) \cdot \vec{\mathbf{n}} - (\mathbf{b} \cdot \vec{\mathbf{n}})\varphi\Big] h - (\mathbf{A} \nabla h) \cdot \vec{\mathbf{n}} \varphi + h \varphi^{\text{a}} \, ds. \tag{5.41}$$

Choosing $h \equiv \text{const} \neq 0$ yields

$$0 = \int_{\Gamma_{\text{a}}} h\Big[(\mathbf{A}^\top \nabla \varphi) \cdot \vec{\mathbf{n}} - (\mathbf{b} \cdot \vec{\mathbf{n}})\varphi + \varphi^{\text{a}}\Big] \, ds, \tag{5.42}$$

hence $(\mathbf{A}^\top \nabla \varphi) \cdot \vec{\mathbf{n}} - (\mathbf{b} \cdot \vec{\mathbf{n}})\varphi + \varphi^{\text{a}} = 0$. On the other hand, choosing $\nabla h \neq 0$ (5.41) must still hold. This yields $\varphi = 0$ on Γ_{a} and

$$\varphi^{\text{a}} = -(\mathbf{A}^\top \nabla \varphi) \cdot \vec{\mathbf{n}} = -(\mathbf{A}^\top \nabla \varphi) \cdot \begin{pmatrix} 0 \\ -1 \end{pmatrix} = \frac{1}{2} \upsilon \sigma_\upsilon \rho \frac{\partial \varphi}{\partial \upsilon} + \frac{1}{2} \upsilon \frac{\partial \varphi}{\partial x} = \frac{1}{2} \upsilon \frac{\partial \varphi}{\partial x} \tag{5.43}$$

As $u\Big((\upsilon, x), 0\Big) \approx \exp(-r\tau)$ if $x \to -\infty$ is given and independent of x, we obtain $\frac{\partial \varphi}{\partial x} = 0$ there and thus $\varphi^{\mathrm{a}} = \varphi = 0$ at this boundary. Similarly, we find on Γ_b that

$$0 = \int_{\Gamma_{\mathrm{b}}} \Big[(\mathbf{A}^\top \nabla \varphi) \cdot \vec{\mathbf{n}} - (\mathbf{b} \cdot \vec{\mathbf{n}})\varphi\Big] h - (\mathbf{A}\nabla h) \cdot \vec{\mathbf{n}}\varphi + h\varphi_b \, ds. \tag{5.44}$$

With the same arguments, we obtain $\varphi = 0$ and $\varphi^{\mathrm{b}} = 0$ on Γ_{b} as well. Following the same arguments for Γ_{d} we obtain $(\mathbf{A}^\top \nabla \varphi) \cdot \vec{\mathbf{n}} - (\mathbf{b} \cdot \vec{\mathbf{n}})\varphi + \varphi^{\mathrm{d}} = 0$ with $\varphi = 0$ and thus it reduces to

$$\varphi^{\mathrm{d}} = -(\mathbf{A}^\top \nabla \varphi) \cdot \vec{\mathbf{n}} = -(\mathbf{A}^\top \nabla \varphi) \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} = -\frac{1}{2}\upsilon\sigma_\upsilon^2 \frac{\partial \varphi}{\partial \upsilon} + \frac{1}{2}\upsilon\sigma_\upsilon\rho\frac{\partial \varphi}{\partial x} = -\frac{1}{2}\upsilon\sigma_\upsilon^2\frac{\partial \varphi}{\partial \upsilon}. \tag{5.45}$$

As $u\Big((\upsilon, x), \tau\Big) \approx \exp(-r\tau)$ is given if $\upsilon \to \infty$ and independent of υ, we obtain $\frac{\partial \varphi}{\partial \upsilon} = 0$ there and thus $\varphi^{\mathrm{d}} = \varphi = 0$ at this boundary.

Altogether, the adjoint equation reads

$$\frac{\partial \varphi}{\partial \tau} + \nabla \cdot \mathbf{A}^\top \nabla \varphi - \mathbf{b} \cdot \nabla \varphi - (r + \nabla \cdot \mathbf{b})\varphi = -(u - u_{\mathrm{data}}) \quad \text{on } \Omega, \tag{5.46}$$

with terminal condition $\varphi(\mathbf{x}, T) = 0$ and $\varphi = 0$ on the boundaries Γ_{a}, Γ_{b} and Γ_{d} and the outflow boundary at $\upsilon = 0$. Since, we obtain an terminal condition, we reverse the time again to the original time t and obtain the IBVP given in Definition 15

Definition 15 (Adjoint of the log-transformed Heston model)**.**

$$\frac{\partial}{\partial t}\varphi(\mathbf{x}, t) = \mathcal{L}\Big[\varphi\Big](\mathbf{x}, t), \tag{5.47}$$

and initial condition

$$\varphi(\mathbf{x}, 0) = 0, \tag{5.48}$$

and the boundary conditions

$$x \to -\infty: \quad \varphi\Big((\upsilon, x), t\Big) \sim 0, \tag{5.49}$$

$$x \to \infty: \quad \varphi\Big((\upsilon, x), t\Big) \sim 0, \tag{5.50}$$

$$\upsilon \to \infty: \quad \varphi\Big((\upsilon, x), t\Big) \sim 0, \tag{5.51}$$

with an outflow boundary at $\upsilon \to 0$. The spatial operator $\mathcal{L}$ is given by either

$$\mathcal{L}_{\tilde{\mathrm{A}}}[\varphi] = -\Big(u - u_{\mathrm{data}}\Big) - \frac{1}{2}\upsilon\sigma_{\upsilon}^{2}\frac{\partial^2\varphi}{\partial\upsilon^2} - \upsilon\sigma_{\upsilon}\rho\frac{\partial^2\varphi}{\partial x\partial\upsilon} - \frac{1}{2}\upsilon\frac{\partial^2\varphi}{\partial x^2} - \Big(\sigma_{\upsilon}^2 - \kappa(\mu-\upsilon)\Big)\frac{\partial\varphi}{\partial\upsilon} \\ - \Big(q - r + \frac{\upsilon}{2} + \sigma_{\upsilon}\rho\Big)\frac{\partial\varphi}{\partial x} - \Big(\kappa - r\Big)\varphi, \quad (5.52)$$

in the constant parameter case or in the nonconstant case

$$\mathcal{L}_{\tilde{\mathrm{A}}}[\varphi] = -(u - u_{\mathrm{data}}) - \frac{1}{2}\upsilon\tilde{\sigma_{\upsilon}}^{2}\frac{\partial^2\varphi}{\partial\upsilon^2} - \upsilon\tilde{\sigma_{\upsilon}}\tilde{\rho}\frac{\partial^2\varphi}{\partial\upsilon\partial x} - \frac{1}{2}\upsilon\frac{\partial^2\varphi}{\partial x^2} - \Big(\tilde{\sigma}_{\upsilon}^2 - \tilde{\kappa}(\tilde{\mu}-\upsilon)\Big)\frac{\partial\varphi}{\partial\upsilon} \\ - \Big(q - r + \frac{\upsilon}{2} + \tilde{\sigma}_{\upsilon}\rho\Big)\frac{\partial\varphi}{\partial x} - \Big(\tilde{\kappa} - r\Big)\varphi. \quad (5.53)$$

5.2.2 Derivation of the Gradient and the Gradient Descent Algorithm for Parameter calibration

Let $\boldsymbol{\xi} = (\sigma_{\upsilon}, \rho, \kappa, \mu)$ be the parameters to be identified, since r and D are given by the data. We compute the optimality condition by setting $\mathrm{d}_{\boldsymbol{\xi}}L(u, \boldsymbol{\xi}, \psi) = 0$. Since the boundaries Γ_{a}, Γ_{b} and Γ_{d} are zero, we focus on $\tilde{\Omega}$. In the following, the derivatives with respect to the different parameters are given separately. For σ_{υ} we get

$$\mathrm{d}_{\sigma_{\upsilon}}\Big\langle\mathcal{E}(u,\boldsymbol{\xi}),\psi\Big\rangle = \int_0^T\int_{\tilde{\Omega}} u\Big[-\sigma_{\upsilon}\upsilon\frac{\partial^2\varphi}{\partial\upsilon^2} - 2\sigma_{\upsilon}\frac{\partial\varphi}{\partial\upsilon} - \rho\frac{\partial\varphi}{\partial x} - \rho\upsilon\frac{\partial^2\varphi}{\partial x\partial\upsilon}\Big]\,\mathrm{dz}\,\mathrm{d}\tau. \quad (5.54)$$

Similarly, we obtain for the other derivatives

$$\mathrm{d}_{\rho}\Big\langle\mathcal{E}(u,\boldsymbol{\xi}),\psi\Big\rangle = \int_0^T\int_{\tilde{\Omega}} u\Big[-\sigma_{\upsilon}\frac{\partial\varphi}{\partial x} - \sigma_{\upsilon}\upsilon\frac{\partial^2\varphi}{\partial x\partial\upsilon}\Big]\,\mathrm{dz}\,\mathrm{d}\tau, \quad (5.55)$$

$$\mathrm{d}_{\kappa}\Big\langle\mathcal{E}(u,\boldsymbol{\xi}),\psi\Big\rangle = \int_0^T\int_{\tilde{\Omega}} u\Big[(\mu-\upsilon)\frac{\partial\varphi}{\partial\upsilon} - \varphi\Big]\,\mathrm{dz}\,\mathrm{d}\tau, \quad (5.56)$$

$$\mathrm{d}_{\mu}\Big\langle\mathcal{E}(u,\boldsymbol{\xi}),\psi\Big\rangle = \int_0^T\int_{\tilde{\Omega}} \kappa u\frac{\partial\varphi}{\partial\upsilon}\,\mathrm{dz}\,\mathrm{d}\tau. \quad (5.57)$$

Note that $\mathrm{d}_{\boldsymbol{\xi}}L(u,\boldsymbol{\xi},\psi)[h_{\boldsymbol{\xi}}] = 0$ needs to hold for arbitrary directions $h_{\boldsymbol{\xi}}$. Therefore, we can read off the gradient from the above expressions.

We extend this gradient formulation for time-dependent parameter $\tilde{\boldsymbol{\xi}} = (\tilde{\sigma}_{\upsilon}, \tilde{\rho}, \tilde{\kappa}, \tilde{\mu})$, where u and φ are the solutions of the problems defined in Definitions 14 and 15 with $\mathcal{L}_{\tilde{\mathrm{H}}}$ and

$\mathcal{L}_{\tilde{\mathrm{A}}}$. The gradient is then time-dependent as well and given by

$$\mathrm{d}_{\tilde{\sigma}_\upsilon}\left\langle \mathcal{E}(u,\tilde{\boldsymbol{\xi}}),\psi\right\rangle = \int_{\tilde{\Omega}} u\Big[-\tilde{\sigma}_\upsilon \upsilon\frac{\partial^2\varphi}{\partial\upsilon^2} - 2\tilde{\sigma}_\upsilon\frac{\partial\varphi}{\partial\upsilon} - \tilde{\rho}\frac{\partial\varphi}{\partial x} - \tilde{\rho}\upsilon\frac{\partial^2\varphi}{\partial x\partial\upsilon}\Big]\,\mathrm{dz}\,\mathrm{d}\tau, \tag{5.58}$$

$$\mathrm{d}_{\tilde{\rho}}\left\langle \mathcal{E}(u,\tilde{\boldsymbol{\xi}}),\psi\right\rangle = \int_{\tilde{\Omega}} u\Big[-\tilde{\sigma}_\upsilon\frac{\partial\varphi}{\partial x} - \tilde{\sigma}_\upsilon \upsilon\frac{\partial^2\varphi}{\partial x\partial\upsilon}\Big]\,\mathrm{dz}\,\mathrm{d}\tau, \tag{5.59}$$

$$\mathrm{d}_{\tilde{\kappa}}\left\langle \mathcal{E}(u,\tilde{\boldsymbol{\xi}}),\psi\right\rangle = \int_{\tilde{\Omega}} u\Big[(\tilde{\mu}-\upsilon)\frac{\partial\varphi}{\partial\upsilon} - \varphi\Big]\,\mathrm{dz}\,\mathrm{d}\tau, \tag{5.60}$$

$$\mathrm{d}_{\tilde{\mu}}\left\langle \mathcal{E}(u,\tilde{\boldsymbol{\xi}}),\psi\right\rangle = \int_{\tilde{\Omega}} \tilde{\kappa}u\frac{\partial\varphi}{\partial\upsilon}\,\mathrm{dz}\,\mathrm{d}\tau. \tag{5.61}$$

Solving the first-order optimality condition all at once is difficult due to the forward-backward structure. Therefore, in the following, we propose a gradient descent algorithm. For a given initial parameter set $\boldsymbol{\xi}^{\mathrm{init}}$, we can solve the state equation for the Heston model with constant control variable $\boldsymbol{\xi}$ (5.6) or time-dependent parameter $\tilde{\boldsymbol{\xi}}$ (5.7). With the state solution at hand, we can compute the corresponding adjoint equation with $\boldsymbol{\xi}$ using (5.52) or (5.53) using $\tilde{\boldsymbol{\xi}}$. Then we have all the information we need to compute the gradient and update the parameter set with a gradient step. The procedure is outlined in the Algorithm 5.

Algorithm 5: The gradient descent method for Heston parameter calibration.

```
Result: calibrated parameters for Heston model
initialize parameters;
while $\|\text{gradient}\| > \epsilon$ do
    solve the problem given in Definition 14 either with $\mathcal{L}_{\mathrm{H}}$ (5.6) or $\mathcal{L}_{\tilde{\mathrm{H}}}$ (5.7);
    solve the adjoint equation in Definition 15 either with $\mathcal{L}_{\mathrm{A}}$ (5.52) or $\mathcal{L}_{\tilde{\mathrm{A}}}$ (5.53);
    compute the gradient;
    line search for step size with the projected Armijo rule (5.22) ;
    update the parameter set;
end
```

Chapter 6

Numerical Results

Within the numerical results, we use the mean square error (MSE), a relative improvement percentage and an average function. On the spatial grid with $\mathbf{M}$, we get in total M_G grid points, computed by

$$M_G = \sum_{i=1}^{d} (\mathrm{M}_i + 1). \tag{6.1}$$

Now let U denote the computed discrete solution of the grid $\mathbf{M}$, and U_i the $i-$th gridnode, with $g = 1, \ldots, M_G$, we define the discrete ℓ_2 as

$$\ell_2 : \quad \left\| \mathrm{U} \right\|_2 = \left(\sum_{g=1}^{M_G} \mathrm{U}_g^2 \right)^{\frac{1}{2}}, \tag{6.2}$$

For the MSE, let U_{ref} denote a discrete reference solution and U the corresponding numerical approximation,

$$\text{MSE} = \frac{1}{M_G} \left\| U_{\text{ref}} - \mathrm{U} \right\|_2. \tag{6.3}$$

For the relative improvement we compare the percentage reduction of the cost functional w.r.t. a given parameter set $\boldsymbol{\xi}^{\text{init}}$, which in the calibration produces the optimized parameter set $\boldsymbol{\xi}^{\text{opt}}$, with the cost functional $\mathcal{J}(\mathrm{U}(\boldsymbol{\xi}_{\text{opt}}), u_{\text{data}})$. This improvement is denoted by

$$\mathtt{r}(\boldsymbol{\xi}^{\text{init}}) = 100 \cdot \left(1 - \frac{\mathcal{J}(\mathrm{U}(\boldsymbol{\xi}^{\text{opt}}), u_{\text{data}})}{\mathcal{J}(\mathrm{U}(\boldsymbol{\xi}^{\text{init}}), u_{\text{data}})} \right). \tag{6.4}$$

Last, we use the mean for the average improvement of $\boldsymbol{\xi}^{\text{init}}$ over N_{cases} test cases

$$\text{Mean} = \frac{\sum_{n=1}^{N_{\text{cases}}} \mathtt{r}_n(\boldsymbol{\xi}^{\text{init}})}{n_{\text{cases}}} \tag{6.5}$$

The numerical results begin with an analysis of the penalty term within the American put option pricing with the Black-Scholes model. We then discuss the Heston model and the different hierarchical approaches. Within the numerical results of the Heston model, we begin with a discussion of the different closure conditions for the model. We then focus on the sparse grid combination technique and the combination with the Parareal, and its extensions are also applied to the pricing of American put options. Since the Parareal is already introduced with the sparse grid combination technique as a spatial discretization, we start with the numerical results of the combination technique with a subsequent temporal discretization. Finally, we analyze the introduction of the space mapping approach as a calibration method. Therefore, we must first discuss the gradient descent algorithm, followed by the numerical results for the space mapping itself.

6.1 The Penalty Term for the Black-Scholes Model

For the numerical setup of the penalty term from Section 4.2, we use a uniform grid spacing w.r.t. to space and time. For the numerical temporal solver, we use the θ-method. We consider the example of pricing American put options from Nielson et al. [83]. All results are computed on an Intel® Core™ i7-5557U CPU running at 3.10 GHz. We choose $z^{\min} = -4$, $z^{\max} = 4$, $M = 5000$, and use the parameter sets from Table 6.1. Due to the initial guess, the only unknown parameter is $\hat{a}$. Since a deterministic expression for $\hat{a}$ is a goal of our future research, the penalty parameter $\hat{a}$ is obtained by optimization. The optimization is done by minimizing the MSE of the solution corresponding to $\hat{a}$. The MSE between the solution of the PSOR algorithm and the solution of our problem. Since the PSOR method does not use a penalty term, we also compute the MSE of the solution of the penalized PDE

$$\frac{\partial w}{\partial \hat{\tau}}(z,\hat{\tau}) - \frac{\partial^2 w}{\partial z^2}(z,\hat{\tau}) - \frac{\max\left(\hat{\phi}(z) - w(z,\hat{\tau}), 0\right)}{p_c} = 0, \tag{6.6}$$

using a well known penalty term [35] with the penalty constant p_c. Since the example parameter set 2 is widely used in research, we compare the free boundary value of parameter set 2 with the free boundary solution of Nielson [83], Fazio [29] and Company et al. [17]. The value for the free boundary solution obtained by Nielson is $S_f^N = 0.8622$, Fazio obtained $S_f^F = 0.86274$, and the free boundary value of Company et al. is 0.8628, while our approach with a finer optimization gives 0.86269 and $\hat{a} = 10.11 \times 10^{-4}$. This comparison illustrates the high accuracy of this method with state-of-the-art research.

Example	T	K	r	σ_s	N_τ	$\hat{a}(\times 10^{-4})$	s_f	MSE
1	3	100	0.08	0.2	1000	7.5	81.87	9.6×10^{-3}
					2500	7.4	82.00	6.2×10^{-3}
2	1	1	0.1	0.2	1000	10.2	0.862	5.2×10^{-5}
					2500	10.0	0.863	3.3×10^{-5}
3	0.05	10	0.1	0.25	1000	8.0	9.158	3.5×10^{-5}
					2500	8.5	9.142	3.5×10^{-5}
4	0.1	100	0.1	0.3	1000	5.5	86.59	1.2×10^{-3}
					2500	5.4	86.87	7.6×10^{-4}
5	1	100	0.1	0.4	1000	2.6	66.49	1.4×10^{-2}
					2500	2.63	66.60	9.2×10^{-3}
6	0.05	50	0.1	0.4	1000	3.0	42.61	3.8×10^{-4}
					2500	3.1	42.61	2.5×10^{-4}

Table 6.1: Numerical results for the time dependent penalty term.

Our numerical results illustrate the accuracy of the method. The best results are obtained by the sample sets with low volatility and short maturity. The observation of the short maturity is based on the fact that the number of points is different. The dependence on the volatility is caused by the simplification of the term p, since we cancel $\sigma_s^2/2$ and include $2/\sigma_2^2$ in $\hat{a}$. We observed that the differences are in the range between the estimated free boundary value and the final free boundary value. They are caused by the time-dependent movement of the free boundary position. There are several ways to analyze this approach in detail. As approximation formulas for the initial guess one can choose the formulas in [108, 93, 27].

6.2 Sparse Grids for American Put Options in the Heston Model

For the American put option pricing problem with the Heston model as the underlying model, we use the LCP representation. The time discretization is done with $N_\tau = 100$ and the different ADI-IT schemes: the DO-IT scheme (2.23), the CS-IT scheme (2.24), the mCS-IT scheme (2.25) and finally the HV-IT scheme (2.26). Furthermore, let $\theta > 0$ be a given real parameter, depending on the stability constraints of the method, we choose $\theta = 0.5$ for DO-IT and CS-IT, $\theta = \frac{1}{3}$ for mCS-IT and for HV-IT the choice is $\theta = \frac{1}{2} + \frac{1}{6}\sqrt{3}$, cf. [50, 51]. To reduce memory and runtime, we use a more intelligent implementation with the same stability and accuracy as the naive implementation [95].

To approximate the spatial derivatives, we use second-order finite differences. We introduce $\mathbf{x} = (\mathrm{x}_1, \mathrm{x}_2) \in [0,1]^2$ and consider a uniform grid for $\mathbf{x}$, due to the choice of $\boldsymbol{\gamma}$ we get a highly non-uniform mesh of s and υ with grid points concentrated around s^{spot} and υ^{spot}, see Figure 4.1. For the sparse grid for $\mathbf{x}$ we set $|l|_1 = 9, \mathbf{l}^{\min} = 3$ and for the grid transformation to $\mathbf{y} = (s, \upsilon)$ we use

$$\mathbf{y}^{\min} = (0,0), \quad \mathbf{y}^{\max} = (3K, 3), \quad \mathbf{y}^0 = (s^{\text{spot}}, \upsilon^{\text{spot}}) \quad \boldsymbol{\gamma} = (2,2). \tag{6.7}$$

Due to our choice of $\boldsymbol{\gamma} = (2,2)$ we get a very nonuniform grid in s and υ direction.

Since numerical experiments for the Heston model have shown that for efficiency reasons it is sufficient to use only half as many spatial grid points in the volatility direction as in the asset direction: $\mathrm{M}_1 = \mathrm{M}_2/2$, cf. [39]. Therefore, for the sparse grids we introduce an *limitation* for $\mathbf{l}$, which leads to a reduced grid resolution in the volatility direction. Due to the special setting for the grid points, where the number of grid points is given by $(\mathrm{M}_1, \mathrm{M}_2) = (2^{\mathrm{l}_1}, 2^{\mathrm{l}_2})$, we can easily adapt the restriction on the number of grid points for the volatility by setting $\mathrm{l}_1 > \mathrm{l}_2$. Since we use sparse grids, we solve the Heston PDE on several different grids using the same spatial approximations for the derivatives, namely central difference quotients of order two in each direction. In addition, we consider the forward and backward second-order difference quotients at the boundaries for the variance. Note that the mixed derivative at the boundaries for $\upsilon = 0$ is zero, as is the diffusion term, so it is treated trivially. First, we focus on the analysis of the limitation on the number of grid points in the sparse grids. Therefore, we consider the parameter set

$$T = 0.25,\ K = 10,\ \kappa = 5,\ \mu = 0.16,\ \sigma_\upsilon = 0.9,\ \rho = 0.1,\ r = 0.1,\ D = 0. \tag{6.8}$$

This parameter set satisfies the Feller condition and is widely used in the literature, cf. [11, 39, 54, 55, 84]. Since Haentjens and in't Hout [39] also solved this example set with ADI-IT methods, but on a full grid structure, we compare our results with their solution. The accuracy of the model is shown in Table 6.2 and Table 6.3. Our solution of this test set was computed without smoothing the initial data because we use the spot price for the grid transformation and therefore the strike price doesn't match a grid point. As expected, the results obtained with the limited sparse grid setting are in the same accuracy range as the common set, the main advantage of the limitation being the reduction in runtime. To compare the runtime between the full sparse grid and the reduced sparse grid, we compute a reference solution using the Crank-Nicolson scheme and initial data smoothing. We further discretize using a second-order stencil on a grid of $(129, 33)$ grid points and $10,000$ time steps. Figure 6.1 shows the effect of reducing the grid resolution in the volatility direction on the runtime for sparse grid settings for the test set (6.8) in comparison to the accuracy. The reduced spare grid approach has a smaller runtime up until an accuracy of $\mathcal{O}(10^{-5})$, then the full grid approach needs a smaller runtime, as the reduced number of grid points and thus the reduced information effects the accuracy. We used Julia as the computational language and ran the computations on an Intel(R) Core(TM) i7-8700K CPU @ 3.70 GHz.

$v^{\text{spot}} = 0.0625$					
s^{spot}	8	9	10	11	12
[39]	2.0000	1.1081	0.5204	0.2143	0.0827
without limitation					
DO-IT	2.0011	1.1095	0.5203	0.2131	0.0821
CS-IT	2.0011	1.1095	0.5202	0.2131	0.0820
mCS-IT	2.0011	1.1093	0.5199	0.2129	0.0821
HV-IT	2.0012	1.1101	0.5215	0.2136	0.0818
with limitation					
DO-IT	2.0006	1.1085	0.5176	0.2132	0.0821
CS-IT	2.0006	1.1085	0.5176	0.2131	0.0820
mCS-IT	2.0005	1.1083	0.5172	0.2130	0.0821
HV-IT	2.0005	1.1091	0.5188	0.2136	0.0816

Table 6.2: Fair prices for the different spot asset and spot volatility $v^{\text{spot}} = 0.0625$ for the parameter set (6.8) with and without limitation.

$\upsilon^{\text{spot}} = 0.25$					
s^{spot}	8	9	10	11	12
[39]	2.0788	1.3339	0.7962	0.4486	0.2433
without limitation					
DO-IT	2.0787	1.3339	0.7962	0.4481	0.2430
CS-IT	2.0787	1.3338	0.7961	0.4481	0.2430
mCS-IT	2.0785	1.3334	0.7956	0.4476	0.2427
HV-IT	2.0792	1.3346	0.7928	0.4371	0.2415
with limitation					
DO-IT	2.0786	1.3336	0.7961	0.4483	0.2431
CS-IT	2.0786	1.3336	0.7961	0.4483	0.2430
mCS-IT	2.0783	1.3331	0.7955	0.4479	0.2428
HV-IT	2.0791	1.3343	0.7935	0.4368	0.2270

Table 6.3: Fair prices for the different spot asset and spot volatility $\upsilon^{\text{spot}} = 0.25$ for the parameter set (6.8) with and without limitation.

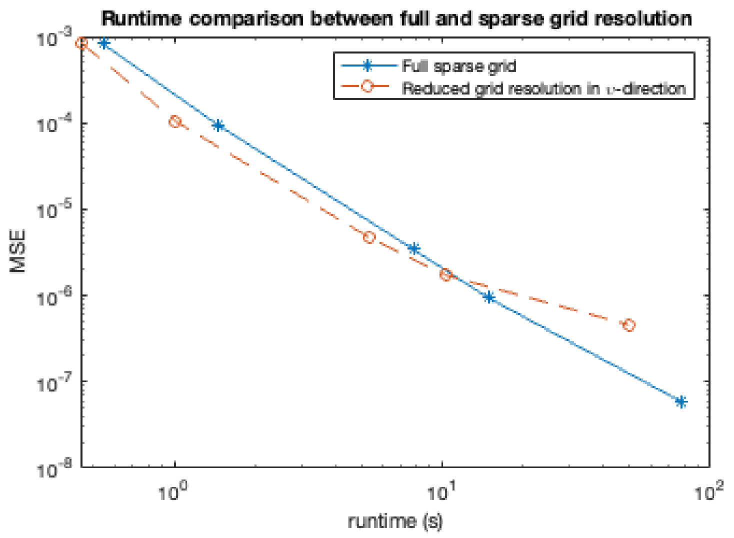


Figure 6.1: Runtime comparison between the sparse grid solution with and without limitation for the parameter set (6.8). The runtime of mCS is shown, where the line with the star represents the runtime of the sparse grid solution without any limitation and the dashed line with the circles corresponds to the runtime for the sparse grid with reduced grid resolution.

6.3 Parareal and the Combination with the Sparse Grids for the Heston Model

We analyze the effect of combining the Parareal with the sparse grids approach, first the effect of reducing the grid resolution in the volatility direction and then the approaches with the reuse of the sparse grid intermediate results. For both sets we use the following financial parameter set

$$T = 0.25,\ K = 10,\ \rho = 0.1,\ r = 0.1,\ \kappa = 5,\ \mu = 0.16,\ \sigma_v = 0.9.$$

as before in 6.2 The set of parameters is often used and is therefore chosen to provide a comparison for the results [11, 38, 39, 84]. We start with the analysis of the effect of reducing the grid resolution in the volatility direction on the accuracy as well as the application of the Parareal to the runtime. We use (6.7) for the grid transformation and set

$$|\mathbf{l}|_1 = 12,\ \mathbf{l}^{\min} = 3,$$

for the sparse grid as well as

$$N_\tau = 16,\ N_{\mathcal{F}} = 100,\ N_{\mathcal{G}} = 25,\ N_P = 3,$$

for the Parareal. Table 6.4 contains the computed Put option prices for different grid resolutions, for each resolution the results are very close to the reference values obtained in [38]. Furthermore, we see that even for very small volatility values and a high reduction in resolution, the results are comparable to the sparse grid solution, including solutions with $l_1 = l_2$, which requires almost twice the number of grid points as the restricted sparse grids and thus twice the computational time.

		$v^{\text{spot}} = 0.0625$					
s^{spot}		8	9	10	11	12	
[38]		2.0000	1.1081	0.5204	0.2143	0.0827	Grids
$\mathbf{l}_{\text{diff}}$	0	2.0000	1.1078	0.5202	0.2138	0.0821	13
	1	2.0000	1.1078	0.5202	0.2138	0.0821	11
	2	2.0000	1.1075	0.5202	0.2138	0.0821	9
	3	2.0000	1.1076	0.5201	0.2137	0.0821	7

Table 6.4: Fair prices for the American put option for the different spot assets for the parameter sets compared to reference values computed by the Parareal using sparse grids.

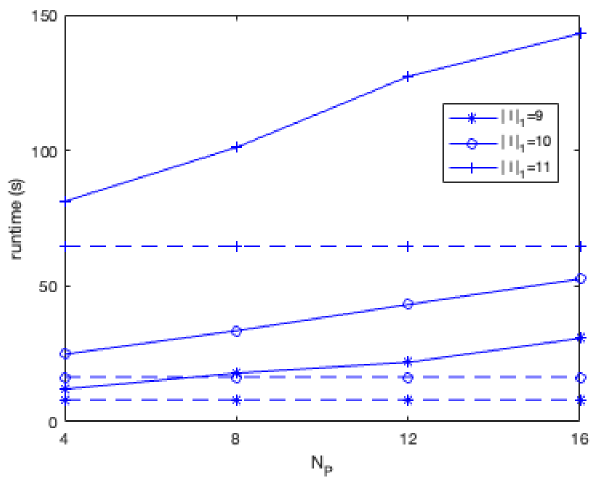


Figure 6.2: Runtime for the sparse grids with and without the Parareal. The dashed line corresponds to the constant serial runtime using sparse grids and the solid line represents the runtime for the Parareal with sparse grids with 4, 8, 12 and 16 parallel processors and $\mathbf{l}_{\text{diff}} = 0$.

Figure 6.2 shows the runtime results for different parallel processors using the same parameter set as before, but with different $|\mathbf{l}|_1$ values. We observe that the sparse grid technique is more efficient than the combination with the Parareal, due to the increased communication time. To underline this fact, we observe that the runtime increases almost linearly with the number of processors. Note that we choose $\mathbf{l}_{\text{diff}} = 0$ for a fair comparison, since the increase of $\mathbf{l}_{\text{diff}} > 1$ is only suitable for the Parareal. Using such an increased sparse grid as the underlying grid structure for the Parareal, we would get a smaller runtime. This is just one of many improvement strategies that can be applied to get a benefit even for smaller problems.

For the second analysis of the effect of reusing the intermediate results of the combination technique, we set the sparse grid level to $m = |\mathbf{l}|_1 = 13$ with $\mathbf{l}^{\min} = 3$ to get a large computational effort to show the effects of our ideas. For the grid transformation to $\mathbf{y}$, we use (6.7) and fix $s^{\text{spot}} = 10$ and $\upsilon^{\text{spot}} = 0.0625$. The time parameters for the Parareal are $N_{\text{Serial}} = 1200$ and $N_{\mathcal{G}} = 25$. From the equation (3.47) we get for 1, 2 and 3 iterations the following optimal number of processors

$$k = 1 : N_P^* \approx 2.44, \quad k = 2 : N_P^* \approx 2.82, \quad k = 3 : N_P^* \approx 2.99,$$

using $\beta^{\text{com}} = 8\ GT/s$, $c(13) = 0.22\ s$, $c(12) = 0.07\ s$, and $c(11) = 0.02\ s$. Besides the optimal number of processors, we test other numbers of processors to qualify the theoretical results.

Table 6.5 shows the accuracy results for the Parareal denoted by "Original", and the improved algorithms using the intermediate results of either the fine or coarse solver, denoted by "Fine solver" and "Coarse solver" respectively. The accuracy is determined by the MSE between the reference solution and our approximation. The reference solution is computed by the sparse grid approach using the modified Craig-Sneyd scheme, i.e. the results are not affected by either the underlying grid structure or the temporal solver.

The accuracy results show that using the intermediate results of the fine solver increases the accuracy compared to the original algorithm, regardless of the number of processors and the number of iterations. On the other hand, the accuracy resulting from the use of the coarse solver is in the same accuracy range as the Parareal and depends strongly on the number of processors and the number of iterations. This behavior results from the incorporation of the less accurate results of the coarse solver into the result of the fine solver at each iteration. Since the parallelism of the sparse grid computation in the coarse solver does not affect the accuracy, the accuracy with and without this parallelism is the same and is therefore not shown.

Table 6.6 and Table 6.7 show the runtime obtained by a benchmark time function. The runtime results show that using the coarse intermediate results significantly reduces the runtime for a small number of iterations. Using the fine results reduces the runtime only for a small number of iterations compared to the number of processors. The additional use of parallelism in the computation of the sparse grid results in the coarse solver is only feasible in combination with a large number of processors and a relatively high number of iterations. All results are computed on an Intel(R) Core(TM) i7-8700K CPU @ 3.70 GHz using the Julia programming language.

		MSE		
Processors	Iterations	Original	Fine solver	Coarse solver
2	1	6.6789	1.2412	7.7833
3	1	5.6220	0.9744	3.8067
3	2	3.3258	0.6123	10.322
4	1	4.6047	0.7494	2.2978
4	2	3.3196	0.5671	6.1924
4	3	2.0428	0.4046	8.9811

Table 6.5: Accuracy results of the Parareal and the adapted algorithms with the reuse of the intermediate results with the serial computation of the solution on a sparse grid, scaled by 10^{-7}.

		Parareal		
Processors	Iterations	Original	Fine solver	Coarse solver
2	1	258.249	269.187	326.535
3	1	192.820	220.193	164.886
3	2	355.484	543.789	277.481
4	1	202.859	196.057	188.723
4	2	405.618	385.204	321.038
4	3	415.204	415.362	416.229

Table 6.6: Runtime in seconds of the Parareal and the adapted algorithms with the reuse of the intermediate results of the sparse grid combination technique.

		Parareal and Sparse Parallelism		
Processors	Iterations	Original	Fine solver	Coarse solver
2	1	417.688	307.761	281.320
3	1	273.177	285.214	163.901
3	2	425.464	465.500	379.796
4	1	213.585	223.042	176.093
4	2	351.261	332.840	271.047
4	3	485.966	404.208	370.041

Table 6.7: Runtime in seconds of the Parareal and its adapted versions with the additional parallelism of the serial computation of the coarse solver.

6.4 Boundary Conditions for the Heston Model

Table 6.8 summarizes the different boundary cases for the log-transformed Heston model presented in Section 4.4. The initial condition has to be adjusted by interpolation w.r.t. the boundary conditions, and we use finite differences as well as the HV scheme with $\theta = 0.75$ for the discretization of the European put option problem.

We compare the numerical results with Heston's closed-form solution [46] by calculating the MSE over the entire domain including the boundary itself, instead of only using the region of interest, as it is done in [40]. For the simulation, we consider two different parameter sets denoted by P1 and P2, see Table 6.9 and five different grids resulting from Table 6.10. Note that P1 is taken from [66] with strike K set to 1.

Boundary cases	$\upsilon^{\min}$	$\upsilon^{\max}$	$x^{\min}$	$x^{\max}$
B1	(a)	(c)	(c)	$\exp(-r\tau_n)$
B2	(a)	(d)	(d)	$\exp(-r\tau_n)$
B3	(a)	(e)	(e)	$\exp(-r\tau_n)$
B4	(a)	(f)	(f)	$\exp(-r\tau_n)$
B5	(b)	(c)	(c)	$\exp(-r\tau_n)$
B6	(b)	(d)	(d)	$\exp(-r\tau_n)$
B7	(b)	(e)	(e)	$\exp(-r\tau_n)$
B8	(b)	(f)	(f)	$\exp(-r\tau_n)$

Table 6.8: Different test cases for the boundary conditions for the log-transformed Heston model from the different equations.

Parameter case	$x^{\min}$	$x^{\max}$	$\upsilon^{\min}$	$\upsilon^{\max}$	θ	T	K	r	σ_υ	μ	κ	ρ
P1 [66]	-7	3	0.01	1	0.75	0.05	1	0.1	0.5	0.07	5	-0.5
P2 [76]	-7	3	0.01	1	0.75	1	1	0.05	0.3	0.2	2	-0.5

Table 6.9: Two parameter sets with their reference for the analysis of the effect of different boundary conditions.

Discretization set	D1	D2	D3	D4	D5
M_1	20	40	80	160	320
M_2	10	20	40	80	160
N_τ	1	4	16	64	256

Table 6.10: Different discretization grids for the computation of the solution w.r.t. the eight different boundary cases.

The numerical results show that all considered boundary cases converge, see Figure 6.3. The plot shows that the influence of the different parameter sets is small. The boundary cases B4 and B8 give the best results with respect to both test cases, i.e., using the extrapolation at $\upsilon^{\max}$ approximates the solution better than the Dirichlet BCs.

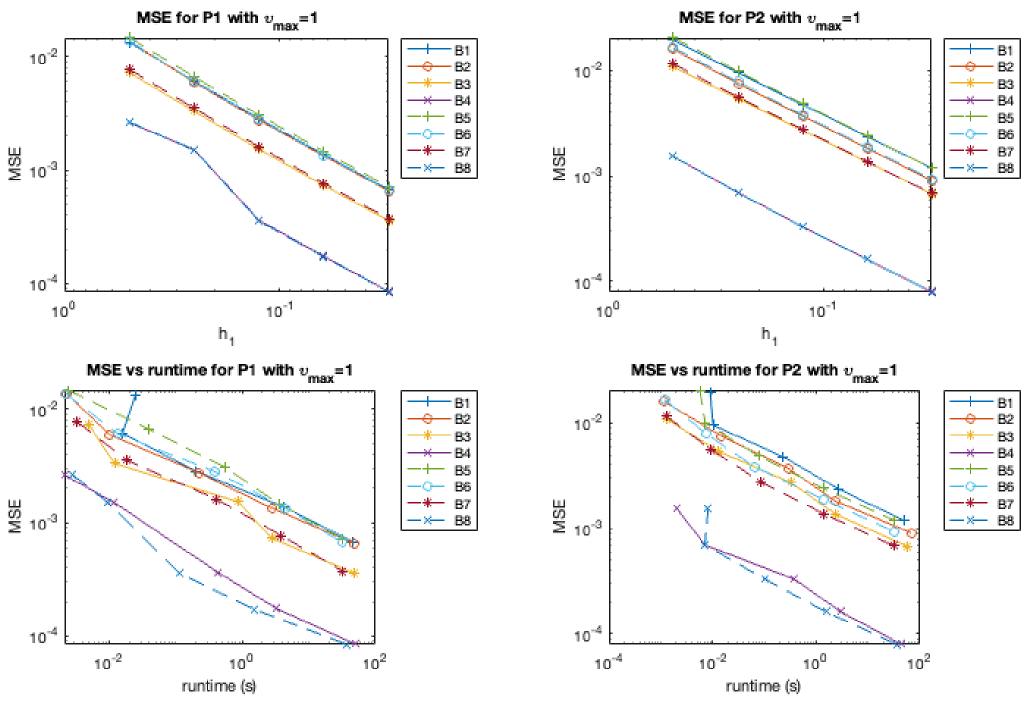


Figure 6.3: Visualization for the MSE presented in Table 6.11 and Table 6.12 as well as MSE vs runtime (s).

Using the extrapolation also for $\upsilon^{\min}$ further improves the MSE, see Table 6.11 and Table 6.12. The increased computational cost can be neglected, as Figure 6.3 shows.

If one is restricted to using Dirichlet BCs, the best choice for $\upsilon^{\max}$ is the BC proposed by Heston in combination with an exponential fit of $x^{\max}$.

P1	D1	D2	D3	D4	D5
B1	13.305	6.092	2.839	1.367	0.671
B2	13.666	5.924	2.751	1.331	0.655
B3	7.224	3.300	1.523	0.731	0.358
B4	**2.618**	**1.514**	**0.362**	**0.174**	**0.085**
B5	14.630	6.635	3.051	1.456	0.711
B6	13.746	6.029	2.803	1.349	0.662
B7	7.720	3.502	1.604	0.763	0.372
B8	**2.650**	**1.500**	**0.355**	**0.172**	**0.085**

Table 6.11: MSE scaled by 10^3 between the semi-analytical solution from Heston and the approximation using the different boundary conditions (Table 6.8) for the different grids corresponding to Table 6.10 and the two parameter cases from Table 6.9.

P2	D1	D2	D3	D4	D5
B1	19.364	9.582	4.768	2.380	1.189
B2	15.972	7.586	3.698	1.827	0.909
B3	10.885	5.439	2.720	1.361	0.681
B4	**1.532**	**0.698**	**0.331**	**0.162**	**0.080**
B5	20.544	9.928	4.862	2.404	1.195
B6	16.695	7.882	3.817	1.878	0.932
B7	11.577	5.637	2.773	1.374	0.684
B8	**1.531**	**0.697**	**0.331**	**0.162**	**0.080**

Table 6.12: MSE scaled by 10^3 between the semi-analytical solution from Heston and the approximation using the different boundary conditions (Table 6.8) for the different grids corresponding to Table 6.10 and the two parameter cases from Table 6.9.

6.5 Gradient Descent Algorithm for the Heston Model

Following [15], we use the values

$$K = 1.0,\ r = 0.1,\ \boldsymbol{\xi}^{\text{ref}} = (5.0, 0.07, 0.5, -0.5), \tag{6.9}$$

to generate an artificial u_{data} for each time step τ_n. For the discretization, we use the parameters $\mathrm{M}_1 = 79$, $\mathrm{M}_2 = 39$, $N_\tau = 59$. As bounds for the projected Armijo rule for $\boldsymbol{\xi} = (\sigma_v, \rho, \kappa, \mu)$, we set $0 < \kappa < 8$, $0 < \mu < 1$, $0 < \sigma_v < 1$, $-1 < \rho < 1$. Note that the projected Armijo rule ensures that the Feller condition holds within each optimization step. So we are in the case of an outflow boundary. We set the maximum iteration value for the calibration to 20. For the initial guesses $\boldsymbol{\xi}^{\text{init}} = (\sigma_v^{\text{init}}, \rho^{\text{init}}, \kappa^{\text{init}}, \mu^{\text{init}},)$ we used generated random numbers within a maximum percentage difference from $\boldsymbol{\xi}^{\text{ref}}$. The calibrated parameters are denoted by cal. We use four different percentages 10, 25, 50 and 75 and generate five sets for each, denoted by T1, T2, T3, T4 and T5. The initial parameters as well as the calibrated parameters in the constant case are given in Table 6.13 for κ, Table 6.14 for μ, Table 6.15 for σ_v, and Table 6.16 for ρ.

ρ^{init}	T1	T2	T3	T4	T5	ρ^{cal}	T1	T2	T3	T4	T5
10	-0.550	-0.485	-0.520	-0.520	-0.495	10	-0.483	-0.547	-0.404	-0.460	-0.384
25	-0.410	-0.490	-0.605	-0.585	-0.380	25	-0.370	-0.390	-0.690	-0.829	-0.159
50	-0.555	-0.465	-0.350	-0.685	-0.725	50	-0.258	-0.242	-0.207	-0.441	-0.847
75	-0.695	-0.205	-0.775	-0.655	-0.150	75	-0.743	-0.972	-0.990	-0.990	-0.437

Table 6.16: Five initial values sets for ρ^{init} for the different percentages and the corresponding calibrated values ρ^{cal} for C0.

κ^{init}	T1	T2	T3	T4	T5	κ^{cal}	T1	T2	T3	T4	T5
10	4.60	5.15	4.50	4.70	4.50	10	4.60	5.15	4.51	4.70	4.50
25	4.75	4.60	5.25	5.90	4.45	25	4.75	4.60	5.24	5.88	4.45
50	4.10	3.25	3.55	2.90	7.00	50	4.11	3.26	3.55	2.94	6.95
75	5.70	7.30	7.00	7.50	6.60	75	5.69	7.29	6.85	7.40	6.60

Table 6.13: Five initial values sets for κ^{init} for the different percentages and the corresponding calibrated values κ^{cal} for C0.

μ^{init}	T1	T2	T3	T4	T5	μ^{cal}	T1	T2	T3	T4	T5
10	0.0679	0.0637	0.0735	0.0714	0.0714	10	0.0598	0.0727	0.0575	0.0632	0.0577
25	0.0721	0.0777	0.0616	0.0630	0.0868	25	0.0655	0.0610	0.0720	0.0836	0.0615
50	0.1001	0.0406	0.0840	0.0448	0.0434	50	0.0476	0.0148	0.0484	0.0136	0.0882
75	0.0742	0.0532	0.0448	0.0399	0.0490	75	0.0847	0.1069	0.1005	0.1064	0.0928

Table 6.14: Five initial values sets for μ^{init} for the different percentages and the corresponding calibrated values μ^{cal} for C0.

σ_v^{init}	T1	T2	T3	T4	T5	σ_v^{cal}	T1	T2	T3	T4	T5
10	0.465	0.545	0.510	0.515	0.480	10	0.451	0.559	0.484	0.502	0.455
25	0.575	0.570	0.505	0.415	0.480	25	0.565	0.546	0.523	0.455	0.429
50	0.580	0.360	0.630	0.345	0.655	50	0.501	0.292	0.581	0.261	0.713
75	0.705	0.470	0.235	0.350	0.565	75	0.718	0.562	0.390	0.489	0.630

Table 6.15: Five initial values sets for σ_v^{init} for the different percentages and the corresponding calibrated values σ_v^{cal} for C0.

We observe that the calibration leaves κ almost untouched, while the other parameter values are significantly changed. This can be explained by the structure of the drift term $\kappa(\mu - v)$, since κ and μ are multiplied. As an optimization measure, we compute the relative reduction (6.4) of the cost functional using $\boldsymbol{\xi}^{\text{init}}$ and $\boldsymbol{\xi}^{\text{cal}}$. Table 6.17 shows the relative reduction of the cost functional for the different test cases for $\boldsymbol{\xi}^{\text{init}}$.

C0	T1	T2	T3	T4	T5
10	*43.12*	77.38	65.22	*63.22*	*52.75*
25	58.46	78.38	62.14	66.06	64.53
50	83.09	20.84	52.76	30.98	*72.62*
75	16.23	72.89	82.89	82.09	80.43

Table 6.17: Relative reduction of the cost functional computed with 6.4 using the different test cases for ξ^{init} and ξ^{cal} in the constant calibration setting (C0).

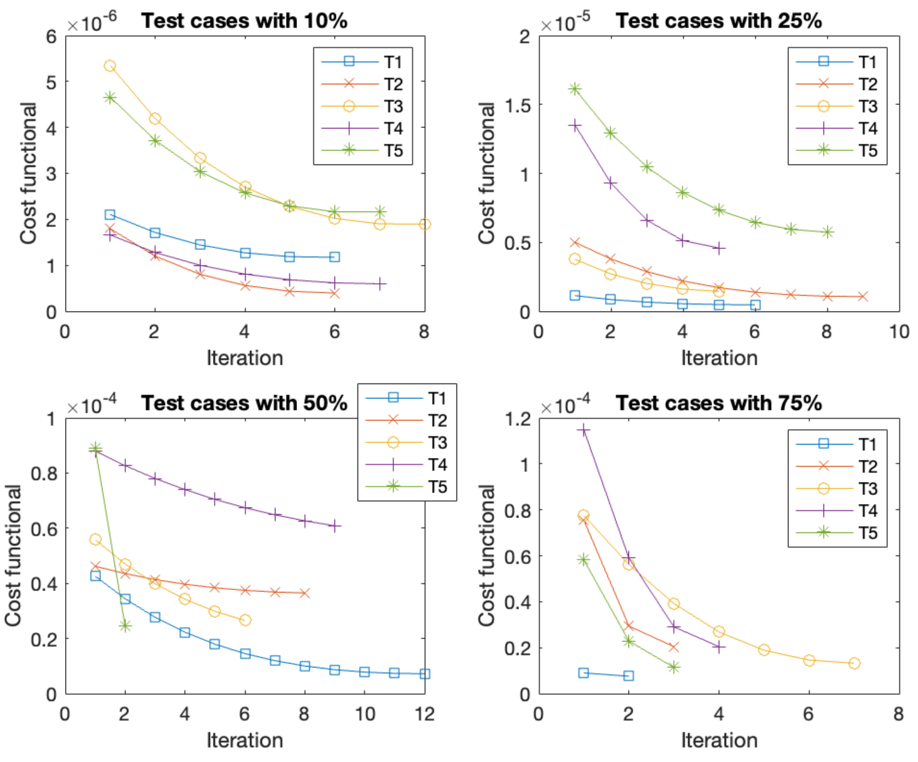


Figure 6.4: Cost functional evolution per iteration for the test cases within the constant parameter calibration (C0).

Since the calibrated values vary across the test cases, it is reasonable to assume that we find only local minima. Nevertheless, the results are remarkable. Since in the real market the parameters are not considered constant, we improve the approach by considering different parameters and some parameter sets as time-dependent. From the relative change in the constant calibration setting, we select the following (additional) test cases, which are listed in Table 6.18. The table also contains the links to the cost functional reduction tables and cost functional evolution figures for the different cases.

	κ	μ	σ_v	ρ
C0	constant	constant	constant	constant
C1	time-dependent	constant	constant	constant
C2	constant	time-dependent	constant	constant
C3	constant	constant	time-dependent	constant
C4	constant	constant	constant	time-dependent
C5	time-dependent	time-dependent	time-dependent	time-dependent
C6	constant	time-dependent	time-dependent	time-dependent
C7	constant	time-dependent	constant	time-dependent

Table 6.18: Different scenario cases for the calibration setting.

For each case of the time-dependent test cases u_{data} is generated as before and $\boldsymbol{\xi}^{\text{init}}$ is assumed to be constant and thus also the same as before.

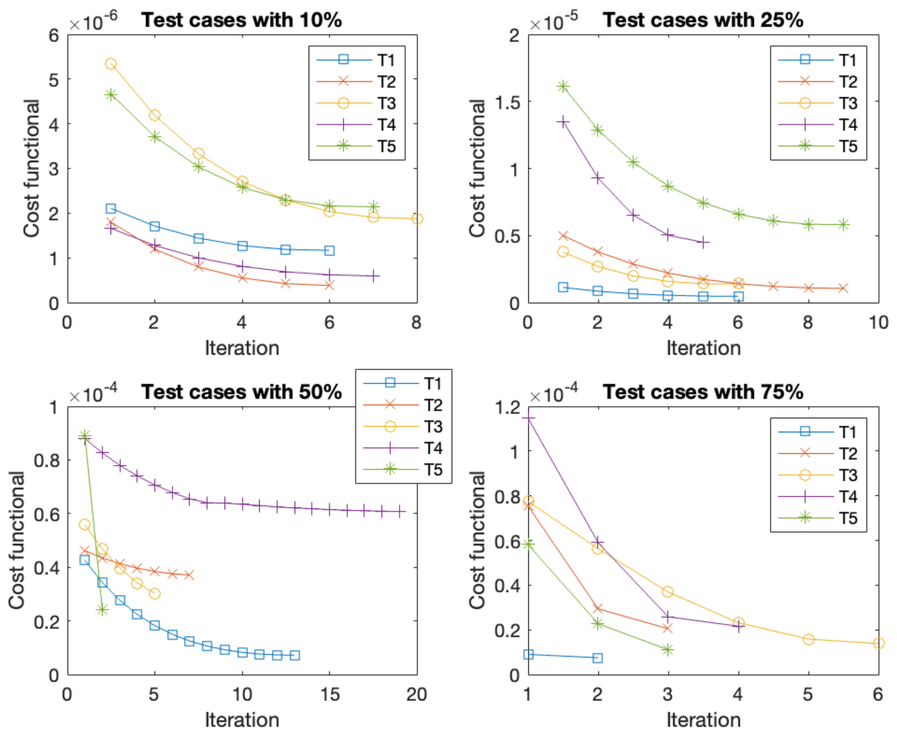


Figure 6.5: Cost functional evolution per iteration for different test sets in scenario C1.

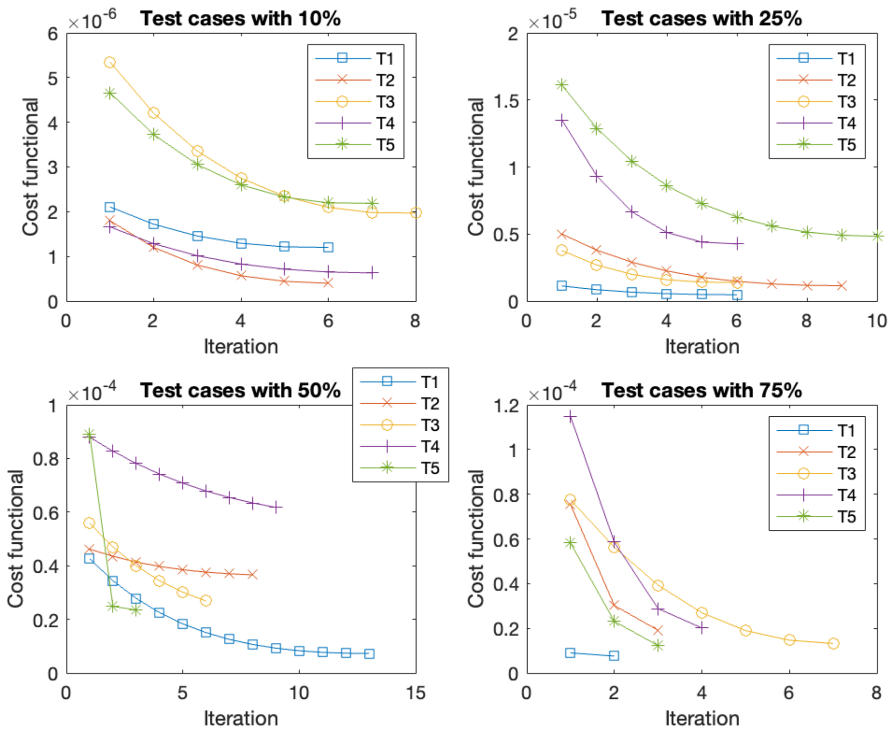


Figure 6.6: Cost functional evolution per iteration for different test sets in scenario C2.

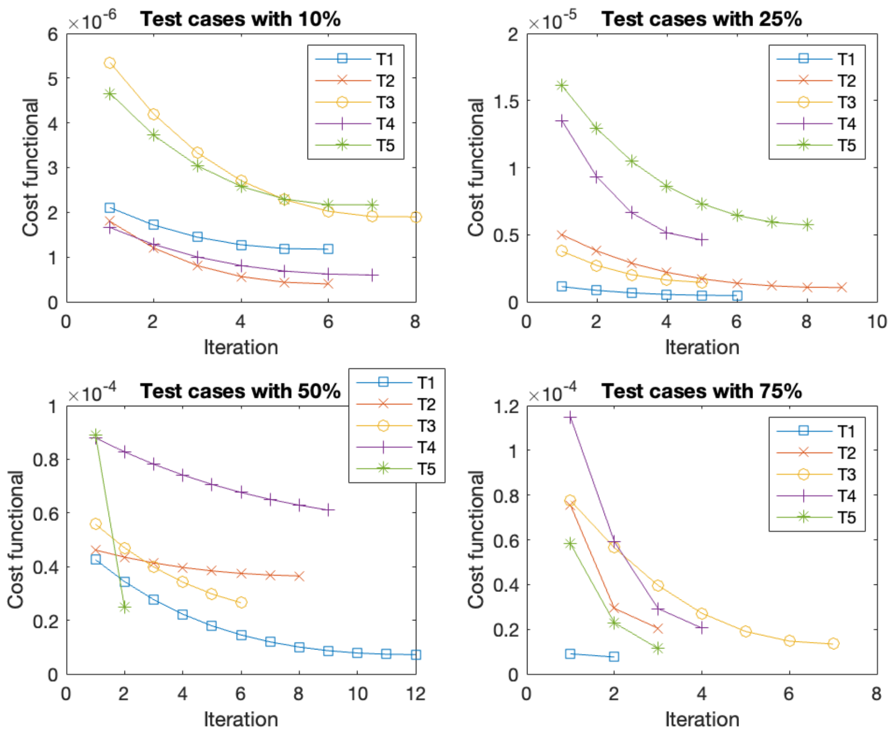


Figure 6.7: Cost functional evolution per iteration for different test sets in scenario C3.

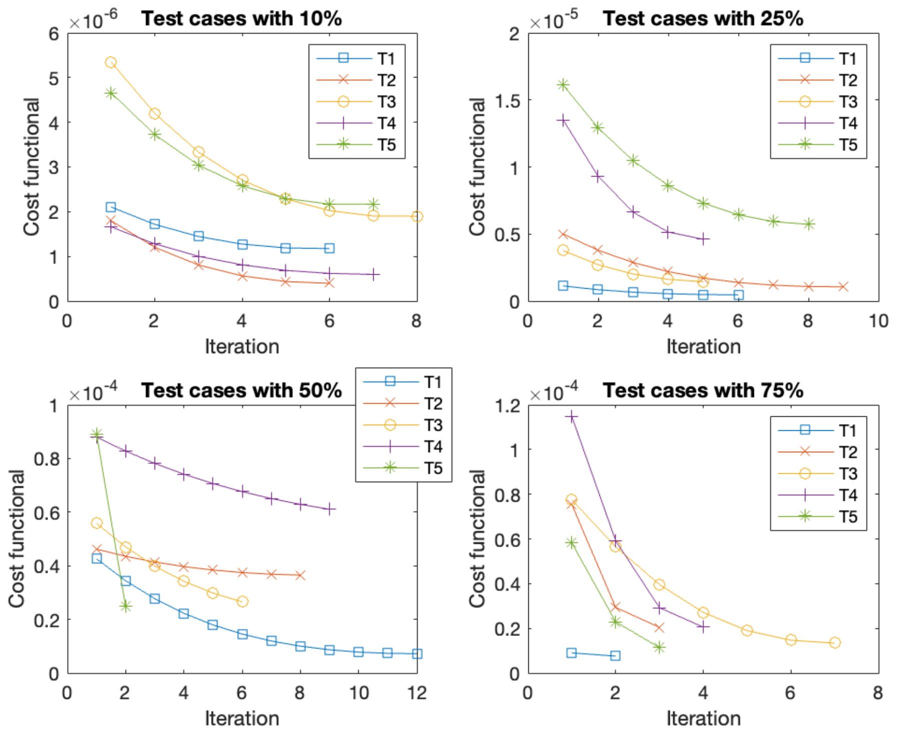


Figure 6.8: Cost functional evolution per iteration for different test sets in scenario C4.

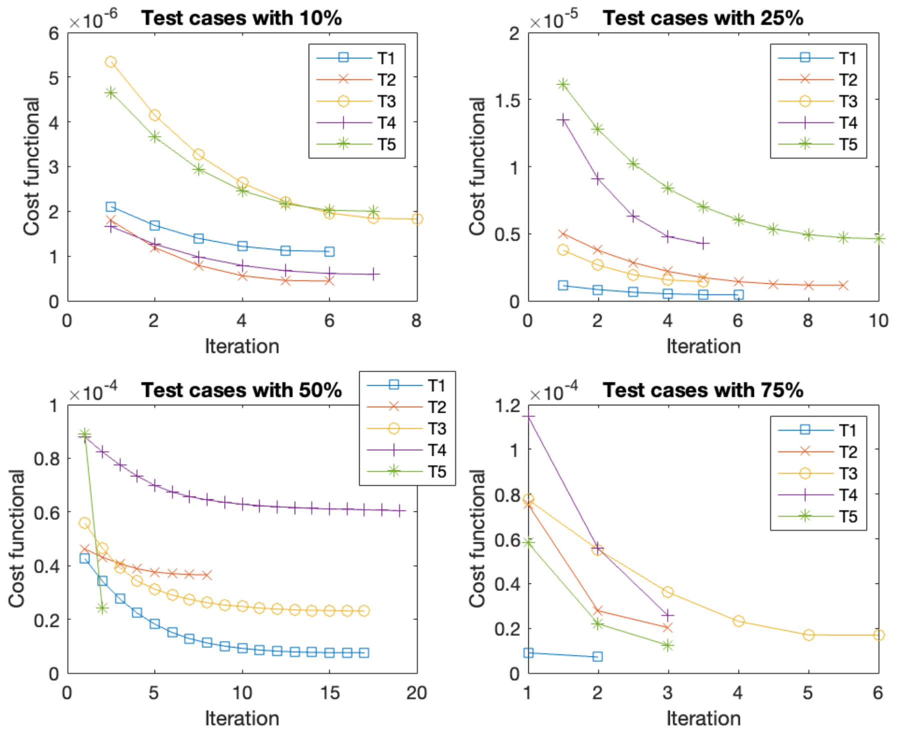


Figure 6.9: Cost functional evolution per iteration for different test sets in scenario C5.

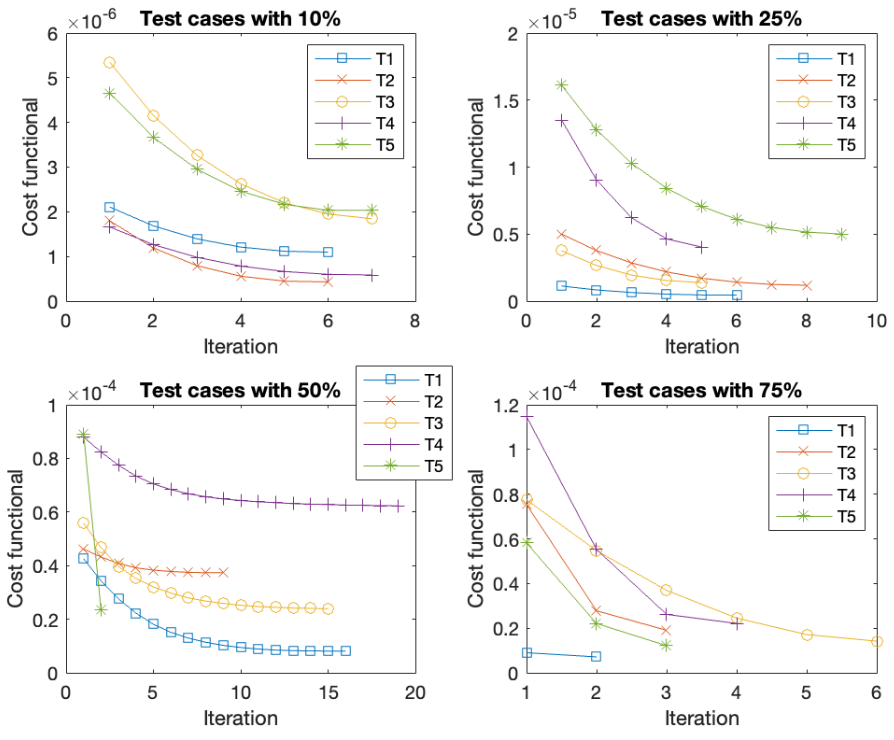


Figure 6.10: Cost functional evolution per iteration for different test sets in scenario C6.

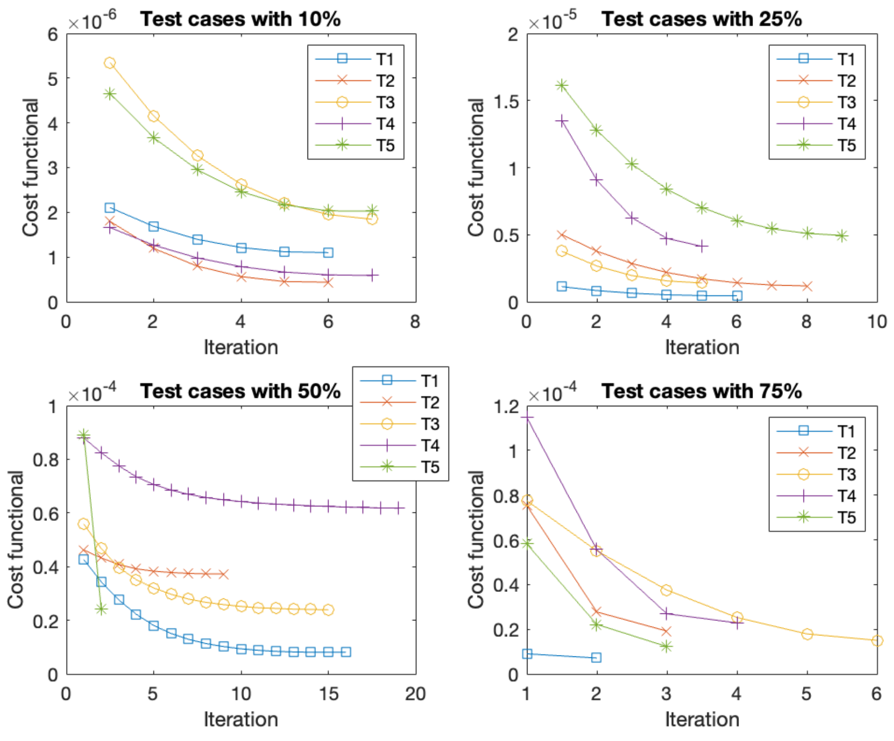


Figure 6.11: Cost functional evolution per iteration for different test sets in scenario C7.

C1	T1	T2	T3	T4	T5
10	44.48	78.75	64.91	64.12	54.03
25	58.97	78.98	62.96	66.77	63.96
50	83.13	19.59	45.71	33.45	72.84
75	16.59	72.80	82.14	81.13	80.94

Table 6.19: Relative cost function reduction for the C1 calibration.

C2	T1	T2	T3	T4	T5
10	42.96	77.85	63.17	62.09	53.05
25	58.03	77.05	63.09	68.46	70.00
50	82.90	20.79	51.89	29.79	73.66
75	16.05	74.64	82.95	82.43	78.89

Table 6.20: Relative cost function reduction for the C2 calibration.

In general, the Figures 6.4–6.11 show the same cost functional evolution. The first test cases with the initial guess closest to the reference set have the smallest initial cost functional value. As the distance to the reference set increases, so does the initial value of the cost functional. Accordingly, the final cost function value is higher for the sets with a more distant initial guess. However, the 75 % deviation within the initial guess shows the greatest cost functional reduction within the first few steps. This is due to the existence of different minima, as these sets find the closest minima instead of the optimal minima. Overall, the cost functional reduction per iteration decreases as the number of iterations increases. The number of iterations within the different test cases with 10 % deviation is the same for all scenarios, except for T2 where it varies between 7 and 8. The scenarios with 75 % deviation give the same number of iterations in cases T1, T2 and T5, for cases T3 and T4 we observe iteration numbers with a total difference of one from each other. For the deviation of 25 % we observe the same number of iterations for T1, a difference of one iteration for T2, T3 and T4, and a difference of two for the last case T5. The largest difference in the number of iterations is found in the 50 % deviation. While T5 has the same number of iterations and T2 has a total difference of 2 for all cases. For the other test cases, the Table 6.26 shows the number of iterations. These are the only cases

C3	T1	T2	T3	T4	T5
10	44.22	77.89	64.50	64.04	53.52
25	58.49	78.76	61.62	65.92	64.50
50	83.05	20.97	52.49	30.55	72.23
75	15.96	72.91	82.71	82.10	80.44

Table 6.21: Relative cost function reduction for the C3 calibration.

C4	T1	T2	T3	T4	T5
10	49.03	75.90	66.62	66.49	56.81
25	60.42	77.71	62.64	68.50	62.96
50	78.54	18.24	52.63	30.29	74.03
75	20.89	73.59	81.75	80.49	79.74

Table 6.22: Relative cost function reduction for the C4 calibration.

C5	T1	T2	T3	T4	T5
10	47.63	75.50	65.79	64.45	57.08
25	60.70	76.67	62.36	68.51	71.33
50	82.32	20.89	58.63	31.72	72.96
75	20.79	73.10	78.14	77.71	78.76

Table 6.23: Relative cost function reduction for the C5 calibration.

C6	T1	T2	T3	T4	T5
10	47.91	76.08	65.50	64.77	56.43
25	60.27	76.39	63.37	70.16	68.91
50	80.94	19.19	57.15	29.38	73.71
75	20.84	74.74	81.70	80.74	79.10

Table 6.24: Relative cost function reduction for the C6 calibration.

C7	T1	T2	T3	T4	T5
10	47.79	75.66	65.50	64.66	56.48
25	60.21	76.37	62.80	69.43	69.29
50	81.08	19.35	57.16	30.05	73.02
75	20.62	74.55	80.59	80.09	78.89

Table 6.25: Relative cost function reduction for the C7 calibration.

where we see an increase in the number of iterations due to considering time-dependent parameters, since the number of iterations increases when μ is considered time-dependent. These are highlighted in bold in the table. If we look at the corresponding Figures 6.6, 6.9, 6.10 as well as 6.11, we observe that the reduction of the cost functional in the end is very small, s.t. the gain of ten iterations forward is almost negligible. Thus, in all cases, 10 iterations are sufficient for a sufficient reduction of the cost functional.

Test cases	C0	C1	**C2**	C3	C4	**C5**	**C6**	**C7**
T1	12	12	**13**	12	12	**17**	**16**	**16**
T3	6	6	**5**	6	6	**17**	**15**	**15**
T4	9	9	**19**	9	9	**19**	**19**	**19**

Table 6.26: Number of iterations for the different case scenarios for cases T1, T3, and T4 within the 50 % percentage difference in the initial parameter set. The bold cases illustrate the scenarios where μ is considered time-dependent.

Comparing the relative cost functional values for each case gives an absolute difference between the reductions, see Table 6.27. We observe that in 9 cases the difference is less than three, while in eight cases it is between three and five, and only in three cases it is greater than five. These are T1 with a deviation of 10 % with a difference of 5.91, a difference of 8.37 for T5 with a deviation of 25 %, and T4 with a deviation of 50 % with the highest difference of 12.93. Overall, we observe that the reductions between C0–C4 and C5–C7 are comparable, while a larger difference can be seen between these two case sets.

Difference	T1	T2	T3	T4	T5
10	5.91	2.39	2.12	3.27	4.33
25	2.67	2.61	1.75	4.24	8.37
50	4.59	2.73	12.92	4.07	1.41
75	4.93	1.94	4.81	4.29	2.85

Table 6.27: Absolute difference between the largest and smallest relative cost functional reduction.

To quantify the general behavior for the different calibrations, we compute the average relative cost functional using (6.5) and summarize the results in Table 6.28. Note, that all cost function reduction averages are huge. We observe that a time-dependent calibration for only one parameter (C1–C4) doesn't improve the cost functional reduction significantly. The first slight improvement can be found by using at least two time-dependent parameters (C5–C7). Surprisingly, C7, where κ is the only variable calibrated as a constant, gives the best calibration results, even though it has only twice the smallest cost functional reduction and never the largest, and C5, where all parameters are calibrated as time-dependent, gives the least improvement when considering combinations of time-dependent parameter calibrations. The fact that u_{data} is generated with constant parameters and the best case considers time-dependent parameters indicates that time dependence is a good way to overcome the local minima. These results are in line with the literature [81, 96].

Case Setup	C0	C1	C2	C3	C4	C5	C6	C7
Mean	61.30	61.31	61.49	61.34	61.86	62.25	62.36	66.12
Table	6.17	6.19	6.20	6.21	6.22	6.23	6.24	6.25
Figure	6.4	6.5	6.6	6.7	6.8	6.9	6.10	6.11

Table 6.28: Average cost functional reduction in percentage for the different cases over the 20 test cases, a list for the corresponding relative cost functional reduction and the figures of the cost functional evolution.

6.6 Space Mapping for the Heston Model

From the Put options on the Nikkei 300 index on December 31, 2012, we get one s^{spot} and five different sets, each with the following parameters r, q, and K. Thus, we specify K1, K2, K3, K4 and K5 for the different market data sets. Since the maturity must be small to use the European option price as a proxy for the Asian option price, we focus on $T = 0.25$. For the spatial discretization we set $\mathbf{x} = (v, x)$ with $\mathbf{x}^{\max} = (1, \log(1.2) \cdot s^{\text{spot}}))$, $\mathbf{M} = (100, 120)$ and $\mathbf{x}^{\min} = \mathbf{x}_{\max}/\mathbf{M}$. We also choose $v^{\text{spot}} = 0.05$. As a result of this discretization, the strike price (and thus the kink in the payoff-function) is approximated at a grid point. Therefore, we smooth the initial condition using the operator from Kreiss et al. [65] The time discretization uses a uniform time discretization, with $N_\tau = 170$. We consider six different initial guesses for the algorithm, see Table 6.29; each parameter set satisfies the Feller condition.

First, we focus on the gradient descent algorithm. For the algorithm, we set the iteration maximum to 51 and the terminal condition to $J(u_c, \boldsymbol{\xi}_{\text{c}}) < 10^{-3}$. Since we are using a gradient-based algorithm, we can only expect to converge to a local minimum, so to evaluate the descent over iterations we use the relative reduction of the cost functional with the initial guess $\boldsymbol{\xi}_{\text{c}}^{\text{init}}$ from Table 6.29 and $\boldsymbol{\xi}^{\text{cal}}$. Table 6.30 shows the cost functional reduction of the last and/or optimal cost function value. The bold values indicate cases

Guess	κ	μ	σ_v	ρ
1	3.0	0.3	0.1	-0.2
2	5.0	0.6	0.2	-0.3
3	4.5	0.8	0.5	-0.15
4	2.0	0.4	0.45	-0.2
5	4.0	0.5	0.15	-0.35
6	3.5	0.35	0.5	-0.5

Table 6.29: Different initial guess sets for the initial coarse model calibration.

where the maximum number of iterations was reached. We observe that the gradient descent algorithm calibrates the parameter $\boldsymbol{\xi}_\mathrm{c}$ almost perfectly, even if we can't guarantee to find the global minimum. In Table 6.31, which shows the optimal value of the cost functional, we observe that for most test cases we reach the terminal condition of the gradient algorithm. The cases where the condition is not reached at the iteration maximum are shown in bold. The cases with the highest cost functional values correspond to the smallest cost function reduction. To illustrate the cost functional reduction per iteration, Figure 6.12 shows the value for the first 10 iterations. We observe that the maximum number of iterations can be significantly reduced depending on the desired accuracy.

K \Guess	1	2	3	4	5	6
1	99.97	99.91	99.98	99.66	99.99	**99.95**
2	99.97	99.92	99.98	99.81	99.98	99.96
3	99.97	**99.51**	**99.69**	99.50	99.98	**99.93**
4	99.97	**99.75**	**99.97**	99.72	99.96	**99.87**
5	**99.96**	99.92	99.98	99.92	**96.15**	**99.49**

Table 6.30: Cost functional reduction for the initial calibration of the coarse model to obtain $\boldsymbol{\xi}_\mathrm{c}^*$.

K \Guess	1	2	3	4	5	6
1	0.888	0.861	0.578	0.808	0.289	**1.058**
2	0.928	0.845	0.680	0.603	0.401	0.892
3	0.933	**4.638**	**9.914**	0.516	0.461	**1.702**
4	0.906	**1.639**	**1.248**	0.678	0.707	**3.542**
5	**1.031**	0.928	0.988	0.689	**56.160**	**17.323**

Table 6.31: Cost functional value for the optimal calibration of the coarse model $\mathcal{J}(u_\mathrm{c}(\boldsymbol{\xi}_\mathrm{c}^*), u_\mathrm{data})$. (scaled by 10^3).

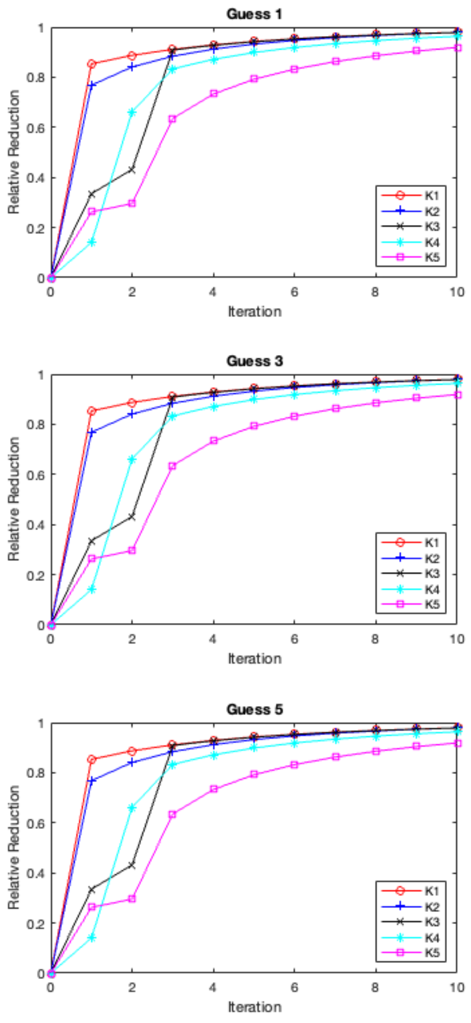
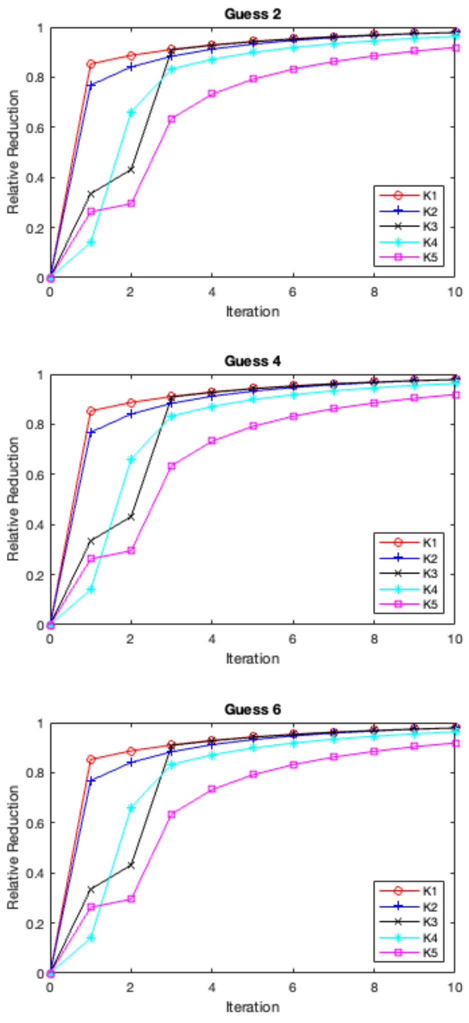


Figure 6.12: Cost functional evolution for the first 10 iterations for the initial calibration for the coarse model.



These results show that the presented gradient descent algorithm is a viable choice for calibration to real market data. For a more detailed analysis of the gradient descent algorithm, see [14]. Within the space mapping, we limit ourselves to a maximum of four iterations, since the evaluation of the fine model is expensive. As a calibration measure we again use the relative cost functional reduction, the results are presented in Table 6.32. The results show that the space mapping approach has only slightly lower reduction rates as the gradient descent algorithm itself, except for K5. One can improve this value by choosing a smaller υ^{spot}, as can be observed in K5a, where we choose $\upsilon^{\text{spot}} = 0.03$. Note that the choice of υ^{spot} is limited by the grid structure. In addition, the calibration to υ^{spot} can be included in the gradient descent algorithm.

Table 6.33 shows the total cost functional reduction relative to the initial guess and Table 6.32 shows the optimal cost functional value. Similar to the gradient algorithm, we observe that when the cost functional reduction is small, the cost functional value is larger.

K \Guess	1	2	3	4	5	6
1	98.57	98.75	99.07	96.73	99.52	99.89
2	99.66	99.21	99.96	72.41	98.93	99.22
3	98.01	99.64	99.98	98.33	91.67	99.39
4	92.42	99.20	93.42	98.44	98.05	99.74
5	28.22	53.71	64.48	52.74	42.48	71.84
5a	44.01	46.58	43.52	61.80	73.26	79.14

Table 6.32: Cost functional reduction the calibration of the space mapping with for $\boldsymbol{\xi}_{\text{c}}^{\text{init}}$ and $\boldsymbol{\xi}_{\text{f}}^{*}$.

Since $\mathcal{J}(u_{\text{c}}(\boldsymbol{\xi}_{\text{c}}^{*}); u_{\text{data}}) \approx \mathcal{J}(u_{\text{f}}(\boldsymbol{\xi}_{\text{f}}^{0}); u_{\text{data}})$ can be significantly worse than $\mathcal{J}(u_{\text{f}}(\boldsymbol{\xi}_{\text{c}}^{\text{init}}); u_{\text{data}})$, we present two figures. One figure shows the reduction of the cost functional per iteration, w.r.t. the initial guess $\boldsymbol{\xi}_{\text{c}}^{\text{init}}$, see Figure 6.13, and the other figure shows w.r.t. the optimal calibration parameters resulting from the initial coarse model calibration $\boldsymbol{\xi}_{\text{c}}^{*} = \boldsymbol{\xi}_{\text{f}}^{0}$, see Figure 6.14. The Figures 6.13 and 6.14 and Table 6.34 show that even if $\boldsymbol{\xi}_{\text{f}}^{0}$ results in a higher cost functional value for the space mapping at iteration 0, the space mapping reduces the cost functional significantly, e.g. Guess 1 with K4, Guess 4 with K2 as well as Guess 5 with K3.

K \Guess	1	2	3	4	5	6
1	0.021	0.006	0.012	0.007	0.005	0.001
2	0.004	0.002	0.001	0.003	0.009	0.009
3	0.012	0.001	0.022	0.011	0.015	0.0110
4	0.027	0.010	0.2010	0.024	0.014	0.007
5	1.159	1.124	1.480	1.290	1.110	1.046
5a	0.574	0.641	1.001	0.699	0.593	0.492

Table 6.33: Cost functional value for the optimal calibration of the space mapping; $\mathcal{J}(u_{\mathrm{f}}(\boldsymbol{\xi}_{\mathrm{f}}^{*}), u_{\mathrm{data}})$.

K \Guess	1	2	3	4	5	6
1	96.12	92.53	97.73	41.27	93.00	87.99
2	84.12	20.39	86 01	-1069.14	78.50	85.35
3	-28.13	-113.55	64.14	-5.97	-314.86	58.95
4	-317.71	-21.38	49.87	5.85	-100.91	43.96
5	-53.76	-1.80	39.41	10.54	-28.04	35.00

Table 6.34: Cost functional reduction for the initial guess $\boldsymbol{\xi}_{\mathrm{c}}^{\mathrm{init}}$ and the calibration of the coarse model $\boldsymbol{\xi}_{\mathrm{c}}^{*}$.

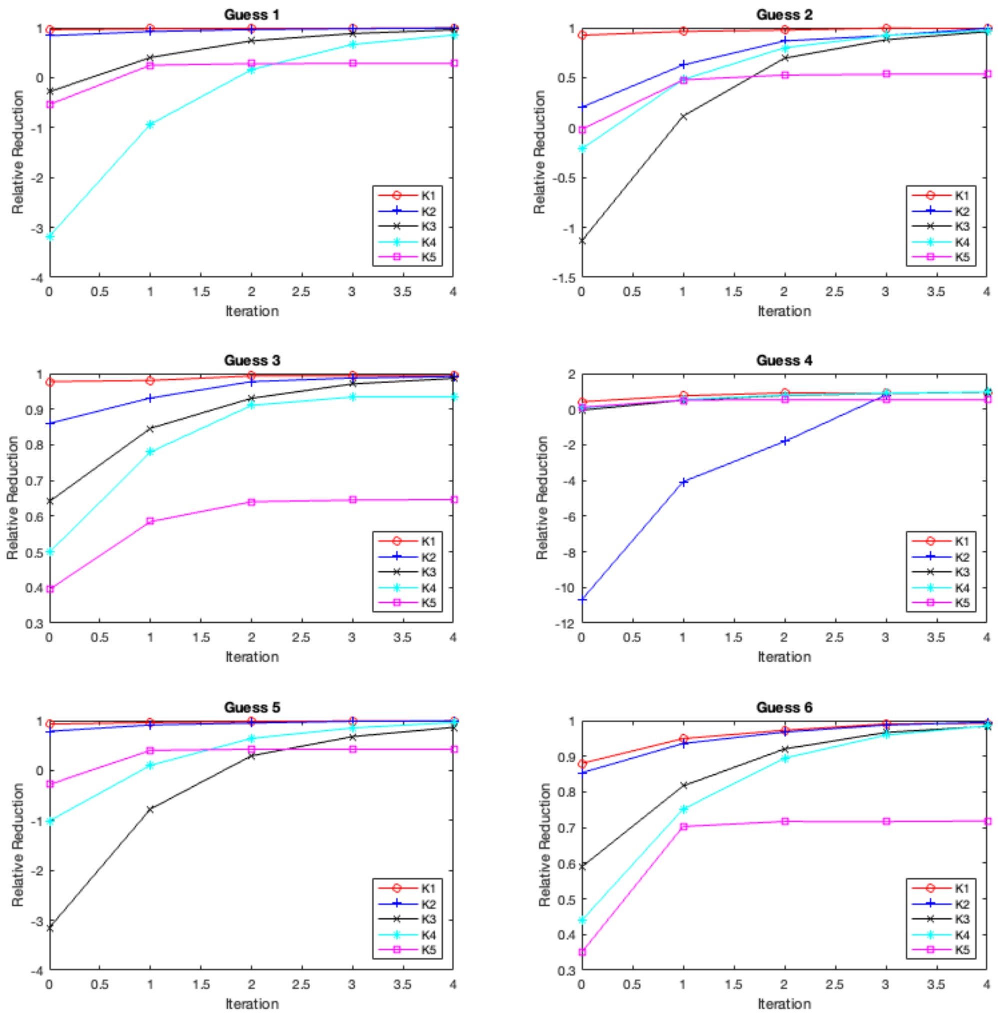


Figure 6.13: Cost functional reduction within the space mapping algorithm for the fine model $\boldsymbol{\xi}_{\mathrm{f}}^{*}$ adjusted to $\boldsymbol{\xi}_{\mathrm{c}}^{\mathrm{init}}$.

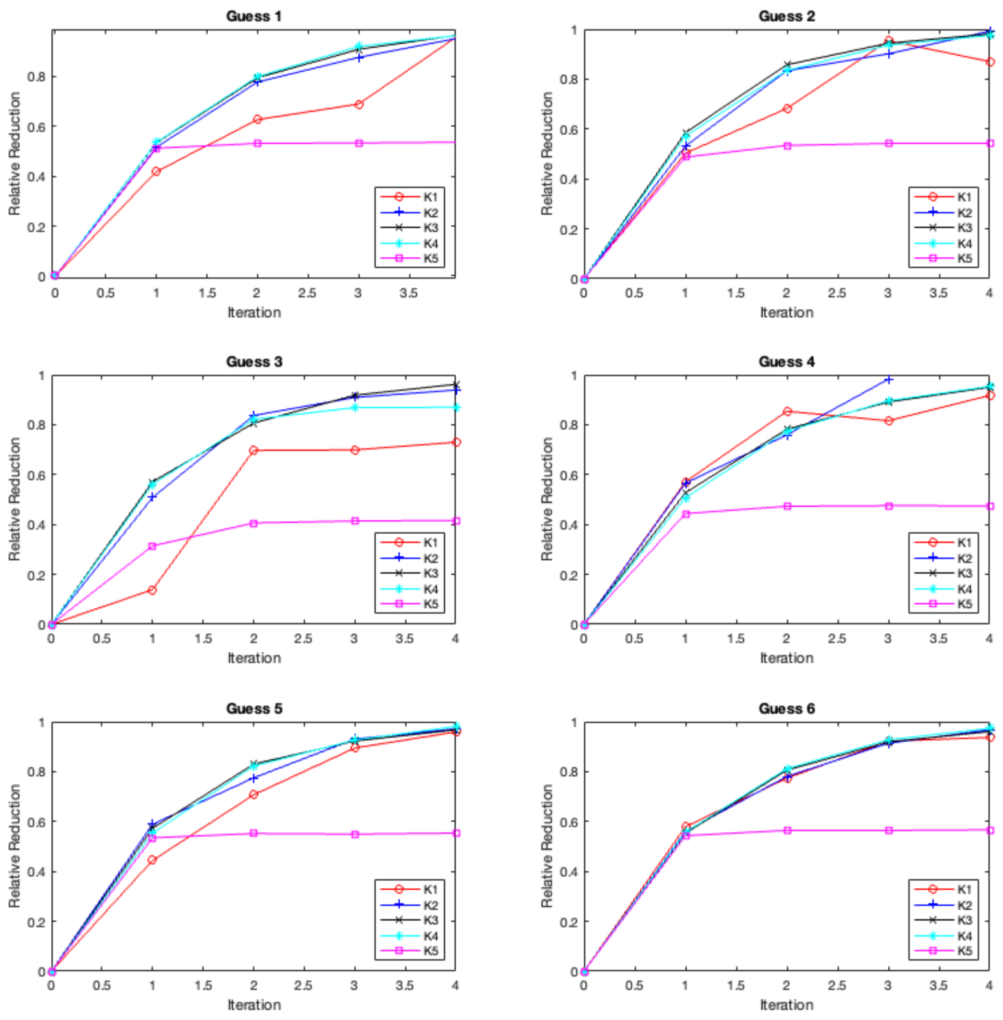


Figure 6.14: Cost functional reduction within the space mapping algorithm for the fine model $\boldsymbol{\xi}_{\mathrm{f}}^{*}$ adjusted to $\boldsymbol{\xi}_{\mathrm{f}}^{0} = \boldsymbol{\xi}_{\mathrm{c}}^{*}$.

Chapter 7

Conclusion and Outlook

In this thesis, hierarchical numerical approaches, as well as different option pricing techniques, have been addressed. First, we give an individual conclusion to the specific numerical test cases and then we summarize conclusions for the application of hierarchical in general. We then present an outlook for each topic individually. This is followed by a more general outlook.

7.1 Conclusion

For the penalty term approach, the numerical results provide clear evidence that the use of a non-constant penalty parameter $\delta(\tau)$ is both feasible and beneficial. In this thesis we have shown that the numerical results of the grid transformation on the spot price with the reduced resolution in the direction of volatility are satisfactory even without smoothing. Even the additional constraint $\mathrm{l}_1 > \mathrm{l}_2$ with $\mathbf{l}_{\mathrm{diff}}$ large, which leads to a high resolution reduction in the direction of volatility, is feasible. As in this setting, the best improvement was achieved by using the spot price as the accumulation point of the transformation (instead of the strike price). Furthermore, we achieved a runtime improvement by reducing the grid resolution in the direction of volatility, even though we have already worked on a sparse grid structure and runtime-optimized implementations. Combining the sparse grid approach with the Parareal algorithm, the numerical results show that for high numbers of processors and iterations, it is recommended to use the intermediate results of the fine solver and to compute the sparse grids in parallel. This statement is true for both accuracy and runtime aspects. For small numbers of iterations, the coarse solver is useful due to the small loss in accuracy compared to the speedup in runtime. Overall, all the ideas presented improve the original Parareal algorithm. The application and the goal of the program will determine which improvement should be chosen.

For the boundary condition we conclude that in case of a free choice it is feasible to use the extrapolation BC for the variance since it has the smallest error in terms of runtime. Especially when the solution depends on the whole domain and not on a single value inside it. If one is limited to Dirichlet BCs due to the numerical method, e.g. sparse grids, one should use the boundary condition B3, see Table 6.8. The gradient descent algorithm as well as the space mapping for the Heston model lead to remarkable relative cost function reductions, although we can expect to find only local minima. Furthermore, the assumption of at least time-dependent parameters within the calibration is better suited to the real market situation, since in the real market almost no parameter is constant. The numerical results show the feasibility of the space mapping approach as a new calibration method in financial research.

Overall, advanced methods based on hierarchical structures are useful within financial research and a combination of them enhances their potential for further improvement.

7.2 Outlook

The inclusion of the free boundary movement in the time-dependent penalty term is a future task. Since a sensitive point of the presented method is the choice of the initial free boundary value, an interesting approach for future research is a detailed analysis of the effect of the choice of the approximation formulas to the solution. As approximation formulas for the initial guess, one can choose the formulas in [108, 93, 27]. Another idea is to consider an iterative scheme. In this idea, the free boundary value of the obtained solution is used as an initial guess for a second iteration of the solution of the penalized American Put problem.

For the sparse grid and Parareal combinations, the ideas can be extended to multidimensional cases, where the improvement is more visible as the total computational cost is higher. The use of the *Multigrid-In-Time (MGRIT)* approach [32, 36] is also possible. Finding the correct boundary conditions for the variance in the Heston model, especially when the Feller condition is not satisfied, is the subject of future research. Furthermore, a study of the effects of artificial boundary conditions on the variance would be beneficial when restricted to small computational domains.

For the space mapping approach, we recognize that there is much room for improvement, such as an extension to time-dependent parameters, see [14] for details on the PDE-constrained optimization problem. In particular, the gradient descent algorithm can be easily adapted to account for time-dependent parameters to improve space mapping for pricing Asian options since they are time-dependent, see [14]. The space mapping approach can be applied to various other hierarchical problems in finance, e.g. model, temporal and spatial, as well as option hierarchies. However, the algorithm can be optimized by using faster or more accurate techniques for solving the Heston PDE and its adjoint equations, as well as for Heston calibration. In addition, further combinations between hierarchical approaches can be invented and analyzed.

References

[1] M. Bakr, J. Bandler, K. Madsen, and J. Søndergaard. Review of the space mapping approach to engineering optimization and modeling. *Optim. Eng.*, 1(3):241–276, 2000.

[2] J. Bandler, Q. Cheng, S. Dakroury, A. Mohamed, M. Bakr, K. Madsen, and J. Sondergaard. Space mapping: the state of the art. *IEEE Trans. Microw. Theory Techn.*, 52(1):337–361, 2004.

[3] F. Black and M. S. Scholes. The pricing of options and corporate liabilities. *J. Polit. Econ.*, 81(3):637–654, 1973.

[4] Z. Bučková, M. Ehrhardt, and M. Günther. Fichera theory and its application in finance. In G. Russo, V. Capasso, G. Nicosia, and V. Romano, editors, *Progress in Industrial Mathematics at ECMI 2014*, pages 103–111, Cham, 2016. Springer.

[5] H.-J. Bungartz and M. Griebel. Sparse grids. *Acta Numerica*, 13:147–269, 2004.

[6] H.-J. Bungartz, M. Griebel, D. Röschke, and C. Zenger. Pointwise convergence of the combination technique for Laplace's equation. *East-West J. Numer. Math.*, 2:21–45, 1994.

[7] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang. *Spectral Methods.* Springer Berlin, Heidelberg, 2006.

[8] P. Carr and D. Madan. Option valuation using the fast Fourier tranform. *J. Comp. Fin.*, 2:61–73, 1999.

[9] S. Cengizci and Ö. Uğur. A computational study for pricing European- and American-type options under Heston's stochastic volatility model: Application of the SUPG-YZβ formulation. *Comput. Econ.*, 2024.

[10] N. Clarke and K. Parrott. The multigrid solution of two factor American put options. Research Report 96-16, Oxford Computing Laboratory, 1996.

[11] N. Clarke and K. Parrott. Multigrid for American option pricing with stochastic volatility. *Appl. Math. Fin.*, 6(3):177–195, 1999.

[12] A. Clevenhaus, M. Ehrhardt, and M. Günther. A parallel sparse grid combination technique using the Parareal algorithm. *IMACM preprint 21/18*, June 2021.

[13] A. Clevenhaus, M. Ehrhardt, and M. Günther. The Parareal algorithm and the sparse grid combination technique in the application of the Heston model. In M. Ehrhardt and M. Günther, editors, *Progress in Industrial Mathematics at ECMI 2021*, pages 477–483, Cham, 2023. Springer International Publishing.

[14] A. Clevenhaus, C. Totzeck, and M. Ehrhardt. A gradient-based calibration method for the Heston model. *Int. J. Comput. Math.*, 101(9-10):1094–1112, 2024.

[15] A. Clevenhaus, C. Totzeck, and M. Ehrhardt. A numerical study of the effects of different boundary conditions on the variance of the Heston model. In K. Burnecki and M. Teuerle, editors, *Progress in Industrial Mathematics at ECMI 2023*, page ??, ??, 2025. Springer International Publishing.

[16] A. Clevenhaus, C. Totzeck, and M. Ehrhardt. A space mapping approach for the calibration of financial models with the application to the Heston model. *arXiv preprint arXiv:2501.14521 (January 2025)*, 2025. submitted to J. Comput. Finance.

[17] R. Company, V. Egorova, and L. Jódar. Solving American option pricing models by the front fixing method: Numerical analysis and computing. *Abstr. Appl. Anal.*, pages Aricle ID: 146745, 9 pages, 2014.

[18] I. Craig and A. Sneyd. An alternating-direction implicit scheme for parabolic equations with mixed derivatives. *Comput. Math. Appl.*, 16(4):341–350, 1988.

[19] C. W. Cryer. The solution of a quadratic programming problem using systematic overrelaxation. *SIAM J. Control*, 9(3):385–392, 1971.

[20] Y. Cui, S. del Baño Rollin, and G. Germano. Full and fast calibration of the Heston stochastic volatility model. *Eur. J. Oper. Res.*, 263 (2):625–638, 2017.

[21] J. Douglas and H. Rachford. On the numerical solution of heat conduction problems in two and three space variables. *Trans. Amer. Math. Soc.*, 82:421–439, 1956.

[22] B. Düring, M. Fournié, and A. Rigal. High-order ADI schemes for convection-diffusion equations with mixed derivative terms. In M. Azaïez, H. El Fekih, and J. S. Hesthaven, editors, *Spectral and High Order Methods for Partial Differential*

Equations - ICOSAHOM 2012, pages 217–226, Cham, 2014. Springer International Publishing.

[23] B. Düring and C. Heuer. High-order compact schemes for parabolic problems with mixed derivatives in multiple space dimensions. *SIAM J. Numer. Anal.*, 53(5):2113–2134, 2015.

[24] B. Düring and J. Miles. High-order ADI scheme for option pricing in stochastic volatility models. *J. Comput. Appl. Math.*, 316:109–121, 2017. Selected Papers from NUMDIFF-14.

[25] D. Echeverría, D. Lahaye, and P. Hemker. Space mapping and defect correction. In *Model Order Reduction: Theory, Research Aspects and Applications.* Springer, Berlin, Heidelberg, 2008.

[26] M. Ehrhardt. *Nonlinear Models in Mathematical Finance: New Research Trends in Option Pricing.* Advances in Mathematical Inequalities. Nova Science Publishers, 2008.

[27] J. Evans, R. Kuske, and J. Keller. American options on assets with dividends near expiry. *Math. Finance*, pages 219–237, 2002.

[28] F. Fang and C. W. Oosterlee. A novel pricing method for European options based on Fourier-Cosine series expansion. *SIAM J. Sci. Comput.*, 31(2):826–848, 2008.

[29] R. Fazio, A. Insana, and A. Jannelli. A front fixing finite difference method for the American put options model, 4 2020. arXiv preprint, arXiv:2004.03595 [math.NA].

[30] A. Ferreiro-Ferreiro, J. García-Rodríguez, L. Souto, and C. Vázquez. A new calibration of the Heston stochastic local volatility model and its parallel implementation on GPUs. *Math. Comput. Simul.*, 177:467–486, 2020.

[31] J. Frank, W. Hundsdorfer, and J. G. Verwer. On the stability of implicit-explicit linear multistep methods. *Appl. Numer. Math.*, 25(2-3):193–205, 1997.

[32] S. Friedhoff, R. Falgout, T. Kolev, S. P. MacLachlan, and J. B. Schroder. A multigrid-in-time algorithm for solving evolution equations in parallel. In *Presented at: Sixteenth Copper Mountain Conference on Multigrid Methods, Copper Mountain, CO, United States, Mar 17 - Mar 22, 2013*, 2013.

[33] M. J. Gander and S. Vandewalle. Analysis of the Parareal time-parallel time-integration method. *SIAM J. Sci. Comput.*, 29(2):556–578, 2007.

[34] P. Glasserman. *Monte Carlo Methods in Financial Engineering: Applications of Mathematics*. Springer, New York, 2004.

[35] M. Günther and A. Jüngel. *Finanzderivate mit MATLAB: Mathematische Modellierung und Numerische Simulation*. Vieweg, 2 edition, 2010.

[36] W. Hackbusch. *Multi-Grid Methods and Applications*. Springer Berlin, Heidelberg, 1985.

[37] T. Haentjens. Efficient and stable numerical solution of the Heston-Cox-Ingersoll-Ross partial differential equations by alternating direction implicit finite difference schemes. *Int. J. Comput. Math.*, 90(11):2409–2430, 2013.

[38] T. Haentjens and K. J. in 't Hout. ADI finite difference schemes for the Heston-Hull-White PDE. *J. Comput. Fin.*, 16:83–110, 2012.

[39] T. Haentjens and K. J. in 't Hout. ADI schemes for pricing American options under the Heston model. *Appl. Math. Fin.*, 22(3):207–237, 2015.

[40] C. Hendricks. *High-Order Methods for Parabolic Equations in Multiple Space Dimensions for Option Pricing Problems*. PhD thesis, Bergische Universität Wuppertal, 2016.

[41] C. Hendricks, M. Ehrhardt, and M. Gunther. High order combination technique for the efficient pricing of basket options. *Acta Math. Uni. Comenianae*, 84(2):243–253, 2015.

[42] C. Hendricks, M. Ehrhardt, and M. Günther. High-order ADI schemes for diffusion equations with mixed derivatives in the combination technique. *Appl. Numer. Math.*, 101:36–52, 2016.

[43] C. Hendricks, M. Ehrhardt, and M. Günther. Error splitting preservation for high order finite difference schemes in the combination technique. *Numer. Math. Theor. Meth. Appl.*, 10(3):689–710, 2017.

[44] C. Hendricks, M. Ehrhardt, and M. Günther. Hybrid finite–difference / pseudospectral methods for the Heston and Heston-Hull-White partial differential equations. *J. Comput. Finance*, 21(5):1–33, 2018.

[45] C. Hendricks, C. Heuer, M. Ehrhardt, and M. Günther. High-order-compact ADI schemes for pricing basket options in the combination technique. In *Novel Methods in Computational Finance*, pages 399–405. Springer, 2017.

[46] S. L. Heston. A closed-form solution for options with stochastic volatility with applications to bond and currency options. *Rev. Fin. Stud.*, 6(2):327–343, 1993.

[47] M. Hinze, R. Pinnau, M. Ulbrich, and S. Ulbrich. *Optimization with PDE Constraints.* Springer, 2009.

[48] B. Horvath, A. Muguruza, and M. Tomas. Deep learning volatility: a deep neural network perspective on pricing and calibration in (rough) volatility models. *Quant. Fin.*, 21(1):11–27, 2021.

[49] J. Hull. *Options, futures, and other derivatives.* Prentice-Hall, Upper Saddle River, NJ, third edition edition, 1997.

[50] W. Hundsdorfer. A note on stability of the Douglas splitting method. *Math. Comput.*, 67(221):183–190, 1998.

[51] W. Hundsdorfer. Stability of approximate factorization with θ-methods. *BIT Numer. Math.*, 39(3):473–483, 1999.

[52] W. Hundsdorfer. Accuracy and stability of splitting with stabilizing corrections. *Appl. Numer. Math.*, 42(1-3):213–233, 2002.

[53] W. Hundsdorfer and J. Verwer. *Numerical solution of time-dependent advection-diffusion-reaction equations.* Springer, Berlin, 2003.

[54] S. Ikonen and J. Toivanen. Operator splitting methods for American option pricing. *Appl. Math. Lett.*, 17(7):809–814, 2004.

[55] S. Ikonen and J. Toivanen. Operator splitting methods for pricing American options under stochastic volatility. *Numer. Math.*, 113(2):299–324, 2009.

[56] K. J. in ’t Hout and S. Foulon. ADI finite difference schemes for option pricing in the Heston model with correlation. *Int. J. Numer. Anal. Mod.*, 7(2):303–320, 2010.

[57] K. J. in ’t Hout and B. Welfert. Stability of ADI schemes applied to convection-diffusion equations with mixed derivative terms. *Appl. Numer. Math.*, 57(1):19–35, 2007.

[58] K. J. in ’t Hout and B. Welfert. Unconditional stability of second-order ADI schemes applied to multi-dimensional diffusion equations with mixed derivative terms. *Appl. Numer. Math.*, 59(3):677–692, 2009. Selected Papers from NUMDIFF-11.

[59] K. J. in 't Hout and B. Welfert. Unconditional stability of second-order ADI schemes applied to multi-dimensional diffusion equations with mixed derivative terms. *Appl. Numer. Math.*, 59(3-4):677–692, 2009.

[60] K. in't Hout and J. Toivanen. Application of operator splitting methods in finance. In R. Glowinski, S. J. Osher, and W. Yin, editors, *Splitting Methods in Communication, Imaging, Science, and Engineering*, pages 541–575. Springer International Publishing, Cham, 2016.

[61] K. J. in't Hout and M. C. Stability of the modified Craig-Sneyd scheme for two-dimenional convection-diffusion equations with mixed derivative terms. *Math. Comp. Simul.*, 81:2540–2548, 2011.

[62] K. J. in't Hout and C. Mishra. Stability of ADI schemes for multidimensional diffusion equations with mixed derivative terms. *Appl. Numer. Math*, 74:83–94, 2013.

[63] D. Khalife, J. Yammine, S. Rahal, and S. Freiha. Pricing Asian and barrier options using a combined Heston model and Monte Carlo simulation approach with artificial intelligence. *Math. Modell. Engrg. Probl.*, pages 1690–1698, 2023.

[64] J. Kienitz, T. McWalter, and R. Sheppard. *PDE Methods for SABR*, pages 265–291. Springer International Publishing, Cham, 2017.

[65] H.-O. Kreiss, V. Thomée, and O. B. Widlund. Smoothing of initial data and rates of convergence for parabolic difference equations. *Commun. Pure Appl. Math.*, 23:241–259, 1970.

[66] P. Kútik and K. Mikula. Diamond-cell finite volume scheme for the Heston model. *(DCDS-S*, 8(5):913–931, 2015.

[67] M. Lauko and D. Ševčovič. Comparison of numerical and analytical approximation of the early boundary of American put options. *ANZIAM J.*, 51(4):430–448, 2010.

[68] I. Leite, J. Yamim, and L. da Fonseca. The DeepONets for finance: An approach to calibrate the Heston model. In G. Marreiros et al., editors, *Progress in Artificial Intelligence*, pages 351–362, Cham, 2021. Springer.

[69] R. J. LeVeque. *Numerical Methods for Conservation Laws.* Birkhäuser, Basel, 1992.

[70] J.-L. Lions, Y. Maday, and G. Turinici. Résolution d'EDP par un schéma en temps «pararéel». *Compt. Rend. Acad. Sci. - Ser. I - Math.*, 332(7):661–668, 2001.

[71] S. Liu, A. Borovykh, L. Grzelak, and C. Oosterlee. A neural network-based framework for financial model calibration. *J. Math. Ind.*, 9:1–28, 2019.

[72] S. Liu, C. Oosterlee, and S. M. Bohte. Pricing options and computing implied volatilities using neural networks. *Risks*, 7(1):16, 2019.

[73] M. Luskin and R. Rannacher. On the smoothing property of the Crank-Nicolson scheme. *Appl. Anal.*, 14(2):117–135, 1982.

[74] G. Marchuk. Splitting and alternating direction methods. In *Handbook of Numerical Analysis*, volume 1, pages 197–462. Elsevier, 1990.

[75] N. Marheineke, R. Pinnau, and E. Reséndiz. Space mapping-focused control techniques for particle dispersions in fluids. *Optim. Engrg.*, 13:101–120, 2012.

[76] S. Maurus. A multi-dimensional pde solver for option pricing based on the Heston model and sparse grids. Master's thesis, TU München, 2012.

[77] S. McKee and A. Mitchell. Alternating direction methods for parabolic equations in two space dimensions with a mixed derivative. *Computer J.*, 13:81–86, 1970.

[78] S. McKee, D. Wall, and S. Wilson. An alternating direction implicit scheme for parabolic equations with mixed derivative and convective terms. *J. Comput. Phys.*, 126:64–76, 1996.

[79] F. Mehrdoust, I. Noorani, and A. Hamdi. Two-factor Heston model equipped with regime-switching: American option pricing and model calibration by Levenberg-Marquardt optimization algorithm. *Math. Comput. Simul.*, 204:660–678, 2023.

[80] R. C. Merton. The theory of rational option pricing. *Bell J. Econ. Manage. Sci.*, 4:141–183, 03 1973.

[81] S. Mikhailov and U. Nögel. Heston's stochastic volatility model: Implementation, calibration and some extensions. *Willmott Magazine*, pages 74–79, 2003.

[82] M. Minion. A hybrid parareal spectral deferred corrections method. *Commun. Appl. Math. Comput. Sci.*, 5(2):265–301, 2010.

[83] B. F. Nielsen, O. Skavhaug, and A. Tveito. Penalty and front-fixing methods for the numerical solution of American option problems. *J. Comput. Fin.*, 5(4):69–97, 2002.

[84] C. W. Oosterlee. On multigrid for linear complementarity problems with application to American-style options. *Elec. Trans. Numer. Anal.*, 15:165–185, 2003.

[85] D. Peaceman and H. Rachford. The numerical solution of parabolic and elliptic differentail equations. *J. Soc. Ind. Appl. Math.*, 3:28–41, 1955.

[86] H. Pham. *Continuous-time Stochastic Control and Optimization with Financial Applications.* Springer Berlin, Heidelberg, 2009.

[87] D. Pooley, K. Vetzal, and P. Forsyth. Convergence remedies for non-smooth payoffs in option pricing. *J. Comp. Fin.*, 6(4):25–40, 2003.

[88] R. Rannacher. Finite element solution of diffusion problems with irregular data. *Numer. Math.*, 42(2):309–327, 1984.

[89] C. Reisinger. Analysis of linear difference schemes in the sparse grid combination technique. *IMA J. Num. Ana.*, 33(2):544–581, 2013.

[90] T. Schiekofer. *Die Methode der Finiten Differenzen auf dünnen Gittern zur Lösung elliptischer und parabolischer partieller Differentialgleichungen.* PhD thesis, Universität Bonn, 1998.

[91] R. Seydel. *Tools for Computational Finance.* Springer London, 2017.

[92] S. A. Smolyak. Quadrature and interpolation formulas for tensor products of certain classes of functions. In *Doklady Akademii Nauk*, pages 1042–1045. Russian Academy of Sciences, 1963.

[93] R. Stamicar, D. Ševčovič, and J. Chadam. The early exercise boundary for the American put near expiry: Numerical approximation. *Canad. Appl. Math. Quaterly*, 7(4):427–444, 1999.

[94] D. Tavella and C. Randall. *Pricing Financial Instruments: The Finite Difference Method.* Wiley Series in Financial Engineering. Wiley, 2000.

[95] L. Teng and A. Clevenhaus. Accelerated implementation of the ADI schemes for the Heston model with stochastic correlation. *J. Comput. Sci.*, 36:101022, 2019.

[96] L. Teng, M. Ehrhardt, and M. Günther. On the Heston model with stochastic correlation. *Int. J. Theor. Appl. Fin.*, 6, 2016.

[97] L. Teng, M. Ehrhardt, and M. Günther. Numerical simulation of the Heston model under stochastic correlation. *Int. J. Financial Studies*, 6(1), 2017.

[98] V. Thomée and L. Wahlbin. Convergence rates of parabolic difference schemes for non-smooth data. *Math. Compt.*, 28(125):1–13, 1974.

[99] C. Totzeck and R. Pinnau. Space mapping-based receding horizon control for stochastic interacting particle systems: dogs herding sheep. *J. Math. Ind.*, 10:1–19, 2019.

[100] F. Tröltzsch. *Optimale Steuerung partieller Differentialgleichungen.* Springer, 2009.

[101] J. Verwer, E. Spee, J. Blom, and W. Hundsdorfer. A second-order Rosenbrock method applied to photochemiacal dipersion problems. *SIAM . Sci. Comput*, 20:1456–1480, 1999.

[102] D. Ševčovič. Analysis of the free boundary for the pricing of an American call option. *Europ. J. Appl. Math.*, 12(1):25–37, 2001.

[103] D. Ševčovič. Transformation methods for evaluating approximations to the optimal exercise boundary for linear and nonlinear Black-Scholes equations. *Nonlinear Models in Mathematical Finance: New Research Trends in Option Pricing*, pages 153–198, 2008.

[104] S. Wang, X. Q. Yang, and K. L. Teo. Power penalty method for a linear complementarity problem arising from American option valuation. *J. Optim. Theor. Appl.*, 129(2):227–254, 2006.

[105] M. Williams. *Uncontrolled Risk: Lessons of Lehman Brothers and How Systemic Risk Can Still Bring Down the World Financial System.* McGraw Hill LLC, 2010.

[106] P. Wilmott. *The Theory and Practice of Financial Engineering.* John Wiley & Sons Ltd., Chichester, UK, 1998.

[107] C. Zenger. Sparse grids. Technical report, Institut für Informatik, October 1990.

[108] S.-P. Zhu. A new analytical approximation formula for the optimal exercise boundary of American put options. *Int. J. Theor. Appl. Fin.*, 09(07):1141–1177, 2006.

[109] R. Zvan, P. Forsyth, and K. R. Vetzal. Penalty methods for American options with stochastic volatility. *J. Comput. Appl. Math.*, 91(2):199–218, 1998.