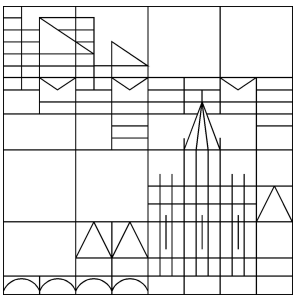
Using POD methods for Option Pricing with Diffusion Models

Master thesis
in partial fulfillment of the requirements
for the Master of Science degree

submitted by

Elena Schnell

at the



Universität Konstanz

Faculty of Sciences

Department of Mathematics and Statistics

1st Supervisor: Herr Prof. Dr. Volkwein
2nd Supervisor: Herr Prof. Dr. Kupper

Konstanz, March 2015

Contents

Chapter 1

Introduction

Jump-diffusion models are nowadays widely used in financial modeling. In contrast to the famous Black-Scholes model, in which sample paths are assumed to be continuous, they are able to capture discontinuities in the underlying assets. These "jumps" have been empirically proven and therefore been incorporated into the model dynamics. Jump-diffusion models combine a jump process and a diffusion process. They are therefore able to represent asset price movements including "market crashes" more realistically. Also realistic volatility skews can be constructed. This comes at the cost of an incomplete market, which makes perfect hedging impossible. Hence hedging has to be reconsidered and new approaches need to be taken in order to measure these "risks that one cannot hedge even by continuous time trading"[1].

Further improvements of the jump-diffusion models include the use of a local volatility function. Here the volatility is not constant, but can be fitted to quoted market prices. Using these models option prices can be calculated by solving a so-called partial-integro differential equation (PIDE). The arising non-local terms make jump-diffusion models numerically challenging. Furthermore using implicit methods for the time discretization results in dense matrices. After we introduce the model, on which our analysis is based, the main part of this thesis consists of the numerical approach. An efficient solution is given by the use of a preconditioned generalized minimum residual (GMRES) algorithm.

In practical applications often exotic options are of interest. For the determination of their fair price, the model must be calibrated correctly in order to be further used in e.g. a Monte-Carlo simulation. Model calibration is out of scope of this thesis, but represents a further application of the methods mentioned here. It involves a constrained optimization problem, where in a non-linear least-squares

[1] Quote from Cont [8], p. 325

formulation the error between the market prices and model prices is minimized. In order to solve the optimization problem efficiently, model order reduction (MOR) is applied. Using proper orthogonal decomposition (POD) a low-dimensional model is derived, which drastically reduces the effort of solving the optimization problem.

In the following section we will illustrate how POD works by giving an example of image compression. Note that all calculations were done in MATLAB on a desktop PC with a Intel Core i7-920 2.66 GHz processor and 15GB RAM.

1.1 Motivation

Nowadays simulation is present in many different fields. It can be used to validate experiments or give predictions of processes. As these models often include many variables, the goal of MOR is to simplify them while making a decent computational effort and an adequate memory capacity for the calculations possible. Realistic simulations however require the model error to be small. Operational MOR tackles this task by incorporating physical characteristics into the model before performing the calculations. Compact MOR uses detailed insight of the problem and the solutions to design efficient models with reduced complexity.

Often, as it is in our case, we do not have a-priori information about the problem. First this information must be obtained. "Model Order Reduction tries to quickly capture the essential features of a structure. This means that in an early stage of the process, the most basic properties of the original model must already be present in the smaller approximation. At a certain moment the process of reduction is stopped. At that point all necessary properties of the original model must be captured with sufficient precision."[2]

In our case we deal with a PIDE, where we apply the finite element (FE) method as space discretization. In a reduced order model the FE basis functions are replaced by a few problem-specific functions. These do not span the whole function space, which is usually H_0^1, but a lower-dimensional subspace. We use the example of image compression to demonstrate the functioning of POD.

Figure 1.1 shows our original image in the bottom right. It is a black and white picture with 1920 x 1080 pixels. Each shade can be represented with a number between 0 (black) and 1 (white). Hence our image can be described by a matrix of dimension 1920×1080, each entry containing information about one pixel. In total

[2]Quote from Schilders [32], p.8

this leads to 2.07mn numbers.

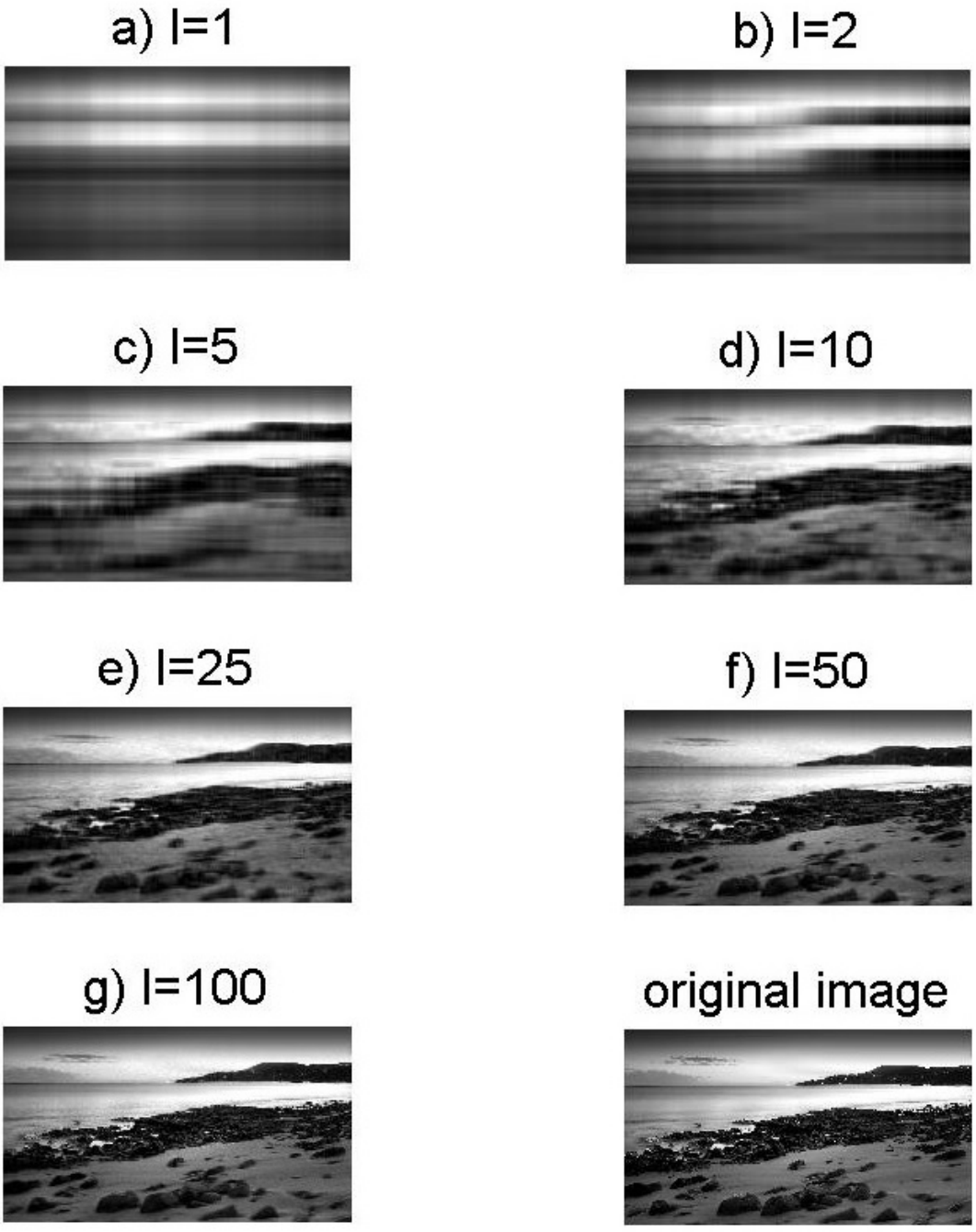


Table 1.1: Approximation of the original image with different numbers l of basis functions

Each column of the matrix can be interpreted as a linear combination of unit vectors, which in this example represent the basis functions. We want to replace these by more complex functions containing the important information. As we can see the essential feature of the picture is the bright top and the dark ground. Hence

advanced basis functions should represent this. They are obtained using singular value decomposition (SVD). Then the matrix columns can be represented as a linear combination of this new basis. As the image is represented in a compressed way, it needs much less memory.

Using only the first basis function $l = 1$ we need to store its values (1920×1 numbers) as well as the weights for each column (1080×1 numbers). This corresponds to 0,14 % of the original memory storage. Figure 1.1 shows the results of the image approximation for different numbers l of basis functions. While in the first picture, only the basic structure of the original image is reproduced, more and more details can be reconstructed by increasing the number of basis functions. For $l = 100$, while only needing 14,47 % of the original memory space, the original picture is already well recovered.

The given example illustrates the concept of MOR in a way that is understandable even for people without mathematical background. It shows that in some cases only few characteristics are needed to describe the model. In our case the model will be the option pricing equation. The solution at one time step corresponds to one column of the matrix described in the example and the finite element basis functions correspond to the unit vectors. We will replace this basis by a POD basis consisting of only a few, e.g. 10, functions. As the considered equation is parabolic, the solution in one time step does not differ much from the solution in the previous time step. In general the solutions are smooth, making a good approximation by a reduced basis possible.

1.2 Outline

This thesis consists mainly of four parts: the derivation of the considered PIDE, the numerical solution of it, the application of POD and the consideration of random input data. In the following we will outline each part.

In Chapter 2 we first give an introduction to options. Our focus will be on European call options. Afterwards two pricing models are considered. Firstly, the Black-Scholes model is described. As this model suffers from well-known weaknesses, we secondly present an extension of it, Merton's jump diffusion model. Our final model of choice will be the jump-diffusion model with a local volatility function as introduced in [1]. Then our focus will lie on the derivation of the PIDE, which can be used to price options. We end up with a Dupire-like version of it dependent on maturity and strike price. As certain parameters in the PIDE are unknown, we state

furthermore the calibration problem and its importance in pricing more advanced options. On this the need of a very efficient solution of the PIDE is found.

Chapter 3 focuses on the numerical solution of the PIDE. First, a weak formulation in a weighted Sobolev space is established and an existence and uniqueness theorem invoked. Afterwards, the finite element method is used as spatial discretization with basis functions in the space H_0^1. The non-local integral term yields hereby a dense stiffness matrix. As the problem is known to be stiff the use of an implicit scheme is preferred as time discretization. In order to achieve second order convergence a damping procedure (cf. [29]) is applied in order to smooth oscillations arising from the non-smooth initial condition. The implicit scheme yields dense linear systems of equations, which are taken care of by a preconditioned GMRES method. Finally the efficiency of the presented algorithm is shown with numerical results.

As mentioned earlier the calibration problem requires a fast computation of the objective function. Even though we do not go further into details regarding model calibration, we will present in Chapter 4 a technique to obtain a reduced order model. Through POD an advanced basis of the solution space can be obtained, which afterwards can be used in a POD Galerkin approach. As this approximation introduces an error, we are interested in its bounds. A-priori error estimates are derived. It is shown that with a full basis the error for the Crank-Nicolson scheme is of order $\mathcal{O}\Delta T^4$. This result is confirmed with numerical experiments.

Chapter 5 gives insight into the case of random input data following [4]. In our case the jump intensity is considered as stochastic variable with assumptions on its distribution. Further numerical approaches such as multiple snapshot sets and a greedy algorithm are described. Lastly, these methods are implemented and a comparison between the standard POD method and the POD-greedy algorithm is performed.

In the end, Chapter 6, we give a conclusion, where we summarize the results and mention further applications.

Chapter 2

Option Pricing

Since the 1970's the role of private finance in the economy has increased tremendously, not only its share of the economy but also in its influence on the real economy and society. In developed economies financial services has become a key industry with a large share of the Gross Domestic Product. The number and types of financial derivatives have grown enormously. A derivative is a financial instrument whose value is derived from its underlying asset. Derivatives are mainly used for hedging, i.e. the mitigation of risk in the underlying, or speculation, the try of earning profit.

In this chapter, we will first give an introduction to options. Options are formally defined and the payoff of a European call is considered (Section 2.1). To determine the fair value of such an option, two option pricing models are discussed: the famous Black-Scholes-model (cf. [3]), where the asset price is modeled as exponential Brownian motion (Section 2.2.1), and its extension, Merton's jump diffusion model (cf. [24]), where jumps are added to the pure diffusion process in order to account for large movements of the asset price (Section 2.2.2). In Section 2.3 the stochastic differential equation (SDE) describing the underlying's behavior in Merton's jump diffusion model is transformed to a PIDE in order to compute option prices. Since these are sensitive to the parameters included in the model, the efficient calibration of model parameters is of great importance and is discussed in Section 2.4.

2.1 Introduction to Options

In contrast to futures or forward contracts, options give the holder the right, but not the obligation, to employ a certain transaction by the maturity date. Options can be traded on exchanges or over-the-counter. Underlying assets include stocks, stock indices, bonds, currencies, commodities, interest rates, and the weather. Options

can be distinguished by the right to buy and sell the underlying, as well as the specific time.

Definition 2.1.1 (Option)
An option gives the holder the right to buy (call option) or sell (put option) the underlying asset from the writer of the option at the strike price up to maturity (American option) or at the maturity date (European option).

Another option style is Bermudan, which allows one to exercise the option only at specific prescribed times before the maturity date. Put and call options are basic options and referred to as "plain Vanilla options". Other options are called "exotic options" and have more complicated payoff functions. Examples for path-dependent exotic options include barrier options (payoff depends on whether or not the underlying price has touched a prescribed bound), Asian options (payoff is determined by an average underlying price) and look-back options (strike price is the lowest or highest underlying price over some period). For more details on options we refer to [19]. We will focus on European call options in this thesis. Often the value of the American counterpart can be derived from the European option price.

Let us now consider the payoff function of a European call option with strike price K and maturity T. Let $S = S(t)$ be the value of the underlying asset at time t. If at maturity the price of the underlying exceeds the strike price, i.e. $S > K$, a rational option holder will exercise the option and receive $S - K$. If $S \leq K$ holds, it is cheaper to buy the underlying on the market. Therefore the option will not be exercised and the payoff is zero. Let $C_t = C(t, S)$ be the value of the option at time t. The payoff function at maturity is

$$C_T = h(S_T) := \max\{S_T - K, 0\} =: (S_T - K)^+. \tag{2.1}$$

Due to the typical form, which is illustrated in figure 2.1, h is also called hockey stick function.

Since the payoff at maturity is non-negative, it is clear that the option must cost a price at acquiring. Option pricing models try to estimate this fair value. To do so they make assumptions about the behavior of the underlying.

2.2 Option Pricing Models

In this section we model the uncertain underlying price. We start with the market assumptions needed to derive the Black-Scholes model. Then the concept of risk-

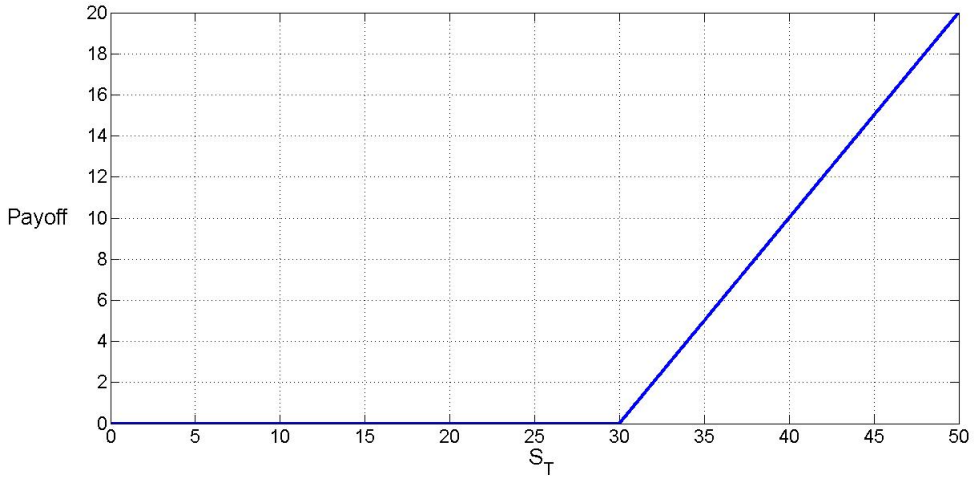


Figure 2.1: Payoff of a European call option with strike price K=30

neutral valuation is applied. We discuss the models' strengths and weaknesses. Afterwards, we analyze Merton's jump diffusion model, which is a modification of the Black-Scholes model: what could be improved and which weaknesses do remain.

2.2.1 Black-Scholes model

The first breakthrough in option pricing was the model derived by Fisher Black and Myron Scholes in 1973 (cf. [3]). In the model the underlying stock price is assumed to follow an exponential Brownian motion. They derived a closed form solution for the price of European call and put options. This led to a boom in options trading. The Black-Scholes model is one of the fundamental concepts of modern financial theory and is the origin of many options pricing models that are used today. Due to its simplicity, the model is still used today. Although, well-known inconsistencies such as the "volatility skew" exist.

The Black-Scholes model is derived under the assumption of an "ideal" market.

Assumption 2.2.1 (Market assumptions)
In the Black-Scholes model the market is assumed to incorporate the following characteristics:

a) The short-term interest rate is constant over time.

b) The log return of the stock price follows a Gaussian random walk in continuous time with a variance rate proportional to the square root of the stock price. The variance rate of the return on the stock is constant.

c) The stock pays no dividends.

d) There are no transaction costs.

e) The market is frictionless.

f) There are no penalties for short selling.

g) Options are of European option style.

h) Arbitrage is not possible.

Without further notice we shall assume throughout the whole thesis that Assumption 2.2.1 holds.

We will now define a standard Brownian motion, which is the driver of the underlying asset price process in the Black-Scholes model.

Definition 2.2.2 (Standard Brownian motion)
A stochastic process $W = \{W_t;\, t \geq 0\}$ *adapted to a filtration* $\mathcal{F} = \{\mathcal{F}_t;\, t \geq 0\}$ *is called standard Brownian motion on a probability space* $(\Omega, \mathcal{F}, \mathcal{P})$ *if*

(a) $W_0 = 0$ $\mathcal{P}$*-almost sure,*

(b) W *has independent increments, i.e.* $W_t - W_s$ *and* $\mathcal{F}_s$ *are independent for all* $0 \leq s < t$,

(c) W *has stationary and normally distributed increments, i.e.* $W_t - W_s \sim N(0, t-s)$ *for all* $0 \leq s < t$,

(d) the mapping $t \to W_t$ *is continuous* $\mathcal{P}$*-almost sure.*

Throughout this thesis $\mathcal{P}$ shall denote the physical probability measure.

Let W_t be a standard Brownian motion, μ the expected rate of return on the stock per year and σ the volatility of the rate of return per year. The following SDE describes the dynamics of the stock price in the Black-Scholes model

$$\frac{dS_t}{S_t} = \mu\, dt + \sigma\, dW_t. \tag{2.2}$$

Note that the return of the stock is modeled, not the stock price. The uncertain stock return consists of two parts: the deterministic amount of $\mu\, dt$ and the random amount of $\sigma\, dW_t$.

The market offers a riskless asset, called bond B_t, which earns the risk-free rate r

$$dB_t = r\, B_t\, dt.$$

The solution to this ordinary differential equation is $B_t = B_0\, e^{r\, t}$. In the market unlimited riskless investment is possible with continuously compounded interest return r.

We will now define the risk-neutral measure or also called equivalent martingale measure (EMM), which we need for risk-neutral valuation.

Definition 2.2.3 (Equivalent Martingale Measure)
Let $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{0\leq t\leq T}, \mathcal{P})$ be a filtered probability space. A probability measure $\mathcal{Q}$ on $(\Omega, \mathcal{F}_T)$ is called a risk-neutral measure or martingale measure if the discounted asset price process X is a $\mathcal{Q}$-martingale, i.e.

$$E_{\mathcal{Q}}[X_t] < \infty \quad \textit{and} \quad X_s = E_{\mathcal{Q}}[X_t|F_s] \quad \textit{for all} \quad 0 \leq s \leq t \leq T.$$

If in addition

$$\mathcal{Q} \sim \mathcal{P} \textit{ on } \mathcal{F}_T, \textit{i.e.} \quad \mathcal{Q}(A) = 0 \Leftrightarrow \mathcal{P}(A) = 0 \quad \textit{for all} \quad A \in \mathcal{F}_T,$$

holds, $\mathcal{Q}$ is called an equivalent martingale measure.

The next theorem, the Fundamental Theorem of Asset pricing (FTAP), states the condition under which an EMM exists.

Theorem 2.2.4 (Fundamental Theorem of Asset Pricing)
Let $(\Omega, \mathcal{F}, \mathcal{P})$ be a given probability space. The market model is arbitrage-free if and only if the set of all equivalent martingale measures to $\mathcal{P}$ is non-empty.

Proof. See [14]. □

Under the no-arbitrage Assumption 2.2.1 h) Theorem 2.2.4 guarantees that an EMM $\mathcal{Q}$ exists.

The following risk-neutral pricing formula gives us an formula on how to calculate the price of a European call option. Note that in the Black-Scholes framework the market is complete, which means every contingent claim is attainable.

Theorem 2.2.5 (Risk-Neutral Pricing Formula)
In an arbitrage-free complete market arbitrage prices of contingent claims are their

discounted expected values under the risk-neutral (equivalent martingale) measure $\mathcal{Q}$.

Proof. See [14]. □

Under the EMM $\mathcal{Q}$ the discounted asset price $\tilde{S}_t = S_t/B_t = e^{-rt}S_t$ is a martingale. We calculate

$$\begin{aligned} d\tilde{S}_t &= e^{-rt}dS_t - rS_t e^{-rt}dt \\ &= \sigma\tilde{S}_t\left(\frac{\mu - r}{\sigma}dt + dW_t\right). \end{aligned}$$

For the drift to be zero, we define

$$W_t^{\mathcal{Q}} = \theta t + W_t \quad \text{with} \quad \theta \coloneqq \frac{\mu - r}{\sigma}$$

where $W_t^{\mathcal{Q}}$ is a standard Brownian motion under $\mathcal{Q}$ and the EMM $\mathcal{Q}$ is characterized through the Radon-Nikodym derivative

$$\frac{d\mathcal{Q}}{d\mathcal{P}}|_t = exp\left(-\theta^2 t - \theta W_t\right)^{1}.$$

Applying Itô's lemma to (2.2), changing from the physical probability measure $\mathcal{P}$ to the EMM $\mathcal{Q}$ and integrating, we arrive at

$$S_t = S_0\, e^{(r-\frac{\sigma^2}{2})\,t+\sigma W_t^{\mathcal{Q}}}. \tag{2.3}$$

Stochastic processes which fulfill 2.3 are called 'exponential Brownian motion'.

The risk-neutral pricing formula states that the value of a European option is the discounted expected value of the payoff at maturity under the EMM

$$C_t = e^{-r\,(T-t)}\,E_{\mathcal{Q}}[(S_T - K)^+]. \tag{2.4}$$

Equation (2.4) can be used to derive a closed-form solution for the price of a European call option.

Theorem 2.2.6 (Black-Scholes formula)
The price of a European call option with asset price S *at time* t*, maturity* T*, strike*

[1] cf. p.276 [14]

price K, interest rate r and volatility σ in a Black-Scholes model is given by

$$C^{BS}(t, S; T, K, r, \sigma) = S\,\Phi(d_1) - \frac{K}{e^{r(T-t)}}\,\Phi(d_2), \quad 0 \leq S,\, 0 \leq t < T, \tag{2.5}$$

where Φ is the cumulative probability distribution for the standardized normal distribution

$$\Phi(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} e^{-s^2/2} ds, \quad z \in \mathbb{R},$$

and

$$d_{1/2} = \frac{\ln(\frac{S}{K}) + (r \pm \frac{\sigma^2}{2})(T-t)}{\sigma\sqrt{T-t}}.$$

Proof. See [3]. □

The closed-form solution for European call and put option prices is one of the advantages of the Black-Scholes model. It is easy to calculate and a useful approximation to real option prices. But only European-style options can be priced with this formula. The model is very restrictive. Dividends are excluded in the model. Through the use of a standard Brownian motion, paths of the underlying are assumed to be continuous (cf. 2.2.2 d)). That means that discontinuities in the stock price are ruled out. But it is known that different financial assets exhibit jumps. Exchange rates for example frequently jump. This can be the response to the actions of central banks. Stock market crashes, like e.g. in 1987, provide also evidence that large market movements do happen. Empirical studies, e.g. [5], confirmed a jump component in the underlying S&P 500 index process.

Another weakness of the Black-Scholes model is the constancy of parameters. In practice, interest rates vary not only by tenor, but also over time. The volatility is the only parameter in the Black-Scholes formula, which is not given or directly observable. Vanilla options are often quoted in the so-called implied volatility, i.e. the volatility with which the Black-Scholes model price equals the market price. Since the Black-Scholes model price of a European call option is monotonic increasing in σ, the implied volatility is unique.

Remark 2.2.7 (Implied Volatility)
Let $C^*(t, S_t; T, K) \in]\,(S_t - K\,e^{-r\,(T-t)})^+, S_t\,[$ be the market price of a European call option with (asset price S_t at time t), maturity T, strike price K, interest rate r. Then there exists a unique volatility $\sigma_{impl} > 0$ with

$$C^*(t, S_t; T, K) = C^{BS}(t, S_t; T, K, r, \sigma_{impl})^2. \tag{2.6}$$

[2]Due to the lack of a closed-form solution, the implied volatility has to be calculated via numerical methods.

Proof. See [8]. □

According to the Black-Scholes model the volatility of options on the same underlying with different maturities and strike prices is the same. Studies have shown that this is not the case. Implied volatilities strongly depend on strike price and maturity. When we plot the implied volatility against the strike price two patterns are typical depending on the market. Foreign currency options for example exhibit the 'volatility smile'. In-the-money or out-of-the money options have a higher implied volatility than at-the-money options[3]. This smile is more pronounced for short-term maturity options and flattens out as maturity increases. For stock and stock index options on the other hand exists a negative relationship between implied volatility and strike price. With constant maturity the implied volatility decreases as strike price increases. That means that out-of-the money put options trade at higher implied volatilities than out-of-the money call options. This 'volatility skew' became apparent after the crash of 1987 and ever since characterizes these markets. The skew is often attributed to the fear of large downward market movements. With increasing maturity the steepness of the skew normally decreases. Volatility 'term structure' describes the dependence of implied volatilities on maturity: implied volatility increases with maturity.

The volatility smile sometimes is explained by the 'fat tail phenomenon'. Extreme tails of assets' return distributions contain more relative frequency than would be expected under the Gaussian hypothesis. Furthermore distributions exhibit small skewness and higher kurtosis than the normal distribution. For more details we refer to the study of [13].

To capture this feature of options, volatility skews, extensions to the Black-Scholes model were made. They can be grouped into three approaches.

One approach consists of the 'stochastic volatility models'. Here the variance of the stochastic process of the underlying is a random variable itself. Famous models are the Heston model (cf. [16]) and Hull-White model (cf. [18]). From a numerical point of view these models are hard to handle even though for the Heston model a closed-form solution exists.

Another approach consists of the 'local volatility models' developed by Dupire [11] and Derman and Kani [10]. Here the volatility is a deterministic function of time and stock price. Dupire showed the existence of a forward equation where the call option price is a function of maturity and strike price. This allows to deduce

[3] A call option is 'out-of-the money' when $S_t < K$ and 'in-the money' when $S_t > K$ (vice versa for put options). Options are 'at-the money' when $S_t \approx K$.

the local volatility from known option prices and therefore precise fitting to quoted market prices. Another advantage is the models' simplicity and the fact that they are easy to handle from a numerical point of view.

Both mentioned models are able to produce a multitude of volatility skews, but only by incorporating unrealistic mechanisms.

The third approach consists of the is the so-called 'jump-diffusion models'. Jumps are added to the diffusion process in the Black-Scholes model. These models provide probability distributions that typically fit market data better (than those of the simple diffusion process) at the cost of an incomplete market model. With a jump-diffusion model various volatility skews can be generated. We analyze here Merton's Jump Diffusion model.

2.2.2 Merton's Jump-Diffusion model

Merton [24] proposed in 1976 a model to describe the asset price as a Lévy process of jump-diffusion type. Lévy processes are a more general concept of Brownian motion.

Definition 2.2.8 (Lévy process)
A cadlag[4] *stochastic process* $X = (X_t)_{t\geq 0}$ *adapted to a filtration* $\mathcal{F} = \{\mathcal{F}_t; t \geq 0\}$ *on a probability space* $(\Omega, \mathcal{F}, \mathcal{P})$ *with values in* $\mathbb{R}^d$ *is called a Lévy process if*

(a) $X_0 = 0$ $\mathcal{P}$*-almost sure,*

(b) *X has independent increments, i.e.* $X_t - X_s$ *and* $\mathcal{F}_s$ *are independent for* $0 \leq s < t$,

(c) *X has stationary increments, i.e. the law/distribution of* $X_{t+h} - X_t$ *does not depend on t,*

(d) *X is stochastic continuous, i.e.* $\forall\, \epsilon > 0 \quad lim_{h\to 0}\, \mathcal{P}(|X_{t+h} - X_t| \geq \epsilon) = 0.$

Note that the distribution of the increments is not specified, i.e. increments do not need to be normally distributed. Another difference to a Brownian motion is that the sample paths are not continuous. Jumps are permitted, but only at random times.

The compound Poisson process belongs to the class of Lévy processes[5].

[4] right-continuous and with left limits
[5] Proof: cf. Proposition 3.3 [8]

Definition 2.2.9 (Compound Poisson process)
A compound Poisson process with intensity $\lambda > 0$ and jump size distribution f is a stochastic process $X_t = \sum_{i=0}^{N_t} Y_i$, where the jump sizes Y_i are independent and identically distributed (i.i.d) with distribution f and (N_t) a Poisson process with intensity λ, independent from $(Y_i)_{i\geq 1}$.

Remark 2.2.10 (Counting process)
N_t is a counting process. It counts the number of times that jumps occurred up to time t and is finite almost sure. For every $t > 0$ N_t follows a Poisson distribution with intensity λt. The jump sizes are determined by the distribution of Y_i.

In Merton's jump diffusion model there is a positive chance of non-local changes in the asset price. Due to announcements or the arrival of new information jumps in the asset price can occur. The following dynamics model the asset price $S(t)$ under the physical probability measure $\mathcal{P}$

$$\frac{dS}{S_{t_-}} = \mu\, dt + \sigma\, dW_t + d(\sum_{i=1}^{N_t}(e^{Y_i} - 1)), \tag{2.7}$$

where W_t is a standard Brownian motion, N_t a Poisson process with intensity λ, μ the expected rate of return on the stock, $\sigma > 0$ the volatility of the expected rate of return and $S_{t_-} = \lim_{u \nearrow t} S_u$. The compound Poisson process $\sum_{i=1}^{N_t}(e^{Y_i} - 1)$ adds jumps to the process in the Black-Scholes model. It is assumed that the compound Poisson process is independent of the Brownian motion. The stock price evolves like a exponential Brownian motion between jumps and at each jump the stock price is multiplied by e^{Y_i}. This is $e^{Y_i} - 1$ is the percentage change in the stock price if a jump occurs. Let τ^i be a point in time, where a jump occurs, then

$$S_{\tau^i} = S_{\tau^i_-}\, e^{Y_i}.$$

To define the model completely, the distribution of jump sizes must be specified. Merton assumes that the jump sizes e^{Y_i} are log-normally distributed, i.e.
$Y_i \sim N(\mu_j, \sigma_j^2)$ i.i.d. with $\mu_j \in \mathbb{R}$, $\sigma_j > 0$ and density function

$$f(y) = \frac{1}{\sqrt{2\pi}\sigma_j} \exp\left\{-\frac{(y-\mu_j)^2}{2\sigma_j^2}\right\}. \tag{2.8}$$

Now we will establish the risk-neutral dynamics in Merton's jump diffusion model. FTAP 2.2.4 guarantees the existence of an EMM, but in an incomplete market the EMM is not unique. The next lemma will be of help in establishing an EMM and

the distribution of the jump sizes under that EMM.

Theorem 2.2.11
Let $\mathcal{P}$ and $\mathcal{Q}$ be equivalent measures. Consider a probability measure space $(\Omega, F, (F_t), P)$ such that F_t is the natural filtration generated by a standard Brownian motion W_t and a compound Poisson process $\sum_{i=0}^{N_t} J_i$ with intensity $\lambda > 0$ and a Radon-Nikodym derivative given by

$$\frac{d\mathcal{Q}}{d\mathcal{P}}|_t = exp\left(-\theta^2 t - \theta W_t + \sum_{i=0}^{N_t}(\gamma J_i + \nu) - \lambda k' t\right), \tag{2.9}$$

where $\gamma \in \mathbb{R}, \nu \in \mathbb{R}$ and $k' = e^{\nu} E_{\mathcal{P}}[e^{\gamma J}] - 1$. Then the standard Brownian motion W_t has drift $-\theta$ under the measure $\mathcal{Q}$ and the compound Poisson process $\sum_{i=0}^{N_t} J_i$ under the measure $\mathcal{Q}$ has a new intensity rate $\tilde{\lambda} = \lambda(1 + k')$ and a new distribution for the jump-sizes, where the moment generating function is given by

$$M_{\mathcal{Q}}(x) = \frac{M_{\mathcal{P}}(x + \gamma)}{M_{\mathcal{P}}(\gamma)} \tag{2.10}$$

Proof. cf. Lemma 1 [6]. □

We know that under an EMM the discounted price process is a martingale. Hence an EMM can be determined by the choice of the parameters γ, ν in the Radon-Nikodym derivative (2.9), with θ determined by the martingale condition after γ, ν have been set. We choose $\gamma = 0, \nu = 0$. Then $k' = 0$ and according to Theorem 2.2.11 the intensity of the compound Poisson process under $\mathcal{Q}$ is $\tilde{\lambda} = \lambda$. We calculate

$$\begin{aligned} d\tilde{S}_t &= e^{-rt} dS_t - r S_t e^{-rt} dt \\ &= \sigma \tilde{S}_t \left(\frac{\mu - r}{\sigma} dt + dW_t + \frac{1}{\sigma} d(\sum_{i=1}^{N_t} (e^{Y_i} - 1) \right) \end{aligned}$$

and find θ

$$\theta := \frac{\mu - r + \lambda\xi}{\sigma}, \quad \text{where} \quad \xi = E_{\mathcal{Q}}[e^Y - 1].$$

Then $W_t^{\mathcal{Q}} = \theta t + W_t$ is a standard Brownian motion under $\mathcal{Q}$ and the EMM $\mathcal{Q}$ is characterized through the Radon-Nikodym derivative 2.9. The distribution of the jump sizes is characterized by 2.10. Through our choice of γ, ν the distribution of Y under $\mathcal{Q}$ corresponds to the distribution under $\mathcal{P}$, i.e. $Y \sim N(\mu_j, \sigma^2)$. Therefore

the average jump size is

$$\xi = E_{\mathcal{Q}}[e^Y - 1] = \int_{\mathbb{R}} (e^y - 1)\, f(y)\, dy = e^{\mu_j + \frac{\sigma^2}{2}} - 1.$$

Under the EMM $\mathcal{Q}$ the risk-neutral stock price process is given by

$$\frac{dS}{S_{t_-}} = (r - \lambda\xi)\, dt + \sigma\, d\tilde{W}_t + d(\sum_{i=1}^{N_t}(e^{Y_i} - 1)). \tag{2.11}$$

Merton's jump diffusion model is an exponential Lévy model with solution

$$S_t = S_0\, e^{X_t},$$

where X_t is the Lévy process

$$X_t = (r - \frac{\sigma^2}{2} - \lambda\xi)\, t + \sigma\, \tilde{W}_t + \sum_{i=1}^{N_t} Y_i.$$

Merton uses a PDE approach to develop a semi-analytical solution expressed in terms of a series of weighted Black-Scholes model prices conditional on the number of jumps.

Theorem 2.2.12 (Merton semi-analytical solution)
Given the Merton jump-diffusion model specified by (2.7) *and* (2.8) *with volatility* $\sigma > 0$*, jump intensity* λ*, mean jump size* μ_j *and volatility of the jump size* σ_j*. Then today's price of a European call option with maturity* T*, strike price* K*, interest rate* r *and current stock price* S_0*, is given by*

$$C^M(0, S_0) = \sum_{n=0}^{\infty} e^{-\bar{\lambda}T} \frac{(\bar{\lambda}T)^n}{n!}\, C^{BS}(t, S_0; T, K, \bar{r_n}, \bar{\sigma_n}^2) \tag{2.12}$$

where $\bar{\lambda} = \lambda(\xi + 1)$, $\bar{r_n} = r - \lambda\xi + \frac{n\ln(\xi+1)}{T}$, $\bar{\sigma_n}^2 = \sigma^2 + \frac{n\sigma_j^2}{T}, \xi = e^{\mu_j + \frac{\sigma_j^2}{2}} - 1$ *and* $C^{BS}(0, S_0; T, K, \bar{r_n}, \bar{\sigma_n}^2)$ *is today's Black-Scholes price with interest rate* $\bar{r_n}$ *and volatility* $\bar{\sigma_n}^2$ *(see 2.5).*

Proof. Cf. [24]. □

This semi-analytical solution is an advantage of Merton's jump diffusion model, since jump-diffusion models are known to be numerically challenging. This model is consistent with the observed skewness and the leptokurtosis in the distribution of stock prices. By setting the mean of the jump process to be negative, steep short-term volatility skews, which are typical in practice, can be produced quite easily.

One weakness of the model is that while fitting the few parameters to market prices often large errors occur.

To overcome remaining weaknesses combinations of models have been proposed. Andersen [1] argues that "the most reasonable model of stock prices would likely include both stochastic volatility and jump diffusion [...]. From the perspective of the financial engineer, such a model would, however, not necessarily be very attractive as it would be difficult to handle numerically and slow to calibrate accurately to quoted prices"[6]. We will follow his approach of a jump-diffusion model with local volatility. Through the jump component we are able to generate volatility skews and with the local volatility we can fit quoted market prices accurately. In fact with this model we overcome the unrealistic appearance of the local volatility surface in the case of a pure diffusion model when we try to incorporate steep short-term skews and obtain a reasonable volatility function.

The stock price is modeled by the following SDE

$$\frac{dS}{S_{t_-}} = (r(t) - \lambda\xi)\,dt + \sigma(t, S_{t_-})\,d\tilde{W}_t + d\left(\sum_{i=1}^{N_t}(e^{Y_i} - 1)\right), \tag{2.13}$$

where $\sigma > 0$ is a bounded time- and state-dependent deterministic local volatility function. The constant risk-free rate r is generalized to a time-dependent interest rate $r(t)$, which is given by the market.

So far we established analytical and semi-analytical formulas in different models for the price of a European call option. We have to remark that analytical solutions can only be established for European-style options. For pricing other options e.g. exotic ones, we have to use numerical methods. Main approaches here are Monte-Carlo methods and the numerical solution of the PIDE's. We will pursue the latter.

2.3 Option Pricing using Partial Differential Equations

Due to the local volatility function no analytical solution is available. For a numerical solution, we transform SDE (2.13) into a PIDE. In the end we will work with a log-moneyness version of this PIDE. We first we obtain the following backward PIDE.

[6]Quote from Andersen [1], p. 233

Theorem 2.3.1 (Backward PIDE)
Let a jump-diffusion model be given where the stock S evolves according to (2.13). *Then the price $C(t,S)$ of a European call option on the underlying S with maturity T and strike price K satisfies the following PIDE*

$$\begin{aligned} &C_t(t,S) + (r(t) - \lambda\xi)\, S\, C_S(t,S) + \frac{\sigma^2(t,S)}{2}\, S^2\, C_{SS}(t,S) \\ &- (r(t) + \lambda)\, C(t,S) + \lambda \int_{\mathbb{R}} C(t, yS) f(y)\, dy = 0 \\ &(t,S) \in [0,T) \times (0,\infty) \\ &\textit{with end condition} \quad C(T,S) = (S-K)^+ \quad S \in (0,\infty), \end{aligned} \tag{2.14}$$

where $\lambda > 0$ is the constant jump intensity and f the density function of the jump size distribution.

Proof. See [1]. □

A partial differential equation with an integral term needs to be solved in order to obtain the call price. Normally the call price today, i.e. $C(0,S_0)$, is desired. When we set r constant and $\lambda = 0$, which means no jumps in the underlying, we obtain the well-known Black-Scholes equation, a linear parabolic equation of second order.

$$\begin{aligned} &C_t(t,S) + \frac{\sigma^2}{2}\, S^2\, C_{SS}(t,S) + r\, S\, C_S(t,S) - r\, C(t,S) = 0, \\ &(t,S) \in [0,T) \times (0,\infty) \\ &\text{with end condition} \quad C(T,S) = (S-K)^+ \quad S \in (0,\infty) \end{aligned} \tag{2.15}$$

To solve (2.15) we need boundary conditions. Economic arguments lead to $\lim_{S\to 0} C(t,S) = 0$ for all $t \in [0,T]$ and $\lim_{S\to\infty} \frac{C(t,S)}{S} = 1$ for all $t \in [0,T]$. With these conditions we can solve (2.15).

Note that in the Black-Scholes model the risk free rate is constant. Even when it is generalized to $r(t)$ as in (2.14), it is given at the market. The strike price K and maturity T are fixed in the contract. Hence in general unknown variables in (2.14) are the volatility $\sigma(t,S)$, the jump intensity λ and the jump density f (we specified f). These can be derived from market prices. Since we want to use the PIDE to calculate new option prices, the accurate calibration of model parameters is crucial. The parameters are calibrated such that the error between market prices and prices

resulting from the model is minimized. Formulating this as an optimization problem, the PIDE has to be solved for each pair of maturity and strike (T_i, K_i) making one function evaluation itself numerically expensive. To avoid this we will show the existence of a forward equation with T and K as variables and t and S occurring in the initial condition. Then the PIDE has to be solved only once for a function evaluation in a least-square formulation.

Theorem 2.3.2 (Forward PIDE)
Let $\tilde{D}(T,K)$ be the solution of the PIDE

$$\begin{aligned}
&\tilde{D}_T(T,K) - \frac{\sigma^2(T,K)}{2} K^2 \tilde{D}_{KK}(T,K) + (r(T) - \lambda\xi)\, K \tilde{D}_K(T,K) \\
&+ \lambda(\xi+1)\tilde{D}(T,K) - \lambda \int_{\mathbb{R}} \tilde{D}(T, K/e^y)\, e^y f(y)\, dy = 0 \\
&(T,K) \in [t_0, T_{\max}) \times (0,\infty) \\
&\text{with initial condition} \quad \tilde{D}(t_0,K) = (S_0 - K)^+ \quad K \in (0,\infty).
\end{aligned} \tag{2.16}$$

Then the following holds:

$$\tilde{D}(T,K) = C(t_0, S_0),$$

where $C(t,S)$ is the solution of (2.14) *with maturity T and strike price K.*

Proof. See [1]. □

In literature the terms moneyness, i.e. K/S_0, and log-moneyness, i.e. $\log(K/S_0)$, are frequently used. Because of numerical reasons we eliminate K from the PIDE via a variable transformation.

Theorem 2.3.3
Let $D(T,x)$ be the solution of

$$\begin{aligned}
&D_T(T,x) - \frac{\bar{\sigma}^2(T,x)}{2} D_{xx}(T,x) + \left(r(T) - \lambda\xi + \frac{\bar{\sigma}^2(T,x)}{2} \right) D_x(T,x) \\
&+ \lambda(\xi+1) D(T,x) - \lambda \int_{\mathbb{R}} D(T, x-y)\, e^y f(y)\, dy = 0 \\
&(T,x) \in [0, T_{\max}) \times (-\infty,\infty) \\
&D(0,x) = (1-e^x)^+ =: D_0(x) \quad x \in (-\infty,\infty).
\end{aligned} \tag{2.17}$$

Then

$$\tilde{D}(T,K) = S_0 D\left(T, \ln\left(\frac{K}{S_0}\right)\right)$$

solves (2.16) *with $\sigma(T,K) = \bar{\sigma}(T, \ln\left(\frac{K}{S_0}\right))$.*

Proof. Consider the variable transformation $x = \ln\left(\frac{K}{S_0}\right)$. Then the derivatives of $\tilde{D}$ are given by

$$\tilde{D}_T = S_0 D_T, \quad \tilde{D}_K = \frac{S_0}{K} D_x, \quad \tilde{D}_{KK} = \frac{S_0}{K^2} D_{xx} - \frac{S_0}{K^2} D_x.$$

A scaling of the PIDE as well as the initial condition by S_0^{-1} yields the desired result. □

In the next chapter we will solve this PIDE numerically. First we give a brief summary of model calibration in the next section.

2.4 Calibration of Model Parameters

To estimate the value of exotic options with a given model, first the model parameters have to be calibrated carefully. This is usually done via a non-linear least-squares-formulation where the error between quoted market prices D_i^M for pairs of maturity and strike (T_i, K_i), $i = 1, \ldots, M$, and model prices $D(T_i, K_i)$ is minimized:

$$(\sigma^*, \lambda^*, f^*) = argmin \sum_{i=1}^{M} |D_i^M - D(T_i, K_i)|^2 \quad \text{s.t. } D \text{ solves (2.16).} \tag{2.18}$$

In a jump-diffusion model the jump density function f, the jump intensity λ and the local volatility function $\sigma(\cdot, \cdot)$ have to be calibrated. In our case we specified the jump density function. We assume that we know the mean jump size and volatility of the jump sizes as well as the jump intensity. Then the local volatility function is implied. The efficient numerical solution of PIDE (2.16) is the basis for solving the constrained optimization problem. Concrete methods for an efficient and accurate model calibration are beyond the scope of this thesis.

Chapter 3

Numerical Solution

In this chapter the details of the numerical solution of PIDE (2.17) are carried out. In Section 3.1 we first derive a variational formulation and prove that under certain assumptions a unique solution exists. When we restrict the infinite space domain on a bounded interval, we have to take into account that due to the non-local integral term in PIDE (2.17) the boundary conditions have to be expanded (cf. Section 3.1). In Section 3.2 we discretize the problem. We use a Galerkin method with piecewise linear functions on the spatial variable. The non-local integral term leads to dense matrices which have to be taken into account. The time domain is discretized using the implicit Crank-Nicolson scheme. We use a damping procedure to smooth out the irregular initial condition. The arising dense linear systems of equations are solved with a preconditioned GMRES method. In the last Section 3.3 numerical results are presented.

3.1 Weak Formulation

In order to be able to apply the finite element method, we first need to derive a variational formulation of PIDE (2.17). This will be the topic of this section. We furthermore prove the existence of a unique solution of the variational problem.

First we note that the initial condition in (2.17) is not L^2-integrable, since $\lim_{x\to-\infty} D_0(x) = 1$. Analogous to [23] we will therefore work with weighted function spaces.

Definition 3.1.1 (Weighted Sobolev spaces)
We define the weighted Sobolev spaces

$$H^1_{-\mu}(\mathbb{R}) := \{v \in L^1_{loc}(\mathbb{R}) : v(x)e^{-\mu|x|}, v'e^{-\mu|x|} \in L^2(\mathbb{R})\}$$

with inner product

$$\langle v, w\rangle_{H^1_{-\mu}(\mathbb{R})} \coloneqq \langle v, w\rangle_{L^2_{-\mu}} + \langle v^{'}, w^{'}\rangle_{L^2_{-\mu}}$$

and

$$L^2_{-\mu}(\mathbb{R}) \coloneqq \{v \in L^1_{loc}(\mathbb{R}) : v(x)e^{-\mu|x|} \in L^2(\mathbb{R})\}$$

with inner product

$$\langle v, w\rangle_{L^2_{-\mu}} \coloneqq \int_{\mathbb{R}} v(x)w(x)e^{-2\mu|x|}dx.$$

Remark 3.1.2
The spaces $L^2_{-\mu}(\mathbb{R})$ and $H^1_{-\mu}(\mathbb{R})$ are Hilbert spaces with their induced norms.

It is easy to show that $D_0(\cdot) \in H^1_{-\mu}(\mathbb{R})$ for all $\mu > 0$. Let w be an arbitrary test function in $C^\infty_0(\mathbb{R})$. When we multiply (2.17) with $w(x)e^{-2\mu x}$, where $\mu > 0$ is fixed, and integrate over $\mathbb{R}$, we get

$$\begin{aligned}
&\int_{\mathbb{R}} D_T(T,x)w(x)e^{-2\mu x}dx - \int_{\mathbb{R}} \frac{\bar\sigma^2(T,x)}{2} D_{xx}(T,x)w(x)e^{-2\mu x}dx \\
&+ \int_{\mathbb{R}} \left(r(T) - \lambda\xi + \frac{\bar\sigma^2(T,x)}{2}\right) D_x(T,x)w(x)e^{-2\mu x}dx \\
&+ \int_{\mathbb{R}} \lambda(\xi+1)\, D(T,x)w(x)e^{-2\mu x}\, dx \\
&- \lambda \int_{\mathbb{R}}\int_{\mathbb{R}} D(T,x-y)\, w(x)e^{-2\mu x}\, e^y\, f(y)\, dy\, dx = 0 \\
&\text{with initial condition } \int_{\mathbb{R}} D(0,x)w(x)e^{-2\mu x}dx = \int_{\mathbb{R}} D_0(x)w(x)e^{-2\mu x}dx.
\end{aligned}$$

Integrating the second term by parts we arrive at the following equation

$$\int_{\mathbb{R}} D_T(T,x)w(x)e^{-2\mu x}dx + a^{-\mu}(T; D(T,\cdot), w(\cdot)) = 0,$$

where for each constant $\mu > 0$ and $T > 0$ the bilinear form
$a^{-\mu}(T;\cdot,\cdot) : H^1_{-\mu}(\mathbb{R}) \times H^1_{-\mu}(\mathbb{R}) \to \mathbb{R}$ is defined via

$$\begin{aligned}
a^{-\mu}(T; v, w) &\coloneqq \int_{\mathbb{R}} \frac{\bar\sigma^2(T,x)}{2} v^{'}(x)w^{'}(x)e^{-2\mu x}dx \\
&+ \int_{\mathbb{R}} \left(r(T) - \lambda\xi + \frac{\bar\sigma^2(T,x)}{2} + \frac{(\bar\sigma^2(T,x))_x}{2} - \bar\sigma^2(T,x)\,\mu\, sgn(x)\right) v^{'}(x)w(x)e^{-2\mu x}dx \\
&+ \int_{\mathbb{R}} \lambda(\xi+1)\, v(x)w(x)e^{-2\mu x}\, dx - \lambda \int_{\mathbb{R}}\int_{\mathbb{R}} v(x-y)\, w(x)e^{-2\mu x}\, e^y\, f(y)\, dy\, dx.
\end{aligned} \tag{3.1}$$

Here λ and ξ are given constants and r and σ are assumed to be sufficiently regular.

Next we define a function space containing the solutions of the weak formulation of (2.17).

Definition 3.1.3
Let V be a Hilbert space, V^* its dual space and $a, b \in \mathbb{R}$. Then we define

$$W([a,b],V) \coloneqq \{u \in L^2((a,b),V) : u^{'} \, in \, L^2((a,b),V^*)\}.$$

We can now formulate the weak formulation of PIDE 2.17

Definition 3.1.4 (Weak formulation)
Find $D \in W([0,T_{\max}], H^1_{-\mu}(\mathbb{R}))$ such that for all $T \in (0,T_{\max}]$

$$\begin{aligned} &\frac{d}{dT}\langle D(T,\cdot), w\rangle_{L^2_{-\mu}(\mathbb{R})} + a^{-\mu}(T; D(T,\cdot), w) = 0 \quad \forall w \in H^1_{-\mu}(\mathbb{R}) \\ &\textit{with initial condition} \\ &\langle D(0,\cdot), w\rangle_{L^2_{-\mu}(\mathbb{R})} = \langle D_0(\cdot), w\rangle_{L^2_{-\mu}(\mathbb{R})} \quad \forall w \in H^1_{-\mu}(\mathbb{R}) \end{aligned} \tag{3.2}$$

holds.

We want to show that under certain assumptions a unique solution of the above weak formulation exists. For this we will make use of the following theorem.

Theorem 3.1.5 (Unique solution)
Let (V,H,V^*) be a Gelfand triple and $D_0 \in H, F \in L^2([0,T_{\max}],V^*)$. If the bilinear form $a(T;\cdot,\cdot) : V \times V \to \mathbb{R}$ is continuous and V-elliptic for all $T \in [0,T_{\max}]$, then for all $T \in (0,T_{\max}]$

$$\begin{aligned} &\frac{d}{dT}\langle D(T,\cdot), w(\cdot)\rangle_H + a^{-\mu}(T; D(T,\cdot), w(\cdot)) = \langle F(T; w(\cdot)), w(\cdot)\rangle_H \quad \forall w \in V \\ &\textit{with initial condition} \\ &\langle D(0,\cdot), w(\cdot)\rangle_H = \langle D_0(\cdot), w(\cdot)\rangle_{L^2_{-\mu}(\mathbb{R})} \quad \forall w \in V \end{aligned} \tag{3.3}$$

has a unique solution $D \in W([0,T_{\max}],V)$.

Proof. See [9]. □

We will now state the assumptions under which the weak formulation 3.2 has a unique solution.

Assumption 3.1.6
We assume that for each $T \in [0,T_{\max}]$ $\sigma(T,\cdot)$ is continuously differentiable on $\mathbb{R}$.

Furthermore constants $r_{\max}, \sigma_{\min}, \sigma_{\max}, \sigma_{der}$ *exist such that*

$$\begin{aligned} 0 \leq r(T) \leq r_{\max} \, \forall T \in [0, T_{\max}] \\ 0 < \sigma_{\min} \leq \sigma(T, x) \leq \sigma_{\max} \, \forall (T, x) \in [0, T_{\max}] \times \mathbb{R} \\ |\sigma_x(T, x)| \leq \sigma_{der} \, \forall (T, x) \in [0, T_{\max}] \times \mathbb{R}. \end{aligned}$$

Lemma 3.1.7
The normal density function f *satisfies for* $\mu > 0$

$$\int_{\mathbb{R}} e^{y+\mu|y|} \, y \, f(y) \, dy < \infty. \tag{3.4}$$

Proof. It is clear because f decays exponentially with e^{-y^2}. □

Remark 3.1.8
It is important to note that the normal density function specified by Merton for the jump sizes satisfies 3.1.6 b).

With the above assumptions we can establish properties of the bilinear form $a^{-\mu}$, which ensure a unique solution of the weak formulation problem.

Theorem 3.1.9
If assumptions 3.1.6 hold, then there exist constants $c_{cont}, c_{ell} > 0, c \in \mathbb{R}$ *independent of* $T \in [0, T_{\max}]$ *such that for all* $T \in [0, T_{\max}]$ *the bilinear form* $a^{-\mu}$ *is continuous, i.e.*

$$|a^{-\mu}(T; v, w)| \leq c_{cont} \, ||v||_{H^1_{-\mu}(\mathbb{R})} ||w||_{H^1_{-\mu}(\mathbb{R})} \quad \forall v, w \in H^1_{-\mu}(\mathbb{R}),$$

and the Garding inequality, i.e.

$$a^{-\mu}(T; v, w) + c||v||^2_{L^2_{-\mu}(\mathbb{R})} \geq c_{ell} \, ||v||^2_{H^1_{-\mu}(\mathbb{R})} \quad \forall v, w \in H^1_{-\mu}(\mathbb{R}),$$

holds.

Proof. This result is derived in [34]. □

The boundedness and weak coercivity of the bilinear form $a^{-\mu}$ guarantees the existence and uniqueness of a solution of 3.2.

Theorem 3.1.10
The weak formulation (3.2) *possesses a unique solution* $D \in W([0, T_{\max}], H^1_{-\mu}(\mathbb{R}))$*, if Assumption 3.1.6 holds.*

Proof. Let us define a new bilinear form $\tilde{a}^{-\mu}$ on $L^2_{-\mu}(\mathbb{R})$ for all $T \in [0, T_{\max}]$ via

$$\tilde{a}^{-\mu}(T; v, w) = a^{-\mu}(T; v, w) + C \langle v, w \rangle_{L^2_{-\mu}(\mathbb{R})}.$$

It is clear that $\tilde{a}^{-\mu}$ is continuous and coercive on $H^1_{-\mu}(\mathbb{R})$. With the Gelfand triple $(H^1_{-\mu}(\mathbb{R}), L^2_{-\mu}(\mathbb{R}), H^1_{-\mu}(\mathbb{R})^*)$ 3.1.5 states, that there exists a unique solution $\tilde{D} \in W([0, T_{\max}], H^1_{-\mu}(\mathbb{R}))$ for all $T \in (0, T_{\max}]$ of

$$\frac{d}{dT}\langle \tilde{D}(T, \cdot), w(\cdot) \rangle_{L^2_{-\mu}(\mathbb{R})} + \tilde{a}^{-\mu}(T; \tilde{D}(T, \cdot), w(\cdot)) = 0 \quad \forall w \in H^1_{-\mu}(\mathbb{R})$$

with initial condition

$$\langle \tilde{D}(0, \cdot), w(\cdot) \rangle_{L^2_{-\mu}(\mathbb{R})} = \langle D_0(\cdot), w(\cdot) \rangle_{L^2_{-\mu}(\mathbb{R})} \quad \forall w \in H^1_{-\mu}(\mathbb{R}).$$

Then $D(T, x) = e^{CT}\tilde{D}(T, x)$ solves the variational problem (3.2). □

Localization

Since PIDE (2.17) is posed on the whole real line, the space domain is infinite. For a numerical solution we must truncate the PIDE to a bounded space domain $\Omega = (\underline{x}, \bar{x})$. This process is called 'localization'. To do so we have to analyze the behavior of the solution at the boundaries. It is easier if we consider $\tilde{D}(T, K)$ before the variable transformation when K takes extreme values close to zero or to infinity.

If the strike price K approaches infinity, it is almost sure that the underlying asset price will be lower than K. Then the option holder will not exercise the option and the payoff and therefore the value of the option are zero

$$\tilde{D}(T, K) \to 0 \quad \text{for} \quad K \to \infty.$$

On the other hand if K tends to zero, the underlying asset price will almost surely be higher than the strike price K and the option holder will almost surely exercise the option at maturity. Under the no-arbitrage assumption the value of a claim equals the value of a perfect hedge. The option writer will have to deliver the underlying at maturity a.s.. To hedge this risk he has to buy the underlying at $t = 0$ for the stock price S_0. At maturity $t = T$ when he delivers the option, he receives the strike price K from the option holder. Discounting K, this is worth $Ke^{-\int_0^T r(\tau)d\tau}$ at $t = 0$. The value of the hedge and therefore the call is

$$\tilde{D}(T, K) \to S_0 - Ke^{-\int_0^T r(\tau)d\tau} \quad \text{for} \quad K \to 0.$$

After the variable transformation $x = \ln(\frac{K}{S_0})$ and $\tilde{D}(T,K) = S_0 D(T,x)$ we arrive at the following boundary conditions

$$D(T,\underline{x}) \approx 1 - e^{\underline{x} - \int_0^T r(\tau)d\tau}, \quad D(T,\bar{x}) \approx 0. \tag{3.5}$$

The integral term in PIDE (2.17) is non-local. This means that in order to evaluate the PIDE at x we need to know D not only in a neighborhood of x but for all values to which the process can jump to from x: $\{x-y : y \in supp(f)\}$. Since the normal density function f specified in (2.8) has support $\mathbb{R}$ (as is the case in many models), values of D at all points of $\mathbb{R}$ are used to evaluate the PIDE at x. In the case of a PDE like the Black-Scholes-equation the boundary conditions (3.5) would be used as Dirichlet boundary conditions and error estimates could be derived using the maximum principle. In our case the boundary conditions must be extended to $\{x - y : x \in [\underline{x}, \bar{x}], y \in supp(f)\}$, i.e. $\mathbb{R}\backslash\Omega$.

Let $D^b(T,x)$ be a function in the solution space fulfilling the boundary conditions (3.5) on $(-\infty, \underline{x}]$ and $[\bar{x}, \infty)$. Then we can rewrite the variational formulation in the following way:

Lemma 3.1.11
Let $D^b \in W([0,T_{\max}], H^1_{-\mu}(\mathbb{R}))$ fulfill the boundary conditions (3.5) on $(-\infty, \underline{x}]$, $[\bar{x}, \infty)$ and $D \in W([0,T_{\max}], H^1_{-\mu}(\mathbb{R}))$ be a solution of (3.2). Then
$D^h \in W([0,T_{\max}], H^1_{-\mu}(\mathbb{R}))$ defined via $D^h := D - D^b$ solves

$$\begin{aligned}
&\frac{d}{dT}\langle D^h(T,\cdot), w(\cdot)\rangle_{L^2_{-\mu}(\mathbb{R})} + a^{-\mu}(T; D^h(T,\cdot), w(\cdot)) = F^{-\mu}(T; w(\cdot)) \quad \forall w \in H^1_{-\mu}(\mathbb{R}) \\
&\textit{with initial condition} \\
&\langle D^h(0,\cdot), w(\cdot)\rangle_{L^2_{-\mu}(\mathbb{R})} = \langle D^h_0(\cdot), w(\cdot)\rangle_{L^2_{-\mu}(\mathbb{R})} \quad \forall w \in H^1_{-\mu}(\mathbb{R})
\end{aligned} \tag{3.6}$$

where $D^h_0(\cdot) := D_0(\cdot) - D^b(0,\cdot) \in H^1_{-\mu}(\mathbb{R})$ and $\forall T \in (0, T_{\max}]$
$F^{-\mu}(T;\cdot) := -\frac{d}{dT}\langle D^b(T,x), \cdot\rangle_{L^2_{-\mu}(\mathbb{R})} - a^{-\mu}(T; D^b(T,x), \cdot) \in (H^1_{-\mu}(\mathbb{R}))^*$.

Proof. By construction of D^h. □

It is clear that $F^{-\mu} \in (H^1_{-\mu}(\mathbb{R}))^* \forall \mu > 0$. By construction of D^h the following holds

$$D^h(T,x) \to 0, \quad \text{for} \quad x \to \pm\infty. \tag{3.7}$$

In a localized problem, the solution $\bar{D}$ will be defined only on the bounded computational domain Ω. But then the non-local integral becomes meaningless, unless

we extend $\bar{D}$ beyond its domain of definition. The value of the non-local integral depends on the extension and differs with it. Considering (3.7) we will assume that $\bar{D} \equiv 0$ on $\mathbb{R}\backslash\Omega$. Then we can define a localized problem on $H_0^1(\Omega)$ in the following way.

Definition 3.1.12 (Localized variational problem)
Find $\bar{D} \in W([0, T_{\max}], H_0^1(\Omega))$ *such that for all* $T \in (0, T_{\max}]$

$$\begin{aligned} &\frac{d}{dT}\langle \bar{D}(T,\cdot), w(\cdot)\rangle_{L^2} + a(T; \bar{D}(T,\cdot), w(\cdot)) = F(T; w(\cdot)) \quad \forall w \in H_0^1(\Omega) \\ &\textit{with initial condition} \\ &\langle \bar{D}(0,\cdot), w(\cdot)\rangle_{L^2} = \langle D_0^h(\cdot), w(\cdot)\rangle_{L^2} \quad \forall w \in H_0^1(\Omega) \end{aligned} \tag{3.8}$$

holds. Here $a(T;\cdot,\cdot) : H_0^1(\Omega) \times H_0^1(\Omega) \to \mathbb{R}$ *is defined via* $a(T; v, w) \coloneqq a^0(T; \tilde{v}, \tilde{w})$, *where* $\tilde{v}, \tilde{w}$ *denote the extension of v,w by zero to all of* $\mathbb{R}$, *i.e.* $v|_{\mathbb{R}\backslash\Omega} = w|_{\mathbb{R}\backslash\Omega} = 0$. *Analogously* $F(T,\cdot) : H_0^1(\Omega) \to \mathbb{R}$ *is defined via* $F^0(T,\cdot)$.

Remark 3.1.13
The localized variational problem has the following properties:

a) $D_0^h|_\Omega \in H_0^1(\Omega)$ *by construction of* D^b.

b) The linear operator $L^{-\mu}$ *is well-defined for* $\mu = 0$ *under Assumption 3.1.6.*

c) We identify the Gelfand triple as $V = H_0^1(\Omega), H = L^2(\Omega), V^* = H^{-1}(\Omega)$. *Then the localized variational problem admits a unique solution.*

Remark 3.1.14 (Localization error)
The restriction of $\mathbb{R}$ *to* Ω *introduces a localization error* $e \coloneqq \tilde{\bar{D}} - D^h$). *Matache [23] considers a backward PIDE with constant volatility. Using a-priori estimates Matache [23] showed that the localization error decays exponentially with the size of the truncation domain.*

3.2 Discretization

In this section we will discretize (3.8). First we discretize $H_0^1(\Omega)$ using the finite element method in Section 3.2.1. The resulting ordinary differential equations are discretized with a Crank-Nicolson scheme in Section 3.2.2. To account for the dense matrices, an approach using preconditioned GMRES method is carried out in Section 3.2.3.

3.2.1 Spatial Discretization

To approximate the solution of the localized problem, we use a special Galerkin method, the finite elements method. We approximate $H_0^1(\Omega)$ by a finite-dimensional subspace V_G of continuous, piecewise linear functions and determine the coefficients of the corresponding approximate solution. The Galerkin problem is then defined as: find $D_G \in W([0, T_{\max}], V_G)$ such that for all $T \in (0, T_{\max}]$

$$\begin{aligned} &\frac{d}{dT}\langle D_G(T,\cdot), w(\cdot)\rangle_{L^2} + a(T; D_G(T,\cdot), w(\cdot)) = F(T; w(\cdot)) \quad \forall w \in V_G \\ &\text{with initial condition} \\ &\langle D_G(0,\cdot), w(\cdot)\rangle_{L^2} = \langle D_0^h(\cdot), w(\cdot)\rangle_{L^2} \quad \forall w \in V_G \end{aligned} \tag{3.9}$$

holds.

Let us consider a uniform mesh $\Omega_h \coloneqq \{i\Delta x : i = 1, \cdots n_x\}$ on Ω such that $\underline{x} =: x_0 < x_1 < \cdots < x_{n_x+1} \coloneqq \bar{x}$. Now we use Lagrange finite elements and define

$$V_G \coloneqq \{w \in C(\Omega) : w|_{(x_i, x_{i+1})} \in P^1((x_i, x_{i+1})) \quad \text{for} \quad i = 1, \cdots, n_x, w|_{\partial\Omega} = 0\}. \tag{3.10}$$

Here $P^k(\Omega)$ is the space of polynomials with degree less or equal to k on Ω. It is clear that $V_h \subset H_0^1(\Omega)$ since it consists of continuous piecewise C^1 functions. Then $\{\Phi_i\}_{i=1}^{n_x}$ with

$$\Phi_i(x) = \begin{cases} \frac{(x - x_{i-1})}{\Delta x} & x_{i-1} \le x < x_i \\ \frac{(x_{i+1} - x)}{\Delta x} & x_i < x \le x_{i+1} \\ 0 & \text{else} \end{cases}$$

is a nodal basis of V_G. Note that the basis functions have local support, which leads to sparse matrices. We can express D_G in terms of the Galerkin ansatz functions

$$D_G(T, x) = \sum_{i=1}^{n_x} \alpha_i(T)\, \Phi_i(x) \approx \bar{D}(T, x), \tag{3.11}$$

where the coefficient functions $\alpha_i(\cdot)$ have to be determined. It is enough to ensure (3.9) for the basis functions. If we plug in 3.11 in 3.9, we get the following semi-discretized problem.

Remark 3.2.1 (Semi-discretized problem)
Let V_G be a Galerkin approximation to $H_0^1(\Omega)$ as defined in (3.10)*. Then the semi-discretized problem of* (3.8) *is to find coefficient functions*

$\alpha_i(\cdot) \in H^1([0, T_{\max}])$, $i = 1, \cdots, n_x$ *such that for all* $T \in (0, T_{\max}]$

$$\begin{aligned} &\sum_{i=1}^{n_x} \dot{\alpha}_i(T)\langle \Phi_i, \Phi_j \rangle_{L^2} + \sum_{i=1}^{n_x} \alpha_i(T) a(T; \Phi_i, \Phi_j) = F(T; \Phi_j) \quad \forall j = 1, \cdots, n_x \\ &\textit{holds with inital condition} \quad \sum_{i=1}^{n_x} \alpha_i(0)\langle \Phi_i, \Phi_j \rangle_{L^2} = \langle D_0^h, \Phi_j \rangle_{L^2} \quad \forall j = 1, \cdots, n_x \end{aligned} \tag{3.12}$$

To simplify the above semi-discretized problem we will rewrite it in vectorial form.

Remark 3.2.2 (Vectorial semi-discretized problem)
Let us define the mass matrix M *via* $M_{ij} = \langle \Phi_j, \Phi_i \rangle_{L^2}$ *and the stiffness matrix* $A(T)$ *via* $A_{ij}(T) = a(T; \Phi_j, \Phi_i)$. *Also let* $F_j(T) = F(T; \Phi_j), B_j = \langle D_0^h, \Phi_j \rangle_{L^2}$ *hold for* $i, j = 1, \cdots, n_x$. *Then the equivalent matrix form of 3.12 is given by*

$$\begin{aligned} M\dot{\alpha}(T) + A(T)\alpha(T) = F(T), \quad T \in (0, T_{\max}] \\ M\alpha(0) = B, \end{aligned} \tag{3.13}$$

with $M, A \in \mathbb{R}^{n_x \times n_x}, B, \alpha(T), F(T) \in \mathbb{R}^{n_x}$.

We now examine above given matrices closely. First we note that the local support of the basis functions Φ_i leads to a sparse mass matrix. Particularly, M is symmetric and of tridiagonal structure with the entries

$$M_{ii} = \frac{2}{3}\Delta x, \quad M_{i,i+1} = M_{i+1,i} = \frac{1}{6}\Delta x.$$

Due to the integral term in (2.17), the time-dependent stiffness matrix $A(T)$ is more complicated. For an analysis we split $A(T)$ into two parts: the part A^I resulting from the double integral in the bilinear form, and the time-dependent part A^{NI} containing the remaining terms in the bilinear form

$$A(T) = A^{NI}(T) + A^I(T).$$

For the local volatility function $\sigma(T, x)$, which depends on x, a numerical integration method has to be used. We will apply Gaussian integration of order one. Then the

entries of $A^{NI}(T)$ are as follows

$$\begin{aligned}
A^{NI}_{ii}(T) &= \frac{1}{2\Delta x}(\sigma^2_{i-1}(T) + \sigma^2_i(T)) \\
&\quad + \frac{1}{4}(\sigma^2_{i-1}(T) - \sigma^2_i(T) + \sigma^2_{x,i-1}(T) - \sigma^2_{x,i})(T)) + \frac{2}{3}\lambda(1+\xi), \\
A^{NI}_{i,i+1}(T) &= -\frac{1}{2\Delta x}\sigma^2_i(T) + \frac{1}{2}(r - \lambda\xi + \frac{\sigma^2_i(T)}{2} + \frac{\sigma^2_{x,i}(T))}{2}) + \frac{\Delta x}{6}\lambda(1+\xi), \\
A^{NI}_{i+1,i}(T) &= -\frac{1}{2\Delta x}\sigma^2_i(T) - \frac{1}{2}(r - \lambda\xi + \frac{\sigma^2_i(T)}{2} + \frac{\sigma^2_{x,i}(T))}{2}) + \frac{\Delta x}{6}\lambda(1+\xi),
\end{aligned} \tag{3.14}$$

where $\sigma^2_i(T) \coloneqq \sigma^2(T, x_i + \frac{\Delta x}{2})$ and $\sigma^2_{x,i}(T) \coloneqq (\sigma^2(T, x_i + \frac{\Delta x}{2}))_x$.

Let us now consider the matrix A^I. It is not sparse, but we can establish a special structure, called 'Toeplitz'. Considering the local support of the basis functions and defining $g(y) \coloneqq e^y f(y)$, we get

$$\begin{aligned}
A^I_{i,j} &= -\lambda \int_{x_{i-1}}^{x_{i+1}} \int_{x-x_{j+1}}^{x-x_{j-1}} \Phi_j(x-y)\Phi_i(x)g(y)\,dy\,dx \\
&= -\lambda \int_{x_{i-1}}^{x_{i+1}} \int_{x-x_{j+1}}^{x-x_{j-1}} \Phi_{j+1}(x+\Delta x-y)\Phi_{i+1}(x+\Delta x)g(y)\,dy\,dx \\
&= -\lambda \int_{x_i}^{x_{i+2}} \int_{z-x_j}^{z-x_{j+2}} \Phi_{j+1}(z-y)\Phi_{i+1}(z)g(y)\,dy\,dz \\
&= A^I_{i+1,j+1} \quad \forall i,j = 1,\cdots,n_x-1,
\end{aligned}$$

where we used the variable transformation $z = x + \Delta x$. Note that the fact $\Phi_i(x) = \Phi_{i+1}(x + \Delta x)$ only holds for equidistantly distributed grid points.

Remark 3.2.3

For an uniform mesh matrix A^I has Toeplitz structure,

$$A^I = \begin{pmatrix} a_0 & a_1 & \cdots & a_{n_x-1} \\ a_{-1} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & a_1 \\ a_{-n_x+1} & \cdots & a_{-1} & a_0 \end{pmatrix},$$

i.e. $A^I_{ij} = a_{j-i} \quad i,j = 1,\cdots,n_x, \quad a_{-n_x+1},\cdots,a_{n_x-1} \in \mathbb{R}$.

Due to the inner integral and the translation $\Phi_i(x-y)$ and the fact that f has support on whole $\mathbb{R}$, usually none of the entries of A^I equal zero. The double integral has to be approximated via numerical integration. Therefore we divide the two-dimensional area into four squares each of size $\Delta x \times \Delta x$ and use again Gaussian

quadrature of order one. Thus

$$A_{ij}^I = -\lambda \frac{\Delta x^2}{4} \left(g((i-j-1)\Delta x) + 2g((i-j)\Delta x) + g((i-j+1)\Delta x) \right).$$

Even though $A^{NI}(T)$ is sparse (tridiagonal structure) and A^I is of Toeplitz structure, the matrix $A(T) = A^{NI}(T) + A^I$ has none of the properties. The local volatility function $\sigma(T, x)$ is in general space-dependent and hence destroys the constancy on the diagonals of A^{NI}.

We now turn to the matrix $F(T)$. Since f has support $\mathbb{R}$, the double integral arising in $F(T)$ is not restricted to a bounded interval:

$$\begin{aligned}
&- \lambda \int_{\mathbb{R}} \int_{\mathbb{R}} D^b(T, x-y)\Phi_i(x)g(y)\, dy\, dx \\
&= -\lambda \int_{x_{i-1}}^{x_{i+1}} \int_{x-\underline{x}}^{\infty} D^b(T, x-y)\Phi_i(x)g(y)\, dy\, dx \\
&= -\lambda(1 - e^{\underline{x} - \int_0^T r(\tau)d\tau}) \int_{x_{i-1}}^{x_{i+1}} \Phi_i(x) \int_{x-\underline{x}}^{\infty} g(y)\, dy\, dx.
\end{aligned}$$

In the case of Merton's choice of $f \sim N(\mu_j, \sigma_j^2)$, we use quadratic completion and a transformation onto the standard normal distribution to verify

$$\begin{aligned}
\int_{x-\underline{x}}^{\infty} g(y)dy &= e^{\mu_j + \sigma_j^2/2} \int_{x-\underline{x}}^{\infty} \frac{1}{\sqrt{2\pi\sigma_j^2}}\, e^{-\frac{(y-(\mu_j\sigma_j^2))^2}{2\sigma_j^2}}\, dy \\
&= e^{\mu_j + \sigma_j^2/2} \int_{\frac{x-\underline{x}-(\mu_j+\sigma_j^2)}{\sigma_j}}^{\infty} \frac{1}{\sqrt{2\pi}}\, e^{-\frac{z^2}{2}}\, dz \\
&= e^{\mu_j + \sigma_j^2/2} N\left(\frac{-x + \underline{x} + \mu_j + \sigma_j^2}{\sigma_j} \right),
\end{aligned}$$

where $N(\cdot)$ is the cumulative distribution function of the standard normal distribution and therefore $N(-x) = 1 - N(x)$ for all x in $\mathbb{R}$ holds. The calculation of $F(T)$ and B takes place through numerical integration with Gauss quadrature of order one.

3.2.2 Time Discretization

For the time discretization of (3.12) we use the θ-method. This is a finite differences method of order two. Consider an initial value problem given

$$\begin{aligned}
\dot{y}(t) &= f(t, y(t)) \\
y(0) &= y_0.
\end{aligned}$$

Then for any $\theta \in [0,1]$ the discretization with a θ-method is given through

$$\frac{y_{k+1} - y_k}{\Delta t} = \theta f(t_{k+1}, y_{k+1}) + (1-\theta) f(t_k, y_k),$$

where Δt is the step size, $t_k \coloneqq k\Delta t$ and the solution y is approximated via $y(t_k) \approx y_k$. The forward Euler scheme ($\theta = 0$) is an explicit method. All methods with $\theta \neq 0$ are implicit. Special cases are the Crank-Nicolson method ($\theta = \frac{1}{2}$) and the backward Euler $\theta = 1$. We will use a combination of both. We will now define the fully discretized problem.

Remark 3.2.4 (Fully discretized problem)
Let $\theta \in [0,1]$ be given. Define a uniform mesh on $[0, T_{\max}]$ via $T_k \coloneqq k\Delta T, k = 1, \cdots, n_T$ such that $T_{n_T} = T_{\max}$. Then the fully discretized problem to 3.12 consists in finding $\alpha^k \in \mathbb{R}^{n_x}$ for all $k = 0, \cdots, n_T$ such that

$$\begin{aligned}(M + \Delta T\theta A(T_{k+1}))\alpha^{k+1} &= (M - \Delta T(1-\theta)A(T_k)), \alpha^k \\ &\quad + \Delta T(\theta F(T_{k+1}) + (1-\theta)F(T_k)) \\ M\alpha^0 &= B.\end{aligned} \tag{3.15}$$

Diffusion problems often contain irregular initial or boundary data, which destroy the solution's regularity locally. This may result in a reduced global accuracy of the discretization scheme. In our case we have a non-smooth initial condition which leads to oscillations. It is known that the vectorial problem (3.13) is very stiff. An unstable explicit Euler scheme would require a restrictive step size condition to guarantee sufficient accuracy. The implicit Crank-Nicolson method is the preferred choice considering the quadratic convergence which leads to a smaller time discretization error. To smooth irregularities Rannacher [29] approach is a damping procedure which preserves the order of the discretization: the first two initial time steps use backward Euler half-time stepping followed by Crank-Nicolson full-time stepping to yield full second order convergence away from the singularities and not increase the computational costs. Giles and Carter [15] modify Rannacher's start-up procedure to four initial backward Euler full-time or quarter-time steps.

3.2.3 Efficient solution of the fully discretized problem

In the fully discretized problem (3.15) we need to solve a linear system of equations, $\tilde{A}x = b$, in each time step. We established before that the stiffness matrix $A(T)$ is neither sparse nor Toeplitz or symmetric , and so the matrix $\tilde{A}$ is as well not sparse, Toeplitz or symmetric. Through splitting of $A(T)$ we can achieve a matrix vector

product in $\mathcal{O}(n_x \log_2 n_x)$ (cf. [34]).

We will use the generalized minimum residual iteration proposed by Saad and Schultz [31] for solving non-symmetric linear systems of equations. This algorithm belongs to the Krylov subspace methods (for details cf. [30] and [20]).

In the next section we will show numerical results for Merton's jump diffusion model using the following parameters

$$\begin{aligned} &\underline{x} = -5,\ \bar{x} = 5,\ T_{\max} = 2,\ r = 0.03,\ \Delta T = 0.01, \\ &\sigma = 0.3,\ \lambda = 0.5,\ \mu_j = 0,\ \sigma_j = 0.5, \end{aligned} \tag{3.16}$$

where $T_{\max}$ is expressed in years and r and σ are annualized.

The condition number is an important quantity which influences the numerical solution of a linear system of equations. Often the convergence behavior of the solver depends on the condition number. While a large condition number indicates poor convergence behavior, a small condition number signals good convergence behavior. Table 3.1 displays the condition number of the matrix $\tilde{A}$, $\kappa_2(\tilde{A})$, for different spatial step sizes Δx. We see that the condition number grows with decreasing spatial step size. In fact, when Δx is halved, $\kappa_2(\tilde{A})$ quadruples.

Δx	$\kappa_2(\tilde{A})$	$\kappa_2(P^{-1}\tilde{A})$
0.02	2.584	1.003
0.01	9.334	1.003
0.005	36.332	1.003
0.0025	144.325	1.003

Table 3.1: Condition number for different step sizes Δx of the unpreconditioned and the preconditioned system

Furthermore we look at the eigenvalue distribution. In the upper graph of Figure 3.1 the eigenvalue distribution of $\tilde{A}$ is displayed for the spatial step sizes $n_x = 499$ and $n_x = 999$. Here the imaginary part is omitted, because it is insignificantly small. The eigenvalues are evenly spread over an interval, which broadens with the number of spatial steps.

We will use a preconditioner to reduce the condition number and improve the eigenvalue distribution. The rate of convergence for most iterative linear solvers, including GMRES, increases with decreasing condition number. A preconditioner P should fulfill $||I - P^{-1}\tilde{A}||_2 = \rho < 1$. We will use $P := M + \Delta T \theta A^{NI}(T_{k+1})$, which satisfies this requirement. Since M and $A^{NI}(T_{k+1})$ are tridiagonal matrices, a

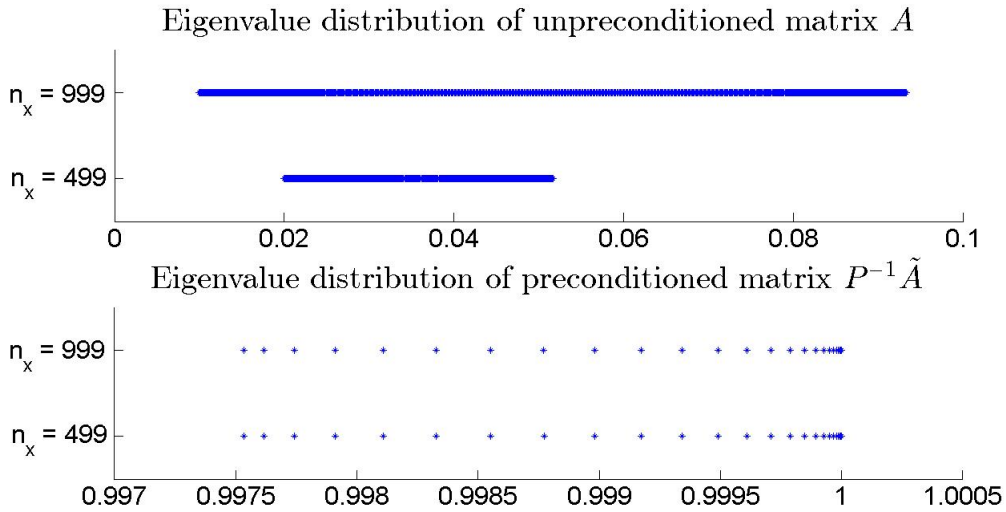


Figure 3.1: Eigenvalue distribution of the unpreconditioned and preconditioned matrix for different space step sizes Δx

linear system is solved in $\mathcal{O}(n_x)$. Applying the preconditioner the condition number $\kappa_2(P^{-1}\tilde{A})$ is reduced to almost 1. In Table 3.1 we see that the condition number of the preconditioned system is mesh-independent. In the lower graph of Figure 3.1 we see that the location of the eigenvalues is independent of the number of spatial steps and that they accumulate around one. The numerical results in the next section underpin the efficiency of the preconditioner. We remark that as initial guess for the linear system of equations, the solution from the previous time step is used. Since we consider a parabolic problem, the solution in time step T_k will not differ much from the solution in the previous time step T_{k-1} and this approach seems natural.

3.3 Numerical results

In this section numerical results for solving PIDE (2.17) are provided. We will first summarize the procedure which needs to be taken in order to compute option prices.

Algorithm 1

---

a) Space discretization: finite element method using a uniform mesh on the spatial domain

b) Time discretization: Crank-Nicolson method with Rannacher smoothing (first time step using four backward Euler quarter steps)

c) Solving dense linear systems of equations $\tilde{A}x = b$: using GMRES method with the sparse part of $\tilde{A}$ as a preconditioner

Figure 3.2 shows a finite element solution with parameter setting according to 3.16 and $\Delta x = 0.0025$ and $\Delta T = 0.01$. For $T = 0$ we see the non-smooth initial condition. Due to the strong smoothing properties of parabolic problems the kink at $x = 0$ is smoothed out as maturity increases.

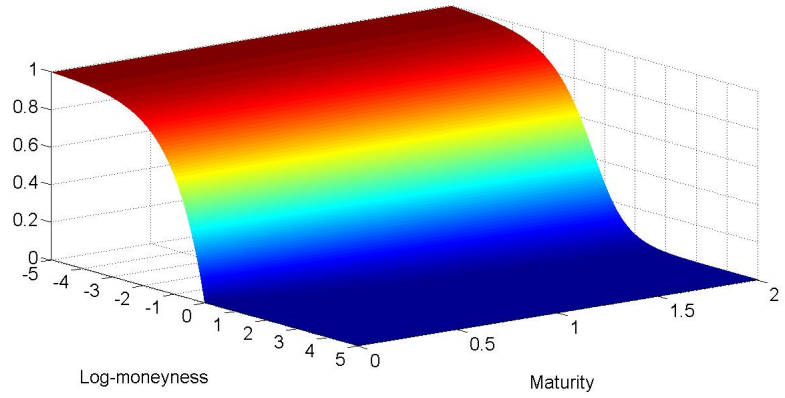


Figure 3.2: FE-solution with Rannacher smoothing $\Delta x = 0.0025, \Delta T = 0.01$

We use a constant volatility instead of a time- and space-dependent volatility to be able to compare the finite element solution with the semi-analytical solution stated in (2.12), which is the exact pricing formula for European call options in Merton's jump diffusion model. Tests have shown that the convergence of the infinite series is fast and a truncation at $n = 50$ provides sufficient accuracy.

Figure 3.3 shows the pointwise error between the semi-analytical solution and the finite element solution with Crank-Nicolson discretization (left graph) and Rannacher smoothing (right graph) on the domain $\Omega = [-1.5, 4]$. We point out that the error increases when we reach the lower boundary which negatively effects the speed of convergence. Due to the irregular initial condition oscillations arise, especially at the non-differentiable point $x = 0$. As expected Rannacher smoothing reduces the oscillations significantly.

The $L^2(\Omega)$-error for maturities $T = 1$ and $T = 2$ between the semi-analytical solution and the finite element solution with Crank-Nicolson discretization and Rannacher smoothing, respectively, is displayed in Table 3.2. Here the spatial step size Δx is fixed and the time step size ΔT varies. The convergence order is computed based on the log-log plot of the $L^2(\Omega)$-error against the time step size ΔT. We

discretization		$L^2(\Omega)$-error Crank-Nicolson		$L^2(\Omega)$-error Rannacher smoothing	
Δx	ΔT	$T = 1$	$T = 2$	$T = 1$	$T = 2$
0.00125	0.08	1.81e-4	1.07e-4	1.78e-5	9.55e-6
	0.04	6.08e-5	3.55e-5	4.23e-6	2.35e-6
	0.02	2.04e-5	1.14e-5	1.04e-6	7.01e-7
	0.01	6.10e-6	3.01e-6	2.48e-7	4.21e-7
convergence order		1.6	1.7	2.0	1.9

Table 3.2: $L^2(\Omega)$-error at $T = 1$ and $T = 2$ between closed-form solution and finite element solution with Crank-Nicolson and Rannacher smoothing time stepping for different time step sizes ΔT and fixed space step size Δx

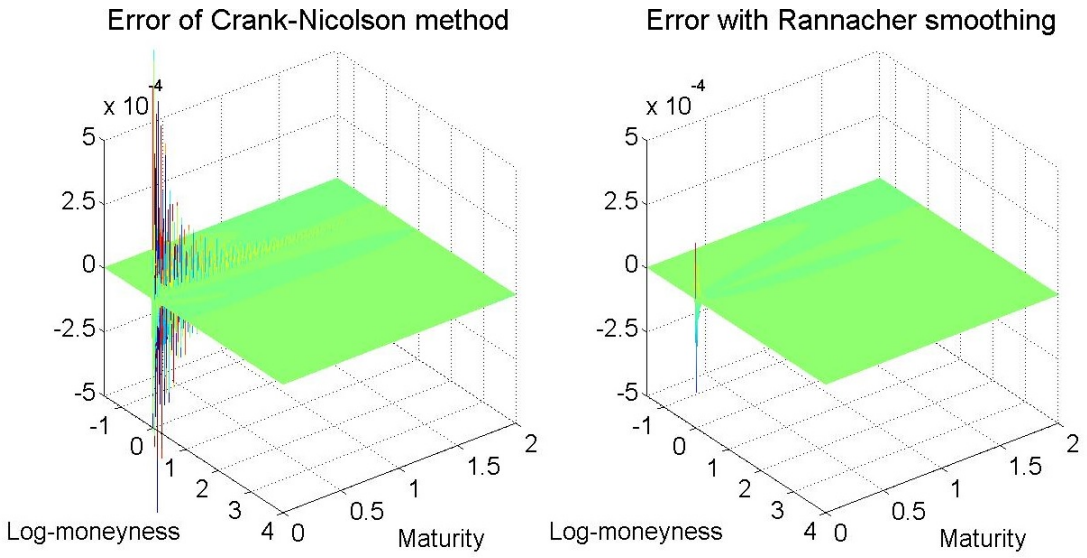


Figure 3.3: Pointwise error between FE-solution and closes-form solution of the Merton model with $\Delta x = 0.0025, \Delta T = 0.01$

see that the convergence order can be improved through the Rannacher damping procedure. While the Crank-Nicolson method does not show quadratic convergence, Rannacher smoothing does. Note that we set the error tolerance for the GMRES algorithm to $1e-8$.

discretization		$L^2(\Omega)$-error		effort	
Δx	ΔT	$T=1$	$T=2$	Time	iter
0.0025	0.08	1.77e-5	9.49e-6	4.95	4.2
	0.04	4.16e-6	2.30e-6	9.89	3.8
	0.02	9.90e-7	7.03e-7	17.94	3.0
	0.01	2.67e-7	4.82e-7	36.37	3.0

Table 3.3: $L^2(\Omega)$-error at $T = 1$ and $T = 2$ between closed-form solution and finite element solution with Rannacher smoothing, computing times and GMRES-iterations per time step for different time step sizes ΔT and fixed space step size Δx

Table 3.3 is structured the same way. The quadratic convergence of the Rannacher smoothing method $O(\Delta T^2)$ in time is confirmed. In the third column the computational effort is shown. The calculation is due to the non-local integral term expensive. The computing time increases linearly with decreasing time step size. In the last column the average number of GMRES-iterations needed per time step is displayed. For the initial grid, which is broad, on average 4 iterations per time step are needed. When we decrease the time step, the GMRES method's first guess

discretization		$L^2(\Omega)$-error		effort	
Δx	ΔT	$T=1$	$T=2$	Time	iter
0.02	0.01	1.41e-5	1.54e-5	1.04	3.0
0.01		3.44e-6	3.87e-6	3.17	3.0
0.005		8.14e-7	1.05e-6	9.25	3.0
0.0025		2.67e-7	4.82e-7	36.46	3.0

Table 3.4: $L^2(\Omega)$-error at $T = 1$ and $T = 2$ between closed-form solution and finite element solution with Rannacher smoothing, computing times and GMRES-iterations per time step for different spatial sizes Δx and fixed time step size ΔT

improves and on average only 3 iterations are needed.

Table 3.3 is designed analogous to the previous table, except that now ΔT is kept fixed and Δx changes. Here the quadratic convergence $O(\Delta x^2)$ is reported. We can see that for all spatial steps the average number of iterations is 3. The calculation time increases over-linearly with decreasing spatial step size.

The finite element solutions with Rannacher smoothing and a jump intensity $\lambda = 0.01$ and $\lambda = 1$ respectively are displayed in Figure 3.4. We can see how the parameter λ effects the shape of the solution.

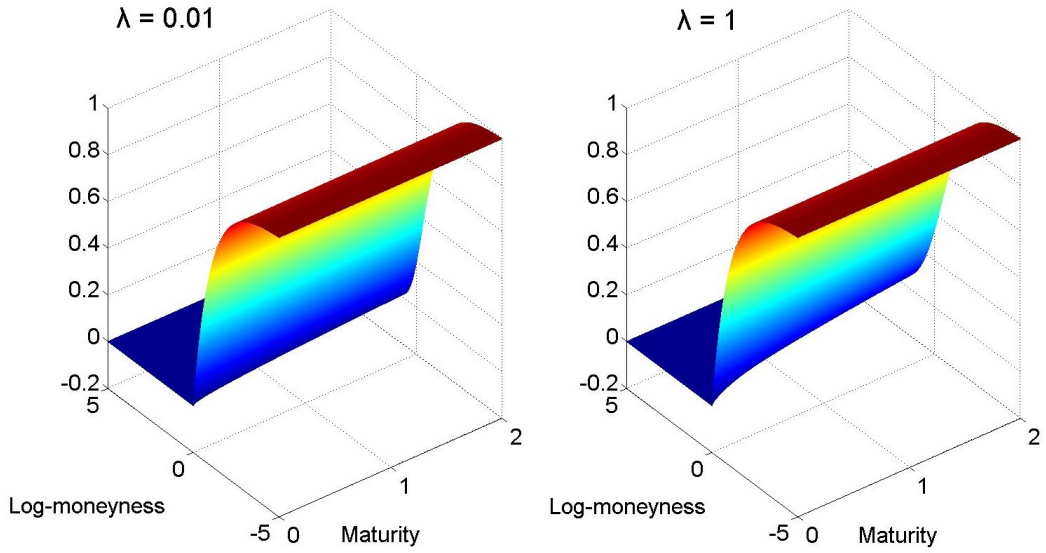


Figure 3.4: FE-solution for different jump intensity with $\Delta x = 0.0025, \Delta T = 0.01$

The difference between these is displayed in Figure 3.5. We can see that the difference is increasing when we approach $x = 0$ from either side and when maturity is increasing.

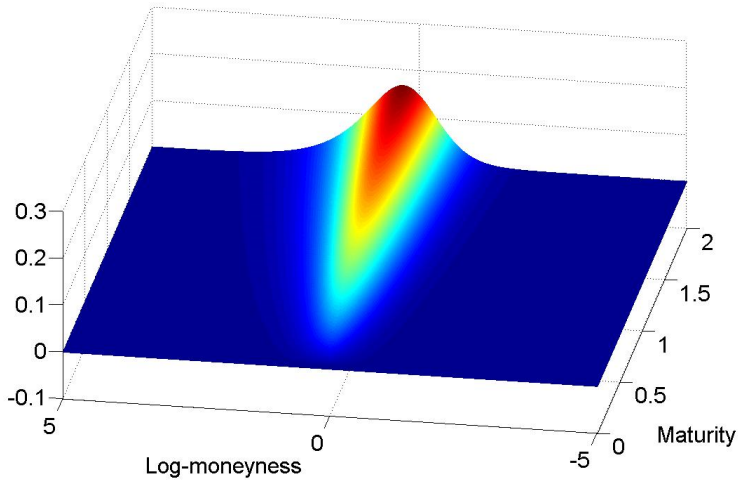


Figure 3.5: Difference between FE-solution with $\lambda = 5$ and $\lambda = 0.01$ for $\Delta x = 0.0025, \Delta T = 0.01$

Overall, numerical results proof efficiency of the implemented algorithm.

Chapter 4

Model Order Reduction via POD

The numerical results of the previous section showed that the implemented algorithm is efficient. But until now we have not taken into account the computational effort. Since we want to solve the PIDE for a multitude of different maturities and strike prices, numerical calculation can take several hours. Therefore we need to reduce the computational cost. An approach is model order reduction. The idea of MOR is to approximate a high-dimensional, complex model by a model with a low-dimensional state space. This reduced model has a significantly lower computational complexity and can hence be solved in adequate time. Naturally the accuracy will decrease. Still, the approximation error should be kept small. The goal of the reduced order model is to capture the essential properties of the original model.

In the proposed finite element method in section 3.2.1 the whole solution space $H_0^1(\Omega)$ is approximated. But often (as in our case of a parameter-dependent differential equation) a lower-dimensional subspace already holds the solutions. The basis functions $\{\phi_i\}_{i=1}^{n_x}$ spanning V_G were chosen out of convenience rather than consideration of the actual problem. We will use proper orthogonal decomposition to find special basis functions, which represent a reduced basis spanning the mentioned lower-dimensional subspace. According to [17] POD was developed in the field of fluid mechanics in the context of turbulence. Other applications include image processing, signal analysis and data compression. POD is one famous model order reduction method and can be applied to non-linear initial value problems. It is strongly connected to singular value decomposition (SVD). For details we refer to [2]. There Antoulas describes the POD method among other things in the chapter of SVD-based approximation methods. The POD problem is developed as an constrained optimization problem and can be solved in a Lagrangian framework. The optimal basis functions are extracted from an information set, stored in vectors called "snapshots", available through experimental data or numerical simulation.

The basis functions are optimal in the sense that they approximate the given information set best. They are subsequent used in a Galerkin approach that yields a low-dimensional model. Summarizing our approach, model order reduction via POD: Solving the PIDE with the finite element method (or using the closed-form solution (2.12)), we obtain snapshots. Then optimal basis functions are obtained through POD. These are used as a reduced basis in a Galerkin approach.

The outline is as follows. First we describe the POD method (Section 4.1). Afterwards a-priori error estimates are derived in Section 4.2. In Section 4.3 we will show numerical results and the efficiency of MOR regarding the reducing of computing time.

4.1 Proper Orthogonal Decomposition

In this section we give an introduction to POD. Hereby we follow closely [36]. POD is also known as Principal Component Analysis or Karhunen-Loeve Decomposition.

Let H be a real, separable Hilbert space with inner product $\langle \cdot, \cdot \rangle_H$ and the induced norm $|| \cdot ||_H \coloneqq \sqrt{\langle \cdot, \cdot \rangle_H}$. Given snapshots $y_i \in H$, $i = 1, \cdots, n$, span the snapshot set $Y \coloneqq span\{y_1, \cdots, y_n\}$ with dimension $d > 0$. The goal of POD is to find an orthonormal basis $\{\Psi_1, \cdots, \Psi_d\}$ of Y with the following characteristic: For each $l \in \{1, , \cdots, d\}$ the (partial) POD basis $\Psi_1, \cdots, \Psi_l$ of rank l is optimal in the sense of representing an average element of Y/in the mean the columns $\{y_i\}_{j=1}^n$ of Y as a linear combination by an orthonormal basis of rank l for any other set of orthonormal vectors $\{\hat{\Psi}_i\}_{i=1}^l$.

First we note that the projection of any element $v \in Y$ onto the space spanned by orthonormal functions $\{\Psi_i\}_{i=1}^l$ can be computed through

$$\bar{v} = \sum_{j=1}^{l} \langle v, \Psi_j \rangle_H \Psi_j.$$

Now we will formally define a POD basis.

Definition 4.1.1 (POD Basis)
Let $y_1, \ldots, y_n \in H$ be given snapshots. Let the dimension of the snapshot set $Y = span(y_1, \ldots, y_n)$ be $d > 0$. Then, for any $l \in \{1, \ldots, d\}$ the solution $\{\psi_i\}_{i=1}^l$ of

$$\min_{\psi_1, \ldots, \psi_l \in Y} \sum_{i=1}^{n} \gamma_i \left\| y_i - \sum_{i=1}^{l} \langle y_i, \psi_j \rangle_H \psi_j \right\|_H^2 \quad s.t. \langle \psi_j, \psi_k \rangle_H = \delta_{jk} \quad \forall j, k = 1, \ldots, l \quad (4.1)$$

with weights $\gamma_i > 0$, $i = 1, \cdots, n$ is called 'POD basis of rank l'.

In the above definition the POD basis is characterized through the minimization of the projection error. We can establish an equivalent maximization problem. The POD basis can also be defined as the functions which capture most features of the model's structure.

Lemma 4.1.2
Let $y_1, \cdots, y_n \in H$ be given snapshots. Let the dimension of the snapshot set $Y = span(y_1, \ldots, y_n)$ be $d > 0$. Then, for any $l \in \{1, \ldots, d\}$ the solution $\{\psi_i\}_{i=1}^{l}$ of

$$\max_{\psi_1,\ldots,\psi_l \in Y} \sum_{i=1}^{n} \gamma_i \sum_{j=1}^{l} \langle y_i, \Psi_j \rangle_H^2 \quad s.t. \langle \Psi_j, \Psi_k \rangle_H = \delta_{jk} \quad \forall j, k = 1, \ldots, l. \tag{4.2}$$

solves (4.1) *and vice versa.*

Proof. For every summand in (4.1) we have

$$\begin{aligned}
||y_i - \sum_{j=1}^{l} \langle y_i, \psi_j \rangle_H \psi_j||_H^2 &= \left\langle y_i - \sum_{j=1}^{l} \langle y_i, \psi_j \rangle_H \psi_j, y_i - \sum_{j=1}^{l} \langle y_i, \psi_j \rangle_H \psi_j \right\rangle_H \\
&= \langle y_i, y_i \rangle_H - 2 \left\langle y_i, \sum_{j=1}^{l} \langle y_i, \psi_j \rangle_H \psi_j \right\rangle_H + \left\langle \sum_{j=1}^{l} \langle y_i, \psi_j \rangle_H \psi_j, \sum_{j=1}^{l} \langle y_i, \psi_j \rangle_H \psi_j \right\rangle_H \\
&= \langle y_i, y_i \rangle_H - 2 \sum_{j=1}^{l} \langle y_i, \psi_j \rangle_H^2 + \sum_{j=1}^{l} \langle y_i, \psi_j \rangle_H \sum_{k=1}^{l} \langle y_i, \psi_k \rangle_H \langle \psi_j, \psi_k \rangle_H \\
&= ||y_i||_H^2 - \sum_{j=1}^{l} \langle y_i, \psi_j \rangle_H^2.
\end{aligned}$$

Thus it is clear that minimizing 4.1 is equivalent to maximizing the last term in above equation. □

We will now define a weighted inner product which is used to replace the inner product in the n_x-dimensional finite element space V_G by an inner product in $\mathbb{R}^{n_x}$. Out of convenience we will write m instead of n_x.

Definition 4.1.3 (Weighted Inner Product)
Let $W \in \mathbb{R}^{m \times m}$ be a symmetric, positive definite matrix. Then the weighted inner product $\langle \cdot, \cdot \rangle_W$ on $\mathbb{R}^m$ is defined via

$$\langle v, u \rangle_W := v^T W u \quad \forall v, u \in \mathbb{R}^m.$$

Remark 4.1.4
Consider two elements $u^h, v^h \in V^h$ with coefficients $\mathrm{u} = (\mathrm{u}_i)_{i=1}^m$ and

$\mathrm{v} = (\mathrm{v}_i)_{i=1}^m \in \mathbb{R}^m$, *respectively, s.t.*

$$u^h(x) = \sum_{i=1}^m \mathrm{u}_i \phi_i(x), \quad v^h(x) = \sum_{i=1}^m \mathrm{v}_i \phi_i(x).$$

Then we have

$$\langle u^h, v^h \rangle_H = \langle \mathrm{u}, \mathrm{v} \rangle_W \quad \textit{and} \quad ||u^h||_H = ||\mathrm{u}||_W,$$

where W is the symmetric, positive definite mass matrix
$W_{ji} = \langle \phi_i, \phi_j \rangle_H$ *for* $i, j = 1, \ldots, m$.

We replace in (4.1) and (4.2) the inner product $\langle \cdot, \cdot \rangle_H$ with the weighted inner product $\langle \cdot, \cdot \rangle_W$ and use the weighted inner product from now on.

We will solve the constrained optimization problem (4.2) in a standard Lagrangian framework. It can be shown that the necessary first-order optimality conditions are transformable to an eigenvalue problem.

Theorem 4.1.5 (Optimality conditions)
The necessary first-order optimality conditions for (4.2) *are given by the symmetric eigenvalue problem*

$$W Y \Gamma Y^T W \psi_i = \lambda_i W \psi_i \quad \textit{for} \quad i = 1, \ldots, l, \tag{4.3}$$

where $\Gamma = diag(\gamma_1, \ldots, \gamma_n) \in \mathbb{R}^{n \times n}$.

Proof. We introduce the Lagrangian $\mathcal{L} : \mathbb{R}^m \times \ldots \times \mathbb{R}^m \times \mathbb{R}^{l \times l}$ through

$$\mathcal{L}(\psi_1, \ldots, \psi_l, \Lambda) = \sum_{i=1}^l \gamma_i \sum_{j=1}^n \langle y_j, \psi_i \rangle_W^2 + \sum_{i,j=1}^l \lambda_{ij} (\delta_{ij} - \langle \psi_i, \psi_j \rangle_W).$$

Then, the first-order optimality conditions are given by

$$\frac{\partial \mathcal{L}}{\partial \psi_k}(\psi_1, \ldots, \psi_l, \Lambda) = 0 \quad \forall k \in \{1, \ldots, l\}. \tag{4.4}$$

Writing this out, we get

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \psi_k}(\psi_1, \ldots, \psi_l, \Lambda) &= \frac{\partial}{\partial \psi_k} \sum_{i=1}^n \gamma_i \sum_{j=1}^l (y_i^T W \psi_j)^2 + \sum_{i,j=1}^l \lambda_{ij} (\delta_{ij} - \psi_i^T W \psi_j) \\
&= \sum_{i=1}^n \gamma_i \cdot 2 \cdot (y_i^T W \psi_k)\, y_i^T W - \sum_{i=1}^l \lambda_{ik} \psi_i^T W - \sum_{j=1}^l \lambda_{kj} \psi_j^T W^T \\
&= \sum_{i=1}^n y_i^T W \, \gamma_i \cdot 2 \cdot (y_i^T W \psi_k) - 2 \sum_{i=1}^l (\lambda_{ik} + \lambda_{ki}) \psi_i^T W \quad \forall k = 1, \ldots, l.
\end{aligned}$$

Thus we arrive with $W^T = W$ at

$$\sum_{i=1}^{n} W^T y_i \, \gamma_i \, y_i^T W \psi_k = \sum_{i=1}^{l} (\lambda_{ik} + \lambda_{ki}) W^T \psi_i \quad \Leftrightarrow \tag{4.5}$$

$$W \, Y \, \Gamma \, Y^T \, W \psi_k = \sum_{i=1}^{l} (\lambda_{ik} + \lambda_{ki}) W \psi_i. \tag{4.6}$$

Now we proceed by induction. We assume that for $l \geq 1$ the first-order optimality conditions are given by

$$W \, Y \Gamma Y^T W \psi_k = \lambda_k W \psi_k \quad \text{for all} \quad k \in \{1, \ldots, l\}. \tag{4.7}$$

Note that due to the fact that W is invertible (since it is symmetric and positive definite by assumption) (4.7) is equivalent to

$$Y \Gamma Y^T W \psi_k = \lambda_k \psi_k \quad \text{for all} \quad k \in \{1, \ldots, l\}. \tag{4.8}$$

For $l = 1$ we fulfill (4.7) with $\lambda_1 = 2\lambda_{11}$

$$W \, Y \, \Gamma \, Y^T \, W \psi_1 = \lambda_1 W \psi_1.$$

Now we have to show that the first-order optimality conditions for a POD basis of rank $l + 1$ are given by

$$W \, Y \, \Gamma \, Y^T \, W \psi_k = \lambda_k W \psi_k \quad \text{for all} \quad k \in \{1, \ldots, l + 1\}.$$

By our induction hypothesis (4.7) this holds for $k \in \{1, \ldots, l\}$. Thus we only have to prove

$$W \, Y \, \Gamma \, Y^T \, W \psi_{l+1} = \lambda_k W \psi_{l+1}.$$

Due to (4.6) we know

$$W \, Y \, \Gamma \, Y^T \, W \psi_{l+1} = \sum_{i=1}^{l+1} (\lambda_{i,l+1} + \lambda_{l+1,i}) W \psi_i. \tag{4.9}$$

The POD basis functions $\{\psi_i\}_{i=1}^{l+1}$ are orthonormal with respect to the weighted product, thus $\langle \psi_{l+1}, \psi_j \rangle_W = 0$ for all $j = 1, \ldots, l$. Using (4.8), the symmetry of

$Y\,\Gamma\,Y^T\,W$ and (4.9) we get for any $j \in \{1,\dots,l\}$

$$\begin{aligned} 0 = \lambda_j \langle \psi_{l+1}, \psi_j \rangle_W &= \langle \psi_{l+1}, Y\Gamma Y^T W \psi_j \rangle_W \\ &= \langle Y\Gamma Y^T W \psi_{l+1}, \psi_j \rangle_W \\ &= \sum_{i=1}^{l+1} (\lambda_{i,l+1} + \lambda_{l+1,i}) \langle \psi_i, \psi_j \rangle_W \\ &= (\lambda_{j,l+1} + \lambda_{l+1,j}), \end{aligned}$$

and thus

$$\lambda_{j,l+1} = -\lambda_{l+1,j}. \tag{4.10}$$

Inserting (4.10) into (4.9) delivers

$$\begin{aligned} W\,Y\,\Gamma\,Y^T\,W\psi_{l+1} &= \sum_{i=1}^{l} (\lambda_{i,l+1} - \lambda_{i,l+1}) W\psi_i + 2\lambda_{l+1,l+1} W\psi_{l+1} \\ &= \lambda_{l+1} W\psi_{l+1}, \end{aligned}$$

when we set $\lambda_{l+1} = 2\lambda_{l+1,l+1}$. □

We will provide a short introduction to SVD (cf. [36]). Let $Y = [y_1,\dots,y_n] \in \mathbb{R}^{m\times n}$ be a given matrix with rank $d \le \min\{m,x\}$. SVD guarantees the existence of orthogonal matrices $\Psi = [\psi_1,\dots,\psi_m] \in \mathbb{R}^{m\times m}$ and $\Phi = [\phi_1,\dots,\phi_n] \in \mathbb{R}^{n\times n}$ and real numbers $\sigma_1 \ge \dots \ge \sigma_d > 0$ such that

$$\Psi^T Y \Phi = \begin{pmatrix} \mathrm{D} & 0 \\ 0 & 0 \end{pmatrix} =: \Sigma \in \mathbb{R}^{m\times n},$$

where $D = diag(\sigma_1,\dots,\sigma_d)$ and '0' denote zero matrices of appropriate dimension. The matrix Y can be factorized as $Y = \Psi\Sigma\Phi^T$. The columns $\{\psi_i\}_{i=1}^d$ of Ψ and $\{\psi_i\}_{i=1}^d$ of Φ, respectively satisfy

$$Y\phi_i = \sigma_i\psi_i \quad \text{and} \quad Y^T\psi_i = \sigma_i\phi_i \quad \text{for all} \quad i = 1,\dots,d.$$

Furthermore they are eigenvectors of YY^T and Y^TY, respectively with eigenvalues $\lambda_i = \sigma_i^2 > 0$ for $i = 1,\dots,d$. The columns $\{\psi_i\}_{i=d+1}^m$ of Ψ and $\{\psi_i\}_{i=d+1}^n$ of Φ are eigenvectors of YY^T and Y^TY, respectively with eigenvalue 0 (if $d < m$ respectively $d < n$).

The following theorem describes how the POD basis can be computed via SVD.

Theorem 4.1.6 (SVD)
Let $Y \in \mathbb{R}^{m\times n}$ be a given matrix with rank $d \leq \min\{m, x\}$, W a symmetric, positive matrix and $\Gamma = diag(\gamma_1, \ldots, \gamma_n)$ with $\gamma_i > 0$ for all $i = 1, \ldots, n$. Further let $\bar{Y} = W^{1/2} Y \Gamma^{1/2}$ and $l \in \{1, \ldots, d\}$. The singular value decomposition of $\bar{Y}$ is given by $\bar{Y} = \bar{\Psi}\bar{\Sigma}\bar{\Phi}^T$. Here $\bar{\Psi} = [\bar{\psi_1}, \ldots, \bar{\psi_m}] \in \mathbb{R}^{m\times m}$ and $\bar{\Phi} = [\bar{\phi_1}, \ldots, \bar{\phi_n}] \in \mathbb{R}^{n\times n}$ are orthogonal matrices and

$$\bar{\Sigma} = \bar{\Psi^T} \bar{Y} \bar{\Phi} = \begin{pmatrix} D & 0 \\ 0 & 0 \end{pmatrix} \in \mathbb{R}^{m\times n},$$

where $D = diag(\sigma_1, \ldots, \sigma_d)$ and $\sigma_1 \geq \ldots \geq \sigma_d > 0$. Then the vectors

$$\psi_i = W^{-1/2} \bar{\psi_i}, \quad i = 1, \ldots, l$$

solve (4.2) *(are a POD basis of rank l).*

Proof. Cf. [36]. □

We now summarize the possibilities we have to compute the POD basis.

Remark 4.1.7
A POD basis can be computed by each of the following methods, having its one strengths and weaknesses.

a) SVD
Computing the singular value decomposition of $\bar{Y}$, we get $\bar{\Psi}$. In Matlab this is done via

$$[\bar{\Psi}, \bar{\Sigma}, \bar{\Phi}] = svd(\bar{Y}).$$

Note that the whole system is solved.

Then we can calculate the POD basis vectors via

$$\psi_i = W^{-1/2} \bar{\psi_i} i = 1, \ldots, l.$$

b)** $n \times n$ **eigenvalue problem (method of snapshots)
If $n < m$ solve the symmetric eigenvalue problem

$$\begin{aligned} \bar{Y}^T \bar{Y} \bar{\phi_i} = \lambda_i \bar{\phi_i}, \quad i = 1, \ldots, l \quad \Leftrightarrow \\ \Gamma^{1/2} Y^T W Y \Gamma^{1/2} \bar{\phi_i} = \lambda_i \bar{\phi_i}, \quad i = 1, \ldots, l \end{aligned}$$

and set

$$\psi_i = W^{-1/2}\bar{\psi}_i = \lambda_i^{-1/2}W^{-1/2}\bar{Y}\bar{\phi}_i = \lambda_i^{-1/2}Y\Gamma^{1/2}\bar{\phi}_i, \quad i = 1, \dots, l.$$

Note that here neither $W^{1/2}$ nor $W^{-1/2}$ need to be calculated. Matlab offers the functions 'eig' and 'eigs' to compute $\bar{\phi}_i$ and the eigenvalues λ_i. With 'eig' the whole system is solved. Therefore one prefers an iterative method for larger systems. Using 'eigs' we can pass the rank l of the POD basis

$$[\bar{\Phi}, \Lambda] = eigs(\Gamma^{1/2}Y^T W Y \Gamma^{1/2}, l)$$

and only the largest l eigenvalues are computed.

c) $m \times m$ ***eigenvalue problem***

If $m < n$ solve the symmetric eigenvalue problem

$$\begin{aligned} \bar{Y}\bar{Y}^T\bar{\psi}_i = \lambda_i\bar{\pi}_i, \quad i = 1, \dots, l \quad &\Leftrightarrow \\ W^{1/2}Y\Gamma Y^T W^{1/2}\bar{\psi}_i = \lambda_i\bar{\psi}_i, \quad i = 1, \dots, l & \end{aligned}$$

and set

$$\psi_i = W^{-1/2}\bar{\psi}_i \quad i = 1, \dots, l.$$

At last we give a result regarding the minimization error when computing the POD basis. It can be shown that this approximation error can be estimated over the sum of remaining eigenvalues. Therefore the error depends on the eigenvalues or more precisely its magnitude on the decay rate of the eigenvalues.

Theorem 4.1.8 (Truncation error)
Let $y_1, \dots, y_n \in H$ be given snapshots. Let the dimension of the snapshot set $Y = span(y_1, \dots, y_n)$ be $d > 0$ and $\{\psi_i\}_{i=1}^l$ be a POD basis of rank l. Then the following holds for the truncation error

$$\arg\min \sum_{i=1}^{n} \gamma_i ||y_i - \sum_{i=1}^{l} \langle y_i, \psi_j \rangle_H \psi_j||_W^2 = \sum_{i=l+1}^{d} \lambda_i. \tag{4.11}$$

Proof. Since the POD basis of rank d span Y, we can write

$$y_i = \sum_{j=1}^{d} \langle y_i, \psi_j \rangle_W \psi_j, \quad \forall i = 1, \dots n.$$

Using the result in the proof of lemma 4.1.2, we get

$$\begin{aligned}
\sum_{i=1}^{n} \gamma_i ||y_i - \sum_{i=1}^{l} \langle y_i, \psi_j \rangle_H \psi_j||_W^2 &= \sum_{i=1}^{n} \gamma_i \left(\langle y_i, y_i \rangle_W - \sum_{j=1}^{l} \langle y_i, \psi_j \rangle_W^2 \right) \\
&= \sum_{i=1}^{n} \gamma_i \left(\left\langle \sum_{i=1}^{d} \langle y_i, \psi_j \rangle_W \psi_j, y_i \right\rangle_W - \sum_{j=1}^{l} \langle y_i, \psi_j \rangle_W^2 \right) \\
&= \sum_{i=1}^{n} \gamma_i \left(\sum_{i=1}^{r} \langle y_i, \psi_j \rangle_W^2 - \sum_{j=1}^{l} \langle y_i, \psi_j \rangle_W^2 \right) \\
&= \sum_{i=1}^{n} \sum_{j=l+1}^{r} \gamma_i \langle y_i, \psi_j \rangle_W^2 \\
&= \sum_{j=l+1}^{d} \lambda_j.
\end{aligned}$$

□

The sum $\sum_{i=1}^{d} \lambda_i$ is sometimes referred to as the 'energy' of the system or the snapshot set. Thus it is natural to choose l such that the quotient

$$\mathcal{E}(l) = \frac{\sum_{i=1}^{l} \lambda_i}{\sum_{i=1}^{d} \lambda_i}. \tag{4.12}$$

exceeds a specific boundary, e.g. $\mathcal{E}(l) \geq 99\%$. Even though this criterion is of interest, in practice the dominator in (4.12) is generally unknown.

4.2 Error Estimates

In this section we will provide a-priori error estimates. First we will describe the idea behind reduced order modeling.

Consider V and H two real, separable Hilbert spaces with $\langle \cdot, \cdot \rangle_V$ and $\langle \cdot, \cdot \rangle_H$ and $|| \cdot ||_V$ and $|| \cdot ||_V$ denoting the inner products and induced norms respectively. Let V^* and H^* denote the dual spaces. Then we have the Gelfand triple

$$V \hookrightarrow H \tilde{=} H^* \hookrightarrow V^*$$

with dense embeddings. Furthermore let $T > 0$ define the final time and assume $a : [0, T] \times V \times V \to \mathbb{R}$ and $F \in L^2(0; T, V^*)$ to be given.

In Theorem 3.1.5 we stated that under certain assumptions on the bilinear form the following general case of an initial value problem in parabolic form admits a unique solution.

Remark 4.2.1 (Continuous problem)
For given $y_0 \in H$ find $y \in W([0,T],V)$ such that y solves

$$\begin{aligned} &\frac{d}{dt}\langle y(t), w\rangle_H + a(t; y(t), w) = \langle F(t; w), w\rangle_H \quad \forall w \in V \\ &\textit{with initial condition} \\ &\langle y(0), w\rangle_H = \langle y_0, w\rangle_H \quad \forall w \in V \end{aligned} \tag{4.13}$$

Now assume we have calculated a POD basis $\{\psi_i\}_{i=1}^{l}$ of rank l. We define the subspace $V^l \coloneqq span\{\psi_1, \cdots, \psi_l\}$ and the Galerkin approximation

$$y^l(t) \coloneqq \sum_{i=1}^{l} \eta_i(t)\, \psi_i(x) \approx y(t)$$

with coefficient functions $\eta_i : [0,T] \to \mathbb{R}, i = 1, \ldots, l$. Through a Galerkin projection we get a a low dimensional model for (4.13).

Remark 4.2.2 (Reduced order model)
For given $y_0 \in H$ find $y^l \in W([0,T],V)$ such that

$$\begin{aligned} &\frac{d}{dt}\langle y^l(t), w\rangle_H + a(t; y(t), w) = \langle F(t; w), w\rangle_H \quad \forall w \in V^l \\ &\textit{with initial condition} \\ &\langle y^l(0), w\rangle_H = \langle y_0, w\rangle_H \quad \forall w \in V^l \end{aligned} \tag{4.14}$$

holds.

Now we can apply a time discretizaztion to (4.14). We use an equidistant time grid

$$t_i = i\Delta t \quad \forall i = 0, \cdots, n, \quad \Delta t = \frac{T}{n}.$$

Using the θ-method we will end up with the following discretized problem.

Remark 4.2.3 (Discretized reduced order model)
For given $y_0 \in H$ find $\{y_i^l\}_0^n \subset V^l$ such that

$$\begin{aligned}
&\langle \bar{\partial} y_i^l, w\rangle_H + \theta\, a(t_i; y_i^l, w) + (1-\theta)\, a(t_{i-1}; y_{i-1}^l, w) = \\
&\theta \langle F(t_i; w), w\rangle_H + (1-\theta) \langle F(t_{i-1}; w), w\rangle_H \quad \forall w \in V^l, i = 1, \cdots, n \\
&\textit{with initial condition} \\
&\langle y_0^l, w\rangle_H = \langle y_0, w\rangle_H \quad \forall w \in V^l
\end{aligned} \tag{4.15}$$

holds.

Note that $\bar{\partial}$ is used as an abbreviation for the finite difference quotient

$$\bar{\partial} y_i^l := \frac{y_i^l - y_{i-1}^l}{dt}.$$

Referencing [9] we state that under further assumptions on a and F problem 4.15 admits a unique solution.

We want to note that in practical applications the solution y often is unknown. Therefore a discretization must be applied to (4.13). In our case we used as spatial discretization a Galerkin approach with basis functions $\{\phi_i\}_{i=1}^{n_x}$. In this case to reduce the dimension we replace the basis functions $\{\phi_i\}_{i=1}^{n_x}$ by a POD basis $\{\psi_i\}_{i=1}^{l}$ with $l \ll n_x$.

We will now describe the arising errors depending on how the POD basis is calculated: with snapshots from the closed-form solution y or obtained from a finite element method y^{FE}.

Error 1

Let y denote the solution of (4.13). Using the solution y at time instances t_i as well as the corresponding finite differences, the POD basis is calculated. Hence the $2n+1$ snapshots are given by

$$\begin{aligned}
\bar{y_i} &= y(t_{i-1}), i = 1, \cdots, n+1 \\
\bar{y_i} &= \bar{\partial} y(t_{i-m-1}), i = n+2, \ldots, 2n+1.
\end{aligned}$$

The POD solution of the discretized reduced order model (4.15) will be called $y^{l,1}$ and the average error is given by

$$ERR_1 = \sum_{i=1}^{n} \gamma_i ||y(t_i) - y_i^{l,1}||_H^2. \tag{4.16}$$

Error 2

Let y^{FE} denote the solution of (4.15), where the space domain is discretized using the finite element method and V^l is replaced with V_G. As snapshots the finite element solution at time instances t_i as well as the corresponding finite differences are used. The POD solution of (4.15) will be called $y^{l,2}$ and the average error is given by

$$ERR_2 = \sum_{i=1}^{n} \gamma_i \, ||y_i^{FE} - y_i^{l,2}||_H^2. \tag{4.17}$$

As we are mainly following [34] in this section, the finite difference quotients were included in the snapshot set, because this simplifies the proof of the error estimates. At the end we will also provide an error bound without including the time derivatives.

For $V^l \subset V$ we define the H-projection operator Π_H^l via

$$\Pi_H^l : V \to V^l \quad \Leftrightarrow \langle P^l u - u, w \rangle_H = 0 \, \forall w \in V^l$$

Furthermore let $S \in \mathbb{R}^{d \times d}$ denote the stiffness matrix, i.e.

$$S_{ij} = \langle \psi_j, \psi_i \rangle_V, \quad i, j = 1, \ldots, d.$$

Schu [34] also assumes that $y_0 \in V$. Then we can state the main results for the average errors. For details we refer to [34].

Theorem 4.2.4 (POD error 1)
Let $y(t)$ be the solution of (4.13), $\{y^{l,1}\}_{i=0}^n$ the solution of (4.15). Then with appropriate constants $C_i (i = 0, 1, 2)$, independent of n, we have

$$\frac{1}{n} \sum_{i=1}^{n} ||y(t_i) - y_i^{l,1}||_H^2 \leq C_0 ||y_(t_0) - \Pi_H^l y(t_0)||_H^2 + C_1 \Delta t^j + C_2 ||S||_2 \sum_{i=l+1}^{d} \lambda_j \tag{4.18}$$

with $j = 2$ for the implicit Euler method assuming $y_{tt} \in L_2([0, T]; H)$ and $j = 4$ for the Crank-Nicolson method assuming $y_{tt} \in L_2([0, T]; H)$ and Δt sufficiently small. Furthermore, for some constant C we have

$$||y_(t_0) - \Pi_H^l y(t_0)||_H^2 \leq nC \sum_{i=l+1}^{d} \lambda_j{}^1.$$

We note that the error consists of two parts: the projection error and the error

[1] Quote of Theorem 4.2.6, p. 69, in Schu [34]. We changed r to d to be consistent with our notation. Note that the weights are $\gamma_i = \frac{1}{n}, i = 1, \ldots, n$.

resulting from the time discretization. When we use the maximum number of POD basis functions, the first one vanishes and our error bound becomes $C_1\Delta t^j$. Hence as expected using Crank-Nicolson discretization instead of implicit Euler results in a better order.

Theorem 4.2.5 (POD error 2)
Let $\{y^{FE}\}_{i=0}^n$ be the finite element solution using the finite element space V_G in the Galerkin approximation. Let $\{y^{l,2}\}_{i=0}^n$ be the solution of problem (4.15) based on the FEM snapshots. Then with appropriate constants $\tilde{C}_0, \tilde{C}_1$ independent of n, we have for the implicit Euler method and,for sufficiently small Δt, also for the Crank-Nicolson method

$$\frac{1}{n}\sum_{i=1}^{n} ||y_i^{FE} - y_i^{l,2}||_H^2 \leq \tilde{C}_0 ||y_0^{FE} - \Pi_H^l y_0^{FE}||_H^2 + \tilde{C}_1 ||S||_2 \sum_{i=l+1}^{d} \lambda_j \tag{4.19}$$

where

$$||y_0^{FE} - \Pi_H^l y_0^{FE}||_H^2 \leq 3n \sum_{i=l+1}^{d} {\lambda_j}^2.$$

In above estimate we compare the POD solution with the finite element solution. Both contain an error induced by the time discretization. Hence using the maximum number of POD basis functions results in an error of zero.

The error estimate without time derivatives is taken from [33]. In the paper Singler considers the nonlinear evolution equation

$$\begin{aligned} &\frac{d}{dt}\langle y(t), w\rangle_H \; + \; a(t; y(t), w) \; + b(y(t), y(t), w) = \langle F(t; w), w\rangle_H \quad \forall w \in V, \\ &y(0) = y_0 \end{aligned} \tag{4.20}$$

where $b(u, v, w) = \langle B(u, v), w)\rangle$ and $B : V \times V \to V^*$ quadratic nonlinear, and the reduced order model

$$\begin{aligned} &\frac{d}{dt}\langle y^l(t), w\rangle_H \; + \; a(t; y(t), w) \; + \; b(y^l(t), y^l(t), w) = \langle F(t; w), w\rangle_H \quad \forall w \in V^l \\ &y^l(0) = y_0^l \end{aligned} \tag{4.21}$$

The POD basis is determined within the continuous framework: for given $y \in L^2(I; H)$ with $I = (a, b), -\infty \leq a < b \leq \infty$ find an orthonormal basis $\{\psi_i\} \subset H$

[2]Quote of Theorem 4.2.8, p. 73, in Schu [34]. We changed r to d and H^{n_x} to V_G to be consistent with our notation. Note that the weights are $\gamma_i = \frac{1}{n}, i = 1, \ldots, n$.

minimizing

$$\int_I ||y(t) - \sum_{i=1}^{l} \langle y(t), \psi_i \rangle_H \psi_i.$$

It can be shown that the POD basis functions are the singular vectors of the compact linear operator $K : L^2(I) \to H$ defined as $Ku = \int_I y(t)u(t)dt$. Furthermore let $P_l^H : H \to V^l$ be defined by: for $x \in H$, $P_l^H x$ minimizes $\inf_{x_l} \in V^l ||x - x_l||_H$.

Theorem 4.2.6
Let $T > 0$, $f \in L^2(0,T;H)$, and $y_0 \in V$, and let y be the exact solutions of the problem (4.20). Let $\{\sigma_k, f_k, \psi_k\} \subset \mathbb{R} \times L^2(0,T) \times H$ be the singular values and singular vectors of the operator $K : L^2(0,T) \to H$. Define $V^l = span\{\psi_1, \ldots, \psi_l\}$. Let y^l be the exact solutions of the problem (4.21). If $||y_0^l||_H$ is bounded independent of l, then there exists a constant C, independent of l, such that

$$||y - y^l||_{L^2(0,T;V)} \leq C_1(T)||y_0^l - P_l^H y_0||_H + C_2(T) \left(\sum_{i>l} \sigma_i^2 ||\phi_i||_V^2 \right)^{\frac{1}{2}}.^3 \tag{4.22}$$

Note that for increasing number of POD basis functions the error bound converges to zero.

4.3 Numerical results

Before we present numerical results for the POD method, we will show how to determine the POD basis in our approach.

Computation of POD Basis

To compute a POD basis, we first need snapshots. In our case these can be either determined by computing the semi-analytical expression (2.12) or through a finite-element solution of the PIDE (2.17). For this we need a time discretization. We use an equidistant grid with points

$$T_i = i\,\Delta T, \quad \Delta T = T/n, \quad i = 0, \ldots, n-1. \tag{4.23}$$

Then the snapshots $y_1, \ldots, y_n$ are given through

$$y_i = D(T_{i-1}), \quad i = 1, \ldots, n, \tag{4.24}$$

[3] Quoting Singler [33], Theorem 6, p.22

where $D(T_i)$ is the solution of the PIDE at T_i. As weights we choose $\gamma_i = 1/n$. Note that the weights represent a certain integral discretization. We could also use other weights, e.g. the trapezoidal rule, i.e.

$$\gamma_1 = \frac{\Delta T}{2}, \quad \gamma_i = \frac{\Delta T}{2}\, i = 2, \ldots, n-1, \quad \gamma_n = \frac{\Delta T}{2}. \tag{4.25}$$

Given weights $\gamma_i > 0$, $i = 1, \ldots, n$ we define $\Gamma = diag(\gamma_1, \ldots, \gamma_n)$. V_G is spanned by the basis functions $\phi_1, \ldots, \phi_{n_x}$. Then we can write

$$y_i(x) = \sum_{j=1}^{n_x} S_{ji}\phi_j(x) \quad \forall i = 1, \ldots, n_x, \tag{4.26}$$

where $S \in \mathbb{R}^{n_x \times n}$ denotes the coefficient matrix and $\bar{Y} = M^{1/2} S \Gamma^{1/2}$ with M representing the symmetric, positive definite mass matrix. Then we have to solve the eigenvalue problem

$$\bar{Y}^T Y v_j = \lambda_j v_j. \tag{4.27}$$

Since we only want to find the l largest eigenvalues, we use MATLAB's iterative solver 'eigs', where the parameter l can be input. Then we get $\lambda_1, \ldots, \lambda_l$ and corresponding eigenvectors $v_1, \ldots, v_l$. Then we are able to compute the coefficient matrix $P \in \mathbb{R}^{n_x \times l}$ via

$$P_{\cdot,j} = \frac{1}{\sqrt{\lambda_j}} S \Gamma^{1/2} v_j$$

and the POD basis functions are given as

$$\psi_j(x) = \sum_{k=1}^{n_x} P_{kj}\phi_k(x) \quad \forall j = 1, \ldots, l.$$

With the POD basis functions we are able to solve a reduced-order model using them in a Galerkin approach. For this we compute the lower-dimension matrices $M_l, A_l(T) \in R^{l \times l}$ and the vectors $B_l, F_l \in \mathbb{R}^l$ via

$$M_l = P^T M P, \quad A_l(T) = P^T A(T) P, \quad B_l = P^T B, F_l(u;T) = P^T F(u,T). \tag{4.28}$$

Then the semi-discretized problem is given as

$$M_l \dot{\eta}(T) + A_l(T)\eta(T) = F_l \tag{4.29}$$

$$M_l \eta(0) = B_l, \tag{4.30}$$

where $\eta = (\eta_i)_{i=1}^{l}$ are the coefficient functions from the Galerkin ansatz

$$y^l(T,x) = \sum_{i=1}^{l} \eta_i(T)\psi_i(x). \tag{4.31}$$

Algorithm 2

a) POD basis computation: Computing the closed-form solution (2.12) at the time grid or solving PIDE 2.17 with the finite element method, we obtain snapshots $y_1, \ldots, y_n$. Using MATLAB's 'svd' solver we compute the POD basis of rank l.

b) Space discretization: Galerkin method using POD basis functions

c) Time discretization: Crank-Nicolson method with Rannacher smoothing (first time step using four backward Euler quarter steps)

d) Solving dense linear systems of equations $\tilde{A}x = b$ using MATLAB's direct solver ' \'.

We will now show numerical results using the same parameters as in (3.16)

$$\underline{x} = -5,\ \bar{x} = 5,\ T_{\max} = 2,\ r = 0.03,\ \Delta x = 0.0025,\ \Delta T = 0.01,$$
$$\sigma = 0.3,\ \lambda = 0.5,\ \mu_j = 0,\ \sigma_j = 0.5.$$

Figure 4.3 shows the eigenvalues in semi-logarithmic scale. Their fast decay is observable: the fifth eigenvalue already has a magnitude of less than 10-5 and the tenth less than 10-10.

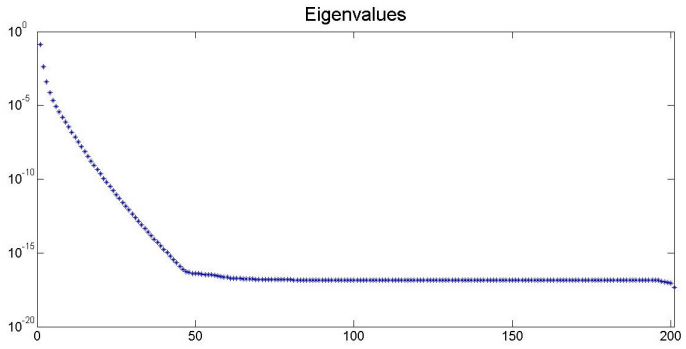


Table 4.1: Eigenvalues in semi-logarithmic scale

In Figure 4.3 we see the actual value of different eigenvalues.

Nr.	Eigenvalue
1	1,34e-1
5	2,22e-5
10	3,34e-7
15	7,25e-9
25	8,97e-12
50	4,10e-17
200	8,59e-18

Table 4.2: Different eigenvalues

In Figure 4.1 POD basis functions for $l = 10$ are displayed. Although the plot of single basis functions does not provide much information, the whole plot shows, that the POD basis functions are mainly active around the origin. This is expected and desired, since our region of interest is around $x = 0$, where the non-smooth initial condition is smoothed out over time.

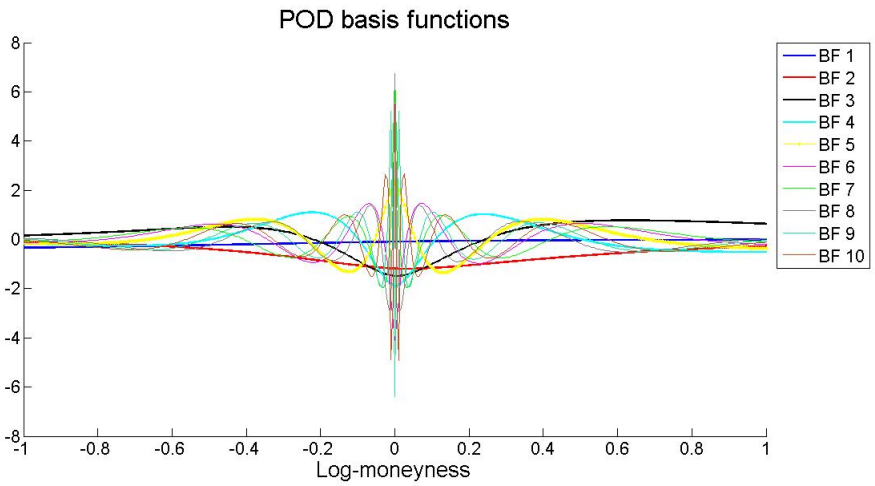


Figure 4.1: POD basis functions for $l = 10$

In Table 4.3 we analyzed the approximation error and sum of remaining eigenvalues for different values of l, the number of POD basis functions. First we used the closed form solution (2.12) to compute snapshots $y(t_0), \ldots, y(t_{n_t})$. Note that we make two errors here: the infinite series is truncated at $n = 50$ and the space steps are finite. We then compute a POD basis, which we use in a Galerkin approach. As time discretization we apply the Crank-Nicolson method. We notice that the error $ERR_1 = \frac{1}{n}\sum_{i=1}^{n} ||y(t_i) - y_i^{l,1}||_H^2$ decays until 1.81e-9 and then stays around this magnitude. This is approximately of order ΔT^4. The remaining sum of eigenvalues $\sum_{i=l+1}^{d} \lambda_i$ decays fast and overall.

We then computed the finite element solution of the PIDE with Crank-Nicolson

	closed form solution POD solution using CN		FE solution using CN POD solution using CN		FE solution using RS POD solution using RS	
l	ERR_1	$\sum_{i=l+1}^{d} \lambda_i$	ERR_2	$\sum_{i=l+1}^{d} \lambda_i$	ERR_2	$\sum_{i=l+1}^{d} \lambda_i$
5	2.68e-8	3.27e-8	2.61e-8	3.44e-8	2.58e-8	3.30e-8
6	7.25e-9	6.36e-9	6.63e-9	8.22e-9	5.85e-9	6.34e-9
7	2.96e-9	1.38e-9	2.59e-9	2.94e-9	1.41e-9	1.35e-9
8	2.00e-9	2.72e-10	1.16e-9	1.15e-9	3.02e-10	2.80e-10
9	1.82e-9	4.86e-11	4.66e-10	4.65e-10	6.63e-11	5.65e-11
10	1.80e-9	8.26e-12	1.92e-10	1.92e-10	1.78e-11	1.16e-11
11	1.80e-9	1.36e-12	7.95e-11	7.98e-11	7.07e-12	2.45e-12
12	1.80e-9	2.18e-13	3.33e-11	3.31e-11	4.29e-12	5.35e-13
13	1.81e-9	3.42e-14	1.37e-11	1.37e-11	3.46e-12	1.21e-13
14	1.81e-9	5.30e-15	5.59e-12	5.59e-12	3.14e-12	2.82e-14
15	1.81e-9	7.04e-16	2.26e-12	2.26e-12	3.01e-12	6.83e-15

Table 4.3: Error ERR_1 between closed form solution and POD solution with Crank-Nicolson time stepping, Error ERR_2 between finite element solution with Crank-Nicolson time stepping and POD solution using Crank-Nicolson time stepping and finite element solution with Rannacher smothing time stepping and POD solution using Rannacher smoothing, and the corresponding sum of remaining eigenvalues for different numbers of POD basis functions l

time stepping y_i^{FE} at time instances t_i. This is used as a new snapshot set to compute the reduced order model solution with Crank-Nicolson time stepping. We observe that now the error $ERR_2 = \frac{1}{n}\sum_{i=1}^{n} ||y_i^{FE} - y_i^{l,2}||_H^2$ decays for all l.

At last we used the finite element solution with Rannacher smoothing as snapshot set. As time discretization for the reduced order model Rannacher smoothing is also used. In the last section of the previous chapter we showed how applying Rannachers damping procedure smoothes out the non-smooth initial condition. This feature influences the reduced order model solution. The remaining sum of eigenvalues decay faster and hence also the error than with Crank-Nicolson time stepping.

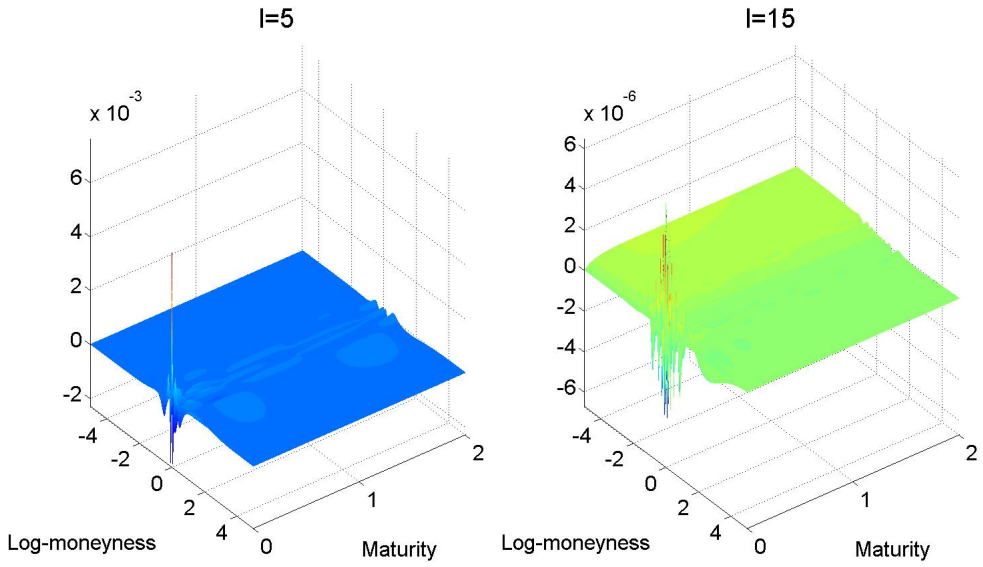


Figure 4.2: Pointwise error between the FE solution and POD solution for different basis sizes l with $\Delta x = 0.00125$ and $\Delta T = 0.005$

Figure 4.2 shows the pointwise error between the FE solution using Rannacher smoothing and POD solution using Rannacher smoothing with $l = 5$ basis functions (left) and $l = 15$ basis functions (right). We can see that the error is concentrated at $T = 0$ and with fewer basis functions the amplitude of the oscillation at $x = 0$ is much higher. Overall, we note that the POD solution can approximate the finite element solution well.

Not only the approximation error is of interest, but also the computation time. Table 4.3 shows a comparison between the finite element and the POD method. While the calculation time of the FE method increases almost quadratically with increasing time or spatial steps, the calculation time of the POD method increases less than linearly. The POD method requires the calculation of the POD basis only

discretization		time	
ΔT	Δx	FEM	POD
0.01	0.005	9.92	0.44
	0.0025	35.98	0.62
	0.00125	138.49	1.01

discretization		time	
ΔT	Δx	FEM	POD
0.005	0.005	17.32	0.84
	0.0025	65.09	1.33
	0.00125	241.68	2.12

Table 4.4: Overall computing times for finite-element method and POD-method with $l = 10$ for different spatial sizes Δx and fixed time step size ΔT

once per discretization. Furthermore the dimension of the system is always reduced to $l \times l$. When we look at the absolute values, the reduction of the computational effort becomes visible. This is especially apparent for a fine grid: with discretization $\Delta T = 0.005$ and $\Delta x = 0.00125$ using the POD method instead of the finite element method decreases the computational time by a factor more than 100.

Chapter 5

Treatment of distributed uncertainties

Up to now we have considered fixed input data and developed a deterministic numerical approach. In our model we chose values for the incorporated parameters. In this chapter we focus on random input data. To be precise we consider the jump intensity λ to be a stochastic variable and make assumptions on the distribution.

In section 5.1 we first present the theory behind distributed uncertainties. In section 5.2 we describe the numerical approach we will use: an extension to the POD method based on a greedy algorithm. In the last section 5.3 we discuss the numerical findings.

5.1 Theory

Nowadays modeling of uncertainties is a of great importance. In particular the effects of uncertainties on the system output need to be analyzed and quantified. Uncertainty is normally introduced through the input data, e.g. model parameters or initial and boundary conditions. In practical applications the input values are usually determined from experiments. Since these experiments possess uncertainties, it is clear that the input data naturally holds uncertainty as well. To cope with model and data uncertainty and to provide numerical accuracy in the presence of it, the design of the system has to be adapted. To be able to compute with the stochasticity involved, we first need to understand how the solution is effected by the uncertainties and then include this into the model system.

Borzi et al. [4] describe PDE constraint optimization with uncertainties involved.

Using PDE's to describe models is a standard procedure. Often this PDE is a constraint in an optimization problem. Model parameters are usually unknown and therefore the implemented system is exposed to a priori uncertainties. In their paper Borzi et al. discuss different methods for solving optimization problems with a PDE constraint subject to stochasticity and show their robustness against this. For an efficient implementation algorithm, structures in the underlying problem need to be taken into account and made use of. Now stochasticity presents a new structure. A probability distribution for the uncertainty is assumed and needs to be taken account of in the model design. In practice,the true distribution is often unknown, so that an estimate for it must be used.

We now consider the following problem

$$\min_{y,u} J(y,u) \quad s.t. \quad c(y,u) = 0. \tag{5.1}$$

Here J denotes the real-valued differentiable objective function and c the PDE constraint with y being the state variable and u the optimization variable. We assume that $c_y(y(u),u)$ is invertible, which implies that for fixed u a unique $y = y(u)$ exists solving the PDE constraint. Let σ denote the uncertainty. We assume that σ is a mapping $\zeta : \mathcal{O} \to \mathbb{R}^n$, with $(\mathcal{O}, \mathcal{A}, \mathcal{P})$ denoting a probability space. Then the expected value of ζ is given through

$$\zeta_o \coloneqq \mathbb{E}[\zeta] = \int_{\mathcal{O}} \zeta(\omega) d\mathcal{P}(\omega).$$

In the following we will discuss three different approaches in dealing with the stochastic uncertainty ζ:

$$\begin{aligned} u_o &\coloneqq \arg\min \{ J(y,u,\zeta_o) \,|\, c(y,u,\zeta_o) = 0 \} \\ \bar{u} &\coloneqq \mathbb{E}[\arg\min_u \{ J(y,u,\zeta) \,|\, c(y,u,\zeta) = 0 \}] \\ \hat{u} &\coloneqq \arg\min_u \mathbb{E}[\{ J(y,u,\zeta) \,|\, c(y,u,\zeta) = 0 \}] \end{aligned} \tag{5.2}$$

In general the methods lead to different solutions, i.e. $u_o \neq \bar{u} \neq \hat{u} \neq u_o$.

In the first design, the expected value ζ_o of the random variable ζ is used. Thus, this method performs a deterministic optimization within which no unknown or stochastic variables are calculated. We did this in section 4.3. In the second method the optimization needs to be done for different values of the stochastic variable in order to estimate the mean. Already existing PDE constrained optimization solver can be used. In the next section this approach will be implemented. The last method

is out of scope of this thesis. For more details we refer to [4].

For the second method in (5.2) we will first introduce an extension to the POD method described in section 4.1. There we computed the POD basis based on one snapshot set with a fixed value of ζ (the mean). For the second method in (5.2) the solver needs to be applied to the problem for different values of the stochastic variable. Hence, the question arises if the determined POD basis is still well suited. This question will be dealt with in section 5.3. In the next section we describe a method to determine a POD basis for multiple snapshot sets corresponding to different values of ζ.

5.2 Numerical Approach

Following closely [21], we discuss when and how multiple snapshot sets can be used (Section 5.2.1). Afterwards a greedy algorithm is presented with which we can extract a POD basis from the combined snapshot set (Section 5.2.2).

5.2.1 Multiple snapshot sets

The POD method introduced in 4.1 is based on a single snapshot set. There the model parameters were kept fixed. Many applications though generate multiple snapshot sets for different sets of parameters. We now consider the case that for different values of the parameter ζ corresponding snapshot sets are available. This means that for each value $\zeta_1, \dots, \zeta_{N_s}$ a simulation of the system was performed in order to get snapshot sets $Y_1, \dots, Y_{N_s}$, where $Y_j = [y_1^{\zeta_j}, \dots, y_{n_t}^{\zeta_j}]$, $j = 1, \dots, N_s$. We assume that the spatial and time discretization is the same for all parameter values and therefore the snapshot sets have same size $n_x \times n_t$.

Using the standard approach we would apply the POD method straight to the multiple snapshot sets. This means that all snapshot sets are considered equal and one snapshot Y is set up via $Y = [Y_1, \dots, Y_{N_s}]$ having dimension $n_x \times n_t \cdot N_s$. The procedure of calculating the POD basis stays the same, only the weighting matrix $\tilde{\Gamma} = diag(\Gamma, \dots, \Gamma) \in \mathbb{R}^{n_t \cdot n_s \times n_t \cdot n_s}$ is now a block diagonal matrix. As described before one can either perform singular value decomposition or solve an eigenvalue problem. For the eigenvalue problem we need to compute the matrices

$$\bar{Y}^T \bar{Y} = \tilde{\Gamma}^{1/2} Y^T W Y \tilde{\Gamma}^{1/2} \in \mathbb{R}^{n_t \cdot n_s \times n_t \cdot n_s}$$

or respectively

$$\bar{Y}\bar{Y}^T = W^{1/2} Y \tilde{\Gamma} Y^T W^{1/2} \in \mathbb{R}^{n_x \times n_x}.$$

The first matrix can become very large if N_s is large. Therefore using the second matrix can be the better choice if $n_x < n_t \cdot N_s$. Even though the matrices $W^{1/2}$ and $W^{-1/2}$ need to be calculated, this computational effort normally is negligible compared the achieved smaller dimension. However in both cases the computational effort can be very high. Thus we will focus on another approach.

5.2.2 Greedy algorithm

We will now describe the POD-greedy algorithm, which we will use in the next section to determine the POD basis. According to [21] for parametrized systems the POD basis used in ROM is often computed via a greedy algorithm. In time-dependent problems this is a POD-greedy algorithm. The algorithm is shown below.

Algorithm 3: POD-Greedy

Require: N_s snapshot sets $Y_j = [y_1^{\zeta_j}, \ldots, y_n^{\zeta_j}]$ associated with different parameter values ζ_j, initial parameter value ζ_i, initial basis rank l, tolerance ϵ^{POD} for the projection error

1. $\Psi \leftarrow POD_l(Y_i), \quad L = l$
2. Compute projection errors for all n_s snapshot sets

$$E_j = w_j \, ||Y_j - \mathcal{P}_\Psi(Y_j)||_W^2, \quad j = 1, \ldots, n_s$$

3. while $\max(E_j) > \epsilon^{POD}$ do
4. $\quad k \leftarrow \arg\max_{j=1,\ldots,n_s} E_j$
5. $\quad \bar{\Psi} \leftarrow POD_l(Y_k - \mathcal{P}_\Psi(Y_k)), \quad \Psi \leftarrow \Psi \cup \bar{\Psi}, \quad L = L + l$
6. $\quad$ Compute projection errors E_j for all n_s snapshot sets using Ψ
7. end while
8. return POD basis Ψ[1]

[1]This algorithm is taken from [21], p. 50. We included the initial basis rank l in the required inputs. We do not necessarily start the POD-greedy algorithm with the first snapshot set Y_1, but with the snapshot set corresponding to the initial parameter value ζ_i. Therefore we added this parameter to the required inputs. Furthermore we added weights $w_j > 0$ when calculating the projection error.

In the above algorithm POD_l means determining the POD basis with rank l according to (4.1). Each of the approaches discussed in 4.1.7 can be used to extract the l basis functions corresponding to the l largest eigenvalues. We start with an initial snapshot set and calculate the corresponding POD basis with rank l. Note that the snapshot sets are computed beforehand the POD-greedy algorithm, making it computational efficient. This is the main difference to the standard reduced basis method where the snapshot sets are calculated within the algorithm. As next step the approximation error for all snapshot sets is computed. The error is given by

$$E_j = w_j ||Y_j - \mathcal{P}_\Psi(Y_j)||_W^2 = w_j \sum_{i=1}^{n} \gamma_i \left\| y_i^{\zeta_j} - \sum_{i=1}^{l} \langle y_i^{\zeta_j}, \psi_j \rangle_W \psi_j \right\|_W^2 \quad j = 1, \ldots, N_s,$$

where $w_j > 0$ are weights depending on the parameter value ζ_j. Here $\mathcal{P}_\Psi$ is the orthogonal projection onto the subspace spanned by Ψ with respect to the weighted inner product. Note that the errors depend only on the particular snapshot sets which are already computed. Therefore the evaluation can be done parallel making the implementation highly efficient. Our algorithm belongs to the strong greedy algorithms as the true error is calculated and not an suitable error estimate as in the case of weak greedy algorithms (cf. [28]). As long as the maximum error exceeds the error tolerance, in each run l basis functions are added to the POD basis. This means that the basis is build iteratively and the (final) size of the POD basis is not required as input in the algorithm. In order to find a global optimum greedy algorithms make the locally best choice at each step. In our case first the snapshot set which is approximated worst is determined. Then the POD basis is supplemented by the basis functions computed based on the difference between this snapshot set and its projection. Because the basis computation is based on the projection residuals, which are orthogonal to the current basis, the basis is orthonormal in each run. Even if a snapshot set is chosen more than once, this orthonormality is still given. But due to numerical errors, an orthogonalization procedure is practical. In order to improve the numerical results, we chose a Gram-Schmidt-procedure (cf. [26]). After obtaining the POD basis $\bar{\Psi}$ and complementing the POD-basis from the previous run in step 5, the Gram-Schmidt procedure is applied and the basis orthogonalized.

5.3 Numerical results

In this section we will compare the POD basis gained from the standard approach and the POD-greedy algorithm.

We use the following parameters:

$$\underline{x} = -5,\ \bar{x} = 5,\ T_{\max} = 2,\ r = 0.03,\ \Delta x = 0.0025,\ \Delta T = 0.01,$$
$$\lambda_{\min} = 0.01, \lambda_{\max} = 5, \Delta\lambda = 0.01, \sigma = 0.3,\ \mu_j = 0,\ \sigma_j = 0.5$$

For the stochastic parameter λ we now have a grid

$$\lambda_{\min} = \lambda_0 < \lambda_1 < \ldots < \lambda_{n_\lambda - 1} < \lambda_{n_\lambda} = \lambda_{\max} \text{ with } \lambda_i = \lambda_0 + i\,\Delta\lambda,\ i = 1, \ldots, n_\lambda.$$

For the POD-greedy algorithm to be efficient, the multiple snapshot set has to be computed beforehand. Using the finite element method with Rannacher smoothing we first calculated the snapshot sets associated with each λ_i and then build the multiple snapshot set. Note that the computational time needed is very high. For each value of λ the full system has to be solved with n_x discretization steps in space and n_t in time, i.e. dimension 3999×201. One calculation takes approximately half a minute, so the construction of the multiple snapshot took over 5 hours. Furthermore note that also the required memory space is high as Y has dimension 3999×100500.

In order to determine the weights in the POD-greedy algorithm we need to assume a distribution for the jump intensity. We compared two assumptions:

1. uniform distribution $\lambda \sim U[\lambda_{\min}, \lambda_{\max}]$
2. normal distribution $\lambda \sim N(0.5, 2)$

For both runs we set the initial basis rank $l = 2$. For the initialization of the POD-greedy algorithm we calculated the POD basis with the snapshot corresponding to the mean of the distribution. As tolerance for the projection error we set $\epsilon^{POD} = 1e-4 \approx \frac{\min\{\Delta T, \Delta x\}}{2} \cdot \min\{w_j | j = 1, \ldots, n_\lambda\}$.

We first discuss the results assuming 1. λ is uniformly distributed. For the initialization $\frac{\lambda_{\min}+\lambda_{\max}}{2} = \lambda_{250} = 2.5$ was used. The weights in the POD-greedy algorithm are determined from the probability density function of the distribution, i.e. $w_j = \frac{1}{\lambda_{\max}-\lambda_{\min}}, j = 1, \ldots, n_\lambda$. With the standard POD approach we needed to set $l = 4$ in order to reach the projection error tolerance. Figure 5.1 displays in the right panel the initial errors using the POD-greedy algorithm and the final errors E_j and in the left panel the errors for the standard POD approach. The minimum error for the initial errors of the POD-greedy and the standard POD approach is at $\lambda = 2.5$. This is only natural as the initial POD basis was calculated with this value. The error grows monotonically with λ diverging from its mean. As λ is uniformly

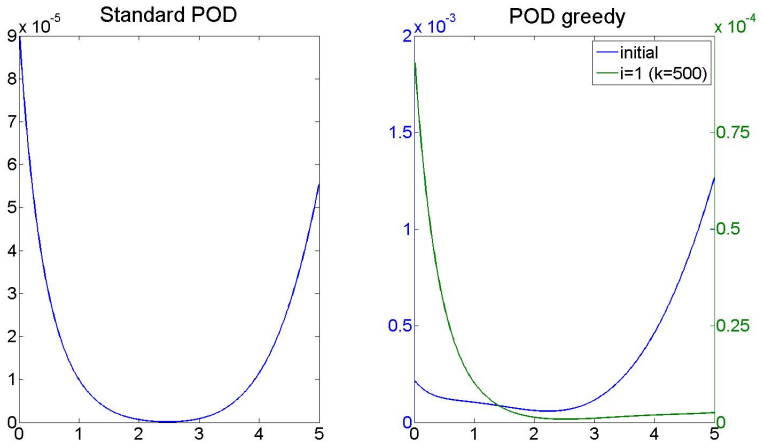


Figure 5.1: Projection errors for the uniform distribution

distributed over $[\lambda_{\min}, \lambda_{\max}]$ a good basis should deliver good approximations in the whole interval. For the POD-greedy approach one iteration, with the maximum error at $k = 500$, was needed to reach the tolerance, resulting in a basis of rank 4. We can see that the error of the final basis is much smoother in $[2, 5]$ than with the standard POD approach. Note that the initial rank of the basis has to be chosen carefully. Increasing l might lead to fewer iterations, but we could also end up with a basis bigger than necessary. Performing the POD-greedy approach with $l = 3$ takes again one iteration, but results with a basis of rank 6.

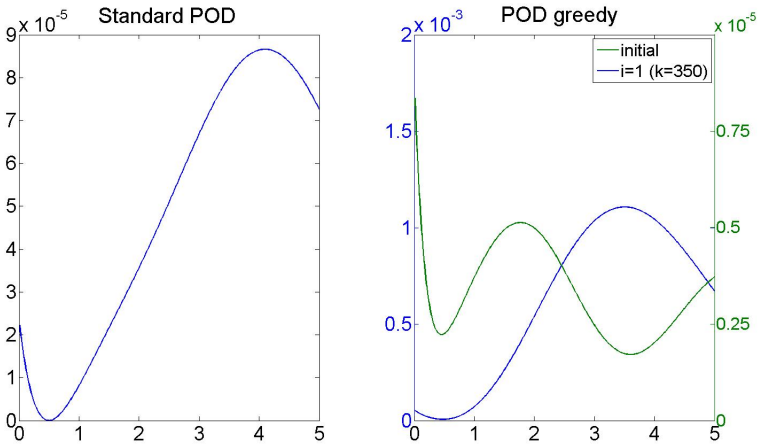


Figure 5.2: Projection errors for the normal distribution

Now we assume 2. λ is normally distributed with mean 0.5 and variance 2. The weights are computed as $w_j = f(\lambda_j)$, $j = 1, \ldots, n_\lambda$, where f is the density function of the normal distribution with mean 0.5 and variance 2 (compare (2.8)). Figure 5.2 is structured the same as Figure 5.1. Here also the minimum error for the initial errors

is at the mean. But since the normal weights are not identical as in the case of the uniform distribution, the POD-greedy basis will be in general computed differently. One iteration was needed, but with now the maximum error at $k = 350$. This yields a basis of rank 4, while with the standard POD approach $l = 6$ basis functions were needed to reach the projection error tolerance. It is clear, that we want to determine the POD basis such, that the projection error is small in a neighborhood of the mean.

After having determined the POD basis (with 1. uniform weights and 2. normally distributed weights), we computed the sample mean
$\mathbb{E}[\arg\min_u \{J(y,u,\zeta \,|\, c(y,u,\zeta) = 0\}]$. For this we need to make an assumption on the distribution of λ. We draw $n_{sim} = 100$ random values of λ from a

a) uniform distribution on $[\lambda_{\min}, \lambda_{\max}]$

b) normal distribution $N(0.5, 2)$

and computed the corresponding weights. For case a) MATLAB offers the command *rand*. In case b) we sampled standard normal distributed random variables $\tilde{\zeta}$ - MATLAB offers the command randn - and transformed these to the normal distribution via $\zeta = 0.5 + 2\tilde{\zeta}$. Then we computed the finite element solution using Rannacher smoothing based on the sampled value. Using each of the 4 determined POD basis, the reduced system was solved for each sampled value of λ and the error

$$ERR_2^{\zeta_j} = \sum_{i=1}^{n} \gamma_i \, ||y_i^{FE,\zeta_j} - y_i^{l,2,\zeta_j}||_W^2, \quad j = 1, \ldots, n_{sim}$$

was computed.

Then the sample mean was calculated via

$$\bar{\mu} = \frac{1}{n_{sim}} \sum_{j=1}^{n_{sim}} w_j \, ERR_2^{\zeta_j}.$$

Error weights	POD-greedy	standard POD
1. uniform	1.07e-5	1.72e-5
2. normal	5.90e-5	4.52e-5

Table 5.1: Sample mean with $n_{sim} = 100$ simulations of a a) uniform distribution using the POD-greedy algorithm and the standard POD method with different weights

Table 5.3 shows the results for case a) uniform distribution. The sample mean was calculated with the POD-greedy basis and the standard POD basis computed

with 1. uniform weights and 2. normal weights. We make several observations. Firstly, as expected the POD-greedy basis computed with normal weights performed worse than the POD-greedy basis computed with uniform weights. When trying to approximate the finite element solution, the POD basis should be computed with the same underlying distribution. Secondly, the POD-greedy algorithm outperforms the standard POD approach when using weights corresponding to the sample mean distribution. Thirdly, we note that when using weights other than from the sample mean distribution, the standard POD approach provides a smaller error than the POD-greedy method.

Error weights	POD-greedy	standard POD
1. uniform	1.59e-5	1.56e-5
2. normal	6.06e-6	1.91e-5

Table 5.2: Sample mean with $n_{sim} = 100$ simulations of a b) normal distribution using the POD-greedy algorithm and the standard POD method with different weights

Table 5.3 is structured the same way. Here a normal distributed was simulated. It gives the same findings as the uniform distribution were.

To show the differences between different values of the initial basis size l, we run the POD-greedy algorithm again with now a lower projection error tolerance $\epsilon^{POD} = 5e-7 \approx \frac{\min\{\Delta T, \Delta x\}^2}{2} \cdot \min\{w_j | j = 1, \ldots, n_\lambda\}$.

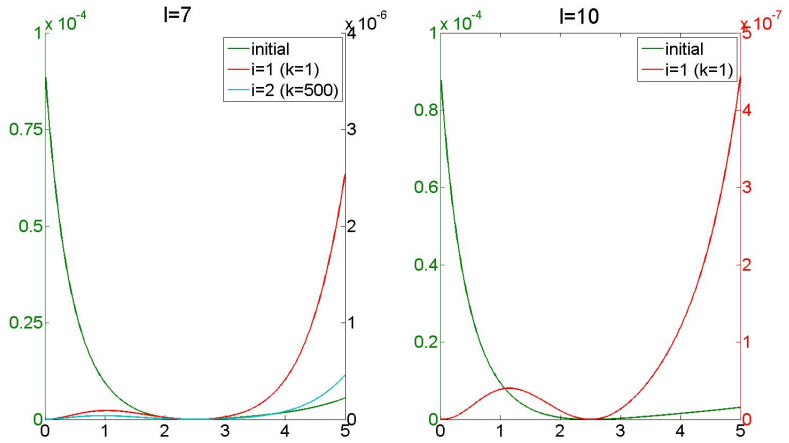


Figure 5.3: Projection errors of the POD-greedy algorithm for the uniform distribution for different initial basis rank sizes l

Figure 5.3 displays the errors in each iteration for $l = 7$ (left panel) and $l = 10$ (right panel). We can see that for the uniform distribution the maximum error is obtained at the same λ_j. With 7 initial basis functions two iterations were needed,

resulting in a basis of rank 21. Using $l = 8, 9$ requires also two iterations yielding basis of bigger size. Hence $l = 10$ is the preferred choice of the initial basis rank as only one iteration is needed, yielding a POD-greedy basis of rank 20.

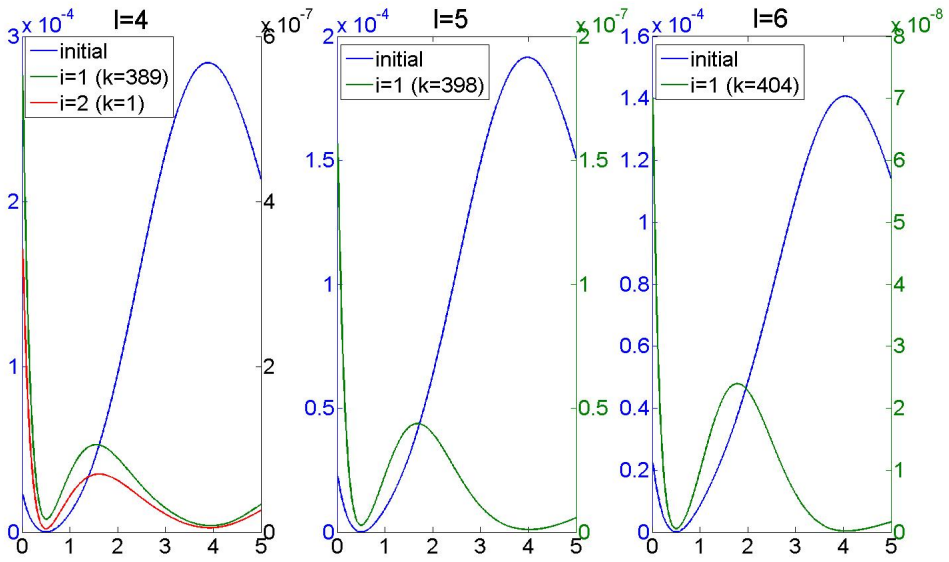


Figure 5.4: Projection errors of the POD-greedy algorithm for the normal distribution for different initial basis rank sizes l

For the normal distribution we compared initial basis sizes $l = 4, 5, 6$. In Figure 5.4 we can see that through the not equal normal weights, the maximum error is obtained for different values of λ, even though they are close together for all three basis sizes. With $l = 4$ three iterations are needed and the final basis has rank 12. Using $l = 5$ and $l = 6$ two iterations are needed yielding a basis of rank 10 and 12 respectively. Hence $l = 5$ is the preferred choice in both computing time and basis rank size.

The initial rank parameter has to be chosen with care considering the computing time of the POD-greedy algorithm it defines and the final size of the basis.

Chapter 6

Conclusion

After giving an introduction to the financial concept of options, we presented as first option pricing model the famous Black-Scholes-model. Here we showed the concept of risk-neutral pricing. The pure diffusion model was a major breakthrough in option pricing theory, but is known to suffer from several drawbacks. Not incorporating the volatility skew should be mentioned here. As an extension to the Black-Scholes model Merton's jump diffusion model was derived. Adding jumps to the simple diffusion process allows for producing volatility skews. Also a semi-analytical solution is available. As our model of choice we equipped the jump diffusion model with a deterministic local volatility function. This makes precise fitting to quoted market prices possible.

For pricing purposes we derived a partial-integro differential equation. Then a variational formulation of the PIDE was established as well as well-posedness and uniqueness of a solution. Here we had to work in weighted Sobolev spaces in order to account for the non-L^2-integrable initial condition arising with European call options.

A major part of this thesis dealt with the numerical solution of the localized problem. First, we applied a Galerkin approach. Via finite elements the spatial variable was discretized. The time variable was discretized using Crank-Nicolson scheme, where we replaced the first four time steps with four quarter implicit Euler time steps to smooth out oscillations. The arising dense linear systems of equations were solved with a preconditioned GMRES-algorithm. Even though the numerical results showed the quadratic convergence in both time and space, the computational cost is still high.

Thus in another important part of the thesis we outlined reduced-order modeling via proper-orthogonal decomposition. Through replacing the n_x basis functions spanning the finite element space$H_0^1(\Omega)$ by only a few POD basis functions we

are able to reduce the computational effort enormously. A-priori estimates for the projection error were furthermore derived. The numerical experiment at the end of Section 5.3 shows the efficiency of the implemented algorithm.

Further numerical approaches included the treatment of λ as a random variable. For different values of the jump intensity snapshots were computed and a multiple snapshot set was build. Based on this a POD-greedy algorithm was applied. Calculations showed that for varying λ the POD basis derived from the greedy algorithm beats the standard POD method in terms of the projection error.

Further applications include model calibration as done in [34] and implementation and analysis of strategy three in (5.2), where the sample mean is minimized.

List of Figures

List of Tables

Bibliography

[1] Andersen, L. and Andreasen, J. (2000). *Jump-diffusion processes: Volatility smile fitting and numerical methods for option pricing.* Review of Derivatives Research 4(3): 231–262.

[2] Antoulas, A. (2005). *Approximation of Large-Scale Dynamical Systems, Advances in Design and Control.* SIAM.

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

[4] Borzi, A., Schulz, V. , Schillings, C. and von Winckel, G. (2010). *On the treatment of distributed uncertainties in PDE constrained optimization.* GAMM-Mitt. 33 (2): 230–246.

[5] Carr, P. and Wu, L. (2003). *What type of process underlies options? A simple robust test.* Journal of Finance, 58:2581–2610.

[6] Cheang, G. H. L. and Chiarella, C. (2011). *A Modern View on Merton's Jump-Diffusion Model.* Quantitative Finance Research Centre, University of Technology Sydney

[7] Cheney, W. and Kincaid, D. R. (2004). *Numerical Methods and Computing.* Brooks/Cole.

[8] Cont, R. and Tankov, P. (2004). *Financial Modelling with Jump Processes.* Chapman & Hall.

[9] Dautray, R. and Lions, J.-L. (1992). *Mathematical Analysis and Numerical Methods for Science and Technology, Vol.5: Evolution Problems I.* Springer.

[10] Derman, E. and Kani, I. (1994). *The Volatility Smile and its Implied Tree.* Quantitative Strategies Research Notes, Goldman Sachs.

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

[12] Eraker, B. (2004). *Do Stock Prices and Volatility Jump? Reconciling Evidence from Spot and Option Prices.* The Journal of Finance, 59(3): 1367–1404.

[13] Fama, E. F. (1965). *The Behavior of Stock-Market Prices* . The Journal of Business, 38(1): 34-105.

[14] Foellmer, H. and Schied, A. (2004). *Stochastic Finance: An Introduction in Discrete Time.* De Gruyter studies in mathematics, Walter de Gruyter.

[15] Giles, M. and Carter, R. (2006). *Convergence analysis of Crank-Nicolson and Rannacher time-marching.* Journal of Computational Finance 9(4): 89–112.

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

[17] Holmes, P., Lumley, J., Berkooz, G. and Rowley, C. W. (2012). *Turbulence, Coherent Structures, Dynamical Systems and Symmetry.* 2nd edn, Cambridge University Press.

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

[19] Hull, J. C. (2012). *Options, Futures and other Derivatives.* 8th edn, Pearson Prentice Hall.

[20] Kelley, C. T. (1995). *Iterative Methods for Linear and Nonlinear Equations.* Society for Industrial and Applied Mathematics.

[21] Lass, O. (2014). *Reduced order modeling and parameter identification for coupled nonlinear PDE systems.* Dissertation, University of Konstanz

[22] Lu, J. (2014). *Affine POD Galerkin schemes for option-pricing in jump-diffusion models.* Master thesis, University of Konstanz

[23] Matache, A.-M., von Petersdorff, T. and Schwab, C. (2004). *Fast deterministic pricing of options on Lévy driven assets.* Mathematical Modelling and Numerical Analysis 38(1): 37–72.

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

[25] Pascucci, A. (2011). *PDE and Martingale Methods in Option Pricing* Springer Verlag.

[26] Prestel, A. (2005). *Linear Algebra I.* Lecture Notes, University of Konstanz

[27] Pyzara, Bylina, B. and Bylina, J. (2011). *The influence of a matrix condition number on iterative methods' convergence.* Computer Science and Information Systems (FedCSIS), Federated Conference on. IEEE 459–464.

[28] Quarteroni, A., Rozza, G. (2014). *Reduced Order Methods for Modeling and Computational Reduction.* Springer Modeling, Simulation and Applications 9: 235-274.

[29] Rannacher, R. (1984). *Finite element solution of diffusion problems with irregular data.* Numerische Mathematik 43: 309–327.

[30] Saad, Y. (2003). *Iterative Methods for Sparse Linear Systems.* 2nd edn, Society for Industrial and Applied Mathematics.

[31] Saad, Y. and Schultz, M. H. (1986). *GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems.* SIAM Journal on Scientific and Statistical Computing 7(3): 856–869.

[32] Schilders, W.H., van der Vorst, H. A., Rommes, J. (2008). *Model order reduction:Theory, Research Aspects and Applications.* Springer Mathematics in Industry Volume 13.

[33] Singler,J.R (2014). New POD expressions, error bounds, and asymptotic results for reduced order models of parabolic PDEs. SIAM J. Numer. Anal. 52(2): 852-876.

[34] Schu, M. (2012). *Adaptive trust-region POD methods and their application in finance.* Ph.D. Thesis, University of Trier.

[35] Sachs, E. W. and Schu, M. (2013). *A priori error estimates for reduced order models in finance.* ESAIM: Mathematical Modelling and Numerical Analysis 47: 449–469.

[36] Volkwein, S. (2013). *Proper Orthogonal Decomposition: Theory and Reduced-order Modelling.* Lecture Notes, University of Konstanz