Technische Universität München
Zentrum Mathematik

Applications of Least-Squares Regressions to Pricing and Hedging of Financial Derivatives

Andreas J. Grau

Vollständiger Abdruck der von der Fakultät für Mathematik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzender:		Univ.-Prof. Dr. Bernd Simeon
Prüfer der Dissertation:	1.	Univ.-Prof. Dr. Rudi Zagst
	2.	Prof. Phelim P. Boyle, Ph.D. (em.), Wilfrid Laurier University, Waterloo, Kanada, (nur schriftliche Beurteilung)
	3.	Univ.-Prof. Dr. Hans-Joachim Bungartz

Die Dissertation wurde am 12.12.2007 bei der Technischen Universität eingereicht und durch die Fakultät für Mathematik am 05.02.2008 angenommen.

Acknowledgements

This thesis would not have been possible without the support of numerous people. First, I would like to thank my supervisor Prof. Dr. Rudi Zagst for assigning challenging tasks to me, discussing my ideas patiently and providing assistance in order to making this thesis readable. He even left me enough room for my unconventional ideas. I thank Prof. Dr. Peter Forsyth, Prof. Dr. Ken Vetzal and Prof. Dr. Jan Kallsen for discussions with valuable input and thought-provoking impulses. This is especially true for my co-supervisor Prof. Dr. Phelim Boyle and the third referee of this thesis Prof. Dr. Hans Bungartz.

As well, I would like to thank my collegues Dr. Stefan Dirnstorfer, Christina Niethammer and Christoph Hänle for making the time working on my dissertation enjoyable.

Last, but not least, a special thanks goes to my parents and my brother for their constant full support from back home.

Contents

Introduction

The introduction of financial options delivered a valuable contribution to the efficiency of the markets in the world. Investors seeking risks - speculators - can use financial options to obtain large effects with little money. Investors avoiding risks - hedgers - can now buy insurances for their portfolios at reasonable prices. In book 1, Chapter 11 of Politics, Aristotle already tells the story of Thales of Miletus (624-547 BC) basically buying an option on olive crop. But, it took until the 1970s where large volumes of financial options were traded at derivatives exchanges. Today, the underlying problem of pricing and hedging options is well known and several approaches of its solution have been proposed. Assuming a simple complete market without any transaction cost, the Black-Scholes model has been most successful since its introduction 1973 [17].

Despite the beauty and simplicity of the Black-Scholes model, the efficient evaluation of many exotic options remains challenging. It turned out quickly that analytic solutions e.g. from Merton [87] are by far not sufficient for the evaluation of traded securities. Consequently, a large variety of procedures has been developed for the solution of the governing partial differential Equation (PDE). Direct solvers are e.g. a finite differences method by Schwartz [103], the finite element method and the finite volume methods by Forsyth and Vetzal [50] resp. Zvan et al [124] as well as a mesh-less method by Li et al [80]. A popular solver is the Cox-Ross-Rubinstein method (CRR) [36], which discretizes the asset price process by a binomial tree and solves for the option price by a simple recursion, which is easy to implement. But, the CRR method does not have as good convergence properties as the other PDE methods.

A different approach solving for option prices in the Black-Scholes model focuses on the underlying stochastic differential equation (SDE). This is done by simulating Monte Carlo paths of the underlying asset and computing option prices as some expected value, presented first by Boyle [21, 23, 53]. While option features such as an early exercise can easily be evaluated in a PDE solver (cp. Forsyth and Vetzal [50]), this is hard for Monte Carlo methods. Carrière [32] presented the first practical Monte Carlo method for the valuation of options with early exercise features in 1996, which became popular after being extended by Longstaff and Schwartz [81] in 2001. This method is called Least-Squares Monte Carlo.

That means the main methods for option valuation are PDE solvers and Monte Carlo simulation. On the one hand, for many pricing problems PDE solvers deliver highly accurate solutions

in little time. This is especially true for low-dimensional pricing problems. But, not all of them are low-dimensional. Especially path-dependent options often require the introduction of additional state variables. Reaching four or five dimensions, the pricing becomes usually infeasible for current PDE and computer technology. On the other hand, Monte Carlo methods can price options independent of the dimension of the pricing problem. But, these methods are converging slowly such that highly accurate solutions often cannot be obtained. Additionally, the valuation of high-dimensional financial derivatives with embedded options like an early exercise is still challenging, even if the method of Carrièr [32, 81] is used.

This work will focus on pricing and hedging of derivatives with Monte Carlo simulation. In some cases, direct numerical PDE solutions will be used as a reference. We will provide insight into the versatile applications of regression methods for the Monte Carlo valuation. As a result, very fast valuation procedures are developed: In some cases the methods developed in this dissertation are the first of its kind which handle specific exotic options. Especially the pricing of a high-dimensional Moving Window Asian option with early exercise and the implementation of a moving window soft-call constraint of convertible bonds are solved for the first time in this thesis.

Prior technology could not cope with the high-dimensional pricing problem together with an early exercise feature. The PDE method can deal with an early exercise feature easily, but high-dimensional problems are unfeasible. Monte Carlo methods can deal with high-dimensional problems, but an early exercise of a high-dimensional option pricing problem is hard to treat correctly in the previous setting.

Another contribution of this thesis is the Simulation-Based Hedging method which connects realistic models for the underlying with suitable pricing and hedging without a detour to a so-called risk-neutral measure. The Simulation-Based Hedging has extraordinary properties: E.g. using the Black-Scholes assumptions its convergence to the Black-Scholes prices is much faster than the comparable Longstaff-Schwartz Least-Squares Monte Carlo [81]. Furthermore, the underlying can follow any real-world process: The algorithm always computes the optimal hedging strategy and thus attains realistic risk-adjusted prices and hedges. This can also be done using multiple hedge instruments.

Consequently, the new Simulation-Based Hedging is a new pricing framework together with a numerical method for the solution to option pricing problems in so called incomplete markets. The whole setting of the framework is new, but related to risk minimization techniques for optimal hedging of financial options presented by several authors [46, 95, 47, 33]). Especially, the setting of Simulation-Based Hedging can be seen as an extension to the variance minimization presented by Schweizer [104] and the presented numerical solution is related to a method presented by Potters et. al. [96] resp. Pochart and Bouchaud [95].

The main results of this dissertation are:

- The usual Monte Carlo method is altered for a quicker evaluation: Extracting the main features of the option's payoff, a simple regression can accelerate the evaluation of path

dependent derivatives significantly.

- A sparse basis for Least-Squares Monte Carlo is presented which allows to price Moving Window Asian Options for the first time. This method is extended to the evaluation of convertible bonds with complex rights of holders and issuers.
- A powerful Simulation-Based Hedging method has been developed, which determines option prices and optimal hedges based on physical simulations of the underlying. This method is an order of magnitude faster than the state of the art Least-Squares Monte Carlo and can operate with much less restrictive assumptions on the market than the widely used Black-Scholes model.

Chapter 1

Mathematical Foundations

1.1 Overview

This chapter summarizes the main mathematical tools needed for the pricing and hedging of financial derivatives. Since this thesis focuses on regression methods, we first define what kind of regression is meant and which properties of the method are required. Then, the different choices of regression basis functions are presented. After summarizing the basics for derivatives pricing, the chapter closes with the corresponding numerical implementation.

1.2 Regression Methods

In this section, we review the mathematical properties of different regression methods. We focus on least-squares regressions due to their desirable properties and extend the regression basis to special sparse basis functions in order obtain computational feasible methods for high-dimensional regressions.

1.2.1 Basics

Before we start with the actual topic on regression methods we need to define some terminology. The function $f(x)$ is said to be of class $\mathcal{C}^k$ if the derivatives $\frac{\mathrm{d}f}{\mathrm{d}x}$, $\frac{\mathrm{d}^2 f}{\mathrm{d}x^2}, \ldots, \frac{\mathrm{d}^k f}{\mathrm{d}x^k}$ exist and are continuous. The function $f(x)$ is said to be of class $\mathcal{C}^0$, if it is continuous. The function $f(x)$ is said to be of class $\mathcal{C}^\infty$, or **smooth**, if it has derivatives of all orders.

Approximation by Regression

In the following, we want to show what kind of regressions are useful in the context of option pricing and which properties they have. There are two main applications of the regression for option pricing: One is the function approximation, the other is a variance minimization of a portfolio.

First, we start with the function approximation. Therefore we need to define our setting and what we mean by an approximation.

Assumption 1 A data set $(\mathbf{X}, \mathbf{y})$, $\mathbf{X} \in \mathbb{R}^{n,s}$, $\mathbf{y} \in \mathbb{R}^n$ is provided.

Assumption 2 The rows $\mathbf{x}^i \in \mathbb{R}^s$ of $\mathbf{X} := (\mathbf{x}^1, \ldots, \mathbf{x}^n)^T$ are independent and identically distributed (i.i.d.) realizations of a random vector with a probability density function $p(\mathbf{x})$, which is non-zero everywhere on the cube $\mathbb{D} := [\mathbf{x}_{\min}, \mathbf{x}_{\max}] = ([x_{j,\min}, x_{j,\max}])_{j=1,\ldots,s}$ and zero outside.

Assumption 3 The provided values of $\mathbf{y} = (y^1, \ldots, y^n)^T$ are noisy observations of $f(\mathbf{x}^i)$, $i = 1 \ldots, n$, with

$$y^i = f(\mathbf{x}^i) + \epsilon^i, \; i = 1 \ldots, n$$

where ϵ^i is random with $\mathbb{E}[\epsilon^i] = 0$, independent of $\mathbf{x}^i$.

Assumption 4 The function $f : \mathbb{R}^s \to \mathbb{R}$, $f \in \mathcal{B}$ has a representation $f(\mathbf{x}) = \sum_{j=1}^{\infty} a_j b_j(\mathbf{x})$, $\mathbf{x} \in \mathbb{R}^s$, where $b_j \in \mathcal{B}$, $j = 1, \ldots, \infty$ are bounded basis functions $b_j : \mathbb{R}^s \to \mathbb{R}$ of a vector space $\mathcal{B} \subset \mathcal{C}^1$ with $\|b_j(\mathbf{x})\|_\infty = c_j < \infty$, $\exists \mathbf{x} \in \mathbb{D} : |b_j(\mathbf{x})| > 0$ $j = 1, \ldots, \infty$.

Assumption 1 is clear. Assumption 2 explains that there is one stochastic variable which determines the value $\mathbf{x}^i$ at which a function is evaluated, while Assumption 3 explains that the function value plus a random noise is denoted by y^i. Now, Assumption 4 contains a decomposition of function f into basis functions with a continuous total derivative such that an approximation can be defined properly. Note that Assumption 4 does not impose a restriction on a numerical evaluation using a subset of the basis functions since e.g. any continuous function can be uniformly approximated by polynomials (Stone-Weierstrass Theorem [110]).

Theorem 1.1 *Let Assumptions 1 to 4 be satisfied. Then the mapping given by*

$$< b, f >_r := \int_{\mathbb{D}} b(\mathbf{x}) f(\mathbf{x}) p(\mathbf{x}) \, d\mathbf{x}, \;\; \mathbf{x} \in \mathbb{R}^s \tag{1.1}$$

with functions $b(\mathbf{x}), f(\mathbf{x}) \in \mathcal{B}$ and probability density function $p(\mathbf{x})$ is a scalar product on $\mathcal{B}$ and thus $\mathcal{B}$ is a Euclidian vector space, i.e. a real vector space $\mathcal{B}$ with a corresponding definition of a scalar product.

Proof We can prove Theorem 1.1 simply by comparing the conditions for a scalar product with the corresponding expressions. This is straight forward, such that we omit the details. □

The next thing we need is a stochastic approximation of this scalar product, which we provide by the following theorem.

Theorem 1.2 *Let Assumptions 1 to 4 be satisfied. Then*

$$\lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} f(\mathbf{x}^i) b_j(\mathbf{x}^i) =< f, b >_r$$

holds.

Proof With $\mathbf{x} := (x_1 \ldots, x_s) \in \mathbb{D}$, $\mathbf{x}^i := (x_1^i \ldots, x_s^i) \in \mathbb{D},\ i = 1, \ldots, n$ and indicator function

$$I_{\mathbf{x}}(\mathbf{x}^i) := \begin{cases} 1 & \text{if } x_j^i < x_j \quad \forall\ j = 1, \ldots, s \\ 0 & \text{else} \end{cases}$$

we define the empirical cumulative distribution function $F_n(\mathbf{x})$ of n sample observations $\mathbf{x}^i$ as

$$F_n(\mathbf{x}) := \frac{1}{n} \sum_{i=1}^{n} I_{\mathbf{x}}(\mathbf{x}^i), \mathbf{x} \in \mathbb{D},$$

which means that

$$\int_{\mathbb{D}} f b_j \, \mathrm{d}F_n = \frac{1}{n} \sum_{i=1}^{n} f(\mathbf{x}^i) b_j(\mathbf{x}^i).$$

From the Glivenko-Cantelli Theorem[1] we know that $F_n(\mathbf{x}) \to F(\mathbf{x})$ with true cumulative distribution function $F(\mathbf{x})$ almost surely and uniformly, i.e.

$$\lim_{n\to\infty} \int_{\mathbb{D}} f b_j \, \mathrm{d}F_n \to \int_{\mathbb{D}} f b_j \, \mathrm{d}F$$

holds for all integrants which are bounded and continuous in the domain $\mathbb{D}$. The integrand $f(\mathbf{x}^i) b_j(\mathbf{x}^i)$ is continuous by Assumption 4 and since the integration domain $\mathbb{D}$ is bounded (Assumption 2), $f(\mathbf{x}^i) b_j(\mathbf{x}^i)$ is bounded everywhere on the domain $\mathbb{D}$. Since $p(\mathbf{x})$ is the total first derivative of the cumulative distribution function $F(\mathbf{x})$,

$$\int_{\mathbb{D}} f b_j \, \mathrm{d}F = \int_{\mathbb{D}} b(\mathbf{x}) f(\mathbf{x}) p(\mathbf{x}) \, \mathrm{d}\mathbf{x} =< b, f >_r$$

holds, which completes the proof. □

We defined an approximation for the scalar product, but our goal is to obtain an approximation $\tilde{f}(\mathbf{x}) \approx f(\mathbf{x})$ for any $\mathbf{x}$ in $[\mathbf{x}_{\min}, \mathbf{x}_{\max}]$ based on the set of noisy observations $(\mathbf{X}, \mathbf{y})$. That means, we first have to define what we mean by an approximation.

Definition 1.3 *Let Assumptions 1 to 4 be satisfied. A* ***local basis approximation*** $\tilde{f}^m$ *of the function f induced by the set of samples $(\mathbf{X}, \mathbf{y}),\ \mathbf{X} \in \mathbb{R}^{n,s},\ \mathbf{y} \in \mathbb{R}^n$ with function space $\mathcal{B}^m$ spanned by the basis functions $b_1, \ldots, b_m \in \mathcal{B}$, is given by*

$$\tilde{f}^m(\mathbf{x}) = \sum_{j=1}^{m} \tilde{a}_j^n b_j(\mathbf{x}), \quad \tilde{f}^m \in \mathcal{B}^m \subset \mathcal{B}, \quad \tilde{a}_j^n \in \mathbb{R}, j = 1, \ldots, m$$

[1] See Fahrmeier et al [43], p.315.

with coefficient vector $\tilde{\mathbf{a}}^m = A(\mathbf{X}, \mathbf{y})$, $\tilde{\mathbf{a}}^m = (\tilde{a}_1^n, \ldots, \tilde{a}_m^n)^T$ *iff*

$$\forall \varepsilon \ \exists N(\varepsilon) : \left\| \begin{pmatrix} a_1 \\ \vdots \\ a_m \end{pmatrix} - \begin{pmatrix} \tilde{a}_1^n \\ \vdots \\ \tilde{a}_m^n \end{pmatrix} \right\|_\infty < \varepsilon, \ \ \forall n \geq N(\varepsilon).$$

In the following, we state how one can obtain a suitable function $A(\mathbf{X}, \mathbf{y}))$, which determines the coefficient vector $\tilde{\mathbf{a}}^m$ given the noisy observations and the basis functions of interest $b_j(\mathbf{x}) \in \mathcal{B}^m, j = 1, \ldots, m$, so that we obtain the local basis approximation.

Theorem 1.4 *Let the Assumptions 1 to 4 be satisfied. The* ***local basis approximation*** $\tilde{f}^m(\cdot)$ *of function* $f(\cdot)$ *based on a set* $(\mathbf{X}, \mathbf{y})$ *of* n *noisy observations is given by*

$$\tilde{f}^m(\mathbf{x}) = \sum_{j=1}^{m} \tilde{a}_j^n b_j(\mathbf{x})$$

with $\tilde{\mathbf{a}}^m = (\tilde{a}_1^n, \ldots, \tilde{a}_m^n)^T$ *where*

$$\begin{aligned} \tilde{\mathbf{a}}^m = A(\mathbf{X}, \mathbf{y})) &= \arg\min_{\tilde{\mathbf{a}}^m} \|\mathbf{B}(\mathbf{X})\tilde{\mathbf{a}}^m - \mathbf{y}\|_2 \\ &= \left(\mathbf{B}(\mathbf{X})^T \mathbf{B}(\mathbf{X})\right)^{-1} \mathbf{B}(\mathbf{X})^T \mathbf{y} \end{aligned} \tag{1.2}$$

with

$$\mathbf{B}(\mathbf{X}) := \begin{pmatrix} b_1(\mathbf{x}^1) & \cdots & b_m(\mathbf{x}^1) \\ \vdots & \ddots & \vdots \\ b_1(\mathbf{x}^n) & \cdots & b_m(\mathbf{x}^n) \end{pmatrix}.$$

See Appendix 7.2 for details of the proof.

Lemma 1.5 *Let the Assumptions 1 to 4 be satisfied and* $\tilde{f}^m(\mathbf{x})$ *be defined as in Theorem 1.4. Then,*

$$\lim_{m,n \to \infty} \tilde{f}^m(\mathbf{x}) = \mathbb{E}[y|\mathbf{x}].$$

Proof The order of the limits $n \to \infty$ and $m \to \infty$ is important:

$$\begin{aligned} \lim_{m\to\infty} \left(\lim_{n\to\infty} \tilde{f}^m(\mathbf{x}) \right) &= \lim_{m\to\infty} \left(\lim_{n\to\infty} \sum_{j=1}^{m} \tilde{a}_j^n b_j(\mathbf{x}) \right) = \lim_{m\to\infty} \left(\sum_{j=1}^{m} a_j b_j(\mathbf{x}) \right) \\ &= f(\mathbf{x}) = \mathbb{E}\left[f(\mathbf{x})|\mathbf{x}\right] = \mathbb{E}\left[y - \epsilon|\mathbf{x}\right] \\ &= \mathbb{E}\left[y|\mathbf{x}\right], \end{aligned}$$

since $\mathbb{E}[\epsilon|x] = 0$.

□

Note: In a real application, the quotient $\frac{n^\alpha}{m}$ should be constant in the limiting process; the optimal exponent α depends on the smoothness of f and dimension s.[2]

[2]For a detailed proof and optimal exponent see Stentoft [108].

Variance Minimization by Regression

To get a better understanding of what the local basis approximation can do for a variance minimization, we look at an idea which dates back to 1979 when Ederington [42] showed that a static minimum variance hedge ratio is simply defined as the ratio of the covariance of V and S to the variance of S.[3]

Ederington argues that

$$\beta := \frac{\mathrm{cov}(S,V)}{\mathrm{var}(S)} \tag{1.3}$$

is the optimal hedge for V with some correlated underlying S in a single period market.

Our approach can be seen as a generalization to this idea to $\beta(\mathbf{x})$ as a function of some underlying state $\mathbf{x} \in \mathbb{R}^s$, such that optimal hedges for multi-period markets can be computed. That means, we allow the optimal hedge to be conditional on the current state of the world $\mathbf{x}$, and thus we can compute optimal *dynamic* hedges in the following Chapters.

Given the random variables $\hat{S}$, $\hat{V}$ with $E[\hat{S}|\mathbf{x}] = 0$ and $E[\hat{V}|\mathbf{x}] = 0$, dependent on state $\mathbf{x}$, the function

$$\beta(\mathbf{x}) := \frac{\mathrm{cov}(\hat{S},\hat{V}|\mathbf{x})}{\mathrm{var}(\hat{S}|\mathbf{x})} \tag{1.4}$$

shall be estimated from a sample $\{\hat{S}^i, \hat{V}^i, \mathbf{x}^i\}, i \in \{1,\dots,n\}$. From linear regressions of stochastic variables we know that the definition of Equation (1.4) is the solution to the minimization of[4]

$$\frac{1}{n}\sum_{i=1}^{n}(\hat{V}^i - \beta(\mathbf{x}^i)\hat{S}^i)^2, \tag{1.5}$$

which we can write with some basis functions $b_j(\mathbf{x}^i), j = \{1,\dots m\}$ and $\beta(\mathbf{x}^i) = \sum_j \tilde{a}_j b_j(\mathbf{x}^i)$ as

$$\{\tilde{a}_j\} = \arg\min_{\tilde{a}_j} \left\| \begin{pmatrix} b_1(\mathbf{x}^1)\hat{S}^1 & \cdots & b_m(\mathbf{x}^1)\hat{S}^1 \\ \vdots & \ddots & \vdots \\ b_1(\mathbf{x}^n)\hat{S}^n & \cdots & b_m(\mathbf{x}^n)\hat{S}^n \end{pmatrix} \begin{pmatrix} \tilde{a}_1 \\ \vdots \\ \tilde{a}_m \end{pmatrix} - \begin{pmatrix} \hat{V}^1 \\ \vdots \\ \hat{V}^n \end{pmatrix} \right\|. \tag{1.6}$$

Standard arguments as in the previous section show that with

$$\mathbf{B}_S(\mathbf{x}^1,\dots,\mathbf{x}^n) := \begin{pmatrix} b_1(\mathbf{x}^1)\hat{S}^1 & \cdots & b_m(\mathbf{x}^1)\hat{S}^1 \\ \vdots & \ddots & \vdots \\ b_1(\mathbf{x}^n)\hat{S}^n & \cdots & b_m(\mathbf{x}^n)\hat{S}^n \end{pmatrix}$$

the values $\tilde{a}_j$ can be computed using efficient algorithms which implicitly solve the normal equations

$$\mathbf{B}_S(\mathbf{X})^T\mathbf{B}_S(\mathbf{X})\tilde{\mathbf{a}} = \mathbf{B}_S(\mathbf{X})^T\hat{\mathbf{V}} \tag{1.7}$$

[3] Ederington's example is the hedge of a future but it applies to any derivative V. For more recent research on futures hedges see e.g. Allen et al [3].

[4] Cp. [102]

with $\tilde{\mathbf{a}} = (\tilde{a}_1, \ldots, \tilde{a}_m)^T$ and $\hat{\mathbf{V}} = (\hat{V}^1, \ldots, \hat{V}^n)^T$. Then,

$$\beta(\mathbf{x}^i) = \frac{\operatorname{cov}(\hat{S}, \hat{V}|\mathbf{x} = \mathbf{x}^i)}{\operatorname{var}(\hat{S}|\mathbf{x} = \mathbf{x}^i)} \approx \sum_{j=1}^{m} \tilde{a}_j b_j(\mathbf{x}^i)$$

is the desired result which denotes the optimal hedge ratio based on some state $\mathbf{x}^i$. This result equals the previous result using Theorem 1.4 if we set $\hat{S} = S_{j+1} - E[S_{j+1}|\mathbf{x}_j]$ and $\hat{V} = V_{j+1} - E[V_{j+1}|\mathbf{x}_j]$ such that $E[\hat{S}|\mathbf{x}_j] = 0$ and $E[\hat{V}|\mathbf{x}_j] = 0$ holds and a local basis approximation of $\hat{V}$ by $\hat{S}$ is conducted. In implementations, $E[V_{j+1}|\mathbf{x}_j]$ and $E[S_{j+1}|\mathbf{x}_j]$ can be obtained from the corresponding local basis approximation of V_{j+1} and S_{j+1} as defined in Theorem 1.4.[5]

In total, we obtained a simple method for computing a conditional variance minimization of a portfolio based on regressions. The resulting Equation (1.7) is very similar to Equation (1.2) of Theorem 1.4. Consequently, this method itself can be seen as a local basis approximation of the option V by the underlying asset price S.

The current literature on non-parametric statistics solves slightly different problems, but we still want the refer to this research: A conditional functional principal components analysis is proposed by Cardot [31], who presents the computation of conditional covariances based on kernel smoothers as well as some convergence properties of the approach. Within the GARCH framework, conditional variance functions are computed by Fan and Yao [44]. They perform regressions on squared residuals and they obtain the asymptotic convergence result that without knowing the regression function, their method estimates the conditional variance as well as if the regression functions were given. Finally, non-parametric regressions are presented by several authors. Especially Härdle [60] provides a good overview of different regression and smoothing techniques.

Numerical Solution to the Least-Squares Minimization

Now, we want to summarize how the normal Equations ((1.2) and (1.7)) are solved efficiently. The direct solution by computation of

$$\tilde{\mathbf{a}} = \left(\mathbf{B}^T\mathbf{B}\right)^{-1}\mathbf{B}^T\mathbf{y}$$

leads to an unstable solution. In the following, we will provide some insight into the reasons for this. For more rigorous derivations and detailed error analysis see Higham [62] and the references therein as well as Voss [114].

First of all, we need a definition and a theorem, which allows us to write the solution of the minimization problem efficiently.

[5] This approach with separate estimations of $E[V_{j+1}|\mathbf{x}_j]$, $E[S_{j+1}|\mathbf{x}_j]$ and $\beta(\mathbf{x}^i)$ can be unified into a single step with twice as many basis functions. But, the unification is not useful in practice. The numerical algorithm for estimating the basis coefficients $\mathbf{a}$ is $\approx m^3$ with m denoting the number of basis functions. Estimating the conditional expectation using the local basis approximation first, subtracting the expectation and estimating the coefficients for $\beta(\mathbf{x})$ with $m > 1$ leads to $\approx 3 \cdot m^3$ which is less than $(2m)^3$ for the joint estimation.

Definition 1.6 *Let* $\mathbf{B} \in \mathbb{R}^{n,m}$. *Then the matrix* $\mathbf{B}^\dagger \in \mathbb{R}^{m,n}$ *which is determined by delivering the solution to the minimization problem* $\|\mathbf{Ba} - \mathbf{y}\|_2 = \min!$ *written as* $\mathbf{a} = \mathbf{B}^\dagger\mathbf{y}$ *is called* ***pseudo inverse*** *of* $\mathbf{B}$.

The next theorem tells us, how to obtain the pseudo inverse:

Theorem 1.7 *Let* $\mathbf{B} \in \mathbb{R}^{n,m}$ *have a singular value decomposition*

$$\mathbf{B} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T,$$

where $\mathbf{\Sigma}$ *is a diagonal matrix with the sorted singular values* $\sigma_1 \geq \sigma_2, \ldots, \geq \sigma_m > 0$ *on the diagonal and the orthonormal vectors* $\mathbf{u}^i$, $\mathbf{v}^i$ *as columns in* $\mathbf{U}$ *resp.* $\mathbf{V}$. *Then,*

- $\mathbf{\Sigma}^\dagger = (\sigma_i^{-1}\delta_{i,j}), \delta_{ij} := \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$
- $\mathbf{B}^\dagger = \mathbf{V}\mathbf{\Sigma}^\dagger\mathbf{U}^T$,

where $\mathbf{B}^\dagger$ *is the pseudo inverse.*

Proof A proof of Theorem 1.7 can be found in [114], Satz 5.26, as well as in [85], Satz. 9.22. □

Now, we take a closer look at the stability of the minimization problem. We are considering the linear minimization problem

$$\{a_i\} = \arg\min_{\mathbf{a}} \|\mathbf{Ba} - \mathbf{y}\|_2$$

with $B \in \mathbb{R}^{n,m}$, rank$(\mathbf{B}) = m$, $n \geq m$, and a perturbation thereof

$$\{\tilde{a}_i\} = \arg\min_{\mathbf{a}} \|\mathbf{B}(\mathbf{a} + \Delta\mathbf{a}) - (\mathbf{y} + \Delta\mathbf{y})\|_2,$$

which are only perturbations of the values $\mathbf{y}$ and not of the matrix $\mathbf{B}$.

Let $\mathbf{a} = \mathbf{B}^\dagger\mathbf{y}$ and $\mathbf{a} + \Delta\mathbf{a} = \mathbf{B}^\dagger(\mathbf{y} + \Delta\mathbf{y})$ be the solution by the corresponding pseudo inverse $\mathbf{B}^\dagger$ of $\mathbf{B}$. Then $\Delta\mathbf{a} = \mathbf{B}^\dagger\Delta\mathbf{y}$ holds and from $\|\mathbf{B}^\dagger\|_2 = \frac{1}{\sigma_m}$ follows [6]

$$\|\Delta\mathbf{a}\|_2 \leq \|\mathbf{B}^\dagger\|_2 \cdot \|\Delta\mathbf{y}\|_2 = \frac{1}{\sigma_m}\|\Delta\mathbf{y}\|_2.$$

[6] Which is a direct result from the definition of a matrix norm:

$$\|\mathbf{B}^\dagger\|_2 := \sup_{\|\mathbf{y}\|_2 = 1} \|\mathbf{B}^\dagger\mathbf{y}\|_2 = \sup_{\|\mathbf{y}\|_2 = 1} \sqrt{\mathbf{y}^T\mathbf{U}\mathbf{\Sigma}^\dagger\mathbf{V}^T\mathbf{V}\mathbf{\Sigma}^\dagger\mathbf{U}^T\mathbf{y}} = \sup_{\substack{\|\mathbf{c}\|_2 = 1 \\ \mathbf{c} = \mathbf{u}^T\mathbf{y}}} \sqrt{\mathbf{c}^T(\mathbf{\Sigma}^\dagger)^2\mathbf{c}}$$

$$= \sup_{\substack{\|\mathbf{c}\|_2 = 1 \\ \mathbf{c} = \mathbf{u}^T\mathbf{y}}} \left(\sum_{i=1}^{m} c_i^2 \frac{1}{\sigma_i^2}\right)^{\frac{1}{2}} = \frac{1}{\sigma_m}$$

Furthermore, with $c_i = (\mathbf{u}^i)^T\mathbf{y}$

$$\|\mathbf{a}\|_2^2 \geq \frac{1}{\sigma_1^2}\left\|\sum_{i=1}^{m}(\mathbf{u}^i)^T\mathbf{y}\mathbf{u}^i\right\|_2^2 \tag{1.8}$$

holds[7]. Since $\sum_{i=1}^{m}(\mathbf{u}^i)^T\mathbf{y}\mathbf{u}^i$ is the projection of $\mathbf{y}$ into the subspace $\mathcal{U}$ spanned by the basis $\mathbf{u}^1, \ldots, \mathbf{u}^m$, we can estimate the relative error by

$$\frac{\|\Delta\mathbf{a}\|_2}{\|\mathbf{a}\|_2} \leq \frac{\sigma_1}{\sigma_m} \cdot \frac{\|\Delta\mathbf{y}\|_2}{\|P_{\mathcal{U}}(\mathbf{y})\|_2}, \tag{1.9}$$

with linear projector $P_{\mathcal{U}}(\mathbf{y})$. Since this inequality describes how a relative error of the input to the minimization can perturb the solution, we define the condition of a matrix.

Definition 1.8 *Let $\mathbf{B} \in \mathbb{R}^{n,m}$, rank$(B) = m$, $n \geq m$ have a singular value decomposition $\mathbf{B} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$. Then, we call $\kappa(\mathbf{B}) := \frac{\sigma_1}{\sigma_m}$ the **condition of matrix** $\mathbf{B}$.*

Example 1.9 *Consider the condition of an orthogonal matrix: For any orthogonal matrix $\mathbf{Q}$*

$$\mathbf{Q}^T\mathbf{Q} = \mathbf{I}$$

with identity matrix $\mathbf{I}$ holds, which means that all eigenvalues equal 1 and thus for all singular values $\sigma_i \equiv 1$ holds. Consequently, the condition of matrix $\mathbf{Q}$ is

$$\kappa(\mathbf{Q}) = 1.$$

This condition number can be computed for other matrices and used for other types of problems, too. We will use the condition number of a rectangular matrix $\mathbf{B} \in \mathbb{R}^{n,m}$ to compare two methods of the solution to least-squares regression.

If we now solve the least-squares regression directly by $\mathbf{a} = \left(\mathbf{B}^T\mathbf{B}\right)^{-1}\mathbf{B}^T\mathbf{y}$, the relative error for the solution is bounded by

$$\begin{aligned}
\kappa\left(\mathbf{B}^T\mathbf{B}\right) &= \kappa\left((\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T)^T\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T\right) \\
&= \kappa\left(\mathbf{V}\left(\mathbf{\Sigma}\right)^2\mathbf{V}^T\right) \\
&= \frac{\sigma_1^2}{\sigma_m^2} \\
&= \kappa\left(\mathbf{B}\right)^2,
\end{aligned}$$

with $\kappa(\mathbf{B}) \geq 1$. We can improve this result by the QR factorization

$$\mathbf{B} = \mathbf{Q}\mathbf{R}$$

[7] See Appendix 7.3 for a proof.

with orthogonal matrix $\mathbf{Q} \in \mathbb{R}^{n,n}$ and upper triangular matrix $\mathbf{R} \in \mathbb{R}^{n,m}$. Since $\mathbf{Q}$ is an orthogonal matrix and thus has rank($\mathbf{Q}$) $= n$, $\mathbf{R}$ has the same rank as $\mathbf{B}$: rank($\mathbf{B}$) $=$ rank($\mathbf{R}$) $= m$. Then,

$$\begin{aligned} \mathbf{B}^T\mathbf{B}\mathbf{a} &= \mathbf{B}^T\mathbf{y} \\ \Leftrightarrow (\mathbf{Q}\mathbf{R})^T (\mathbf{Q}\mathbf{R})\,\mathbf{a} &= (\mathbf{Q}\mathbf{R})^T\,\mathbf{y} \\ \Leftrightarrow \mathbf{R}\mathbf{a} &= \mathbf{Q}^T\mathbf{y} \end{aligned}$$

holds. From Example 1.9, we know that the orthogonal matrix $\mathbf{Q}$ has a condition of one: $\kappa(\mathbf{Q}) = 1$. For the condition of matrix $\mathbf{R}$, we consider

$$\begin{aligned} \mathbf{B} &= \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T, \quad \mathbf{R} = \mathbf{U}_R\mathbf{\Sigma}_R\mathbf{V}_R^T \\ \mathbf{B} &= \mathbf{Q}\mathbf{R} = \mathbf{Q}\mathbf{U}_R\mathbf{\Sigma}_R\mathbf{V}_R^T \end{aligned}$$

where $\mathbf{Q}\mathbf{U}_R$ is again an orthogonal matrix. That means, $\mathbf{V} = \mathbf{V}_R$ and $\mathbf{\Sigma} = \mathbf{\Sigma}_R$, which is almost unique.[8] That also means that the upper triangular matrix $\mathbf{R}$ has the same condition as matrix $\mathbf{B}$, i.e. $\kappa(\mathbf{R}) = \kappa(\mathbf{B})$. The relative error of the solution of $\mathbf{a} = \mathbf{R}^\dagger\mathbf{Q}^T\mathbf{y}$ is then about $\kappa(\mathbf{B})$ because in order to obtain this solution, one has to consider the upper triangular matrix $\mathbf{R}_\triangle \in \mathbb{R}^{m,m}$ of $\mathbf{R}$ and the first m entries $y_1, \ldots, y_m$ of $\mathbf{y}$ only, since all other rows of $\mathbf{R}$ are zero. Consequently,

$$\begin{aligned} \mathbf{a} &= \mathbf{R}^\dagger\mathbf{Q}^T\mathbf{y} \\ &= (\mathbf{R}_\triangle)^{-1}\,\mathbf{Q}^T \begin{pmatrix} y_1 \\ \vdots \\ y_m \end{pmatrix} \end{aligned}$$

holds, which can be solved by back substitution without any further error, i.e the solution by normal equations which squares the condition of the problem can increase the error of the solution significantly.

Consequently, the QR factorization leads to a method for the solution of the least-squares problem which is more stable than the naïve solution of the normal equations.

A general description of the least-squares problem can be found in the book of Acton [1], an in depth analysis in the book of Higham [62] and the description of an efficient implementation of the solution by QR decompositions can be found in the Lapack Guide [8].

1.2.2 Basis Functions

A tricky part of the function approximation by regression and in fact the crucial challenge is the careful choice of the basis functions $\mathbf{B}$. As described in the previous section we will use a linear

[8] Here, we quote from Manning and Schütze [86], p. 561: For any given SVD solution, you can get additional non-identical ones by flipping signs in corresponding left and right singular vectors U and V, and, if there are two or more identical singular values, then only the subspace determined by the corresponding singular vectors is unique, but it can be described by any appropriate orthonormal basis vectors. Apart from these cases, SVD is unique.

combination of these basis functions to approximate specific functions, e.g. an estimate for the current option value conditioned on the underlying asset price.

In the following, we will present different choices of basis functions from polynomials to splines and sparse piecewise linear functions. We compare their properties so that we can choose the most suitable ones for each problem.

Polynomials

In some cases the choice of the class of basis functions seems to have little effect on the values computed by Least-Squares Monte Carlo [81]. Consequently, in some cases, it is sufficient to choose the simplest set of basis functions, polynomials.

In a simple approach we could use the full set B_ℓ^{full} of all s-dimensional polynomials up to a certain polynomial degree $\ell = (\ell_1, \ldots, \ell_s) \in \mathbb{N}^s$,

$$B_\ell^{\text{full}}(x_1, \ldots, x_s) := \left\{ \prod_{i=1}^{s} x_i^{g_i} \;\middle|\; g_i \in \mathbb{N}_0 \wedge g_i \leq \ell_i \right\}. \tag{1.10}$$

But, it is easy to see that this construction by its own will quickly exhaust any available computational resources. A setting with 10 dimensions and a maximal polynomial degree of one in each direction $\ell = (1, \ldots, 1)$ already yields as many as $2^{10} = 1024$ basis functions, over which least squares regression has to be performed. A maximal quadratic polynomial degree in each direction leads to $3^{10} = 59049$ basis functions.

That means, a full polynomial basis seems to be useful only in one or two dimensional problems, when the computational efforts are relatively small. But, in these cases, one often wants to decrease the approximation error by adding basis function to the regression. In the case of polynomials, this leads to an ill-conditioned matrix which cannot be efficiently treated by standard methods. Additionally, a refinement does not necessarily lead to uniform convergence (Theorem of Faber, see [116]). A lot more efficient than these global basis functions are local basis functions such as piecewise polynomial splines which show a better convergence, especially at the boundary of the regression domain.

Splines

A useful class of basis functions especially for a single dimension ($s = 1$) is the cubic spline. A spline is a piecewise polynomial function which lives on a decomposition of the interval $[x_0, x_m]$,

$$\Delta : x_0 < x_1 < \ldots < x_m. \tag{1.11}$$

We focus on splines which coincide on each interval $[x_{i-1}, x_i], i = 1, \ldots, m$ with a polynomial of degree 3 and lie in the class of twice continuously differentiable functions C^2. Consider the

functions

$$\begin{aligned} b_i(x, x_i) &= \begin{cases} (x_i - x)^3 & \text{if } x_i \geq x \\ 0 & \text{else} \end{cases}, \quad i = 1, \ldots, m-1 \\ b_m &= x^3 \\ b_{m+1} &= x^2 \\ b_{m+2} &= x \\ b_{m+3} &= 1, \end{aligned}$$

then we see from simple differentiation that a linear combination

$$f^{\text{spline}}(x) = \sum_{i=1}^{m+3} a_i b_i(x) \tag{1.12}$$

is continuous everywhere and has continuous first and second derivatives ($f^{\text{spline}} \in \mathcal{C}^2$). Thus the function $f^{\text{spline}}(x)$ in Equation (1.12) is a cubic spline[9].

Given the function values $y_0, \ldots, y_m$ at the knots $x_0, \ldots, x_m$, it is easy to find the coefficients a_i of f^{spline} by solving

$$\begin{aligned} f^{\text{spline}}(x_i) &= y_i, \quad i = 0, \ldots, m \\ \frac{\mathrm{d}^2}{\mathrm{d}x^2} f^{\text{spline}}(x_0) &= 0 \\ \frac{\mathrm{d}^2}{\mathrm{d}x^2} f^{\text{spline}}(x_m) &= 0 \end{aligned}$$

which leads to natural splines due to the condition that the second derivative at the boundary of the spline is zero.

Piecewise Linear Sparse Grids

As already stated, the exponential growth of the number of basis functions of full grids quickly overextends any computer. Fortunately, a much more efficient selection of basis functions can be constructed, known as sparse grids or combination method [107, 29]. This kind of function basis has been successfully applied in the field of high-dimensional function approximation [52] and many others.

The original idea of sparse grids is based on piecewise linear basis functions which we will call grids. Similar to the full set of m-dimensional polynomials, we define the full grid Ω_ℓ, $\ell = (\ell_1, \ldots, \ell_s)$ which has a possibly different equidistant spacing $\mathbf{h}_\ell := (2^{-\ell_1}, \ldots, 2^{-\ell_s})$ for each dimension of $\mathbf{x} = (x_1, \ldots, x_s)$ and has grid points $\mathbf{x}_{\ell,i} := (x_{\ell_1,i_1}, \ldots, x_{\ell_s,i_s})$, $0 \leq i_j \leq 2^{\ell_j}$ $\forall j \in \{1, \ldots, s\}$. The basis functions and thus the values of such a grid are given by[10]

$$b_{\ell,\mathbf{i}}(\mathbf{x}) := \prod_{j=1}^{s} b_{\ell_j,i_j}(x_j)$$

[9] See [111] for details of this basis spline formulation. Other formulations of this spline basis are also useful, especially for fast evaluation [38, 37] and for higher stability [38, 114].

[10] See [29], p. 10

with index vector $\mathbf{i} := (i_1, \ldots, i_s)$ which denotes the multi-index of $b_{\ell,\mathbf{i}}$ in the grid Ω_ℓ. The required one dimensional basis functions are

$$b_{l_j,i_j}(x_j) := b\left(\frac{x_j - i_j \cdot h_{l_j}}{h_{l_j}}\right)$$

where

$$b(x) := \begin{cases} 1 - |x| & \text{if } x \in [-1,1] \\ 0 & \text{otherwise.} \end{cases}$$

The idea of sparse grids is summarized as follows. Instead of using a full grid Ω_ℓ

$$\Omega_\ell := \text{span}\{b_{\ell,\mathbf{i}}(\mathbf{x}) \ : \ 1 \leq i_j \leq 2^{\ell_j} - 1 \quad \forall j \in \{1, \ldots, s\}\},$$

we combine multiple grids according to a sparse and error optimal scheme Ω_L^{sparse},

$$\Omega_L^{\text{sparse}} := \bigcup_{\sum \ell_i = L} \Omega_\ell. \tag{1.13}$$

Instead of defining a multidimensional level ℓ we use the single sparse level L, that limits the sum of all components $\ell = (\ell_1, \ldots, \ell_s)$. Figure 1.1 presents two- and three-dimensional sparse grids. This kind of combining regular grids has been shown to produce a reasonable sparseness for a wide class of smooth functions [28].

If we compare full and sparse grids, we can see that the computational effort decreases radically while the error rises only slightly: Comparing grids with minimal mesh size $h_L = 2^{-L}$, a full grid has $O(h_L^{-s})$ grid points and a sparse grid only employs $O(h_L^{-1} |\log h_L|^{s-1})$ points. At the same time, the L_2-interpolation error for smooth functions rises from $O(h_L^2)$ to $O(h_L^2 \cdot |\log h_L|^{s-1})$ [29]. In many applications with high-dimensional smooth functions, $L \in \{2, 3, 4\}$ is already sufficient.

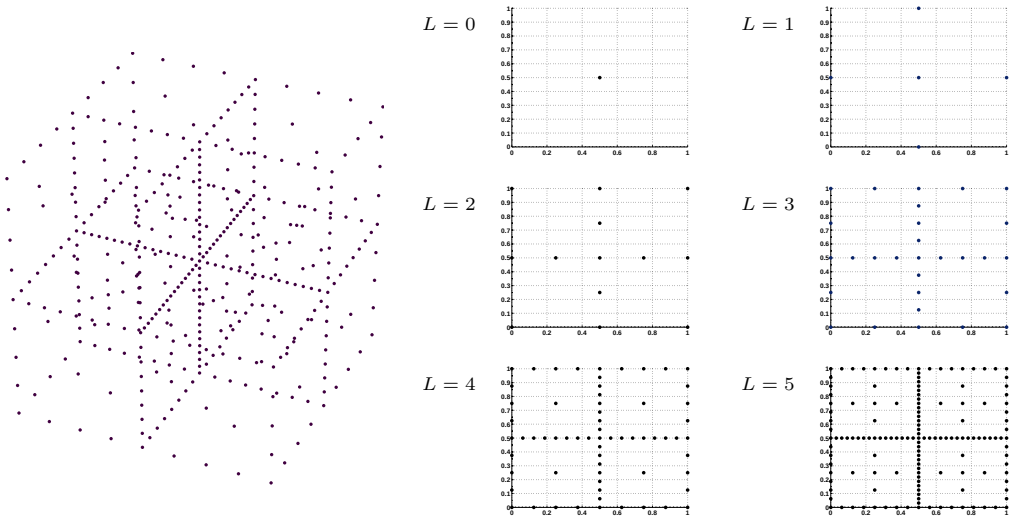

Figure 1.1: *A three-dimensional sparse grid with $L = 5$ is presented on the left and two-dimensional sparse grids from $L = 0, \ldots, 5$ on the right hand side.*

Sparse Polynomial Basis Functions

The presented sparse grid approach uses piecewise linear basis functions supporting the grid nodes. Most of the examples we will consider in the remainder of this dissertation are approximations to continuation value functions for financial options. These functions are usually smooth ($\in C^\infty$) with respect to the state variables. In such a case, differentiable basis functions deliver better results than piecewise linear functions. Consequently, we propose creating a sparse polynomial basis, which is very smooth ($\in C^\infty$). A detailed analysis for this kind of sparse basis can be found in [13] so that we can focus on the main issues.

We combine the idea of a polynomial basis with the idea of sparse grids: instead of using a plain polynomial basis B_ℓ^{full} as in Equation (1.10), we combine the multiple polynomial orders according to the same sparse and error optimal scheme as for the sparse grids. The sparse basis B_L^{sparse},

$$B_L^{\text{sparse}}(x_1, \ldots, x_s) := \bigcup_{\sum \ell_i = L} B_{\beta(\ell)}^{\text{full}}(x_1, \ldots, x_s) \tag{1.14}$$

has many of the properties of the sparse grid with piecewise linear basis functions but it is smooth everywhere.

The polynomial basis sparse level L again limits the sum of all components $\ell = (\ell_1, \ldots, \ell_s)$. Furthermore, the degree of the combined full polynomial bases is transformed by a mapping function β that turns each level into a maximum polynomial degree

$$\beta(\ell) = 2 \cdot (2^{\ell_1} - 1, \ldots, \ldots, 2^{\ell_s} - 1).$$

This transformation cannot be applied to grids because a grid with $\ell_i = 0$ nodes in the ith dimension makes no sense. But, for polynomials, this reduces the number of basis functions. An example for a three dimensional sparse polynomial basis can be found in Figure 1.2.

Example: A sparse polynomial basis with $s = 3, L = 2$

First, we have to compute the full basis polynomials of Equation (1.10). The sparse level $L = 2$ requires the computation of

$$\begin{aligned}
\ell &= (2,0,0) \to B^{\text{full}}_{6,0,0} = \{1, x_1, x_1^2, x_1^3, x_1^4, x_1^5, x_1^6\} \\
\ell &= (1,1,0) \to B^{\text{full}}_{2,2,0} = \{1, x_1, x_2, x_1x_2, x_1^2, x_2^2, x_1^2x_2, x_1x_2^2, x_1^2x_2^2\} \\
\ell &= (1,0,1) \to B^{\text{full}}_{2,0,2} = \{1, x_1, x_3, x_1x_3, x_1^2, x_3^2, x_1^2x_3, x_1x_3^2, x_1^2x_3^2\} \\
\ell &= (0,2,0) \to B^{\text{full}}_{0,6,0} = \{1, x_2, x_2^2, x_2^3, x_2^4, x_2^5, x_2^6\} \\
\ell &= (0,1,1) \to B^{\text{full}}_{0,2,2} = \{1, x_2, x_3, x_2x_3, x_2^2, x_3^2, x_2^2x_3, x_2x_3^2, x_2^2x_3^2\} \\
\ell &= (0,0,2) \to B^{\text{full}}_{0,0,6} = \{1, x_3, x_3^2, x_3^3, x_3^4, x_3^5, x_3^6\}
\end{aligned}$$

for the sparse grid basis and leads to 31 basis functions,

$$\begin{aligned}
B^{\text{sparse}}_2(x_1,x_2,x_3) &= \bigcup_{\sum \ell_i = 2} B^{\text{full}}_{\beta(\ell)} \\
&= \{1, x_1, x_2, x_3, x_1x_2, x_1x_3, x_2x_3, x_1^2, x_2^2, x_3^2, x_1^2x_2, \\
&\quad x_1x_2^2, x_1^2x_2^2, x_1^2x_3, x_1x_3^2, x_1^2x_3^2, x_2^2x_3, x_2x_3^2, x_2^2x_3^2, \\
&\quad x_1^3, x_2^3, x_3^3, x_1^4, x_2^4, x_3^4, x_1^5, x_2^5, x_3^5, x_1^6, x_2^6, x_3^6\}.
\end{aligned}$$

Figure 1.2: *An example for a three-dimensional sparse polynomial basis.*

Thin-Plate Spline Basis Functions

Another alternative to produce a sparse and smooth multivariate approximation function is to use the nodes of a sparse linear grid as basis function nodes of a thin-plate regression spline.

Thin plate splines are radial basis functions defined by the minimization of a smoothness measure in a function space. We directly use the resulting spline basis and define a thin-plate spline $f^{\text{tp spline}}(\mathbf{x}), \mathbf{x} \in \mathbb{R}^s$ as

$$\begin{aligned}
f^{\text{tp spline}}(\mathbf{x}) &= \sum_{i=1}^{m} a_i b_i(\mathbf{x}), \\
b_i(\mathbf{x}) &:= \|\mathbf{x} - \mathbf{x}_i\|^2 \log \|\mathbf{x} - \mathbf{x}_i\|,
\end{aligned}$$

with basis nodes $\mathbf{x}_i$. [11] The main advantage of thin-plate splines is that we can choose the grid nodes $\mathbf{x}_i \in \mathbb{R}^s$ freely. This allows to use random samples as grid nodes as well as the introduction

[11] See [115] for the basic derivation, [118] for regressions with large data samples on thin-plate splines and [93] for option pricing with radial basis functions.

of additional nodes into interesting areas.

1.2.3 Approximation Properties

The theoretical convergence properties of an approximation by the presented basis functions are well analyzed. Uniform convergence can be expected from piecewise-linear splines as well as some thin-plate splines [97]. The approximation properties of a polynomial basis is not as good as of a spline basis, especially the values at the boundary of the approximation domain can have a significant error [62].

Table 1.1 presents an overview of the basis functions presented in this chapter. Regressions on basis functions with a global support usually suffer from artifacts at the boundary of the domain of the samples. Smooth functions are sometimes required due to numerical properties, such that C^∞ functions should be preferred. The number of functions, which depends on the choice of the parameter ℓ and the dimension s, can sometimes grow large for medium size dimensions. Polynomials seem to be useful up to $s = 2$ or $s = 3$. Piecewise linear sparse grids and sparse basis functions can reach $s = 10$ to $s = 20$ and thin-plate splines might still be useful for $s > 20$. The presented B-splines are only useful in a single dimension $s = 1$.

Table 1.1 *Overview of the presented basis function classes.*

Basis Function	**Support**	**# of Functions** $m(L, s)$	**Smoothness**
polynomials	global	$m \in \mathrm{O}(L^s)$	C^∞
B-Splines	(local) global	$m + 3$, $s = 1$ only	C^2
piecewise-linear sparse grids	local	$m \in \mathrm{O}(2^L \lvert \log L \rvert^{s-1})$	C^0
sparse polynomials	global	$m \in \mathrm{O}(2^L \lvert \log L \rvert^{s-1})$	C^∞
thin-plate splines	global	m	C^∞

1.3 Pricing and Hedging in Complete Markets

In the previous sections we saw the basic properties of regression methods and different basis functions. Now, we are presenting the application for which these tools are required: option valuation.

1.3.1 Terminology

Before we present the general framework for option valuation, we focus on the instruments we seek to valuate. All financial options in this thesis depend on a single underlying[12] S at specific dates t_i. We will write the value of the underlying at time t_i as S_{t_i}.

A financial option[13] is a contract between an option issuer and the option holder. An option

[12] The underlying is also called asset, stock, spot price or equity.
[13] A financial option is also called derivative, security or simply option.

gives the holder the right, but not the obligation to receive cash flows in the future dependent on the value of the underlying S.

Consequently options are characterized by cash flows to the option holder. In this section we introduce a few essential instruments, to which we will add others in the subsequent chapters if required.

Definition 1.10 *A **European call (put) option** is the right but not the obligation to receive $S_{t_T} - K$ ($K - S_{t_T}$) at maturity time t_T. The variable K is called strike price. European call (put) options are also called **vanilla options**.*

There are many other ways to formulate option contracts. One of the most common option features is an exercise opportunity by the option's holder. This can be specified for a certain date (discrete) or a specific time interval (continuous). We can also make the payoff dependent on the asset history to get a so called path dependent option.

Definition 1.11 *An **American call (put) option** is the right to receive $S_{t_i} - K$ ($K - S_{t_i}$) at any time t_i from initial time t_0 until maturity time t_T. The variable K is called strike price. This option is called **early exercisable** because the option holder can receive the exercise value $S_{t_i} - K$ ($K - S_{t_i}$) prior to the options maturity time t_T.*

In order to find so-called fair values for these and other options, we proceed with the general pricing framework.

1.3.2 General Framework

The valuation of a derivative security is a common task in mathematical finance. Here, we will provide a brief summary of the model derivation which can also be found in e.g. [117, 64]. For the Black-Scholes model, we need to make some assumptions and simplifications. The basic assumptions are a frictionless market, no transaction costs, risk-less assets earn the risk-free rate r and short selling is allowed without restrictions. Another more conceptual assumption is that the price of the underlying S behaves like a geometric Brownian motion. That means that nobody can foresee future stock prices and the asset evolves as

$$\mathrm{d}S_t = \mu S_t \,\mathrm{d}t + \sigma S_t \,\mathrm{d}W_t \tag{1.15}$$

where μ is the drift rate, σ is the volatility of S and $\mathrm{d}W_t$ is the increment of a Wiener process.

We can now establish a portfolio Π consisting of the security of interest and a short position of ϕ shares. The price of the security clearly depends on the underlying stock price S_t and time t and will be noted as $V(S_t, t)$ or just V_t, i.e.

$$\Pi_t = V(S_t, t) - \phi_t S_t.$$

Using Itô's Lemma [67], we find that the portfolio changes can be described by

$$\begin{aligned} \mathrm{d}\Pi_t &= \mathrm{d}V_t - \phi_t \,\mathrm{d}S_t \\ &= \frac{\partial V_t}{\partial t}\,\mathrm{d}t + \frac{\partial V_t}{\partial S_t}\,\mathrm{d}S_t + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V_t}{\partial S_t^2}\,\mathrm{d}t - \phi_t \,\mathrm{d}S_t. \end{aligned} \tag{1.16}$$

If we chose

$$\phi_t = \frac{\partial V_t}{\partial S_t} \tag{1.17}$$

we can make the portfolio independent of $\mathrm{d}S_t$, the movements of the stock price. That means the portfolio is completely deterministic and inhabits no risk. Using the no-arbitrage principle[14], this risk-free portfolio earns the risk-free rate r. Consequently, the changes in Π_t are

$$\mathrm{d}\Pi_t = r\Pi_t \,\mathrm{d}t. \tag{1.18}$$

Now, we can use (1.16), (1.17) and (1.18) to get the relationship

$$\frac{\partial V_t}{\partial t} + \frac{1}{2}\sigma^2 S_t^2 \frac{\partial^2 V_t}{\partial S_t^2} + rS_t \frac{\partial V_t}{\partial S_t} - rV_t = 0. \tag{1.19}$$

Knowing that at maturity time t_T the option value equals the payoff $P(S_T, t_T)$,

$$V(S_{t_T}, t_T) = P(S_{t_T}, t_T),$$

the partial differential Equation (1.19) can be solved by numerical methods as an initial value problem in backwards time. This kind of reasoning was published in 1973 by Fischer Black and Myron Scholes [17].

1.3.3 Exercisable Options

We now look at a possible early exercise by the holder of the security. By the no-arbitrage assumption, an exercise where the holder obtains the payoff $P(S_t, t)$ leads to

$$V(S_t, t) \geq P(S_t, t), \tag{1.20}$$

i.e. the early exercisable option will always have a value which is greater or equal to the immediate exercise value. If this were not true, an investor would buy the option, exercise immediately and make a risk-less profit.

In order to find a representation of an exercisable option value similar to Equation (1.19), little thought lead to the observation that the portfolio Π can at most earn the risk-free rate if the no-arbitrage assumption should still be possible. Consequently, we get, instead of Equation (1.18), the relationship

$$\begin{aligned} \mathrm{d}\Pi_t &\leq r\Pi_t \,\mathrm{d}t \\ &\Leftrightarrow \\ \frac{\partial V_t}{\partial t} + \frac{1}{2}\sigma^2 S_t^2 \frac{\partial^2 V_t}{\partial S_t^2} + rS_t \frac{\partial V_t}{\partial S_t} - rV_t &\leq 0. \end{aligned} \tag{1.21}$$

[14] No-arbitrage means in this case that risk-less investments earn the risk free rate and investments that earn more than the risk-free rate must by risky.

In the case of an American option the value V is given by the solution to Equations (1.21) and (1.20) where at least one of the inequalities holds with equality on the complete solution. This is an initial value problem or Cauchy problem in backwards time $\tau = t_T - t$ with a free boundary, starting with the terminal condition, i.e. for an American put option with strike K

$$V(S_{t_T}, t_T) = \max(K - S_{t_T}, 0) \; .$$

Numerical schemes for this kind of PDE solution have been presented by several authors. An efficient solution has been presented by Forsyth and Vetzal [50], which will also form the foundation for the PDE reference methods used in this thesis. The next section proceeds with an overview of the numerical methods currently used in the field of derivatives pricing.

1.4 Numerical Methods for Option Valuation

1.4.1 Overview

The challenge that remains after the introduction of the Black-Scholes framework is the efficient valuation of arbitrary derivatives. Even though closed-form solutions of the governing Equation (1.19) can be found for European put and call options, no analytic solutions are available for early exercisable options such as for the American option price which is governed by Equations (1.21) and (1.20). In other cases, the derivation of a closed-form solution might be possible, but its evaluation still might be a challenge or the derivation too complex. In all these cases a numerical tool is required which can deliver accurate option prices.

In the past few decades, several option valuation methods have been proposed. Besides the analytic solution of the Black-Scholes PDE by Merton [87] one of the most common methods is the Cox-Ross-Rubinstein method [36], which discretizes the asset price process by a binomial tree. A simple recursive solution for the option value allows the valuation of early exercisable options. This method can be seen as a special case of the finite differences method by Schwartz [103] which discretizes the Black-Scholes PDE directly and has better convergence properties than the Cox-Ross-Rubinstein method. Another method with wide application is a Monte Carlo method by Boyle [21] which simulates the underlying asset price process under a risk-neutral expectation. Other methods were developed for special cases, e.g. a trinomial model by Boyle [22] for valuation of options with two correlated underlyings. Furthermore, a multinomial-tree method by Andricopoulos et. al [9] which is based on quadrature methods for integration has found some applications. This list is certainly not complete, but it assembles the main methods which are in use in academia and the financial industry.

In the following sections, we focus on numerical option valuation by Monte Carlo methods. As a reference, we will use a solver for the Black-Scholes PDE similar to the finite differences method.

1.4.2 Monte Carlo Methods

In numerous cases, the Monte Carlo method proved useful due to a simple implementation and dimension independent convergence properties. Further details and advanced Monte Carlo methods for option pricing can be found in [53].

Valuation of European Option

As we will see later in this section, the valuation of European style options is equivalent to the integration of the expected terminal option value using the risk-neutral distribution of the underlying stock value at the maturity time t_T of the option. Consequently, we focus on the integration of functions by Monte Carlo methods, first. Assume that an option value V is given by

$$V = e^{-rt_T} \int_{\mathbb{D}} P(S') p^{S_{t_T}}(S') \, \mathrm{d}S' \tag{1.22}$$

where $P(S')$ is the option's payoff function and $p^{S_{t_T}}(S')$ is the risk-neutral probability density function of the stock price at the option's maturity. The area $\mathbb{D} \subseteqq \mathbb{R}^s$ is the domain of integration, where s denotes the dimension of the state space, e.g. the number of underlying instruments defining the payoff at t_T.

The probability density function $p^{S_{t_T}}(S')$ can be obtained by the computation of Green's function to the Black-Scholes PDE. Naturally, it satisfies

$$\int_{\mathbb{R}^s} p^{S_{t_T}}(S') \, \mathrm{d}S' = 1.$$

If we now draw a sample $S^1, \ldots, S^n$ from the distribution given by $p^{S_{t_T}}(S')$, we can compute an estimate $\overline{V}^n$ for the integral given by (1.22),

$$\overline{V}^n = e^{-rt_T} \frac{1}{n} \sum_{i=1}^{n} P(S^i).$$

The variance of this estimate is given by

$$\left(\mathrm{std}[\overline{V}^n]\right)^2 = \frac{1}{n-1} \sum_{i=1}^{n} \left(e^{-rt_T} P(S^i) - \overline{V}^n\right)^2.$$

If we standardize the distribution function of the error, then we get the

$$\text{standardized error} = \frac{V - \overline{V}^n}{\mathrm{std}[\overline{V}^n]} \cdot \sqrt{n} \tag{1.23}$$

which converges with $n \to \infty$ to a standard normal distribution if the third moment $\mathbb{E}\left[|\text{standardized error}|^3\right]$ exists and is finite [15].

[15] This is an immediate result of the Berry-Esseen Theorem [45] resp. the central limit theorem.

For Equation (1.23), we can see that the standard deviation of the error $V - \overline{V}^n$ is equal to

$$\frac{\text{std}[\overline{V}^n]}{\sqrt{n}} \in \text{O}\left(\frac{1}{\sqrt{n}}\right). \tag{1.24}$$

The confidence interval at level α can be approximated by a standard normal distribution:

$$\left[\overline{V}^n - \frac{\text{std}[\overline{V}^n]}{\sqrt{n}}\epsilon(\alpha), \overline{V}^n + \frac{\text{std}[\overline{V}^n]}{\sqrt{n}}\epsilon(\alpha)\right].$$

The value $\epsilon(\alpha)$ can be obtained by inverting the cumulative standard normal distribution Φ such that

$$\text{Prob}(|\text{standardized error}| \leq \epsilon(\alpha)) = 1 - \alpha$$

holds, i.e. $\epsilon(\alpha)$ is the $(1 - \frac{\alpha}{2})$-quantile of the cumulative standard normal distribution. For frequently used confidence levels, the intervals are given by Table 1.2. It is important to note that the confidence limits only depend on the Monte Carlo sample value $\overline{V}^n$ as well as the number of samples n. It does not depend on the dimension s of the state space, which makes this method suitable for the evaluation of high-dimensional option pricing problems.

Table 1.2 *Confidence intervals for different levels of confidence* α.

Confidence level α	Interval
68%	$\overline{V}^n \pm 1.00 \cdot \frac{\text{std}[\overline{V}]}{\sqrt{n}}$
90%	$\overline{V}^n \pm 1.65 \cdot \frac{\text{std}[\overline{V}]}{\sqrt{n}}$
95%	$\overline{V}^n \pm 1.96 \cdot \frac{\text{std}[\overline{V}]}{\sqrt{n}}$
99%	$\overline{V}^n \pm 2.58 \cdot \frac{\text{std}[\overline{V}]}{\sqrt{n}}$

Considering a plain vanilla option in the Black-Scholes framework, the probability density of the terminal stock price values is given by[16]

$$p^{S_{t_T}}(S) = \frac{1}{\sigma S\sqrt{2\pi t_T}} e^{-\frac{(\log(S/S_{t_0}) + (r - (1/2)\sigma^2)t_T)^2}{2\sigma^2 t_T}}. \tag{1.25}$$

But, the inversion of $p^{S_{t_T}}$ in order to draw a sample $S^1, \ldots, S^n$ of terminal asset prices is computationally expensive. A better approach follows from the inspection of the PDE given by Equation (1.19): The Black-Scholes PDE is just the backward Kolmogorov Equation for the process

$$\text{d}S_t = rS_t\,\text{d}t + \sigma S_t\,\text{d}W_t, \tag{1.26}$$

which is the same process as in the derivation of Equation (1.19), except that the drift rate μ in Equation (1.15) is replaced by the risk-free interest rate r.[17]

[16] See [117], p. 94.
[17] See Wilmott [117], p.164 ff for details.

This observation is very important: The dynamics of the process S required for the evaluation of an option with S as an underlying is different from the real dynamics of S. The reason for this is that not the dynamics of S itself are required for the valuation, but the dynamics of a hedged portfolio which consists of a position in the underlying asset S, the option itself V and money in a bank account.

In the following, we will call the real process of S, which is determined by Equation (1.15) as the physical or real-world process, while we call the dynamics for the purpose of option valuation (Equation (1.26)) the risk-neutral process. Furthermore, we will denote expectations which require a risk-neutral dynamic of S_t by $\mathbb{E}_Q[S_t]$, while we will leave $\mathbb{E}[S_t]$ for the expectation under the physical dynamic:

Definition 1.12 *Given an option valuation problem in the Black-Scholes framework.*

a) *The dynamic of the asset S*

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

is called physical or real-world process. A formula, which requires an expectation of S_t using this dynamic is denoted by

$$\mathbb{E}[S_t].$$

b) *The dynamic of the asset S*

$$dS_t = r S_t \, dt + \sigma S_t \, dW_t,$$

is called risk-neutral process. A formula, which requires an expectation of S_t using this dynamic is denoted by

$$\mathbb{E}_Q[S_t].$$

The easiest sampling technique for an Equation as Equation (1.26) is the Euler method. This method samples n trajectories S_t^j, $j = 1, \ldots, n$ at several time steps t_i, $i = 1, \ldots, T$, starting at $S_{t_0}^1 = S_{t_0}^2 = \ldots = S_{t_0}^n =: S_{t_0}$ as

$$S_{t_{i+1}}^j = S_{t_i}^j + r S_{t_i}^j (t_{i+1} - t_i) + \sigma S_{t_i}^j \theta_{i,j} \sqrt{t_{i+1} - t_i} \tag{1.27}$$

with $\theta_{i,j}$ drawn from a standard Normal distribution.

It turns out that a better discretization can be found, which is given by

$$S_{t_{i+1}}^j = S_{t_i}^j \exp\left((r - \frac{1}{2}\sigma^2)(t_{i+1} - t_i) + \sigma \sqrt{(t_{i+1} - t_i)} \theta_{i,j} \right). \tag{1.28}$$

This method is exact in the sense that the distribution of S_{t_T} does not depend on intermediate time steps as in Equation (1.27)[18]. Consequently, only one time step is required for the valuation of a vanilla European option:

$$S_{t_T}^j = S_{t_0} \exp\left((r - \frac{1}{2}\sigma^2)(t_T - t_0) + \sigma \sqrt{(t_T - t_0)} \theta_j \right) \tag{1.29}$$

[18] See Wilmott [117], p. 924 for details.

with θ_j drawn from a standard Normal distribution.

Finally, we summarize the algorithm for pricing a European Put option in the Black-Scholes framework in Table 1.3.

Table 1.3 *Valuation of a European Put option in the Black-Scholes framework with risk-free rate r, volatility σ, maturity time t_T and strike K.*

1. Simulate n risk-neutral asset price trajectories starting at the current price S_{t_0}, $j \in \{1, \dots, n\}$,

$$S_{t_T}^j = S_{t_0} \exp\left((r - \frac{1}{2}\sigma^2)(t_T - t_0) + \sigma\sqrt{(t_T - t_0)}\theta_j \right).$$

2. Compute the average payoff and discount it with the risk-free rate r, i.e.

$$\overline{V}^n = e^{-r(t_T - t_0)} \frac{1}{n} \sum_{j=1}^{n} \max(K - S_{t_T}^j, 0),$$

   which provides an estimate for the option value V, i.e. $V \approx \overline{V}^n$.

Least-Squares Monte Carlo

Since Monte Carlo is a standard method which is used when dimensionality causes numerical difficulties, we extend the method in Table 1.3 to exercisable options. We already saw exercisable options in Section 1.3.3. But another way of formulating this mathematical problem can be used: The price of an exercisable security is the discounted expected value of the payoff at the optimal stopping time.

This can be formulated as an optimal stopping problem (see Carrière [32, Section 4]), with

$$V(S_{t_0}, t_0) = \sup_{\tau \in \mathcal{T}} \mathbb{E}_Q \left[e^{-r(\tau - t_0)} P(S_\tau, \tau) \right], \tag{1.30}$$

where the asset price process S_t evolves in the risk-neutral fashion

$$\mathrm{d}S_t = rS_t \, \mathrm{d}t + \sigma S_t \, \mathrm{d}W_t, \tag{1.31}$$

and $\mathcal{T}$ is the set of all possible stopping times.

This formulation as an optimal stopping problem (1.30) now leads to the idea of Monte Carlo algorithms for pricing American options, because one only has to estimate an optimal stopping (exercise) strategy within the Monte Carlo method for vanilla options. This way, the solution of complex free boundary PDEs as defined by the Inequalities (1.21) and (1.20) is avoided.

One approach for Monte Carlo valuation of exercisable options is to parameterize the region of optimal exercise with a function (see [14] and the references therein). The main other approach is to approximate the conditional expected continuation value with a regression. This method

was first presented by Carrière [32] and is called Least-Squares Monte Carlo. This is the method we will use in the following.

Similar to the previous section, we simulate different asset paths S^j, $j \in \{1, \dots, n\}$, which follow Equation (1.28),

$$S^j_{t_{i+1}} = S^j_{t_i} \exp\left((r - \frac{1}{2}\sigma^2)(t_{i+1} - t_i) + \sigma\sqrt{(t_{i+1} - t_i)}\theta_{i,j}\right)$$

with $\theta_{i,j}$ drawn from a standard Normal distribution.

At each exercise time t_i, the holder decides to exercise the option and get the payoff $P(S_{t_i}, t_i)$ or to continue. In this case, the payoff $P(S_{t_i}, t_i)$ may only depend on the value of S_{t_i} at time t_i, which can easily be extended to a dependence on the complete asset paths history as we will see in later chapters. In order to maximize the option value V, the holder exercises if

$$P(S_{t_i}, t_i) \geq \mathbb{E}_Q[V(S_{t_{i+1}}, t_{i+1})|S_{t_i}, t_i]$$

with $\mathbb{E}_Q[V(S_{t_{i+1}}, t_{i+1})|S_{t_i}, t_i]$ denoting the expected option value under the risk-neutral measure Q if the option is not exercised at time t_i. In the Least-Squares Monte Carlo approach, the value of $\mathbb{E}_Q[V(S_{t_{i+1}}, t_{i+1})|S_{t_i}, t_i]$ is approximated by

$$P^e(S_{t_i}, t_i) \approx \mathbb{E}_Q[V(S_{t_{i+1}}, t_{i+1})|S_{t_i}, t_i]. \tag{1.32}$$

The value $P^e(S_{t_i}, t_i)$ is computed using a least-squares regression of many path-realizations S^j on some basis functions b_k, i.e. the local basis approximation of $V(S_{t_{i+1}}, t_{i+1})$ given S_{t_i} (Theorem 1.4). The regressions start at the time step t_{T-1}, i.e. one step before maturity time t_T. The approximated values are

$$P^e(S_{t_i}, t_i) = \sum_k a^i_k b_k(S_{t_i}) \tag{1.33}$$

with some basis functions b_k and unknown coefficients a^i_k minimizing

$$\left\| \left(\sum_k a^i_k b_k(S^j_{t_i}) - e^{-r(t_{i+1}-t_i)} V^j_{t_{i+1}} \right)_{j=1,\dots,n} \right\|_2 \tag{1.34}$$

where $V^j_{t_{i+1}}$ is the estimate of the option value for time t_{i+1} using the Monte Carlo path realization S^j. The value of $V^j_{t_i}$ is given by the maximum between the estimated value of the unexercised option P^e and the intrinsic value P,

$$V^j_{t_i} = \begin{cases} e^{-r(t_{i+1}-t_i)} V^j_{t_{i+1}} & \text{if } P^e(S^j_{t_i}, t_i) > P(S^j_{t_i}, t_i) \\ P(S^j_{t_i}, t_i) & \text{else} \end{cases}. \tag{1.35}$$

In Section 1.2.1, we saw how the solution to Equation (1.33) respectively Equation (1.34) can be computed.

Given that the option value at maturity time equals the payoff $V^j_{t_T} = P(S^j_{t_T}, t_T)$, a dynamic program solves for all values $V^j_{t_i}$, starting at time t_T and iterating backwards to t_0. Based on the

value $V_{t_0}^j$, we can compute an approximation to the option value, which is known as the in-sample price,

$$V^{in} = \frac{1}{n_1} \cdot \sum_{j=1}^{n_1} V_{t_0}^j,$$

where the asset paths are S^j, $j \in \{1, \ldots, n_1\}$. This approach has an obvious disadvantage. Each of the estimated option values $V_{t_0}^j$ contains information about its future asset path $S_{t_{i+1}}^j$. In order to avoid this property, we compute the out-of-sample option price: We generate additional simulation paths S^l, $l \in \{n_1+1, \ldots, n_1+n_2\}$ but use the coefficients a_k^i which were fitted to the old set of simulation paths S^j, $j \in \{1, \ldots, n_1\}$. Consequently, the out-of-sample value can not depend on the knowledge of the future paths. It is given by

$$V^{out} = \frac{1}{n_2} \cdot \sum_{l=n_1+1}^{n_1+n_2} V_{t_0}^l \tag{1.36}$$

with

$$V_{t_i}^l = \begin{cases} e^{-r(t_{i+1}-t_i)} V_{t_{i+1}}^l & \text{if } P^e(S_{t_i}^l, t_i) > P(S_{t_i}^l, t_i) \\ P(S_{t_i}^l, t_i) & \text{else} \end{cases}, l \in \{n_1+1, \ldots, n_1+n_2\}.$$

Under optimal conditions, the in-sample and the out-of-sample price converge to the correct arbitrage-free price. However, in our computations, we will only compute the out-of-sample value because it is the value for which we can state the exercise policy without information about the future. Furthermore, the expected value of the out-of-sample price V^{out} of an American option is always a lower bound for the option value: The crucial point for the convergence of Least-Squares Monte Carlo simulation is the estimate P^e. We are confined to finite many samples and to finite degrees of freedom in the regressions and will not be able to perfectly represent the real shape of $\mathbb{E}_Q[V(S_{t_{i+1}}, t_{i+1})|S_{t_i}, t_i]$. Thus, a less than optimal exercise strategy is performed and provides a reduced option value.

Notes on the Convergence of Least-Squares Monte Carlo

Figure 1.3 presents American put option value estimates computed with Least-Squares Monte Carlo and different numbers of cubic spline basis functions.

While the Least-Squares Monte Carlo estimates with $n_1 + n_2 = 10^5$ asset paths already reach a minimal error with two basis functions, the estimates with $n_1 + n_2 = 10^7$ asset paths reach a minimal error at 10 basis functions. The remaining error is mainly due to the finite number of time steps ($T = 50$). That means, one has to analyze the optimal number of basis functions based on the number of asset paths. With more than 10 basis functions, the error increases due to some kind of overfitting of the conditional expectation function.

In the following chapters, there will always be a discussion, when and which basis functions b_k are suitable for the specific problem.

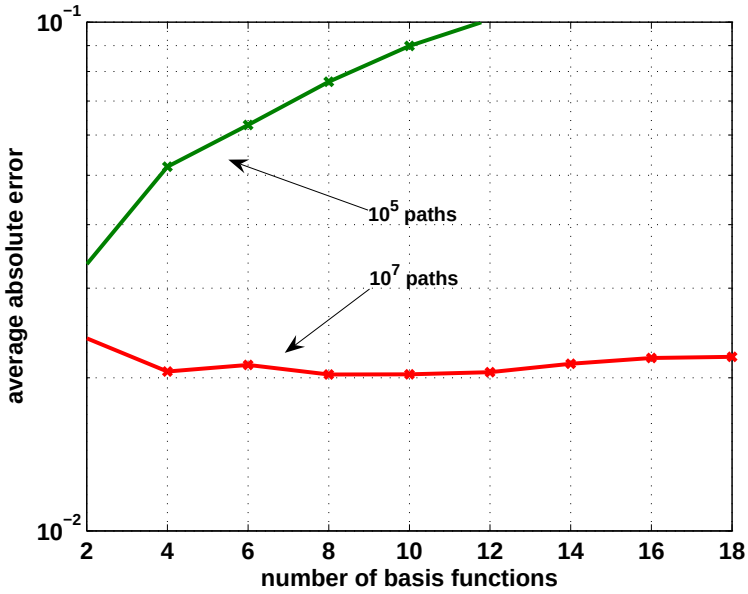

Figure 1.3: *The average error of the Least-Squares Monte Carlo price estimate of an American put option. The put has a strike $K = 100$, the asset price is $S_{t_0} = 100$, volatility $\sigma = 0.4$, risk-free rate $r = 0.05$. Different numbers of basis functions are used within the 50 time steps Least-Squares Monte Carlo. The PDE reference value is 13.66761.*

1.4.3 Direct PDE Methods

There are several direct PDE solver methods. The common ones include finite differences [103, 117], finite elements [50] and finite volume methods [124]. Other methods are Laplace and Fourier transform methods [117] as well as meshless methods [80].

In the following, we present a simple finite element method, which is sufficient for our purposes as a pricing engine. In some cases, this thesis will refer to more advanced techniques. These techniques are mainly based on the methods presented by Forsyth et. al. [50].

European Options

In this section, we want to build a very simple PDE solver for the Black-Scholes Equation. The Black-Scholes PDE is a Cauchy-Problem in backwards time τ where the initial values are given by the payoff at maturity. As usual, with time t we denote the asset price by S_t, the value of the option by V_t, the volatility of the asset by σ and the risk-free rate by r. The solver is an implicit finite volume discretization of

$$\frac{\partial V_\tau}{\partial \tau} = \frac{1}{2}\sigma^2 S_\tau^2 \frac{\partial^2 V_\tau}{\partial S_\tau^2} + rS\frac{\partial V_\tau}{\partial S_\tau} - rV_\tau \tag{1.37}$$

working backwards in time from maturity to present time t_0. This equation is equivalent to Equation (1.19), using backwards time $\tau = t_T - t$. We integrate Equation (1.37) over a finite volume A^i using the discrete values S^i, $i = 0, \ldots, m$ where

$$A^i = \left(\frac{S^{i+1} + S^i}{2}\right) - \left(\frac{S^i + S^{i-1}}{2}\right)$$

with $S^0 = 0$ and sufficiently S^m large[19] That means that the cell boundaries are placed half way between the nodes at

$$S^{i+1/2} = \frac{S^i + S^{i+1}}{2}$$
$$S^{i-1/2} = \frac{S^i + S^{i-1}}{2}.$$

After the integration of Equation (1.37) over the ith cell, we obtain

$$\int_{A^i} \frac{\partial V_\tau}{\partial \tau}\, \mathrm{d}s = \int_{A^i} \frac{\sigma^2}{2} s^2 \frac{\partial^2 V_\tau}{\partial S_\tau^2}\, \mathrm{d}s + \int_{A_i} rs \frac{\partial V_\tau}{\partial S_\tau}\, \mathrm{d}s - \int_{A^i} rV_\tau\, \mathrm{d}s,$$

where V_τ is a function of S_τ and τ, i.e. $V_\tau = V(S_\tau, \tau) = V(S_t, t),\ \tau = t_T - t$. In the following, we denote the value of the option at time τ_j and asset price S^i as

$$V(S^i_{\tau_j}, \tau_j) = V^i_j.$$

Using approximations we get

$$\begin{aligned}
\int_{A^i} \frac{\sigma^2}{2} s^2 \frac{\partial^2 V_\tau}{\partial S_\tau^2}\, \mathrm{d}s &\approx \frac{\sigma^2}{2}(S^i)^2 \int_{A^i} \frac{\partial^2 V_\tau}{\partial S_\tau^2}\, \mathrm{d}s \\
&= \frac{\sigma^2}{2}(S^i)^2 \left(\left(\frac{\partial V_\tau}{\partial S_\tau}\right)^{i+1/2} - \left(\frac{\partial V_\tau}{\partial S_\tau}\right)^{i-1/2} \right) \\
&\approx \frac{\sigma^2}{2}(S^i)^2 \left(\frac{V^{i+1} - V^i}{S^{i+1} - S^i} + \frac{V^{i-1} - V^i}{S^i - S^{i-1}} \right).
\end{aligned}$$

Furthermore, we get for the other terms

$$\begin{aligned}
\int_{A^i} rV\, \mathrm{d}s &\approx rV^i A^i, \\
\int_{A^i} rs \frac{\partial V_\tau}{\partial S_\tau}\, \mathrm{d}s &\approx rS^i \left[V^{i+1/2} - V^{i-1/2} \right] \\
&= rS^i \left[\frac{V^{i+1} - V^{i-1}}{2} \right].
\end{aligned}$$

On the left-hand side of Equation (1.37), we get

$$\int_{A^i} \frac{\partial V_\tau}{\partial \tau}\, \mathrm{d}s \approx A^i \left[\frac{V^i_{j+1} - V^i_j}{\Delta \tau} \right].$$

[19] For a European call option $S^m = e^{10\sigma\sqrt{t_T - t_i}} K$ is enough for many realistic settings [49, p.21].

All the above equations finally lead to

$$A^i\left[\frac{V^i_{j+1}-V^i_j}{\Delta\tau}\right]=\frac{\sigma^2}{2}(S^i)^2\left(\frac{V^{i+1}-V^i}{S^{i+1}-S^i}+\frac{V^{i-1}-V^i}{S^i-S^{i-1}}\right)+rS^i\left[\frac{V^{i+1}-V^{i-1}}{2}\right]-rV^iA^i$$

where we have to define the time level of the right-hand side. Choosing a time level j for the right hand side, we get a so called explicit discretization; choosing a level of $j+1$, the discretization is called implicit. The implicit discretization is more stable so that we will use

$$A^i\left[\frac{V^i_{j+1}-V^i_j}{\Delta\tau}\right]=\frac{\sigma^2}{2}(S^i)^2\left(\frac{V^{i+1}_{j+1}-V^i_{j+1}}{S^{i+1}-S^i}+\frac{V^{i-1}_{j+1}-V^i_{j+1}}{S^i-S^{i-1}}\right)+rS^i\frac{V^{i+1}_{j+1}-V^{i-1}_{j+1}}{2}-rV^i_{j+1}A^i \quad (1.38)$$

in the following. The final algorithm of the PDE solver is obtained by rearranging Equation (1.38) in a matrix $\mathbf{M}$ such that the linear Equation

$$\mathbf{M}\cdot\mathbf{V}_{j+1}=\mathbf{V}_j$$

with the known vector $\mathbf{V}_j=(V^1_j\dots V^m_j)$ and the unknown $\mathbf{V}_{j+1}=(V^1_{j+1}\dots V^m_{j+1})$ can be solved.

Note that the time stepping is conducted in backwards time τ_j, i.e. $\tau_0=t_T-0,\dots,\tau_T=0$. Consequently, the boundary for a put option valuation, i.e. the first and the last element of $\mathbf{V}_j$ can be assumed as

$$\begin{aligned} V^0_{j+1} &= V^0_j(1-r\Delta\tau),\ \text{i.e.}\ \frac{V^0_j-V^0_{j+1}}{V^0_j}=r\Delta\tau \\ V^m_j &= 0\ \forall j. \end{aligned}$$

Other boundary settings are useful, especially a property that the second derivative $\frac{\partial^2 V}{\partial S^2}$ should be zero is often used.

Exercisable Options

Previously, we saw that the valuation of exercisable options can be formulated (Equation (1.21) and (1.20)) as

$$\begin{aligned} V(S_t,t) &\geq P(S_t,t), \\ \frac{\partial V_t}{\partial t}+\frac{1}{2}\sigma^2S_t^2\frac{\partial^2V_t}{\partial S_t^2}+rS_t\frac{\partial V_t}{\partial S_t}-rV_t &\leq 0, \end{aligned}$$

where at least one of the inequalities holds with equality on the complete solution.

In order to solve this setting with a so called free boundary, we can use basically the same Equation as in the case of non-exerciseable options (Equation (1.38)):

$$A^i\left[\frac{\tilde V^i_{j+1}-\tilde V^i_j}{\Delta\tau}\right]=\frac{\sigma^2}{2}(S^i)^2\left(\frac{\tilde V^{i+1}_{j+1}-\tilde V^i_{j+1}}{S^{i+1}-S^i}+\frac{\tilde V^{i-1}_{j+1}-\tilde V^i_{j+1}}{S^i-S^{i-1}}\right)+rS_i\frac{\tilde V^{i+1}_{j+1}-\tilde V^{i-1}_{j+1}}{2}-r\tilde V^i_{j+1}A^i,$$

except that we simply set the node values V_{j+1}^i to the exercise value $P(S^i, t)$ if they are lower than the exercise value (see Wilmott [117, p. 906]):

$$\tilde{V}_{j+1}^i = \max(P(S^i, t), V_{j+1}^i).$$

This solver is sufficient for the purpose of this thesis. However, a better method is given by internal iterations using a penalty method [50].

Chapter 2

The Challenge of Path Dependency

2.1 Overview

A challenging problem in option pricing is the evaluation of path dependant options. This chapter presents a method which can increase the convergence of Monte Carlo pricing [21, 23, 53, 32, 81] significantly. The method is extended such that Monte Carlo simulation and numerical integration methods are combined in a consistent framework called Feature Extraction. As an example for the efficiency of the framework, the computational effort of pricing different types of Parisian and Asian style options is compared to the effort of classical Monte Carlo and PDE pricing. Especially, the fast pricing of a moving window Parisian option is presented using a PDE solver [103, 50] as an efficient tool for the required numerical integration.

While this chapter focuses on options which can only be exercised at maturity time, the later chapters will address options with an early exercise.

2.2 Introduction

For the numerical evaluation of option prices in the Black-Scholes models, three approaches come to mind, namely direct numerical integration, solving the Black-Scholes partial differential Equation (PDE), or Monte Carlo simulation, respectively. Whereas the first two methods are fast and accurate in many cases, they encounter considerable difficulties for path dependent options. If the path dependence involves more than just a single or few additional state variables, the PDE approach may be untractable altogether. Monte Carlo methods, on the other hand, allow for complex path dependencies (high-dimensional problems) but their efficiency is limited by their relatively low convergence rate as presented in the previous chapter.

In this chapter we suggest to combine elements of the Monte Carlo and the direct numerical integration in order to increase the accuracy of the former. We call this method Feature Extraction because the feature of the option's payoff, which is hard to compute by other means is estimated from a Monte Carlo simulation. This feature is then used in a numerical integration method,

which would not have been possible before. In this context, we use a PDE solver as an efficient tool for the numerical integration.

That means for the case of a path dependent options, we proceed in two steps. First, a Monte Carlo simulation is used to replace the given path dependent option with a European-style option which has the same price. In a second step, this hypothetical European option is evaluated by direct numerical integration or by solving the Black-Scholes PDE.

Why does this procedure lead to a higher accuracy? In many concrete cases, the value S_{t_T} of the underlying at expiration contributes considerably to the actual payoff of the option. The variability of the payoff among all paths of the underlying with terminal value S_{t_T} is often much smaller than the variability of the payoff among all conceivable asset price movements. The possibly slow and inaccurate Monte Carlo step in the Feature Extraction contributes only to the variability that cannot be explained by the asset's terminal value S_{t_T}, thus leading to a higher precision in the end.

The main idea of this chapter was first presented by Grau [55]. In contrast to this earlier work, this chapter provides more insight into the method and derives its correctness.

The chapter is organized as follows. In the next section we explain the general idea of the combined approach and compare the relative performance in the case of a discretely sampled Asian option. Subsequently, an iterative extension of the method is applied to moving window Parisian options. The last section concludes.

2.3 Pricing Using Feature Extraction

We denote by $S = (S_t)_{t\in[0,t_T]}$ the price process of an asset in the Black-Scholes model with constant volatility σ and risk-less interest rate r. Our aim is to price an option with a payoff $P(\mathbb{S})$, $\mathbb{S} := \{S_\tau | \tau \in \mathbb{I}\}$, $\mathbb{I} \subseteq \{t_0, t_1, \ldots, t_T\}$ at t_T which may depend on the whole path history of S. The fair initial price of the option is given by

$$V_{t_0} = e^{-rt_T}\mathbb{E}_Q[P(\mathbb{S})],$$

where the expectation is taken under the risk-neutral measure given an initial asset price S_{t_0}. If $\mathbb{E}_Q[P(\mathbb{S})|S_{t_T} = s]$ denotes the conditional expectation of the payoff given that the asset price terminates at s, it follows that

$$V_{t_0} = e^{-r(t_T)} \int_0^\infty \mathbb{E}_Q[P(\mathbb{S})|S_{t_T} = s] \cdot p^{S_{t_T}}(s)\,\mathrm{d}s, \tag{2.1}$$

where $p^{S_{t_T}}$ is the probability density function of S_{t_T} given an initial asset price S_{t_0}, (Equation (1.25)),

$$p^{S_{t_T}}(s) = \frac{1}{\sigma s\sqrt{2\pi t_T}} e^{-\frac{(\log(s/S_{t_0})+(r-(1/2)\sigma^2)t_T)^2}{2\sigma^2 t_T}}.$$

Now, we define the Feature Extraction as a method which separates the option price computation into the two parts in Equation (2.1), namely into the conditional expected payoff function

$\mathbb{E}_Q[P(\$)|S_{t_T} = s]$ and the corresponding probability density function $p^{S_{t_T}}(s)$. In the later Section, we will see Asian options, where $p^{S_{t_T}}(s)$ is known analytically and $\mathbb{E}_Q[P(\$)|S_{t_T} = s]$ has to be estimated numerically. Furthermore, we will see Parisian options, where $p^{S_{t_T}}(s)$ has to be estimated numerically and $\mathbb{E}_Q[P(\$)|S_{t_T} = s]$ is known at each time step of an induction in backwards time. Feature Extraction means to use as much analytical information in an option pricing process as possible.

Suppose that $p^{S_{t_T}}(s)$ is known and $\mathbb{E}_Q[P(\$)|S_{t_T} = s]$ is not known. The intuition behind this method is that we interpret

$$\widetilde{f}(s) := \mathbb{E}_Q[P(\$)|S_{t_T} = s]$$

as the payoff of a hypothetical European-style option, which is computed by Monte Carlo simulation. In a second step, the integral in (2.1) is evaluated directly or by numerical solution of the corresponding PDE. As noted in the introduction, the conditional variance $\mathrm{var}(P(\$)|S_{t_T} = s)$ is typically much lower than the total variance $\mathrm{var}(P(\$))$ which leads to a higher precision compared to an unconditional Monte Carlo simulation.

From the way this algorithm works by integrating the payoff of a hypothetical European option, it is clear that an extension to options with early exercise features is not easy. Another obvious limitation is that the probability density function of the terminal asset price distribution has to be known with a high precision, in a symbolic form at best. Apart from these limitations, the pricing of any option with a payoff that can be computed by a simple forward simulation and an underlying price process with a known probability density can profit from the presented approach.

Let us illustrate the approach in the case of a discretely sampled Asian option in the next section followed by a Parisian option.

2.3.1 A Discretely Sampled Asian Option

An Asian option is a European option with a payoff $P(\$)$ that depends on the average I of the past stock prices and the strike K. For a discretely sampled Asian call option with sample dates $t_0 = 0, t_1, \ldots, t_T$, the payoff P is

$$P(\$) = \max(I(\$) - K, 0),$$

for an Asian call option resp.

$$P(\$) = \max(K - I(\$), 0)$$

for an Asian put option with

$$I(\$) = \frac{1}{T+1} \sum_{i=0}^{T} S_{t_i}.$$

As presented in Chapter 1, in the classical Monte Carlo pricing method, one would estimate the discounted expected payoff by the mean of the payoff of n simulated asset paths

$$\begin{aligned} V_{t_0} &= e^{-r(t_T)}\mathbb{E}_Q[P(\mathbb{S})] \\ &\approx e^{-r(t_T)}\frac{1}{n}\sum_{j=1}^{n} P(\mathbb{S}^j) \end{aligned}$$

with

$$P(\mathbb{S}^j) = \max\left(\left(\frac{1}{T+1}\sum_{i=0}^{T} S_{t_i}^j\right) - K, 0\right),$$

for the Asian call,

$$P(\mathbb{S}^j) = \max\left(K - \left(\frac{1}{T+1}\sum_{i=0}^{T} S_{t_i}^j\right), 0\right),$$

for the Asian put with

$$S_{t_i}^j = S_{t_{i-1}}^j e^{(r-\frac{1}{2}\sigma^2)(t_i - t_{i-1}) + \sigma\sqrt{t_i - t_{i-1}}\theta_{i,j}}, \tag{2.2}$$

where $\theta_{i,j}$, $i = 1 \dots T$, $j = 1 \dots n$, denotes independent realizations of a random variable drawn from a standard normal distribution.

The value V_{t_0} is the no-arbitrage price of the standard Black-Scholes option price model.

As explained in the previous section (see Equation (2.1)), the Asian option pricing problem can be divided into an expected payoff function and the probability density function $p^{S_{t_T}}$ of the asset price at maturity. The PDF is known (see Equation (1.25)) and the expected payoff function can be estimated by Monte Carlo simulations.

In order to compute the expected payoff function in the case of an Asian option, we have to compute the payoff for all possible paths starting at S_{t_0}. Consider a Monte Carlo simulation with stock price paths $\mathbb{S}^j$ and payoffs $P(\mathbb{S}^j)$, $j = 1 \dots n$. As in Section 1.2.1, an estimate for the conditional payoff function $\widetilde{f} \approx \mathbb{E}_Q[P(\mathbb{S})|S_{t_T} = s]$ is generated by a regression on basis functions. In this case, a B-spline $f^{\text{spline}}(S_{t_T}) = \sum_k a_k b_k(S_{t_T})$ proves to be useful. The required regression is given by

$$\min_{f^{\text{spline}}} \sum_{j=1}^{n} \|P(\mathbb{S}^j) - f^{\text{spline}}(S_T^j)\|_2^2 \Leftrightarrow \min_{a_1 \dots, a_m} \sum_{j=1}^{n} \left\| P(\mathbb{S}^j) - \sum_{k=1}^{m} a_k b_k(S_{t_T}) \right\|_2^2 \tag{2.3}$$

i.e.

$$\widetilde{f}(s) = \mathbb{E}_Q[P(\mathbb{S})|S_{t_T} = s] \approx f^{\text{spline}}(s). \tag{2.4}$$

This can easily be done by using the local basis approximation as presented in Chapter 1. The regression is only one dimensional – on the asset price at maturity, so that the spline is the best

choice for the basis. Since this least-squares regression leads to an estimate for the conditional expectation (see Section 1.2.1), this kind of regression exactly produces the results we need. Due to the smoothness of the solution, already a small number of cubic spline basis functions (3-5) generate an acceptable accuracy. A simple implementation can be found in Appendix 7.5.

Figure 2.1 demonstrates the smoothing effect for an Asian option by comparing the conditional payoff of Monte Carlo simulations with the spline approximation of the expected conditional payoff function. On the left, we can see the terminal distribution of the asset price as a histogram and the realized payoff values as dots. Performing a regression on the data of these dots, we obtain the line for the conditional expected value of the payoff on the right of the figure. Instead of using the empirical distribution of the asset price as demonstrated by the histogram on the left, the Feature Extraction now uses the analytic solution shown as probability density on the right.

An alternative to a regression on the asset price is the simulation of only very specific asset paths. Using Brownian bridges, one can basically choose the terminal asset price for each paths. That means, one can determine the conditional payoff on a grid of the values of S_{t_T} and interpolate the values to obtain $\widetilde{f}(S_{t_T})$.

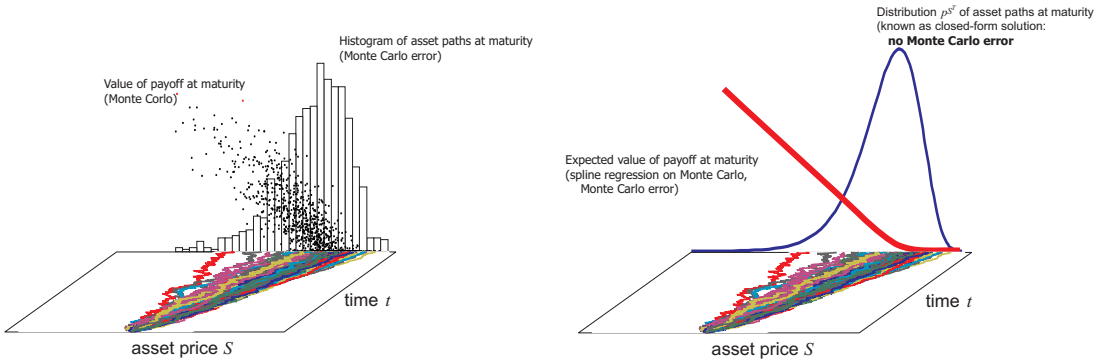

Figure 2.1: *On the left, an Asian option with the conditional payoff of each asset path and the path distribution histogram is presented. The right figure demonstrates the smoothing effect taking place using the PDF and the approximation of the expected conditional payoff function.*

2.3.2 Simple Example

To clarify the algorithm, this section will proceed in a step by step fashion and explain every computational task of the evaluation of the discretely sampled Asian option with the Feature Extraction method. Consider an Asian option with data in Table 2.1. This is a simple Asian put option with three observation dates for the averaging: $t_0 = 0$, $t_1 = 0.5$ and $t_T = t_2 = 1$.

We start the evaluation by simulating the underlying's paths using Equation (2.2)

$$S^j_{t_i} = S^j_{t_{i-1}} e^{(r-\frac{1}{2}\sigma^2)(t_i - t_{i-1}) + \sigma\sqrt{t_i - t_{i-1}}\theta_{i,j}}$$

Table 2.1 *Data of an Asian put option with three averaging sample dates.*

General features	
Independent variable I	$\frac{1}{3}\sum_{i=0}^{2} S_{t_i}$
strike price K	100
risk-free rate r	5% p.a.
current stock price S_{t_0}	100
volatility σ	40% p.a.
maturity time t_T	1 year
observations	every 0.5 years
Payoff at Maturity P	$\max(K - I, 0)$

and the data in Table 2.1. With $S^j_{t_0} = 100$ and random numbers $\theta_{i,j}$, $j = 1\ldots, 10$, $i = 1, 2$, we get 10 asset paths:

j	$S^j_{t_0}$	$S^j_{t_1}$	$S^j_{t_2}$
1	100	81.6340	61.7521
2	100	120.7011	86.3528
3	100	89.1249	84.7387
4	100	118.4613	87.9178
5	100	104.5817	118.0245
6	100	81.0836	86.1567
7	100	58.9702	40.8700
8	100	101.6986	72.1828
9	100	65.8471	65.5679
10	100	119.6538	133.2725

Now, we can compute the payoff of the Asian option for each paths. With

$$P^j(S^j_{t_0}, S^j_{t_1}, S^j_{t_2}) = \max\left(100 - \frac{1}{3}\sum_{i=0}^{2} S^j_{t_i}, 0\right), \quad j = 1, \ldots, 10$$

we obtain

$$\mathbf{P} := \mathbf{P}(S_{t_0}, S_{t_1}, S_{t_2}) = \begin{pmatrix} 18.8713 \\ 0.0000 \\ 8.7121 \\ 0.0000 \\ 0.0000 \\ 10.9199 \\ 33.3866 \\ 8.7062 \\ 22.8616 \\ 0.0000 \end{pmatrix}$$

which can already be used for a price estimate following the traditional Monte Carlo pricing method (cp. Section 1.4.2):

$$\begin{aligned} V_{t_0} &\approx e^{-rt_T}\frac{1}{10}\sum_{j=1}^{10} P^j \\ &\approx e^{-0.05\cdot 1}\cdot 10.3458 \\ &\approx 9.8412. \end{aligned}$$

However, we want to compute the estimate for the option price using the Feature Extraction, where we need a local basis approximation of $\mathbb{E}[P|S_{t_T}]$ which we compute following Theorem 1.4 and Lemma 1.5. For this regression estimate, we need to define our basis functions $b_1(\mathbf{x}),\ldots,b_m(\mathbf{x})$. In this simple case we choose polynomials up to the power of two in $\mathbf{x} := S_{t_T}$ such that $b_1(S_{t_T}) = 1$, $b_2(S_{t_T}) = S_{t_T}$ and $b_3(S_{t_T}) = S_{t_T}^2$. Consequently, the matrix $\mathbf{B} := \mathbf{B}(S_{t_T})$ in the regression is

$$\mathbf{B} = \left(1 \quad S_{t_T}^j \quad (S_{t_T}^j)^2\right)\Big|_{j=1,\ldots,10},$$

which is

$$\mathbf{B} = \begin{pmatrix} 1 & 61.7521 & 3813.3171 \\ 1 & 86.3528 & 7456.8073 \\ 1 & 84.7387 & 7180.6524 \\ 1 & 87.9178 & 7729.5404 \\ 1 & 118.0245 & 13929.7926 \\ 1 & 86.1567 & 7422.9713 \\ 1 & 40.8700 & 1670.3548 \\ 1 & 72.1828 & 5210.3516 \\ 1 & 65.5679 & 4299.1531 \\ 1 & 133.2725 & 17761.5684 \end{pmatrix}$$

in this particular case. That means, that the local basis approximation is given by

$$\tilde{f}(S_{t_T}) = \sum_{k=1}^{3} \tilde{a}_k b_k(S_{t_T})$$

with

$$\begin{aligned} (\tilde{a}_1 \quad \tilde{a}_2 \quad \tilde{a}_3)^T &= \left(\mathbf{B}^T\mathbf{B}\right)^{-1}\mathbf{B}^T\mathbf{P} \\ &= \mathbf{B}^{\dagger}\mathbf{P} \\ &= (81.70 \quad -1.375 \quad 0.005712)^T. \end{aligned}$$

We use this result for the pricing with Equation (2.1), i.e.

$$\begin{aligned} V_{t_0} &= e^{-rt_T} \int_0^\infty f(s) \cdot p^{S_{t_T}}(s)\, \mathrm{d}s \\ &\approx e^{-rt_T} \int_0^\infty \tilde{f}(s) \cdot p^{S_{t_T}}(s)\, \mathrm{d}s \\ &\approx \int_0^\infty e^{-rt_T} \cdot (81.70 - 1.375 \cdot s + 0.005712 \cdot s^2) \cdot \frac{1}{\sigma s \sqrt{2\pi t_T}} e^{-\frac{(\log(s/S_{t_0}) + (r - (1/2)\sigma^2)t_T)^2}{2\sigma^2 t_T}}\, \mathrm{d}s. \end{aligned}$$

The last integral can now be evaluated with numerical methods. In this simple case, we use the trapezoidal rule (see Voss [114, p.47]) with equidistant nodes:

s	$f(s)$	$p^{S_{t_T}}(s)$	F(s):=$e^{-rt_T} f(s) \cdot p^{S_{t_T}}(s)$
25	50.9062	0.0001372	0.0062
50	27.2489	0.0050471	0.1308
75	10.7312	0.0108062	0.1103
100	1.3531	0.0099545	0.0128
125	-0.8852	0.0065309	-0.0055
150	4.0161	0.0036761	0.0140
175	16.0571	0.0019223	0.0294
200	35.2378	0.0097297	0.0326
225	61.5581	0.0004863	0.0285
250	95.0181	0.0002430	0.0220
275	135.6178	0.0001222	0.0158
300	183.3572	0.0000621	0.0108
325	238.2362	0.0000319	0.0072
350	300.2549	0.0000167	0.0048
375	369.4132	0.0000088	0.0031
400	445.7113	0.0000047	0.0020
425	529.149	0.0000026	0.0013
450	619.7264	0.0000014	0.0008
475	717.4434	0.0000008	0.0005
500	822.3002	0.0000004	0.0004

which leaves us with

$$V_{t_0} \approx 25 \left(0.5 \cdot (F(25) + F(500)) + \sum_{i=2}^{19} F(i \cdot 25) \right) \approx 10.69$$

as the Feature Extraction estimate for the Asian option with data in Table 2.1. This value is far from the true value of 6.97 (estimated with 10.000.000 paths) and even further away than the traditional Monte Carlo method with 9.84. But, as we will see in the next section, in realistic settings with many paths and more regression basis functions the Feature Extraction method converges faster to the true solution and delivers more accurate estimates than the traditional Monte Carlo method.

Table 2.2 *Specifications of an Asian option.*

Option type	Asian arithmetic average
Independent variable I	$\frac{1}{124}\sum_{i=1}^{124} S_{t_i}$
Strike K	100
Payoff at Maturity P	$\max(I-K,0)$
Maturity T	0.5 years
Risk free rate r	5% p.a.
Volatility σ	25% p.a.
Daily observations $\Delta_{\text{obs}}t$	1/250 years
no observations at	$t=t_0$, $t=t_T$
Initial asset price S_{t_0}	100

2.3.3 Numerical Examples

The examples in this section are discretely observed Asian call options based on the specifications in Table 2.2.

The value of a classical PDE solution of the Asian option with specifications in Table 2.2 is 4.646 (Grid with 1600 nodes in S and 4000 time steps, all four digits correct). All subsequently reported errors are relative to this value.

In the case of Asian options, the antithetic variables and the variance reduction improve a Monte Carlo estimate significantly. See Boyle [21, 23] for more information on variance reduction and antithetic variables. These improvements can be combined with the method of this chapter.

Figure 2.2 shows the frequency distribution of the error with the different Monte Carlo methods, compared with the new method. In both cases, the antithetic variables and the antithetic variables with control variable, the error distribution of the new method is smaller. The standard deviation for each of the methods is a half using the new method compared with the classical Monte Carlo. That means the new method is an additional improvement of Monte Carlo simulations.

In order to analyze the effect of the Feature Extraction a little further, we are going to separate the error of a Monte Carlo estimate (cp. Equation 1.24)

$$\text{Err} := \frac{\text{std}[\overline{V}^n]}{\sqrt{n}}$$

into separate effects. From (Equation (2.1))

$$V_{t_0} = e^{-r(t_T)} \int_0^\infty \mathbb{E}_Q[P(\$)|S_{t_T}=s]\cdot p^{S_{t_T}}(s)\,\mathrm{d}s$$

we know that we can separate the errors into errors for the estimation of $\tilde{f}(s)\approx\mathbb{E}_Q[P(\$)|S_{t_T}=s]$ and errors for the estimation of $\tilde{p}^{S_{t_T}}(s)\approx p^{S_{t_T}}(s)$. We do not discuss an empirical estimation of a probability density function (PDF) here so that we refer to [20] for details.[1] However, the analytic

[1] The following examples are computed using the Matlab command ksdensity.

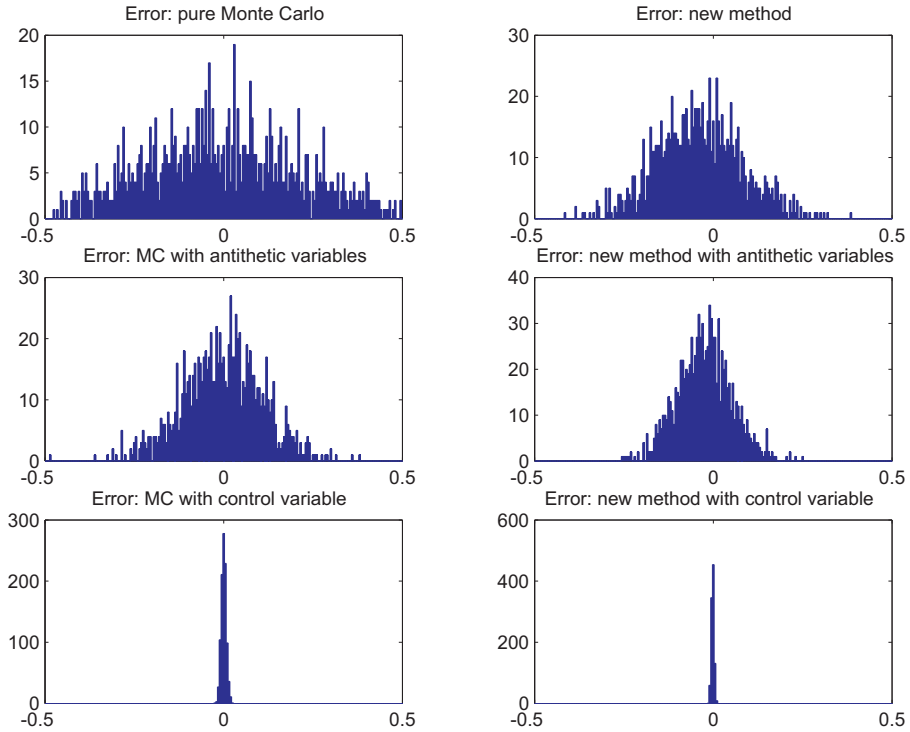

Figure 2.2: *The Asian option with data in Table 2.2: Histograms of the error distribution of different Monte Carlo simulations. The pure Monte Carlo method uses 1,000 asset paths, the antithetic variables uses the same paths twice: once positive, once negative. The variance reduction uses the paths with antithetic variables and the geometric averaging Asian option as correction. The reference value for this asian option is* 4.646.

conditional expectation $f(s)$ for the Asian option in Table 2.2 is not known, which is a problem for the computation of Equation (2.1). Consequently, we compute a highly accurate estimate with 10.000.000 asset paths and use the result instead of an analytic formula for $f(s)$. A corresponding implementation can be found in Appendix 7.5.

Table 2.3 summarizes the results of the different methods for pricing an Asian option with data in Table 2.2. All values are computed using the same set of asset paths, the values in the column A (Reference MC) are values computed with a pure Monte Carlo method. The mean value is computed using 10,000 valuations with 10,000 asset paths simulations each. This mean value is 4.655 ± 0.001 with 95% confidence. This value differs from the reference PDE value by about 0.01 because the Monte Carlo method uses only 125 time steps for the averaging while the PDE method uses 4000.

The next column contains the corresponding values where the option value is based on the Feature Extraction, i.e. the integration with an estimated conditional expectation function $\tilde{f}(s)$. Columns follow, where the integration is also conducted using an estimated probability density $\tilde{p}^{S_{t_T}}(s)$ and combinations of both. The standard deviation (Std) and thus the error of the Fea-

ture Extraction with an analytic PDF and an estimate for the conditional expected payoff function (column B) is much smaller than the value of Reference MC. This was expected since this demonstrates the effectiveness of the method. The Std for integration based on the PDF estimation (column C) is also smaller than the value of Reference MC. This is expected because the conditional expectation in this integration was estimated from a highly accurate estimation with 10,000,000 asset paths. If we now focus on column D, where both functions $\tilde{p}^{S_{t_T}}(s)$ and $\tilde{f}(s)$ are estimated from the sample, we see that the Std is very close to the value of the Reference MC. Again, this is expected because besides smoothing, no additional information was added to the valuation process.

The question arises, how the Std values of the different methods are connected. In the following we give some empirical intuition for the interdependency of the different Monte Carlo errors using the obtained numerical values. Let us assume that the Monte Carlo error Err can be decomposed into

$$\begin{aligned} \text{std}(\text{Err}) &= \text{std}(\text{Err}_p + \text{Err}_f) \\ &= \sqrt{\text{std}(\text{Err}_p)^2 + \text{std}(\text{Err}_f)^2 + 2 \cdot \text{cov}(\text{Err}_p, \text{Err}_f)} \end{aligned}$$

where Err_f is the error due to the estimation of $\tilde{f}(s)$ in Equation (2.4) and Err_p is the error due to the estimation of the probability density $p^{S_{t_T}}$. In our example, we can obtain the numerical estimate for $\text{Err}_p = 0.05700$, for $\text{Err}_f = 0.03701$ and for $\text{Err} = 0.06758$ from row "Std" of Table 2.3. A numerical estimate for the covariance of the two errors $\text{cov}(\text{Err}_p, \text{Err}_f)$ is 0 ± 0.000001, which is effectively zeros. Now, computing

$$\sqrt{\text{std}(\text{Err}_p)^2 + \text{std}(\text{Err}_f)^2 + 2 \cdot \text{cov}(\text{Err}_p, \text{Err}_f)} \approx \sqrt{0.05700^2 + 0.03701^2 + 2 \cdot 0} \tag{2.5}$$

$$\approx 0.06796 \tag{2.6}$$

this corresponds surprisingly well to the value of Reference MC (0.06758). Therefore, it seems to be plausible that the different components of the error add up linearly and that they are uncorrelated. A further detailed analysis of this error splitting remains open to further research, since it is beyond the scope of this dissertation.

However, this numerical example demonstrates that the Feature Extraction really benefits from splitting the error of a Monte Carlo simulation into an error for the expected conditional payoff function and the probability density function. Using the Feature Extraction uses an analytic expression for the probability density function and thus only the error for estimating the expected conditional payoff remains.

Table 2.3 *Values and error estimates are presented of different ways estimating the value of the Asian option in Table 2.2 with $S_{t_0} = 100$, and $n = 10,000$ simulated asset paths. Under Reference MC, the prices were computed using the regular Monte Carlo technique. The other columns are computed using Equation* (2.1) *with estimates for the pdf $p^{S_{t_T}}(s) \approx \tilde{p}^{S_{t_T}}(s)$ resp. the conditional expectation $f(s) \approx \tilde{f}(s)$. Std is the standard deviation of a series of 10.000 option valuations and thus an expected error for a single valuation.*

	A	B	C	D	E
	Reference MC	$\tilde{f}(s)$	$\tilde{p}^{S_{t_T}}(s)$	$\tilde{p}^{S_{t_T}}(s)$ and $\tilde{f}(s)$	highly accurate f(s)
Mean	4.655	4.652	4.687	4.687	4.652
Std	0.06758	0.03701	0.05700	0.06788	0
Systematic error	-	<0.001	≈0.03	≈0.03	<0.001

2.3.4 Summary of the Feature Extraction for European Path Dependent Options

For discretely observed European style path dependent options, the new method can be summarized as follows:

1. Compute the local basis approximation of the expected payoffs $\widetilde{f}(S_{t_T}) = \mathbb{E}_Q[P(\mathbb{S})|S_{t_T} = s]$ for all possible asset path histories $\mathbb{S} := \{S_\tau | \tau \in \mathbb{I}\}$, $\mathbb{I} \subseteq \{t_0, t_1, \ldots, t_T\}$ and terminal asset price S_{t_T} at maturity time t_T. This can be done by Monte Carlo simulations, starting at S_{t_0} and Theorem 1.4.

2. Use the Black-Scholes Equation to solve for the price V_{t_0}, e.g. by a finite differences time stepping of
$$\frac{\partial V(S_t,t)}{\partial t} + \frac{1}{2}\sigma^2 S_t^2 \frac{\partial^2 V(S_t,t)}{\partial S_t^2} + rS\frac{\partial V(S_t,t)}{\partial S_t} - rV(S_t,t) = 0$$
with $\widetilde{f}(S_{t_T})$ as terminal condition. Or solve the Black-Scholes Equation by using the distribution of S_{t_T}, i.e.
$$p^{S_{t_T}}(s) = \frac{1}{\sigma s\sqrt{2\pi(T)}} e^{-\frac{(\log(s/S_{t_0})+(r-(1/2)\sigma^2)t_T)^2}{2\sigma^2 t_T}}.$$
The option price V_{t_0} is then given by
$$V_{t_0} = e^{-rt_T}\int_0^\infty p^{S_{t_T}}(s)\widetilde{f}(s)\,\mathrm{d}s.$$

2.4 Pricing Delayed Barrier Options

The Feature Extraction method can be used for pricing Parisian options with different kinds of knock-out or knock-in conditions. In order to use less computations in the Monte Carlo simulation one can extend the method from one expected payoff function at time T to expected payoff functions at all discrete observation times with different probabilities for each expected payoff function to occur. The resulting algorithm is based on PDE time stepping in order to integrate the expected payoff functions at the different times consistently. In contrast to the previous example, the probability density function of the terminal asset price is unknown and the conditional expected payoff function is known at each time step of the Parisian option observation. This was reversed in the pricing an Asian option.

Definition 2.1 *We define Parisian options as follows:*

*(i) A **consecutive counting Parisian option** is an option which becomes worthless if the underlying stock stays M consecutive days above a barrier level S_B.*

*(ii) A **cumulative counting Parisian option** is an option which becomes worthless if the underlying stock stays M days above a barrier level S_B since the initialization of the Parisian option.*

*(iii) A **moving window Parisian option** is an option which becomes worthless if the underlying stock stays M out of the last N days above a barrier level S_B.*

Note that the moving window Parisian option is a generalization of the consecutive and cumulative counting Parisian option. The consecutive counting Parisian is equivalent to a moving window Parisian option with $N = M$. The cumulative counting Parisian is equivalent to a moving window Parisian with $N \to \infty$.

In the following we will consider a moving window Parisian call option. At each time step we want to consider the fraction of options which is knocked out separately from the options still alive.

In the context of moving window Parisian options we apply the method in the previous sections recursively in time. The payoff function of the option at expiration t_T can be represented as

$$f(\mathbb{S}) = \max(S_{t_T} - K, 0) \cdot I(\mathbb{S}, t_T),$$

where $I(\mathbb{S}, t)$ denotes an indicator variable which equals 0 if the option has been knocked out up to (and including) time t and 1 if it is still alive at t. Obviously, this indicator depends in a complex way on the whole path history $\mathbb{S} := \{S_\tau | \tau \in \mathbb{I}\}$, $\mathbb{I} \subseteq \{t_0, t_1, \ldots, t_T\}$ of the stock price process S. In order to evaluate the initial fair price

$$V_{t_0} = e^{-rT} \mathbb{E}_Q[f(\mathbb{S})],$$

we suggest to compute the conditional expectations

$$\mathbb{E}_Q[f(\$)|S_{t_i} = s, I(\$, t_{i+1}) = 1] \tag{2.7}$$

and

$$\mathbb{E}_Q[f(\$)|S_{t_i} = s, I(\$, t_i) = 1] \tag{2.8}$$

recursively for $i = T-1, \ldots, 0$. Since

$$\mathbb{E}_Q[f(\$)] = \mathbb{E}_Q[f(\$)|S_{t_0} = S_{t_0}, I(\$, t_0) = 1],$$

this eventually leads to the fair option price.

For the first Step (2.7) suppose that

$$\mathbb{E}_Q[f(\$)|S_{t_{i+1}} = s, I(\$, t_{i+1}) = 1]$$

is known by recursion. Note that this is definitely true for $i = T-1$ because

$$\begin{aligned} &\mathbb{E}_Q[f(\$)|S_{t_T} = s, I(\$, t_T) = 1] \\ = \mathbb{E}_Q[\max(S_{t_T} - K, 0) \cdot I(\$, t_T)|S_{t_T} &= s, I(\$, t_T) = 1] \\ &= \max(s - K, 0). \end{aligned}$$

Since S is a Markov process, we have

$$\begin{aligned} &\mathbb{E}_Q[f(\$)|S_{t_i} = s, I(\$, t_{i+1}) = 1] \\ = &\int \mathbb{E}_Q[f(\$)|S_{t_{i+1}} = \widetilde{s}, I(\$, t_{i+1}) = 1] \cdot p^{S_{t_{i+1}}|S_{t_i}=s}(\widetilde{s})\, \mathrm{d}\widetilde{s}, \end{aligned} \tag{2.9}$$

where $p^{S_{t_{i+1}}|S_{t_i}=s}(\widetilde{s})$ denotes the conditional probability density function of $S_{t_{i+1}}$ given that $S_{t_i} = s$, i.e.

$$p^{S_{t_{i+1}}|S_{t_i}=s}(\widetilde{s}) = \frac{1}{\sigma\widetilde{s}\sqrt{2\pi(t_{i+1} - t_i)}} e^{-\frac{(\log(\widetilde{s}/S_{t_i}) + (r - (1/2)\sigma^2)(t_{i+1} - t_i))^2}{2\sigma^2(t_{i+1} - t_i)}}.$$

For the second step, we need an estimate of the conditional probability

$$P_{i,i+1}(s) := \text{Prob}\left(I(\$, t_{i+1}) = 1 | S_{t_i} = s, I(\$, t_i) = 1\right), \tag{2.10}$$

i.e. the probability of survival until time t_{i+1} if the option has not been knocked out until t_i and the underlying price equals s. This conditional probability is determined in the Monte Carlo step of our approach. It is the only instance where simulation is actually needed. Using (2.10) we can determine (2.8) by

$$\begin{aligned} &\mathbb{E}_Q[f(\$)|S_{t_i} = s, I(\$, t_i) = 1] \\ = \;&\mathbb{E}_Q[f(\$)|S_{t_i} = s, I(\$, t_{i+1}) = 1] \cdot P_{i,i+1}(s) \\ &+ \underbrace{\mathbb{E}_Q[f(\$)|S_{t_i} = s, I(\$, t_{i+1}) = 0]}_{=0} \cdot (1 - P_{i,i+1}(s)) \\ = \;&\mathbb{E}_Q[f(\$)|S_{t_i} = s, I(\$, t_{i+1}) = 1] \cdot P_{i,i+1}(s). \end{aligned} \tag{2.11}$$

For many path dependent options, the Feature Extraction can be summarized as follows

1. Compute the probabilities $P_{i,i+1}(s)$ of survival for all values of s and i. This can be done e.g. by Monte Carlo simulations, starting at the asset value S_{t_0}.

2. Use numerical integration, or better, a numerical solver for the Black-Scholes PDE to compute
$$\mathbb{E}_Q[f(\mathbb{S})|S_{t_i} = s, I(\mathbb{S}, t_{i+1}) = 1]$$
from $\mathbb{E}_Q[f(\mathbb{S})|S_{t_{i+1}} = s, I(\mathbb{S}, t_{i+1}) = 1]$ as in Equation (2.9).

3. Compute $\mathbb{E}_Q[f(\mathbb{S})|S_{t_i} = s, I(\mathbb{S}, t_i) = 1]$ from
$$\mathbb{E}_Q[f(\mathbb{S})|S_{t_i} = s, I(\mathbb{S}, t_{i+1}) = 1]$$
and $P_{i,i+1}(s)$ using Equation (2.11) and go back to step 2 if $i \neq 0$.

4. The price of the path dependent option is given by
$$V_{t_0} = e^{-rt_T}\mathbb{E}_Q[f(\mathbb{S})] = e^{-rt_T}\mathbb{E}_Q[f(\mathbb{S})|S_{t_0} = S_{t_0}, I(\mathbb{S}, t_0) = 1].$$

This algorithm can be extended to Parasian, lookback or similar options without large efforts.

2.4.1 Numerical Example: A Parisian Option

In this section, the efficiency of the new method will be compared with the classical Monte Carlo method. Table 2.4 provides the data of the moving window Parisian option used for the calculations.

Table 2.4 *Specifications of a Parisian option.*

Option type	Parisian up-and-out
Payoff at Maturity f	$\max(S_{t_T} - K, 0)$
Strike K	100
Maturity t_T	0.25 years
Risk free rate r	5%
Volatility σ	25%
Barrier Level S_B	120
Daily observations $\Delta_{\text{obs}}t$	1/250 years
Length of observation period N	15 $\Delta_{\text{obs}}t$
Number of observations to event M	5 $\Delta_{\text{obs}}t$
no knock out at	$t = 0$, $t = t_T$

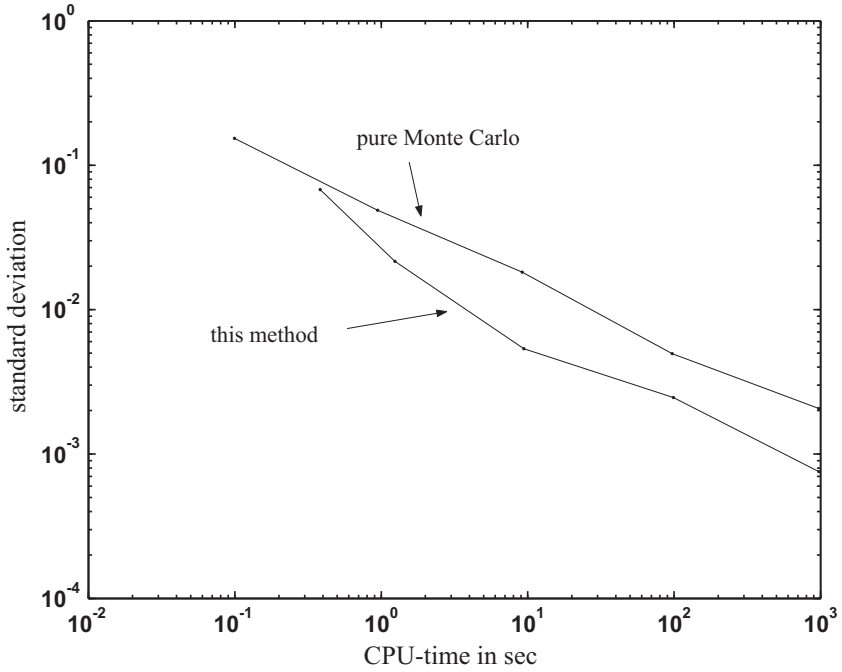

Figure 2.3: *The standard deviation of the value for a Parisian option with specifications in Table 2.4 is plotted versus CPU-time. The standard deviation for each CPU-time is computed using 100 runs of different Monte Carlo simulations. The total number of asset paths reach from about 1,000 to 15,000,000 (PDE discretization: 800 to 3200 nodes in the S, 480 to 1920 time steps, $P_{i,i+1}$-grid: spacing in S direction: 800 to 12800 nodes). The CPU time is the run time of a C/C++ implementation on an Intel Xeon 1.7 GHz computer.*

For this example of the Parisian options, the values of $1 - P_{i,i+1}$ are presented in Figure 2.4. The values are estimated using a Monte Carlo simulation. The options price V_{t_0} is computed using these probabilities, and a linear interpolation between the nodes.

In order to get an idea of the improvement of convergence, a confidence interval for the price of the Parisian option is computed. With a traditional Monte Carlo method and with the Feature Extraction method. In the this example, the Feature Extraction uses a PDE solver for the integration with the probability density function. This is very efficient for the treatment of many sub steps, which are required in the Parisian option case.

In Figure 2.3 the confidence interval is given as the standard deviation of different runs of the pricer.[2] The standard deviation of 100 runs of the new method and the classical Monte Carlo method is presented. The Figure shows that, the Feature Extraction has about half the standard deviation compared with the pure Monte Carlo simulation.

An interesting property one can observe at Figure 2.3 is that the slope of the log(standard deviation) versus log(CPU-time) is about -0.5 of all three methods. That means that the standard deviation is approximately proportional to $(\text{CPU-time})^{-0.5} = \frac{1}{\sqrt{\text{CPU-time}}}$. This result was expected

[2]This corresponds to 68% probability that the option value is within $\overline{V} \pm$ standard deviation (cp. Table 1.2).

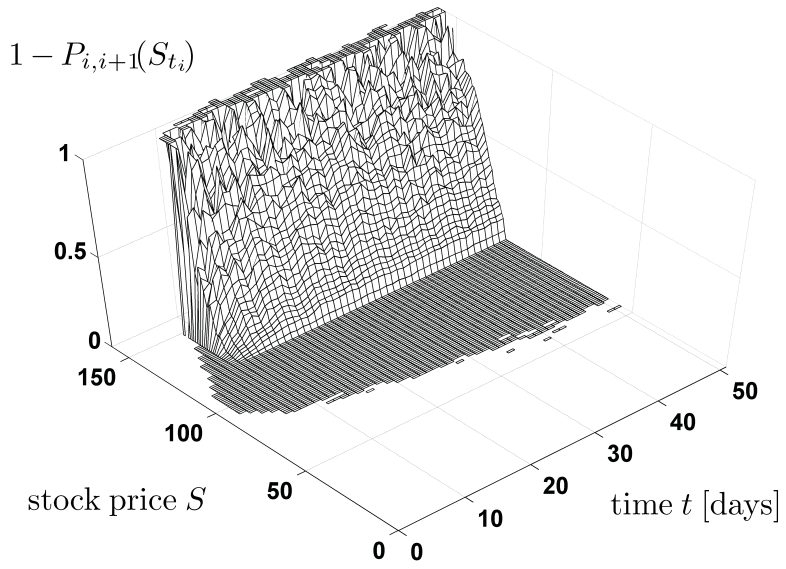

Figure 2.4: *Value of the knock out conditional probability* $1 - P_{i,i+1}(S)$ *of a Parisian option with specifications in Table 2.4 are plotted versus the asset price and time step.*

for the pure Monte Carlo method because we found for this kind of Monte Carlo pricing that is converges with $\frac{1}{\sqrt{\text{number of paths}}}$ (Equation (1.24)) and the CPU-time is proportional to the number of paths. The numerical PDE solution required by the new method needs only a small fraction of time compared to the Monte Carlo sampling thus it has only a minor effect on the CPU-time.

2.5 Summary

This chapter presents a new framework for the valuation of exotic path dependent options, which we call Feature Extraction. The new framework presented is based on separating the pricing problem into two parts. One part with high complexity is solved by a Monte Carlo method, and a second part with low complexity is solved by standard numerical tools (numerical integration, PDE solution). The only problem arising is that even though a PDE method can be used in the process, the Feature Extraction is not easily extended to pricing options with early exercise feature.

Values for different kinds of complex derivatives can be computed. The numerical convergence studies show that the new method is capable of a precise pricing of moving window Parisian options. While it is practically impossible for a pure PDE method to handle a moving window Parisian option with long windows (> 20 observations, see [55]), the new solution can deal with this problem.

The improvement of convergence for Asian options using the new method is comparable with

the improvement for Parisian options. Furthermore, the improvements by the new method can be combined with classical Monte Carlo improvements like antithetic variables and importance sampling in order to increase convergence.

Chapter 3

Moving Window Asian Options

3.1 Overview

In the previous chapter, we saw a method which can increase the speed of pricing path dependent options which do not allow early exercise. Now, we want to see, how regression methods can help pricing exercisable path dependent options.

The pricing of moving window Asian options with an early exercise feature is considered as one of the most complex problems in numerical finance. The computational challenge is created by the unknown optimal exercise strategy and the high dimensionality that is required for its approximation. We use the Least-Squares Monte Carlo approach together with Sparse Grid type basis functions to combine two simple and well established methods. The resulting algorithm provides a convergent and practical method for pricing the moving window Asian options as well as other high-dimensional, exercisable securities, which to our knowledge have not yet been solved with reasonable accuracy.

3.2 Introduction

Methods for pricing a large variety of exotic options have been developed in the past decades. Still, the pricing of high-dimensional American style options remains challenging. The price of this kind of option depends on the complete price path not only on the stock price at the final exercise date. In this chapter, we consider the price of a moving window Asian option (MWAO) with discrete and continuous observations for the computation of the early exercise value[1]. The early exercise value of the MWAO depends on the average value of the underlying stock over a moving period of time, which means that a continuous observation leads to an infinite dimensional problem.

[1]Note that we skip the American in the name for the moving window Asian option with early exercise. We do this, because a moving window would be useless without an early exercise or a knock-out feature. In the remainder of this thesis, we will only consider MWAOs with an early exercise.

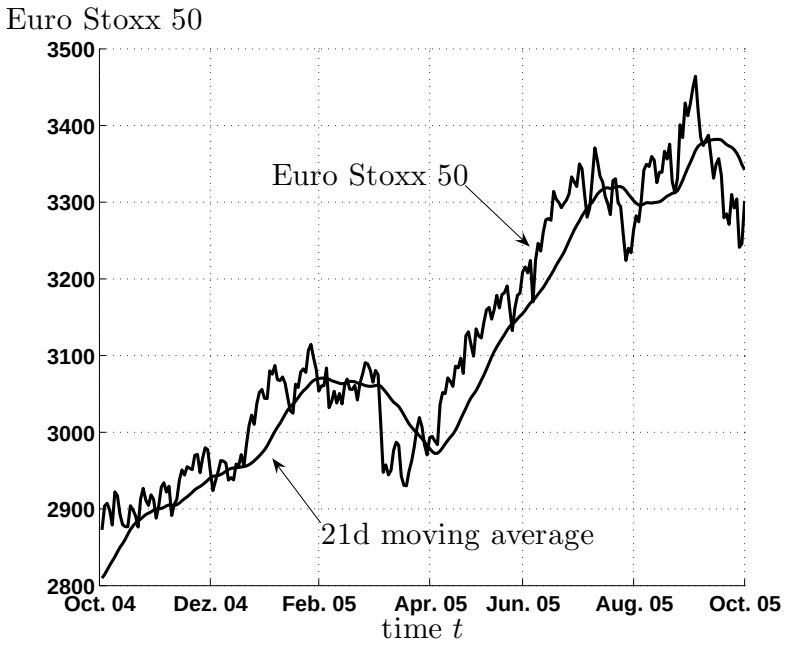

Figure 3.1: *Although being a popular tool for chart analysts, pricing options on a moving average is challenging.*

The idea of computing a moving average value comes from the technical analysis of stock price evolution: Chart analysts use the moving average as an indicator for future stock price movements and they often present charts similar to Figure 3.1. The figure shows a stock-price index and the corresponding moving average. The analysts claim that there is information about the future in such charts. However, we will not discuss whether this is true or not, we will use the moving average in a different setting, as a strike of stock options. This idea is simple and leads to a product which is easy to understand for investors. But, only a few options which have a moving average as a strike or as an underlying are actively traded [70]. More common is the moving average computation in issuer-call features of some fixed income securities [71]. Our algorithm can easily be adapted to these securities, so that we will only present the simple case of MWAOs.

The foundation of almost any option pricing method is layed by the no-arbitrage framework introduced by Black and Scholes [17]. We presented the common methods for valution in this framework in Chapter 1. Especially important to note is that Least-Squares Monte Carlo which was first presented by Carrière [32] was improved by Longstaff and Schwartz [81], who already presented an example of a moving window Asian option with early exercise in their publication. However, the option priced by their mathematical formula solves a much easier problem than indicated by their prosa. Another application of the Least-Squares Monte Carlo to the MWAO

option is presented by Bilger [16]. His method is very limited and computationally extremely expensive. Accurate values can hardly be expected. However, Bilger's approach is closely related to our method, which only uses a different choice of basis functions for the conditional expected option value.

There are virtually no analytic pricing formulas known for American type options so that one has to rely on numerical methods, of which Monte Carlo simulation is among the most common. Alternative approaches are based on the Cox Ross Rubinstein (CRR) [36] binomial tree model, which can easily be adapted to American Asian options by using non-recombining trees. The size of non-recombining trees grows exponentially with the number of time steps, such that accurate results are hardly obtained. Window options in a recombining CRR model have been presented by Lau and Kwok [77] using forward shooting grids but they price Parisian or delayed-barrier options and not averaging options. Zvan, Forsyth and Vetzal present PDE methods for continuously [122] as well as for discretely sampled Asian options [123]. The averaging period in their model is limited to a start at a fixed point in time and cannot be easily adapted to a moving averaging period. Other authors like Wilmott [117] present the MWAO with early exercise as a challenging ("not easy") problem in a PDE framework.

In fact, pricing methods for MWAOs have been described by very few authors besides Longstaff and Schwartz [81] or Bilger [16]. Kao and Lyuu [70] present results for moving average-type options which are traded in the Taiwan market. Their method is based on the CRR model and can handle short averaging periods: the examples include up to 5 discrete observations in the averaging period. To our knowledge, Bilger [16] as well as Kao and Lyuu [70] are pioneers in the treatment of MWAOs with early exercise features.

Related to the MWAOs is the problem of multi-asset Asian options. An interesting approach using Markov transition matrices on low distortion grids has been presented by Berridge and Schumacher [15]. Their method seems to be promising for problems with medium dimensionality (4 to 10) and should be applicable to moving window Asian options. An implementation of their method is much more complex and less flexible than ours. Work on European Asian option contracts has been conducted by several authors, e.g. Kemna and Vorst [71] and Shao and Roe [106].

As the main extension to Least-Squares Monte Carlo we propose the utilization of sparse grids type basis functions in the regression, which allows for an accurate option valuation of up to 20 discrete observations on prevailing hardware. The idea of this technique was originally discovered by Smolyak [107] and was rediscovered by Zenger [121] for PDE solutions in 1990. It has been applied to many different topics since then, such as integration [19] or Fast Fourier Transformation [59]. Recently, sparse grids have been used for finite element PDE solutions by Bungartz [28], interpolation by Bathelmann et al [13], and clustering by Garcke et al [52]. They also have been applied to PDE option pricing by Reisinger [100]. An extensive overview of sparse grid methods is provided by Bungartz and Griebel [29].

This chapter is structured as follows: First we formulate the problem of moving window Asian option pricing and explain why it is computationally challenging. It follows a brief description of the Least-Squares Monte Carlo and the introduction of sparse grids to the framework. We show some numerical examples that demonstrate the method's effectiveness. Finally we apply an extrapolation technique to further reduce the error originating from the discrete observations and other limiting parameters. A paper version of this chapter is also available [40].

3.3 Moving Window Asian Option

In this section, we work out the details of a moving window Asian option and present some similar derivatives. The MWAO is a simple option that makes use of the moving average as it is plotted in many stock price charts. Similar to an American option which pays the difference between the current underlying price and a fixed strike, the MWAO pays the difference between the current stock price and the floating moving average. Since the computation of moving averages is well established in chart analysis, this option could be accepted by the market, despite its computational difficulties. Having derived a precise mathematical formulation for the price of an MWAO, we will be able to understand its computational challenge. Other securities which seem to be equally challenging at first sight are already very common and actively traded. We will show, how the valuation of the related securities avoid the computational difficulties of MWAOs. However, MWAOs might be more interesting for investors than the related securities because they have a more intuitive averaging mechanism.

3.3.1 Continuous Version

Before we go into the details of the financial product we set up the process for the underlying variable. As in the previous chapters, we use a standard diffusion process that models the uncertainty in the stock price, according to the formula of Black and Scholes. We denote the stock price at time t with S_t and the option price in dependence of $\mathbb{S}_t := \{S_\tau | \tau \in \mathbb{I}\},\ \mathbb{I} \subseteq [t_0, t]$ with $V_t := V(\mathbb{S}_t, t)$. From the no-arbitrage arguments we know that the option value satisfies the partial differential Equation (Equation (1.19)),

$$\frac{\partial V_t}{\partial t} + \frac{1}{2}\sigma^2 S_t^2 \frac{\partial^2 V_t}{\partial S_t^2} + r S_t \frac{\partial V_t}{\partial S_t} - r V_t = 0$$

with risk-free interest rate r.

Now, the peculiarity of the MWAO is expressed by a boundary condition to the option value V, known as an American constraint. The following condition states the minimum value for the function V_t and has to be satisfied at each time $t > t_0 + t_w$,

$$V_t \geq P(A_t, S_t), \tag{3.1}$$

$$A_t = \frac{1}{\int_0^{t_w} \alpha(\tau) d\tau} \int_{t-t_w}^{t} \alpha(t-\tau) S_\tau d\tau, \tag{3.2}$$

where P is the option's payoff function that depends on the current stock price S_t and a weighted average A_t of the historic stock prices using the weight function α. The moving average is taken over a window ranging from $t - t_w$ to t. In the following, we will consider the payoff function

$$P(A_t, S_t) = \max(A_t - S_t, 0). \tag{3.3}$$

Hence, the exercise value is greater zero if the stock price falls below its moving average. Effectively this is the case if the stock price drops either quickly or steadily.

The standard value for the weight α is

$$\alpha \equiv 1$$

which results in an arithmetic average.

We will discuss other values in Section 3.4. The difficulty in this pricing Equation is the boundary condition in Equation (3.1) which depends on the whole history of stock prices S within the averaging period $t - t_w \leq \tau \leq t$. In fact, it is almost impossible to represent this integral numerically, unless we discretize the path of S.

3.3.2 Discretization

For the computational implementation of this problem we introduce a number of additional variables that contain samples of historic values of S_{t_i} at different times $t_i \in \{t_0 = 0, t_1, \ldots, t_T\}$, i.e. $\mathbb{S}_t := \{S_\tau | \tau \in \mathbb{I}\}$, $\mathbb{I} \subseteq \{t_0, t_1, \ldots, t_T\}$. The accuracy of this approximation depends on the time resolution of the samples. Thus the boundary condition (3.1) becomes a constraint in terms of historic samples. We assume the last M samples to form the historic window. The condition

$$V_{t_i} = V(\mathbb{S}_t, t_i) \geq P\left(\frac{1}{M}\sum_{j=i-M}^{i} \alpha(i-j) S_{t_j}, S_{t_i}\right) \tag{3.4}$$

holds for $i \geq M$, after an initial incubation. For weight α, we will consider two possible configurations. Since the sample points are used to approximate the integral over the stock price path, we can use the trapezoid method for integration as the preferred method for non-smooth integrands:

$$\alpha_1(t) = \begin{cases} \frac{1}{2} & \text{for } t = 0 \vee t = M \\ 1 & \text{otherwise} \end{cases}. \tag{3.5}$$

A simpler method is sometimes closer to reality. With a constant α we do not optimally approximate the continuous integral, but might do better at modeling the practical implementation of such an option. In a realistic setting, this option has predefined dates at which the stock price is fixed and considered in an equally weighted arithmetic average. That means we require a weight function α with

$$\alpha_2(t) = \begin{cases} 1 & \text{for } t < M \\ 0 & \text{for } t = M \end{cases}. \tag{3.6}$$

Our method for the valuation of the option uses the presented discretization and a quadrature of either α_1 or α_2, depending on the setting. The valuation proceeds backwards in time, starting at maturity t_T, where condition (3.4) holds with equality. Then, we solve for the option value at current time and current stock price $V_{t_0} = V(S_{t_0}, t_0)$.

For low values of M this procedure can be rephrased in a PDE setting and solved numerically by standard methods. Without going into details, we recommend a method that is based on a finite volume discretization of the Black-Scholes PDE according to the model of Zvan et al [123]. However, due to the "curse of dimensions" it is traditionally thought that a function with more than three or four dimensions is extremely hard to discretize.

3.4 Related Problems

As we have seen, the moving window Asian option is a derivative with the moving average as one of its underlyings. In order to determine its price correctly, the full history of previous prices has to be considered, which leads to an arbitrary number of relevant dimensions. Despite its intuitive definition the moving average presents a serious computational challenge. This section distinguishes the MWAO from other similar derivatives for which straight-forward implementations or even analytical formulas were derived. Since all the difficulties originate from the averaging mechanism A_t, we will focus on some alternative averaging styles.

3.4.1 Asian American Option

The Asian American option (AAO) is very similar to the moving window Asian option. It differs in the time horizon over which the average is evaluated. While the MWAO has a moving window with constant length, the AAO has a window that increases in time. The averaging window always starts at t_0 and ends at the current time t. This slight difference considerably simplifies the computational procedure. In the following we will briefly show that this pricing problem can be solved in two dimensions.

Consider an asset price process S with an asset price at time t of S_t. The moving average A_t^{AAO} is given by

$$A_t^{AAO} = \frac{1}{t} \int_0^t S_\tau \, \mathrm{d}\tau. \tag{3.7}$$

Differentiating this expression with respect to time t, we obtain

$$\mathrm{d}A_t^{AAO} = \frac{1}{t} S_t dt - \frac{1}{t} A_t^{AAO} \, \mathrm{d}t \tag{3.8}$$

which does not depend on any historic stock price. Only the current stock price and the previous average is required.

3.4.2 Exponential Weight

There exists another version of the moving window Asian option for which a good Markovian approximation of the update formula can be constructed. It uses the variable a as a decay factor which determines how much less old stock prices are weighted compared to newer values. Consider an exponentially weighted average for the payoff

$$V_t \geq P\left(A_t, S_t\right), \tag{3.9}$$

$$A_t^{Exp} = \frac{1}{\int_0^t \alpha(\tau)d\tau} \int_0^t \alpha(t-\tau)S_\tau \, \mathrm{d}\tau, \tag{3.10}$$

with

$$\alpha(t) = a\exp(-at). \tag{3.11}$$

The average theoretically depends on all previous prices, which makes it difficult to implement in practice. However, a simple update formula is available by differentiation of the expression A_t^{Exp} with respect to time,

$$\mathrm{d}A_t^{Exp} = \left(\frac{a}{1-e^{-at}}(S_t - A_t^{Exp})\right) \mathrm{d}t. \tag{3.12}$$

Since this special case assigns virtually no weight to very old asset prices, the method can be seen as a rough approximation to the MWAO in Equation (3.1) with $\alpha(t) = a\exp(-at)$. This kind of approximation is presented by Longstaff and Schwartz [81].

3.4.3 Moving Window Asian Option

The previous paragraphs presented simple update formulas for averages A_t of Asian options. A similar update formula can not be constructed for the MWAO[2]. The complete set of historic asset prices in the window is relevant to the exercise decision of MWAOs.

To see that the problem of the MWAO is different from the presented Asian options, we reconsider the averaging function in Equation (3.2) with a weight function $\alpha = 1$:

$$A_t = \frac{1}{t_w} \int_{t-t_w}^{t} S_\tau \, \mathrm{d}\tau.$$

Differentiating this expression with respect to time t leads to

$$dA_t = \frac{1}{t_w}\left(S_t - S_{t-t_w}\right) \mathrm{d}t,$$

which depends on the asset price at two different times. An optimal exercise strategy has to consider the two values S_t, S_{t-t_w} and all asset prices in between. The reason for this is that all the values $S_{t_i}, t > t_i > t - t_w$ will be used in the computation of future moving averages, which are required in the computation of the expected value of continuation. Since there are infinite many asset prices S_{t_i}, the computation of the optimal exercise strategy is hard.

[2]Recall that we defined MWAO to be a moving window Asian option with an early exercise feature.

3.5 Numerical Procedure

The algorithm that is proposed in this chapter is effectively combining three individual techniques which are well established in their respective fields. We combine Monte Carlo simulation, least squares regression and sparse grids to a practical method for American option valuation. Especially in quantitative finance the technique called sparse grids does not yet fully live to its potential. One of the purposes of this article is to demonstrate the flexibility and the simplicity of sparse grids. Since all the individual components of our algorithm have been elaborated in full detail by our cited sources, we will just summarize each of the components' main aspects.

3.5.1 Simulation

As noted in the previous chapters, the standard method which is used when dimensionality causes numerical difficulties is Monte Carlo simulation. As we will see, this approach does not resolve our issue but will provide the framework for our algorithm. Again, we simulate different asset paths. Each of these paths follows the risk-neutral process, a geometric Brownian motion as in the first chapter (1.26). Recall this process, which is the process underlying the Black-Scholes Equation (1.19),

$$\mathrm{d}S_t = rS_t\,\mathrm{d}t + \sigma S_t\,\mathrm{d}W_t$$

with a risk-less interest rate r, volatility σ and the increment of a Wiener process $\mathrm{d}W_t$. This process is sampled at discrete times $t_i \in \{t_0, t_1, \ldots, t_T\}$ so that each of the n realization S^j, $j \in \{1, \ldots, n\}$ follows as in Equation (1.28)

$$S^j_{t_{i+1}} = S^j_{t_i} \exp\left((r - \frac{1}{2}\sigma^2)(t_{i+1} - t_i) + \sigma\sqrt{(t_{i+1} - t_i)}\theta_{i,j}\right)$$

with $\theta_{i,j}$ drawn from a standard Normal distribution. The price of the MWAO is the discounted expected value of the payoff at the optimal stopping time. The optimal stopping time provides a strategy maximizing the option value without information about the future of the asset path. The optimal stopping time is computed by Least-Squares Monte Carlo as presented in Section 1.4.2. It is important to recall that the numerical procedure always produces a suboptimal exercise strategy, such that the average option value is underestimated.

3.5.2 Choice of Basis Functions

A tricky part of our numerical solution and in fact the crucial challenge is the careful choice of the basis functions b_k in Equation (1.34). As described in the previous section we will use a linear combination of these basis functions to express an estimate for the current option value in dependence of all relevant input parameters.

Implementation

In our implementation, we perform the regressions required by Equation (1.34) on sparse polynomial basis functions as presented in Chapter 1 (Section 1.2.2). We use sparse levels L from 0 to 3 which are sufficient for our purposes. But, we do not perform the regressions on S directly. Instead, we use scaled values of S such that for each simulated path $\$^j$, we compute $x^j = (\gamma_1(S^j_{t_i}), \ldots, \gamma_M(S^j_{t_{i-M}})$, with linear transformation function

$$\gamma_j(S^j_{t_i}) := \frac{S^j_{t_i} - \min(\mathbf{S}_{t_i})}{\max(\mathbf{S}_{t_i}) - \min(\mathbf{S}_{t_i})},$$

such that $x^j \in [0,1]^{M+1}$ lies in a unit cube. Since sparse polynomial basis functions are used, this creates matrices with better condition numbers than without the transformation.

The regression itself is performed by solving the linear least squares problem of Equation (1.34) implicitly via QR-decompositions (cp. Section 1.2.1). Furthermore, the regression is only performed on the paths with a positive exercise value $S^i : P(S^i, t) > 0$. This decreases the computational effort.

3.5.3 Simple Example

Table 3.1 *Specification of a simple moving window Asian option with a floating strike in discrete time.*

Option type	moving window Asian option
Maturity t_T	0.4 years
Risk free rate r	5% p.a.
Volatility σ	40% p.a.
observation frequency $\Delta_{\text{obs}}t$	1/10 years
Length of observation period M	3 observations
Exercise value	$P(\$, t_i) = \max\left(\frac{1}{3}\left(\sum_{j=i-2}^{i} S_{t_j}\right) - S_{t_i}, 0\right)$
Exercise dates	$t_i \in 0.3, 0.4$

To learn more about the implementation, we consider a simple example of a MWAO with few simulated asset paths and only a single early exercise date. The data of this option is presented in Table 3.1.

As always, the Monte Carlo evaluation starts with simulating asset paths. To keep this example simple, we simulate 20 paths S^j, only. The paths $j = 1, \ldots, 10$ are used for an in-sample estimate ($n_1 = 10$) and $j = 11, \ldots, 20$ for an out-of-sample estimate ($n_2 = 10$).

j	$S^j_{t_0}$	$S^j_{t_1}$	$S^j_{t_2}$	$S^j_{t_3}$	$S^j_{t_4}$
1	100	104.0085	80.5268	90.4124	64.3768
2	100	98.1999	91.2582	70.0437	62.3572
3	100	110.3735	118.5273	132.243	111.0303
4	100	98.8469	111.907	96.2535	86.32
5	100	87.1526	80.691	77.7054	78.7367
6	100	110.4539	99.8516	83.3369	103.8844
7	100	78.6144	69.4161	65.5344	63.532
8	100	96.4781	74.4774	90.8101	110.0753
9	100	105.8142	101.2728	96.8907	103.1488
10	100	103.8195	118.0207	101.0886	76.6437
11	100	102.8094	108.1064	121.8964	131.7954
12	100	98.4026	94.6187	76.2072	73.3851
13	100	94.1248	140.5762	150.434	126.2498
14	100	94.6225	81.2551	68.6878	70.1597
15	100	120.4949	135.0204	145.2074	112.4673
16	100	98.0982	89.5304	96.7834	85.1618
17	100	132.8095	125.8557	106.592	107.2087
18	100	96.542	94.6397	105.7335	96.5274
19	100	91.1224	102.0637	88.0567	90.9533
20	100	95.1477	97.4547	78.4171	75.2402

Now, we can compute the value of the option V_{t_4} at maturity time t_4 for each of the paths, i.e.

$$V_{t_4} := P_{t_4}(S_{t_2}, S_{t_3}, S_{t_4}) = \max\left(\frac{S_{t_4} + S_{t_3} + S_{t_2}}{3} - S_{t_4}, 0\right),$$

$$\mathbf{V}_{t_4} = \begin{pmatrix} 14.0619 \\ 12.1958 \\ 9.5699 \\ 11.8402 \\ 0.3077 \\ 0 \\ 2.6288 \\ 0 \\ 0 \\ 21.9406 \\ 0 \\ 8.0185 \\ 12.8368 \\ 3.2078 \\ 18.4311 \\ 5.3301 \\ 6.0101 \\ 2.4395 \\ 2.7379 \\ 8.4638 \end{pmatrix}.$$

This completes the work required at time t_4. We proceed at time t_3, where we compute the immediate exercise value $P^j_{t_3}$ for each path S^j,

$$P_{t_3}(S_{t_1}, S_{t_2}, S_{t_3}) = \max\left(\frac{S_{t_3} + S_{t_2} + S_{t_1}}{3} - S_{t_3}, 0\right),$$

$$\mathbf{P}_{t_3}(S_{t_1}, S_{t_2}, S_{t_3}) = \begin{pmatrix} 1.2369 \\ 16.4569 \\ 0 \\ 6.0823 \\ 4.1443 \\ 14.5439 \\ 5.6539 \\ 0 \\ 4.4352 \\ 6.5543 \\ 0 \\ 13.5356 \\ 0 \\ 12.834 \\ 0 \\ 0 \\ 15.1604 \\ 0 \\ 5.6909 \\ 11.9227 \end{pmatrix}.$$

Following the Least-Squares approach in Section 1.4.2, we have to compute a three dimensional local basis approximation for the expected exercise value $P^e(\$, t_3) \approx \mathbb{E}_Q[V_{t_4}|\$, t_3]$, $\$:=$

$\{S_\tau|\tau \in \mathbb{I}\}$, $\mathbb{I} \subseteq \{t_0, t_1, t_2, t_3\}$ which we solve with a sparse polynomial basis as in Section 1.2.2. The approximation is three dimensional because the known stochastic values at time t_3 which determine V_{t_4} and P_{t_3} are S_{t_3}, S_{t_2} and S_{t_1}.

That means, we compute (Theorem 1.4 and Lemma 1.5)

$$\begin{aligned} P^e(\mathbb{S}, t_i) &\approx \mathbb{E}_Q[e^{-r(t_{i+1}-t_i)}V(\mathbb{S}, t_{i+1})|\mathbb{S}, t_i] \\ &= \sum_{j=1}^{m} \tilde{a}_j b_j(\mathbf{x}), \\ (\tilde{a}_1 \quad \tilde{a}_2 \quad \dots \quad \tilde{a}_m)^T &= \left(\mathbf{B}(\mathbf{X})^T\mathbf{B}(\mathbf{X})\right)^{-1}\mathbf{B}(\mathbf{X})^T\mathbf{y}, \end{aligned}$$

where we identify the state $\mathbf{X} := (x_1 \quad x_2 \quad x_3)^T$, with $x_1 = \gamma_1(S_{t_1})$, $x_2 = \gamma_2(S_{t_2})$, $x_3 = \gamma_3(S_{t_3})$,

$$\gamma_j(S^j_{t_i}) := \frac{S^j_{t_i} - \min(\mathbf{S}_{t_i})}{\max(\mathbf{S}_{t_i}) - \min(\mathbf{S}_{t_i})}$$

and the function values $\mathbf{y} := e^{-r(t_{i+1}-t_i)}\mathbf{V}_{t_4}$ to approximate.

We will not use the three dimensional basis presented in Figure 1.2, the $m = 31$ basis functions

$$\begin{aligned} B_2^{\text{sparse}}(x_1, x_2, x_3) &= \bigcup_{\sum \ell_i = 2} B^{\text{full}}_{\beta(\ell)} \\ &= \{1, x_1, x_2, x_3, x_1x_2, x_1x_3, x_2x_3, x_1^2, x_2^2, x_3^2, x_1^2x_2, \\ &\quad x_1x_2^2, x_1^2x_2^2, x_1^2x_3, x_1x_3^2, x_1^2x_3^2, x_2^2x_3, x_2x_3^2, x_2^2x_3^2, \\ &\quad x_1^3, x_2^3, x_3^3, x_1^4, x_2^4, x_3^4, x_1^5, x_2^5, x_3^5, x_1^6, x_2^6, x_3^6\} \end{aligned}$$

are too many for just 8 asset paths (in-sample and in-the-money). Instead, we will use a sparse polynomial basis with $L = 1$, i.e.

$$\begin{aligned} B_1^{\text{sparse}}(x_1, x_2, x_3) &= \bigcup_{\sum \ell_i = 1} B^{\text{full}}_{\beta(\ell)} \\ &= \{1, x_1, x_2, x_3, x_1^2, x_2^2, x_3^2\}. \end{aligned}$$

Since the basis functions do not include high polynomial degrees, we skip the transformation $\gamma(\mathbf{S})$ of the $\mathbf{S}$ values onto a unit cube, i.e.

$$x_j := \gamma_j(S^j_{t_i}) := S^j_{t_i}.$$

Then, the values of the basis functions using the asset paths S^j which are in the in-sample valuation set and in the money ($P^j_{t_3} > 0$), i.e. $j \in \{1, 2, 4, 5, 6, 7, 9, 10\}$ lead to

$$\mathbf{B}^{\text{in}} := \mathbf{B}^{\text{in}}(S_{t_1}, S_{t_2}, S_{t_3}) = \left(\mathbf{1} \quad S^j_{t_1} \quad S^j_{t_2} \quad S^j_{t_3} \quad (S^j_{t_1})^2 \quad (S^j_{t_2})^2 \quad (S^j_{t_3})^2\right)\Big|_{j\in\{1,2,4,5,6,7,9,10\}},$$

$$\mathbf{B}^{\text{in}} = \begin{pmatrix} 1 & 104.0085 & 80.5268 & 90.4124 & 10817.7681 & 6484.5731 & 8174.3955 \\ 1 & 98.1999 & 91.2582 & 70.0437 & 9643.212 & 8328.0551 & 4906.1259 \\ 1 & 98.8469 & 111.907 & 96.2535 & 9770.7034 & 12523.1715 & 9264.7272 \\ 1 & 87.1526 & 80.691 & 77.7054 & 7595.5709 & 6511.0443 & 6038.1319 \\ 1 & 110.4539 & 99.8516 & 83.3369 & 12200.0661 & 9970.3394 & 6945.0413 \\ 1 & 78.6144 & 69.4161 & 65.5344 & 6180.2308 & 4818.5902 & 4294.7519 \\ 1 & 105.8142 & 101.2728 & 96.8907 & 11196.6402 & 10256.1791 & 9387.8019 \\ 1 & 103.8195 & 118.0207 & 101.0886 & 10778.4862 & 13928.8762 & 10218.9062 \end{pmatrix}.$$

Analog to $\mathbf{B}^{\text{in}}$, the remaining paths (the out-of-sample paths) in the money $j \in \{12, 14, 17, 19, 20\}$ lead to a basis function value matrix $\mathbf{B}^{\text{out}}$ of

$$\mathbf{B}^{\text{out}} = \begin{pmatrix} 1 & 98.4026 & 94.6187 & 76.2072 & 9683.0692 & 8952.6961 & 5807.5343 \\ 1 & 94.6225 & 81.2551 & 68.6878 & 8953.4162 & 6602.3897 & 4718.0082 \\ 1 & 132.8095 & 125.8557 & 106.592 & 17638.3735 & 15839.6651 & 11361.8604 \\ 1 & 91.1224 & 102.0637 & 88.0567 & 8303.2904 & 10416.9973 & 7753.9758 \\ 1 & 95.1477 & 97.4547 & 78.4171 & 9053.0917 & 9497.4135 & 6149.2424 \end{pmatrix}.$$

We are interested in $P^e(\$, t_3)$ of the in-the-money asset path values $S^j_{t_3}$, $j \in \{1, 2, 4, 5, 6, 7, 9, 10, 12, 14, 17, 19, 20\}$, only. These values are obtained by

$$\mathbf{P}^{e,\text{in}}_{t_3} = \mathbf{B}^{\text{in}} \cdot (\tilde{a}_1 \quad \tilde{a}_2 \quad \dots \quad \tilde{a}_m)^T$$

for the in-sample respectively

$$\mathbf{P}^{e,\text{out}}_{t_3} = \mathbf{B}^{\text{out}} \cdot (\tilde{a}_1 \quad \tilde{a}_2 \quad \dots \quad \tilde{a}_m)^T$$

for the out-of-sample paths. Altogether with $\mathbf{P}^e_{t_3} = \begin{pmatrix} \mathbf{P}^{e,\text{in}}_{t_3} \\ \mathbf{P}^{e,\text{out}}_{t_3} \end{pmatrix}$ we can just compute

$$\mathbf{P}^e_{t_3} = \begin{pmatrix} \mathbf{B}^{\text{in}} \\ \mathbf{B}^{\text{out}} \end{pmatrix} \cdot \left((\mathbf{B}^{\text{in}})^\dagger \begin{pmatrix} e^{-r\cdot(t_4-t_3)} V^1_{t_4} \\ e^{-r\cdot(t_4-t_3)} V^2_{t_4} \\ e^{-r\cdot(t_4-t_3)} V^4_{t_4} \\ e^{-r\cdot(t_4-t_3)} V^5_{t_4} \\ e^{-r\cdot(t_4-t_3)} V^6_{t_4} \\ e^{-r\cdot(t_4-t_3)} V^7_{t_4} \\ e^{-r\cdot(t_4-t_3)} V^9_{t_4} \\ e^{-r\cdot(t_4-t_3)} V^{10}_{t_4} \end{pmatrix} \right),$$

where $(\mathbf{B}^{\text{in}})^\dagger$ is the pseudo inverse of $\mathbf{B}^{\text{in}}$ (cp. Theorem 1.7). The exercise decision (Equation (1.35)) is in this case

$$V^j_{t_3} = \begin{cases} e^{-r(t_4-t_3)} V^j_{t_4} & \text{if } P^e(\$^j, t_3) > P^j_{t_3} \\ P^j_{t_3} & \text{else} \end{cases}.$$

Computing the data for all paths leads to

j	$P^e(\mathbb{S}^j_{t_3})$	$P^j_{t_3}$	$e^{-0.05\cdot 0.1}V^j_{t_4}$	$V^j_{t_3}$
1	16.2631	1.2369	**13.9918**	13.9918
2	15.8368	**16.4569**	12.1350	16.4569
3	-	0	**9.5222**	9.5222
4	-73.5746	**6.0823**	11.7811	6.0823
5	9.9375	4.1443	**0.3062**	0.3062
6	-1.2629	**14.5439**	0	14.5439
7	1.301	**5.6539**	2.6157	5.6539
8	-	0	**0**	0
9	1.7157	**4.4352**	0	4.4352
10	-111.2316	**6.5543**	21.8312	6.5543
11	-	0	**0**	0
12	1.9591	**13.5356**	7.9785	13.5356
13	-	0	**12.7728**	12.7728
14	16.9106	12.834	**3.1918**	3.1918
15	-	0	**18.3392**	18.3392
16	-	0	**5.3035**	5.3035
17	-85.4283	**15.1604**	5.9801	15.1604
18	-	0	2.4273	2.4273
19	-23.5981	**5.6909**	2.7242	5.6909
20	-8.5221	**11.9227**	8.4216	11.9227

where the last column contains the option values V_{t_3} at time t_3. Since the option has no further early exercise dates, we can just compute the values of $V^j_{t_0}$ as the discounted values of $V^j_{t_3}$:

$$
\begin{aligned}
V^j_{t_0} &= e^{-r(t_3-t_0)}V^j_{t_3}, \\
\mathbf{V}_{t_0} &= \begin{pmatrix} 13.8525 \\ 16.2931 \\ 9.4275 \\ 6.0218 \\ 0.3031 \\ 14.3392 \\ 5.5977 \\ 0 \\ 4.3911 \\ 6.4891 \\ 13.4010 \\ 12.6457 \\ 3.1600 \\ 18.1567 \\ 5.2507 \\ 15.0096 \\ 2.4032 \\ 5.6343 \\ 11.8041 \end{pmatrix}.
\end{aligned}
$$

The option value V^{in} of the in-sample and the value V^{out} of the out-of-sample set are then the

average values of the corresponding paths estimates $V_{t_0}^j$,

$$\begin{aligned} V^{\text{in}} &= \tfrac{1}{10} \sum_{j=1}^{10} V_{t_0}^j = \quad 7.6392 \\ V^{\text{out}} &= \tfrac{1}{10} \sum_{j=11}^{20} V_{t_0}^j = \quad 8.7029. \end{aligned}$$

In the remainder of this chapter, we will focus on the out-of-sample prices, since on average they represent a lower bound on the true price as discussed in Section 1.4.2.

3.6 Numerical Examples

In order to demonstrate the efficiency of our approach, a numerical case study is provided in this final section. We will focus on a discretely sampled MWAO with properties sketched in Table 3.2. The option is sampled with a regular frequency, e.g. every trading day at a specified time. We will distinguish between two different sample techniques. The first one has a discretely sampled averaging window spanning ten observations and is consequently integrated with α_2 from Equation (3.6). The second one is aimed at an approximation of the continuous-time version of the MWAO and is integrated with α_1 from (3.5).

Table 3.2 *Specifications of a moving window Asian option with a floating strike in discrete time.*

Option type	moving window Asian option
Maturity t_T	0.4 years
Risk free rate r	5% p.a.
Volatility σ	40% p.a.
Daily observations $\Delta_{\text{obs}} t$	$1/250$ years
Early exercise	at each observation with $t \geq 10/250$ years
Length of observation period M	10 days
Exercise value	$P(\$, t_i) = \max\left(\frac{1}{M}\left(\sum_{j=i-M+1}^{i} S_{t_j}\right) - S_{t_i}, 0\right)$

3.6.1 Convergence

To analyze the convergence of the presented pricing algorithm for MWAOs, we will denote the computational result of V^{out} according to Equation (1.36) by $\widetilde{V_a}^i(n, L, M)$. Thus, each Monte Carlo value $\widetilde{V_a}^i$ depends on the number of samples n, the level of the sparse grid function basis L, the number of observations in the window M and the quadrature scheme α_a. Using different sets of random numbers, we compute different $\widetilde{V_a}^i(n, L, M)$ with n, L, a and M fixed in order to get an

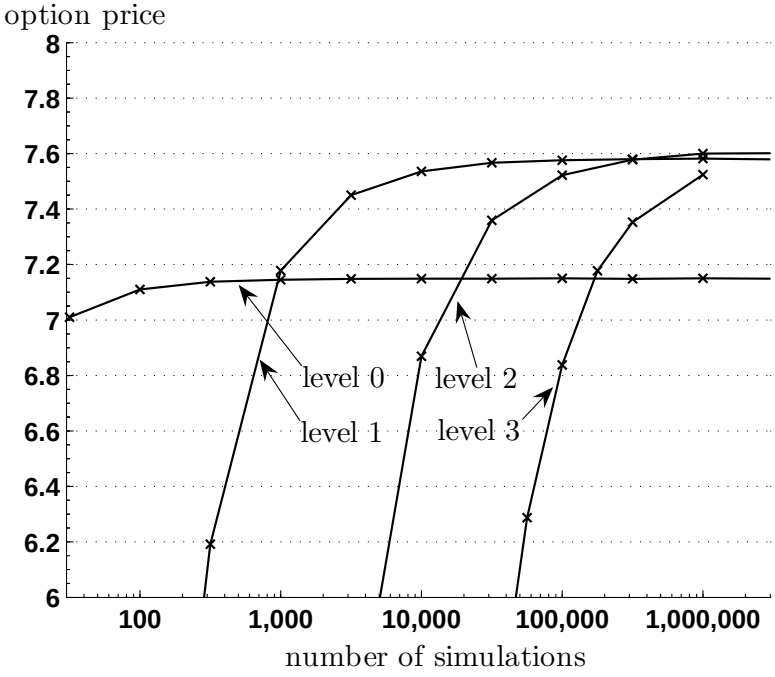

Figure 3.2: *The option value of an MWAO option with data in Table 3.2 estimated by LSMC.*

estimate for the mean

$$\overline{V_a}(n, L, m) = \frac{1}{I} \sum_{i=1}^{I} \widetilde{V_a}^i(n, L, M) \tag{3.13}$$

of I different Monte Carlo prices. The values for I range from 10 to 1000 depending on an estimate of the Monte Carlo error. In most cases I is chosen in a way such that the estimate of the 68%-error is less than 0.001, which means that all presented digits of the option values are correct. The only exception from this rule are two presented option values with many basis functions $L = 3$ and many asset paths $n > 10^5$, where $I = 1$. The computations of more values is too expensive and the error should be already less than 0.01.

The number of samples n per Monte Carlo price results from the in-sample paths $S^1, \ldots, S^{n_1}$ and the out-of-sample paths $S^{n_1+1}, \ldots, S^{n_2}$, i.e. $n = n_1 + n_2$. We use 30% of the sample paths for regressions (n_1) and 70% for valuation out-of-sample (n_2).

Figure 3.2 presents the mean $\overline{V_2}(n, L, 10)$ for different numbers of samples n and different levels L. The values at level $L = 0$ converge quickly to a value of about $\overline{V} = 7.15$ which does not change after 3000 simulations. Using $M = 10$, the level 0 consists of just one basis function and the resulting exercising decision is almost trivial. Level 1 consists of 21 basis functions. This allows for a more sophisticated strategy with a better utilization of the option. After about 100.000 simulations, the option value saturates at 7.58. The level 2 with 241 basis functions results in an even higher value of $\overline{V} = 7.60$ after 1.000.000 simulations. A third level with 2001 basis functions

Table 3.3 *The option value of an MWAO option with data in Table 3.2 estimated by Least-Squares Monte Carlo. The mean of a series of evaluations with level L and a fixed # of samples is denoted by $\overline{V_2}(n, L, 10)$ where as the standard deviation $\sigma(\widetilde{V_2}^i(n, L, 10))$ of this series is denoted by $\widehat{\sigma}$.*

\ **level** L	$L=0$		$L=1$		$L=2$		$L=3$	
# samples n	$\overline{V_2}(n,1,10)$	$\widehat{\sigma}$	$\overline{V_2}(n,2,10)$	$\widehat{\sigma}$	$\overline{V_2}(n,3,10)$	$\widehat{\sigma}$	$\overline{V_2}(n,4,10)$	$\widehat{\sigma}$
3×10^1	7,010	0,499						
1×10^2	7,110	0,234	4,053	0,470				
3×10^2	7,138	0,134	6,192	0,264				
1×10^3	7,145	0,073	7,178	0,114	3,813	0,148		
3×10^3	7,148	0,043	7,450	0,061	5,399	0,083		
1×10^4	7,149	0,022	7,536	0,034	6,869	0,041	3,166	0,069
3×10^4	7,149	0,014	7,567	0,018	7,359	0,018	5,357	0,024
1×10^5	7,150	0,007	7,576	0,010	7,522	0,009	6,841	0,010
3×10^5	7,148	0,005	7,580	0,005	7,578	0,006	7,358	
1×10^6	7,150	0,004	7,582	0,003	7,600	0,002	7,524	
3×10^6	7,149	0,001	7,579	0,001	7,601	0,001		

already exceeds our available computational resources, such that the saturation level could not be computed.

One thing worth mentioning is the initial inferiority of higher levels due to an over-fitted exercise strategy. This effect is based on the fact that a regression with relatively few asset paths on many basis functions is conducted for estimating the optimal early exercise strategy. Now, the basis functions can predict the behavior of the in-sample data set perfectly and deliver early exercise strategies which have knowledge of specific future paths characteristics instead of the average characteristics. In Figure 3.2, the out-of-sample values are presented. For the out-of-sample data set, the trained knowledge of specific in-sample future paths characteristics delivers wrong estimates of the expected path development. Now, the over-fitted exercise strategy is suboptimal and thus delivers lower values than an optimal strategy. The larger the over-fitting effect, the worse is the exercise strategy in the out-of-sample data set.

The corresponding values to Figure 3.2 are presented in Table 3.3. The mean values of a series of valuations is denoted by $\overline{V_2}(n, L, 10)$, the standard deviation of the series is denoted by $\widehat{\sigma}$. For a single evaluation with the Least-Squares Monte Carlo, $\widehat{\sigma}$ can be seen as a measure how close the value is to the mean of many valuations. Contrarily, $\widehat{\sigma}$ does not provide a measure for the error compared with the real value. The mean estimate will be biased towards lower than the real values due to the insufficient estimate of the optimal exercise strategy P^e. An approximation of the MWAO with 1.000.000 sample paths and level two regressions delivers cent accurate estimates. Consequently, the value of an option with properties in Table 3.2 is at least 7.60.

3.6.2 Heuristic Extrapolation

After we have successfully handled the ten-dimensional case we will aim for the infinitely dimensional problem. While the previous option's exercise value depends on the average of ten discretely sampled stock prices we will now consider the continuous integral. The option has a payoff as defined in Equation (3.3). In order to present the optimal approximation we will increase the number of samples in the averaging window and then extrapolate the value based on the obtained convergence properties. The derivative's specification can be found in Table 3.4.

Table 3.4 *Specifications of a moving window Asian option with a floating strike in continuous time.*

Option type	moving window Asian option
Maturity t_T	0.4 years
Risk free rate r	5% p.a.
Volatility σ	40% p.a.
Averaging window length t_w	10 days = $\frac{10}{250}$ years
Early exercise interval	$10/250$ years $\leq t \leq 0.4$ years
Exercise value	$P(\$, t) = \max\left(\frac{1}{t_w}\left(\int_{t-t_w}^{t} S_\tau d\tau\right) - S_t, 0\right)$

For the continuous integral we rely on the trapezoidal quadrature rule α_1 as stated in (3.5). Hence, we compute $\overline{V_1}(n, L, M)$ and analyze the effect of increasing M arbitrarily. Figure 3.3 demonstrates the convergence on level $L = 2$ with 10^6 sample path's. We can clearly recognize the convergence with the number of observation samples within the averaging window M. Despite the converging shape there is still some slope in the curve's final point $\overline{V_1}(10^6, 2, 20) = 8.16$. Extrapolation will lead us to a final result that is about 2% higher than our best finite approximation.

In order to approximate the infinite-dimensional result as accurate as possible we use an extrapolation technique for the Least-Squares Monte Carlo, similar to the Richardson extrapolation [1, 98]. For an extrapolation, we require a convergent, strictly increasing series of option values. Assuming that we knew the order of convergence of the error, we could extrapolate to infinity and solve for the value of the continuously averaging MWAO. Before going into mathematical details we can think of this method as a way of guessing the limit value based on the information known.

The price of our continuous average option has three main sources of error: the number of simulation paths n, the level of the function basis L and the number of integration samples M. The best possible approximation would have to limit each of these parameters towards infinity and compute $\overline{V}(\infty, \infty, \infty)$. We will limit our discussion to $L \in \{0, 1, 2\}$ because this should already give values accurate enough. Furthermore, multidimensional extrapolation is certainly

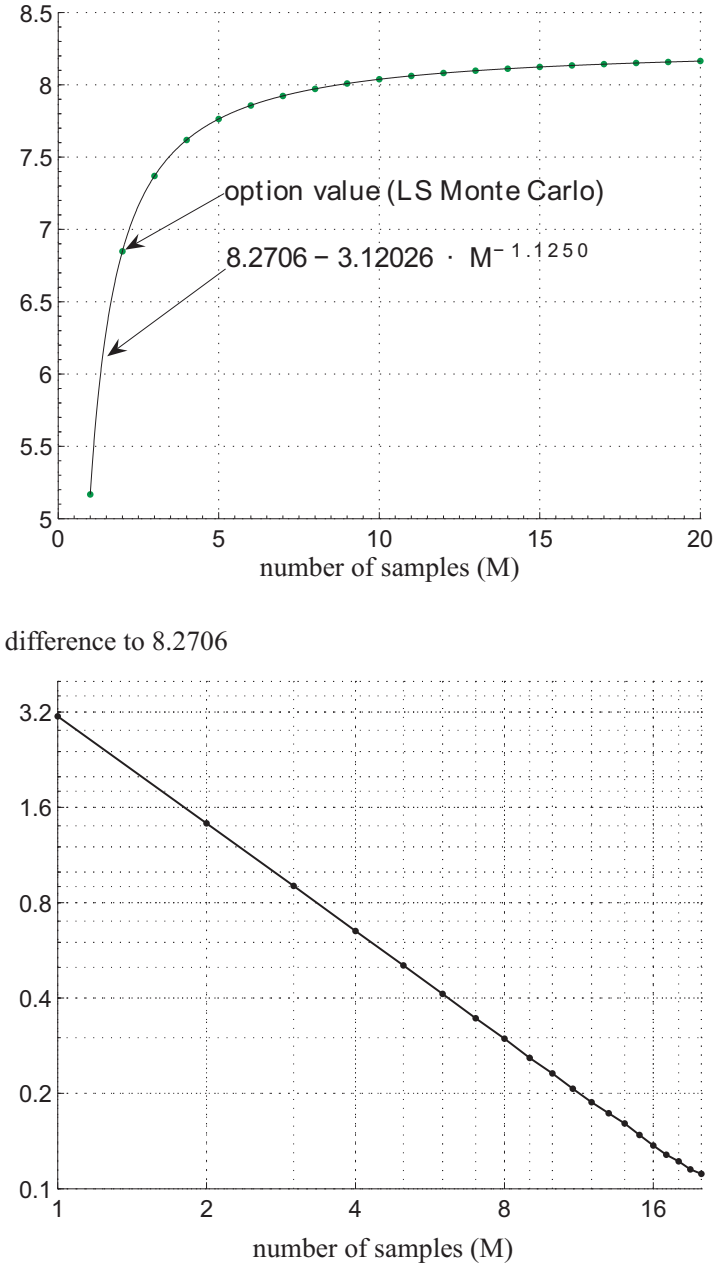

Figure 3.3: *The mean option value* $\overline{V_1}(10^6, 1, M)$ *of an MWAO option with data in Table 3.4 estimated by Least-Squares Monte Carlo together with the function* $8.2706 - 3.12026 \cdot M^{-1.1250}$ *is presented in the upper plot. The plot on the bottom shows the difference of the option values to the extrapolated value,* $8.2706 - \overline{V_1}(10^6, 1, M)$.

something where little experience has been collected so far. We will do extrapolation not only as a mental exercise, but also as a way to justify our finite results which are very close to the presumable infinite limit.

To our knowledge, there has not been any theoretical error analysis of the Least-Squares Monte Carlo for MWAOs. But, we can build on a result from Stone [109]: If a regression function $\theta(x) = \mathbb{E}[Y|X = x]$ is p-smooth, then the L_2-error of a local polynomial kernel estimator converges to zero at a rate of n^{-c} with some fixed value c and n denoting the number of samples of X. This result is related to the Least-Squares Monte Carlo valuation because we use a polynomial basis in order to estimate the conditional expectation and we assume that this is the main source of error. Hence, we can rewrite $\overline{V_a}(n, L, m)$ as

$$\overline{V_a}(n, L, M) \approx \overline{V_a}(\infty, L, M) - (c_0 n^{-c_1}) \tag{3.14}$$

and our empirical data analysis indicates that this is a reasonable guess.

This extrapolation to $n \to \infty$ has little impact, since values based on $n = 10^6$ are already very precise. The difference between $V(10^6, 1, M)$ and $V(\infty, 1, M)$ is just about one cent.

Being able to produce a series of $\overline{V_1}(\infty, L, M)$ we can continue and focus on the next parameter: M. If we want to extrapolate it to infinity, we again have to find the order of the error. We look for a reasonable function which can fit the option values for different M. Figure 3.3 presents the Least-Squares Monte Carlo option values for different M together with the function $8.2706 - 3.12026 \cdot M^{-1.1250}$. The fit of the function is almost perfect so that we get $V(10^6, 1, \infty) \approx 8.27$. The same procedure for $L = 2$ leads to $V(10^6, 2, \infty) \approx 8.30$.

In the end, we can present an informed guess for the value V of a continuously averaging MWAO with properties in Table 3.4:

$$V \approx 8.30. \tag{3.15}$$

This kind of extrapolation can be useful to decrease computational effort or to increase accuracy. However, the correct description of the error and in particular the suitability of the error estimators for the parameters is open for future research.

3.7 Summary

This chapter presents a simple and flexible implementation of a moving window Asian option. Despite being actively traded, no accurate algorithm has been published so far which could extract the derivative's optimal exercise strategy and its precise value. The computational difficulty stems from one of the options underlyings: an either discretely or continuously sampled moving average over a stock price path. This leads to a very high dimensionality in the mathematical definition with an exponential complexity in standard algorithms. We have shown that a straight forward approach to this problem could be found by employing Least-Squares Monte Carlo and a technique called sparse grids, which was specifically developed as a cure to the curse of dimension. The presented approach can be applied to any derivative that has the moving average as an underlying, as it is commonly plotted in stock price charts. We believe that this thesis can increase the acceptance of such products. Now, there is a simple algorithm for the simple derivative.

Chapter 4

Callable Convertible Bonds

4.1 Overview

After the solution to the Moving Window Asian Option pricing, we will push the Least-Squares Monte Carlo a little further. This will allow us to determine its efficiency for complex exercise and call features. Convertible bonds inhabit many of these complex rights for the holder and the issuer, which makes them a suitable choice for the test of Least-Squares Monte Carlo.

Most methods for valuing convertible bonds assume that the bond is continuously and instantly callable by the issuer. However, in practice convertible bonds can often be called only if advance notice is given to the holders. In this chapter, we develop an accurate PDE method for valuing convertible bonds with a finite notice period as a reference. Then, we present a Least-Squares Monte Carlo method capable of pricing convertible bonds and compare the two methods. Example computations are presented which illustrate the effect of varying notice periods, and moving window call constraints. It appears that a low-dimensional sparse basis can be used to obtain reasonably accurate prices even in the case of the moving window call constraint.

4.2 Introduction

The market for convertible bonds has been growing rapidly in the past few years [12]. Convertibles can be thought of as normal corporate bonds with embedded call options on the issuer's stock. Having properties of both stocks and bonds, convertibles can be an attractive choice for investors. Studies suggest that the historical average return of convertibles has been roughly the same as that for the equity market, but convertibles have tended to have lower risk [113, 83]. From the standpoint of the issuing firm, a convertible can be attractive for several reasons [26]. They are particularly desirable in situations where the risk of the issuer is hard to evaluate and its investment policy is somewhat unpredictable. The prototypical issuer is a relatively small firm with high growth potential and risk. Such firms are often cash-constrained and willing to give the embedded call option to investors in exchange for lower coupon payments on their debt.

Convertibles incorporate a variety of features. The instrument might be convertible into shares of the issuing company or in some cases into shares of a different company. Usually convertibles may be converted by the holder at any time. Often, these bonds can be sold back (or "put") to the issuer at specific dates for a guaranteed price. In addition, the issuer may have the right to redeem or call back the convertible at a specified call price. If the issuer does so, the investor can choose between receiving the call price or converting into shares. The ability of the issuer to call back the issue is frequently restricted by "soft" and "hard" call constraints. A hard call constraint prohibits calling the issue during the initial life of the contract. A soft call constraint requires that the issuer's stock price remains above a discretely observed trigger level before the issue can be called, e.g. the convertible cannot be called until the underlying has been m out of n days above the trigger level. In addition, and what is our main focus here, the issuer usually has to give notice some period in advance (e.g. 1 month) of calling the issue.

A long-standing puzzle with regard to convertibles is the "delayed call" phenomenon. This has been discussed by many authors [26, 78, 58, 4, 5, 119]. As shown in [65], assuming that the issuing firm's management is acting in the interests of the existing shareholders, it is optimal to call the convertible as soon as its value is equal to the call price. However, the observed behavior of firms is not consistent with this in that companies often wait until the convertible value is far above the call price before calling. The optimality of the policy of calling immediately after the convertible value reaches the call price depends on various other assumptions, and so a variety of explanations have been proposed to account for the difference between the theoretically optimal policy and that observed in practice. One possibility is the dilution effect[1] [74, 72, 69, 73], another possibility is the effect of the notice period (see [10] and references therein) on which we will focus.

A detailed lattice method for valuing convertibles with notice periods is presented in [78]. In [56], we develop a PDE method for pricing convertible bonds with a call notice period. The main focus of the work in [56] was to compare the pricing results for a call notice period to the approximations developed by Butler [30], under both the Tsiveriotis and Fernandes [112] model and the Ayache, Forsyth, Vetzal (AFV) model [11, 12, 7, 119, 84].

Since it now appears that the standard reduced-form pricing model for convertible bonds uses the AFV assumptions, (see, for example, the most recent version of [64]), we will consider only models of the AFV type in this thesis.

The main focus of this chapter is on methods for pricing complex path-depedent call and put features of convertible bonds. We will first present a brief overview of the numerical PDE method used for pricing convertible bonds using the AFV [11] model. In particular, this one factor PDE model can be used to price complex put and call features, including calls with a notice period.

Although the PDE method is very general, the valuation of convertible bonds with complex call constraints is sometimes not feasible due to memory and computational restrictions. Con-

[1]Dilution is the effect that the relative share of the existing stock holders declines if a holder of a convertible bond decides to exercise into new shares instead of taking the bond's face value at maturity time.

sequently, we present a Monte Carlo method that is capable of pricing securities with a variety of call constraints. In most cases, computing values of convertible bonds using the Monte Carlo method is not nearly as efficient as our PDE method. But, the Monte Carlo method allows us to analyze properties of convertibles which we cannot model efficiently in a PDE framework.

Convertible bonds often have complex call trigger features. For example, a call may be issued (with notice) only if the underlying was above a trigger level for 20 out of the last 30 days. In the following, we will refer to this contract feature as moving window call protection. In this case, the numerical PDE solution would require a solution on a thirty dimensional grid, which is clearly infeasible. The only real possibility for pricing such a feature is by means of a Monte Carlo method.

Most previous work on Monte Carlo methods for pricing convertible bonds [27, 6] relies on a parameterization of the optimal stock price level for issuing a call. It is difficult to correctly account for the different cash flows and call constraints in this setting so that we choose a more rigorous non-parametric approach in this work which is similar to the one used in [84].

For the assessment of complex trigger level call constraints, we present a Monte Carlo method based on least-squares regressions and special basis functions. Our Monte Carlo Method is based on the American option pricing procedure presented in [32] and [81]. To the best of our knowledge, the only previous work which uses this approach for convertibles is presented by Lvov et al [84]. While their focus lies on the general application of Least-Squares Monte Carlo to discretely callable convertibles, we focus on the quantitative comparison of PDE and Monte Carlo for convertibles with continuous call. Additionally, we present how to incorporate common soft call constraints.

We will first compare Monte Carlo and PDE methods for convertible bonds with call notice period features which can be priced using both techniques. We then go on to use the Monte Carlo method for pricing complex features which cannot be priced using PDE methods.

The main results of this chapter are as follows

- Assuming that the issuer uses various rules of thumb for issuing a call notice, we examine the impact of these non-optimal strategies on the bond price.

- We verify that the Least-Squares Monte Carlo method can be used to price a convertible with vanilla call and put provisions. The accuracy of the Monte Carlo method is verified by comparing with an accurate PDE solution. Our results are consistent with those in [84].

- Since the moving window call protection is heavily path-dependent, it would appear that the Monte Carlo method would require a very large number of basis functions, even if sparse grid techniques are used. Our preliminary results indicate that reasonable results can be obtained even with a small number of basis functions for the least-squares regression. This is fortunate, since otherwise, the computation would be infeasible.

The chapter is organized as follows. We first present the standard model for convertible bonds with credit risk and a short summary of new developments in this area. We then derive the equations which take into account, in a rigorous manner, the call notice period. An outline of the numerical algorithms is next presented, followed by some illustrative results. A paper version of this chapter is also available [57].

4.3 Models for Convertible Bonds

Our main focus here is on modeling the call notice period and other call provisions. We will restrict our attention to the case where interest rates are constant. This is in line with current practice since it is commonly believed that the effect of stochastic interest rates on convertible pricing and hedging is small, compared to stochastic stock prices (see [25, 6]). Dilution effects will also be ignored in the following.

4.3.1 No Default Risk

For ease of explanation, consider first the case where we ignore the credit risk of the issuer of the convertible. This is basically the same derivation as for the valuation of vanilla options which we presented in Chapter 1. Recall that we assume that the stock price S evolves according to

$$\mathrm{d}S_t = rS_t\,\mathrm{d}t + \sigma S_t\,\mathrm{d}W_t \tag{4.1}$$

where r is the risk-free interest rate, σ is the volatility, and $\mathrm{d}W_t$ is the increment of a Wiener process. Then, following Equation (1.19), the value of any claim V_t contingent on S_t satisfies

$$\frac{\partial V_t}{\partial t} + \frac{1}{2}\sigma^2 S_t^2 \frac{\partial^2 V_t}{\partial S_t^2} + rS_t\frac{\partial V_t}{\partial S_t} - rV_t = 0. \tag{4.2}$$

Consider the case of a convertible bond which has no put or call provisions and can only be converted at the terminal time t_T. If the convertible has face value F and can be converted into κ shares, then the value of the convertible V is given from the solution to Equation (4.2), with the terminal condition

$$V_{t_T}(S_{t_T}) = \max(F, \kappa S_{t_T}). \tag{4.3}$$

Note that the index of V_{t_T} denotes a time t_T at which we observe the variable V.

4.3.2 Credit Risk

The above model ignores the credit risk of the issuer of the bond, but this is potentially an important effect. Several models for incorporating credit risk have been proposed. Tsiveriotis and Fernandes (T&F) [112] proposed a model in 1998 which was widely adopted. The T&F model was derived in a very heuristic manner, and, as pointed out by Ayache, Forsyth, and Vetzal (AFV) in [11, 12], seems to be inconsistent in some cases. In the following, we will use the AFV model as a

basis for out study. A similar model has been used in [7, 119, 84]. We also note that a simplified form of this model was also suggested in [64].

The Hedged Model (AFV Model)

AFV derive a model, based on a hedging portfolio where the risk due to the normal diffusion process is eliminated, and assuming a Poisson default process. The probability of default in $[t, t+dt]$, conditional on no-default in $[0, t]$ is $p(S_t, t)\ \mathrm{d}t$ with $p(S_t, t)$ the hazard rate of the default process.

This model allows different scenarios in the event of default. Upon default, it is assumed that the stock price jumps according to

$$S_{t^+} = S_{t^-}(1-\eta),\ \ 0 \leq \eta \leq 1$$

where S_{t^+} is the stock price immediately after default, and S_{t^-} is the stock price just before default. Further, the holder of the convertible can choose upon default between:

1. Recovering RX, where $0 \leq R \leq 1$ is the recovery factor. There are various possible assumptions for X, e.g. face value of bond, discounted bond cash flows, or pre-default value of the bond component of the convertible; or
2. Receiving shares worth $\kappa S_{t^+} = \kappa S_{t^-}(1-\eta)$.

For simplicity in the following, we will assume that the recovery rate R as well as the present value of the convertible if we wait until liquidation are zero. This leads to the following partial differential inequality for the convertible value V

$$\frac{\partial V_t}{\partial t} + \frac{\sigma^2}{2} S_t^2 \frac{\partial^2 V_t}{\partial S_t^2} + (r + p\eta) S_t \frac{\partial V_t}{\partial S_t} - (r+p)V_t + p\kappa S_t(1-\eta) \geq 0 \tag{4.4}$$

$$V_t(S,t) \geq \max(B_p(S_t,t), \kappa S_t) \tag{4.5}$$

$$\frac{\partial V_t}{\partial t_t} + \frac{\sigma^2}{2} S_t^2 \frac{\partial^2 V_t}{\partial S_t^2} + (r + p\eta) S_t \frac{\partial V_t}{\partial S_t} - (r+p)V_t + p\kappa S_t(1-\eta) \leq 0 \tag{4.6}$$

$$V_t(S_t,t) \leq \max(B_c(S_t,t), \kappa S_t), \tag{4.7}$$

where either one of (4.4)-(4.5) or (4.6)-(4.7) hold, and one of the inequalities holds with equality at each point in the solution domain. The terminal condition is given in Equation (4.3). Note that the call price $B_c(S_t, t)$ is the price at which the issuer can terminate the convertible and the put price $B_p(S_t, t)$ is the price at which the holder can return the convertible. Inequality (4.5) represents the options of the holder: She can convert into shares worth κS_t or put the option to the issuer for $B_p(S_t, t)$. The value of the convertible cannot drop below these prices because otherwise an investor would buy the convertible, convert (resp. put) immediately and receive a risk-less profit. This is not possible following the no-arbitrage assumption. Inequality (4.7) represents the option of the issuer, who can call the convertible and pay $B_c(S_t, t)$. The value of the convertible will not rise above this price because the issuer calls and thus terminates the convertible as soon

as the convertible reaches the call price. This is in the interest of the existing stock holder (see Ingersoll [66] for details of this reasoning.).

The above Inequalities (4.4)-(4.7) can be derived by constructing a hedging portfolio

$$\Pi_t = V_t - \phi_1 S_t - \phi_2 L_t$$

containing the convertible bond V, ϕ_1 of the underlying stock S_t and ϕ_2 of a plain bond L issued by the same firm as V_t and with zero recovery. An appropriate choice of ϕ_1 and ϕ_2 renders this portfolio risk-free, so that we can perform valuations based on pure hedging arguments, see Appendix 7.4.

Modeling Default Intensity

In order to obtain realistic default behavior of the stock price process, we will use a hazard rate which depends on the stock price S. For the hazard rate $p(S_t, t)$, we use the model suggested in [89] and in [7] where

$$p(S_t, t) = p(S_t) = p_0 \left(\frac{S_t}{S_0}\right)^{\alpha}.$$

The parameters $p_0 > 0$ and $\alpha < 0$ can be calibrated to market data.

As before, we assume in the following, that recovery $R = 0$ and that the stock jumps to zero on default (i.e. $\eta = 1$) for ease of exposition.

4.3.3 Cash Flows, Call and Put Provisions

Convertible bonds usually have a variety of different features which influence their value. In this section we will present the most important features and their effects.

Dividends and Coupons

If a discrete dividend D_i is paid at time $t_{d,i}$, then the usual no-arbitrage arguments imply that

$$V(S_{t^+_{d,i}} - D_i, t^+_{d,i}) = V(S_{t^-_{d,i}}, t^-_{d,i}), \tag{4.8}$$

where $t^-_{d,i}$ is the time immediately before the dividend payment, and $t^+_{d,i}$ is the time immediately after the payment.

Consider coupon payments c_i paid at times $t_{c,i}$. Denote the time immediately before the payment as $t^-_{c,i}$ and immediately after the coupon payment as $t^+_{c,i}$. The price of the convertible then drops according to

$$V(S_{t^+_{c,i}}, t^+_{c,i}) = V(S_{t^-_{c,i}}, t^-_{c,i}) - c_i. \tag{4.9}$$

Clean and Dirty Prices

The call price B_c and the put price B_p in the previous equations include accrued interest. Specifically, let B_{cl}, B_{pl} be the *clean* call and put prices. The actual call (put) price is computed by

$$B_c(\mathbb{S}_t, t) = (B_{cl} + A(t)) \cdot \delta_{\text{call}}(\mathbb{S}_t, t), \tag{4.10}$$

$$B_p(S_t, t) = (B_{pl} + A(t)) \cdot \delta_{\text{put}}(S_t, t), \tag{4.11}$$

where $A(t)$ is the accrued interest, a fraction of the next coupon payment and $\mathbb{S}_t := \{S_\tau | \tau \in \mathbb{I}\}$, $\mathbb{I} \subseteq \{t_0 = 0, t_1, \ldots, t_T\}$ denotes the complete asset paths until time t. If the last payment was at t_{i-1} and the next payment worth c_i is paid at t_i, then the accrued interest $A(t)$ is

$$A(t) = \frac{t - t_{i-1}}{t_i - t_{i-1}} c_i.$$

The function δ_{call} indicates if a call is allowed, δ_{put} indicates if a put is allowed. Different specifications follow.

Specifications of indicator functions δ_{call} and δ_{put}

Another common feature of convertible bonds is the hard call protection which prevents calling in the initial lifetime of the security. A hard call protection during time $[t_0, T_h]$ can be accommodated in our model by defining the set of call times $\mathcal{T}_{\text{call}} = \{t | t > T_h\}$ and the indicator function of callability

$$\delta_{\text{call}}(t) = \begin{cases} 1 & \text{if } t \in \mathcal{T}_{\text{call}} \\ \infty & \text{otherwise} \end{cases}.$$

Note that we set the indicator to ∞ if the convertible is not callable such that we can use it as a multiplier to the call price: In Equation (4.10) the dirty call price $B_c(S_t, t)$ is the product of a call price and the indicator. If the call feature is not allowed, the indicator and thus the dirty call price $B_c(S_t, t)$ are set to infinity. Consequently, Inequality (4.7) is not a binding constraint on the price of the convertible. But, if the call constraint is active, the indicator is set to 1 and thus the convertible value $V(S_t, t)$ cannot exceed the dirty call price. A similar reasoning holds for the other indicator functions defined in this section.

The indicator for the put feature is defined as

$$\delta_{\text{put}}(t) = \begin{cases} 1 & \text{if } t \in \mathcal{T}_{\text{put}} \\ 0 & \text{otherwise} \end{cases}$$

with the set of put times $\mathcal{T}_{\text{put}}$.

In addition to the hard call protection period, some convertibles have a call trigger price $B_{c,\text{trigger}}$ which means that the underlying asset value S_t has to be above the trigger value before a call can be issued

$$\delta_{\text{call}}(\mathbb{S}_t, t) = \begin{cases} 1 & \text{if } t \in \mathcal{T}_{\text{call}} \text{ and } S_t > B_{c,\text{trigger}} \\ \infty & \text{otherwise} \end{cases}$$

with $\mathbb{S}_t = \{S_\tau | \tau \in \mathcal{T}_{\text{call}}\}$ Again, we set the indicator to ∞ if the convertible is not callable so that we can use it as a multiplier.

The most challenging level of complexity in traded convertible securities is a moving window trigger protection. In this case, the asset value S has to be at least M out of the last N days above the trigger $B_{c,\text{trigger}}$ before a call can be issued. We assume that in this case, the asset value is measured discretely. This means that

$$\delta_{\text{call}}(\mathbb{S}, t) = \begin{cases} 1 & \text{if } t \in \mathcal{T}_{\text{call}} \text{ and } \sum_{k=i-N+1}^{i} 1_{S_{t_k} > B_{c,\text{trigger}}} \geq M \\ \infty & \text{otherwise} \end{cases} \tag{4.12}$$

with $\mathbb{S}_t := \{S_\tau | \tau \in \mathbb{I}\}$, $\mathbb{I} \subseteq \{t_0, t_1, \ldots, t_T\}$ and daily observations, i.e. $\forall k : t_k - t_{k-1} = 1$ day.

Protection by Call Notice Periods

We now add the feature that the issuer has to provide advance notice of calling the convertible. In particular, upon the notice being provided, the holder has T_n time units to decide whether to take the face value or to convert into shares. As noted in [30], the issuer is effectively giving the holder a put option on the shares, plus the shares themselves. The longer the notice period, the more valuable is this put option.

The value of the shares plus the put option can be viewed as the value $V^{\text{called},t}$ of a new convertible bond starting at time t, maturing at time $t + T_n$, and having a terminal value of

$$V^{\text{called},t}(S_t, t + T_n) = \max(B_c(S_t, t + T_n), \kappa S_{t+T_n}).$$

Note that the call price B_c includes accrued interest and is set to infinity if no call is allowed (see Section 4.3.3). Based on the assumption that the issuer acts in the interests of the existing shareholders, he has to minimize the market value of the convertible [65]. Consequently, the issuer will call the convertible as soon as $V^{\text{called},t}$ is less than the price of the convertible. That means that in the model for convertibles, we need to replace all conditions with a call price B_c by conditions with $V^{\text{called},t}$.

For the AFV model, the following equations need to be solved

$$\frac{\partial V_t}{\partial t} + \frac{\sigma^2}{2} S_t^2 \frac{\partial^2 V_t}{\partial S_t^2} + (r + p\eta) S_t \frac{\partial V_t}{\partial S_t} - (r + p) V_t + p\kappa S_t (1 - \eta) \geq 0 \tag{4.13}$$

$$V(S_t, t) \geq \max(B_p(S_t, t), \kappa S_t) \tag{4.14}$$

$$\frac{\partial V_t}{\partial t} + \frac{\sigma^2}{2} S_t^2 \frac{\partial^2 V_t}{\partial S_t^2} + (r + p\eta) S_t \frac{\partial V_t}{\partial S_t} - (r + p) V_t + p\kappa S_t (1 - \eta) \leq 0 \tag{4.15}$$

$$V(S_t, t) \leq \delta_{\text{call}}(\mathbb{S}_t, t) \cdot V^{\text{called},t}(S_t, t), \tag{4.16}$$

with $V^{\text{called},t}(S_t, \hat{t}),\ \hat{t} \geq t$ satisfying

$$\frac{\partial V_{\hat{t}}^{\text{called},t}}{\partial \hat{t}} + \frac{\sigma^2}{2} S_{\hat{t}}^2 \frac{\partial^2 V_{\hat{t}}^{\text{called},t}}{\partial S_{\hat{t}}^2} + (r + p\eta) S_{\hat{t}} \frac{\partial V_{\hat{t}}^{\text{called},t}}{\partial S_{\hat{t}}} - (r + p) V_{\hat{t}}^{\text{called},t} + p\kappa S_{\hat{t}} (1 - \eta) \geq 0 \tag{4.17}$$

$$V_{\hat{t}}^{\text{called},t}(S_{\hat{t}}, \hat{t}) \geq \max(B_p(S_{\hat{t}}, \hat{t}), \kappa S_{\hat{t}}), \tag{4.18}$$

with terminal condition

$$V^{\text{called},t}(S_t, t+T_n) = \max(B_c(S_t, t+T_n), \kappa S_{t+T_n}). \tag{4.19}$$

Dividend and coupon payments are accounted for by applying Equation (4.8) resp. (4.9) to the convertible bond value V if dividend time $t_{d,i}$ (coupon times $t_{c,j}$) equals model time t. Furthermore, if the model time $\hat{t}$ of the call value $V^{\text{called},t}$ is equal to a dividend time $t_{d,i}$ resp. a coupon time $t_{c,j}$, then Equations (4.8) resp. (4.9) are applied to $V^{\text{called},t}$. This treatment is presented in Figure 4.1.

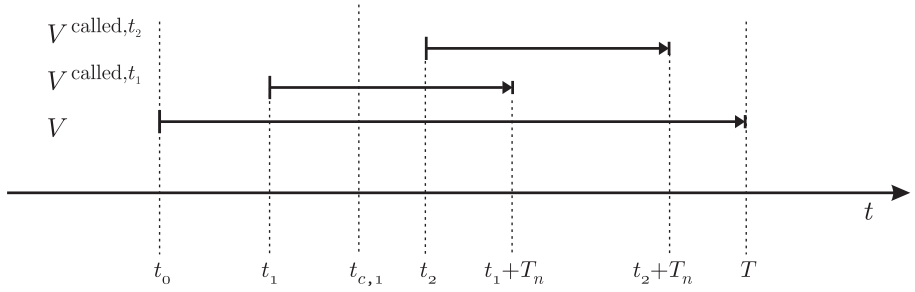

Figure 4.1: *A convertible bond is presented with two call dates: t_1 and t_2. The maturity time of the bond is T, notice time is T_n and a coupon is payed at time $t_{c,1}$. While $V^{\text{called},\,t_1}$ is effected by the coupon payment, $V^{\text{called},\,t_2}$ is not.*

4.4 Numerical Algorithm

This section presents the outline for an accurate PDE model for valuing convertible bonds with call notice periods. Furthermore, we explain the details of a Least-Squares Monte Carlo method which can additionally price moving window soft call constraints. For valuations without a moving window constraint, the PDE method is the superior method due to the slow convergence of the Monte Carlo method. But, the PDE method cannot handle long moving windows which are a common feature of convertible bonds.

In the next sections, we first present a brief outline of a PDE implementation with call notice periods. A Monte Carlo implementation follows, which ignores default and call constraints, so that we can concentrate on the estimation of optimal call and conversion. Then, we extend the Monte Carlo implementation to default and soft call constraints. Finally, we present a detailed description of the regression basis functions used in the Monte Carlo simulation leaving us with a tool capable of solving high-dimensional problems.

4.4.1 PDE Implementation

The PDEs in the AFV models are parabolic Linear Complementarity Problems (LCP) which in general cannot be solved analytically. However, the equations can be solved numerically.

The solution of the LCPs in the AFV case are computed via a discretization in two dimensions: S and t. The solution is generated at discrete values $V(S_i, t_n) = V_{t_n}^j$, $\mathcal{S} = \{S^1, \ldots, S^{\text{imax}}\}$. As is usual in finance, the solution proceeds backwards in time. Given the terminal (payoff) conditions at $t_n = T$, the solution at t_{n-1} is generated using an implicit finite difference scheme. Dividend and coupon payments are included as in Equation (4.8)-(4.9).

The pseudo code provided in Listing 1 illustrates the solution procedure. We assume the existence of a function `discrete_timestep` which, given $\mathcal{V}(t_n) = \{V_{t_n}^1, \ldots, V_{t_n}^{\text{imax}}\}$, does one time step of the implicit solution method to return $\mathcal{V}(t_{n-1}) = \{V_{t_{n-1}}^1, \ldots, V_{t_{n-1}}^{\text{imax}}\}$. See [50] and [12] for implementation details of such a function.

Listing 4.1 *Pseudo code for the numerical algorithm*

```
function vector=discrete_timestep(V_old, S, t, constraint, ...)
  \\This function is a discrete version of the AFV
  \\model. It uses an implicit method to compute the
  \\values V(t − Δt) from V(t) and returns the result
  \\as a vector. The constraint on the values V is
  \\implicitly applied with a penalty method [50].

function vector=convertible_with_notice(V_terminal, S, T, σ, r, ...)
{
  \\Computes the values of a convertible with a notice period
  \\and returns the prices V(S^i)∀i at t = 0 as a vector.

  V=V_terminal;
  for all timesteps from t = T down to t = 0
  {
    if notice to call possible
    {\\solve for the constraint
      B_c=B_cl + accrued_interest(t + T_n);
      V^{called,t}(S^i)=max(B_c, κS^i)∀i;\\the terminal condition
      for all timesteps from t̂ = t + T_n down to t̂ = t
      {
        constraint={V^{called,t}(S^i) ≥ max(B_p(t̂), κS^i)∀i};
        V^{called,t}=discrete_timestep(V^{called,t}, S, t̂, constraint, ...);

        if cash flow occurs between last timestep and t̂
          apply_cash_flow();
      }\\end of inner time-stepping for loop
    }\\end of constraint block
    else \\no call possible
    {
        V^{called,t}(S^i)=∞∀i;
    }
    constraint={(V ≥ max(B_p, κS) ∧ (V ≤ max(V^{called,t}, κ S))};
    V_{t−Δt}=discrete_timestep(V_t, S, t, constraint, ...);

    if cash flow occurs between last timestep and t
      apply_cash_flow();
  }\\end of time-stepping for loop
  return V;
}\\end of function convertible_with_notice
```

An important detail in this implementation is the treatment of cash flows which occur within the notice period. There are usually no details given in the convertible bond contract about what happens if the issuer calls and there is a coupon payment within the notice period. We assume that there is no special treatment in this case and the coupon will be paid as usual. A similar reasoning applies for dividends. Both types of cash flows, coupons and dividends, which are paid at time t_i are applied at $t = t_i$ to calculate $V(\mathcal{S}, t)$ and at $\hat{t} = t_i$ to calculate the value for the constraint $V^{\text{called},t}(\mathcal{S}, \hat{t})$. This allows the holder to obtain the coupon after a notice of call and then convert into shares before the end of the notice period to get the dividend. The algorithm in Listing 1 can be easily adapted for a different treatment of these cash flows.

4.4.2 Monte Carlo Implementation

The numerical solution of the AFV model by finite difference schemes is very efficient for convertibles with and without call notice periods. However, consider the case where the convertible has a discretely observed moving window call protection. In this case, the underlying has to be m out of the last n days above a trigger level before a call can be issued. This would require the solution of an n dimensional PDE. This moving window type feature is computationally extremely challenging using traditional PDE discretization schemes.

We will present a Monte Carlo method based on least-squares regressions similar to the procedures proposed for American type option pricing [32, 81]. Least-Squares Monte Carlo methods for convertible bonds have also been suggested in [84]. Our method is extended to handle the moving window feature by utilization of sparse grid like basis functions.

For explanatory purposes, our first formulation of the Least-Squares Monte Carlo will focus on the exercise decisions leaving out default and call notice periods. We will then present a second formulation which extends the first formulation, by adding provisions for call notice periods and default of the asset. Two sections follow, which explain the basis functions and the specific design choices made for the implementation used in the case study.

Simulation without Default and Call Notice Period

In a Monte Carlo simulation, we simulate different asset paths. Each of these paths follows a geometric Brownian motion as described by Equation (4.1). This process is sampled at discrete times $t_i \in \{t_0, t_1, \ldots, t_T\}$ so that each realization S^j, $j \in \{1, \ldots, s_1\}$ follows

$$S^j_{t_{i+1}} = S^j_{t_i} e^{(r - \frac{1}{2}\sigma^2)(t_{i+1} - t_i) + \sigma\sqrt{(t_{i+1} - t_i)}\theta_{i,j}} \tag{4.20}$$

with $\theta_{i,j}$ drawn from a standardized Normal distribution. The price of the convertible is the discounted expected value of the payoff at the optimal stopping time. The optimal stopping time provides a strategy for maximizing the bond value by optimal conversion of the holder and minimizing the bond value by optimal calling of the issuer.

It is interesting to note that Equation (4.20) is a discretization of the process described by Equation (4.1) which does not introduce any time stepping error by itself, i.e. the distribution of the simulated asset prices S_T does not depend on the number of timesteps n. But, in the PDE case we assume continuous conversion and continuous call. The Least-Squares Monte Carlo can only evaluate convertibles with discrete conversion and discrete call features. This introduces a time stepping error which we analyze in Section 4.5.2.

Least-Squares Monte Carlo for Optimal Decision

The valuation of the convertible proceeds as the valuation of exercisable options in the previous sections. But, the procedure presented in Section 1.3 is generalized to handle the conversion and put by the holder as well as the issuer's call constraints.

As usual, the Monte Carlo valuation begins with a simulation of the underlying by (4.20) forwards in time. At each conversion time t_i, the holder decides to convert the bond and receive the payoff κS_{t_i} or to continue. At the same time, the issuer decides between a call and paying the call price $B_c(\mathbb{S}, t_i)$ or continuation, where

$$\mathbb{S}_{t_i} := \{S_\tau,\ \ \tau \in I\},\ I \subseteq \{t_0, \ldots, t_i\}$$

denotes the complete asset paths until time t_i. In order to maximize the option value V_{t_i}, the holder exercises if

$$\kappa S_{t_i} \geq \mathbb{E}_Q[e^{-r(t_{i+1}-t_i)} V_{t_{i+1}} | \mathbb{S}_{t_i}, t_i],$$

and returns the convertible to the issuer for the put price $B_p(S_t, t)$ if

$$B_p(S_{t_i}, t_i) \geq \mathbb{E}_Q[e^{-r(t_{i+1}-t_i)} V_{t_{i+1}} | \mathbb{S}_{t_i}, t_i] \ \text{ and } \ B_p(S_{t_i}, t_i) \geq \kappa S_{t_i},$$

and the issuer calls if

$$B_c(\mathbb{S}_{t_i}, t) \leq \mathbb{E}_Q[e^{-r(t_{i+1}-t_i)} V_{t_{i+1}} | \mathbb{S}_{t_i}, t_i],$$

with $\mathbb{E}_Q$ denoting the expectation under the risk-neutral measure. In the Least-Squares Monte Carlo, the value of $\mathbb{E}_Q[V_{t_i} | \mathbb{S}_{t_i}, t_i]$ is approximated by

$$P^e(\mathbb{S}_{t_i}, t_i) \approx \mathbb{E}_Q[e^{-r(t_{i+1}-t_i)} V_{t_{i+1}} | \mathbb{S}_{t_i}, t_i],$$

The value $P^e(\mathbb{S}_{t_i}, t_i)$ is computed using a least-square regression on many path-realizations $S^j_{t_i}$. The regressions start at the time step t_{T-1}, i.e. one step before maturity time t_T. The approximated values are

$$P^e(\mathbb{S}_{t_i}, t_i) = \sum_k a^i_k b_k(\mathbb{S}_{t_i}) \tag{4.21}$$

with some basis function $b_k(\mathbb{S}_{t_i})$ and unknown coefficients a^i_k which are determined by a least-squares regression corresponding to the procedure presented in Section 1.4.2. We then determine

a_k^i in Equation (4.21) from

$$\{a_k^i\} = \arg\min_{a_k^i} \left\| \left(\sum_k a_k^i b_k(\$_{t_i}^j) - e^{-r(t_{i+1}-t_i)} V_{t_{i+1}}^j \right)_{j=1,\ldots,s_1} \right\|_2 \tag{4.22}$$

where $V_{t_{i+1}}^j$ is the estimate of the current convertible value of a Monte Carlo path realization $\$_{t_i}^j$ at time t_i. The value of $V_{t_i}^j$ is given as the maximum between the estimated value of the unexercised bond P^e and the exercise value κS_{t_i}, as well as the minimum of the expected bond value P^e and the call price B_c,

$$V_{t_i}^j = \begin{cases} \max(B_p(S_{t_i}^j, t_i), \kappa S_{t_i}^j) & \text{if} \quad P^e(S_{\tau,t_i}^j, t_i) < \max(B_p(S_{t_i}^j, t_i), \kappa S_{t_i}^j) \\ \max(B_c(\$_{t_i}^j, t_i), \kappa S_{t_i}^j) & \text{if} \quad P^e(\$_{t_i}^j, t_i) > B_c(\$_{t_i}^j, t_i) \\ e^{-r(t_{i+1}-t_i)} V_{t_{i+1}}^j & \text{otherwise} \end{cases} . \tag{4.23}$$

Given that the value of the convertible bond at maturity time equals the payoff $V_T^j = F$, a dynamic program solves for all values $V_{t_i}^j$, starting at time T and iterating backwards to t_0. Dividend and coupon payments are included by the application of Equation (4.8) resp. (4.9).

Now, there are essentially three different methods which can provide us with an estimate of the convertibles value $V(S_{t_0}, t_0) = \mathbb{E}_Q[V_{t_0}|S_{t_0}, t_0]$. The first possibility is the computation using the regression function given in Equation (4.21):

$$V \approx P^e(S_{t_0}, t_0) = \sum_k a_k^0 b_k(S_{t_0}).$$

This is especially useful if the asset paths realizations S^j do not start at S_{t_0} but at values spanning an interval around S_{t_0}, e.g. $\forall j : S_{t_0}^j \in [\frac{1}{2}S_0, 2S_0]$. This way, we can easily get estimates for the hedge ratios delta and gamma,

$$\frac{\partial V_t}{\partial S_t} = \sum_k a_k^0 \frac{\partial b_k(S_t)}{\partial S_t},$$
$$\frac{\partial^2 V_t}{\partial S_t^2} = \sum_k a_k^0 \frac{\partial^2 b_k(S_t)}{\partial S_t^2}.$$

But, computing the value of the convertible and the greeks (delta and gamma) this way, the shape of the basis functions $b_k(S_{t_0})$ can introduce a systematic error, which might not be negligible. This error will be especially large if the expected value function $\mathbb{E}_Q[V_{t_0}|S_{t_0}, t_0]$ is not smooth in S_{t_0}.

A second possibility for the computation of $V(S_{t_0}, t_{t_0})$ is given by

$$V^{in} = \frac{1}{n_1} \cdot \sum_{j=1}^{s_1} V_{t_0}^j$$

and a third by

$$V^{out} = \frac{1}{n_2} \cdot \sum_{j=n_1+1}^{n_1+n_2} V_{t_0}^j$$

with $V_{t_i}^j$ as given in Equation (4.23), which is similar to the usual Least-Squares Monte Carlo (Equation (1.36)).

As in Section 1.3, we name this value the out-of-sample price. As already noted, this price has the desirable property that it can present a bound on the true model price for exercisable options. However, in the presence of exercise and call features, the out-of-sample price is neither a lower, nor an upper bound since both, the issuer's call and the holder's conversion strategy, are suboptimal.

The next sections will extend this implementation to default and soft call constraints. After that, we summarize the design choices of the implementation we use for our numerical experiments.

Simulation with Default

Given, that the issuer can go into default, the simulation has to account for the effect of default on the price of the convertible. In general, we could simulate the asset as a combined Brownian motion and jump default process. However, if we examine the PDEs (4.4-4.7) with ($\eta = 1$), we can immediately see that the price V can be computed by simulating the process

$$\mathrm{d}S_t = (r+p)S_t \,\mathrm{d}t + \sigma S_t \,\mathrm{d}W \tag{4.24}$$

and discounting the cash flows back along the path with the effective discount rate $(r+p)$. Note that in general, we must also add in the effective *default cash flow* $p\kappa S_t(1-\eta)\ \mathrm{d}t$ in each timestep $t \to t + \mathrm{d}t$. We will use this approach in the following.

Monte Carlo Valuation with Default and Call Notice

In Section 4.4.2, we discussed the case without a notice period and without default. Now, we want to consider the call constraints discussed in Section 4.3.3. In contrast to the clean call price B_{cl} which is a constant, specified in the convertible bond contract, we defined the call price $B_c(S_t, t)$ to account for accrued interest (Equation (4.10)) and the call trigger constraint (Equation (4.12)). That means, $B_c(S_t, t)$ may depend on the complete asset paths history. Consequently, the optimal call strategy may depend on the recent history of the asset path. This greatly increases the complexity of the convertible bond valuation.

In the case of a defaultable convertible, we discretize the asset price process governed by (4.24) which leads to

$$S_{t_{i+1}}^j = S_{t_i}^j e^{(r+p(S_{t_i}^j)-\frac{1}{2}\sigma^2)(t_{i+1}-t_i)+\sigma\sqrt{(t_{i+1}-t_i)}\theta_{i,j}}\ . \tag{4.25}$$

In contrast to the Equation without default (4.20), this Equation introduces some time stepping error because the default intensity p is not constant. However, the error is small already for a few timesteps as we will see in Section 4.5.2.

As for the non-defaultable contingent claim case, the optimal decisions are solved by regressions. In the Least-Squares Monte Carlo, the value of $\mathbb{E}_Q[V_{t_{i+1}}|\mathbb{S}_{t_i}, t_i],\ \ \mathbb{S}_{t_i} := \{S_\tau,\ \ \tau \in I\},\ I \subseteq$

$\{t_0, \ldots, t_i\}$ is approximated by

$$P^e(\mathbb{S}_{t_i}, t_i) \approx \mathbb{E}_Q[V_{t_{i+1}} | \mathbb{S}_{t_i}, t_i].$$

The value $P^e(\mathbb{S}_{t_i}, t_i)$ is again computed using a least-square regression backwards in time, starting at time t_{i-1}. The approximated values are

$$P^e(\mathbb{S}_{t_i}, t_i) = \sum_k a_k^i b_k(S_{t_{i-m}}, \ldots, S_{t_i})$$

with some m-dimensional basis function b_k and unknown coefficients a_k^i satisfying

$$\{a_k^i\} = \arg\min_{a_k^i} \left\| \left(\sum_k a_k^i b_k(S^j_{t_{i-M}}, \ldots, S^j_{t_i}) - e^{-r(t_{i+1}-t_i)} V^j_{t_{i+1}} \right)_{j=1,\ \ldots, n} \right\|_2 \tag{4.26}$$

where $V^j_{t_{i+1}}$ is the estimate of the current convertible value of a Monte Carlo path realization S^j at times $t_{i-M}, \ldots, t_i$. The value of $V^j_{t_i}$ is given as the maximum between the estimated value of the unexercised bond $P^e(\mathbb{S}_{t_i}, t_i)$ and the exercise value κS_{t_i}, as well as the minimum of the expected bond value $P^e(\mathbb{S}_{t_i}, t_i)$ and the effective value of the call price $V^{\text{called},t_i}(S^j_{t_i}, t_i)$,

$$V^j_{t_i} = \begin{cases} \max(B_p(S^j_{t_i}, t_i), \kappa S^j_{t_i}) & \text{if} \quad P^e(\mathbb{S}_{t_i}, t_i) < \max(B_p(S^j_{t_i}, t_i), \kappa S^j_{t_i}) \\ \max(V^{\text{called},t_i}(S^j_{t_i}, t_i), \kappa S^j_{t_i}) & \text{if} \quad P^e(\mathbb{S}_{t_i}, t_i) > \delta_{\text{call}}(\mathbb{S}_{t_i}, t_i) \cdot V^{\text{called},t_i}(S^j_{t_i}, t_i) \\ e^{-r(t_{i+1}-t_i)} V^j_{t_{i+1}} & \text{otherwise} \end{cases}.$$

The Equation for $V^j_{t_i}$ now also accounts for a call notice period. The algorithm proceeds according to Section 4.3.3, i.e. the call price B_c is substituted by the value of the convertible after a call notice. Since after the notice, the convertible cannot be called again, the value of the convertible after a call notice $V^{\text{called},t}(S^j, t_i)$ is easy to compute. It is the value of a convertible bond with the exact same properties as the original convertible except that the convertible with price $V^{\text{called},t}(S^j_{t_i}, t_i)$ has a maturity time $t + T_n$ and no issuer call options. This value can be estimated e.g. by Least-Squares Monte Carlo or the PDE method (Equations (4.17) to (4.19)).

As in the valuation of a non-defaultable convertible, a dynamic program solves for all values $V^j_{t_i}$, starting at time T and iterating backwards to t_0, given that the value of the convertible bond at maturity time equals the payoff $V^j_T = F$. We can compute the in-sample option price with

$$V^{in} = \frac{1}{n_1} \cdot \sum_{j=1}^{n_1} V^j_{t_0}$$

with asset paths S^j, $j \in \{1, \ldots, n_1\}$. And we compute the out-of-sample option price by

$$V^{out} = \frac{1}{n_2} \cdot \sum_{l=n_1+1}^{n_1+n_2} V^j_{t_0}$$

with $V^j_{t_0}$ as given in Equation (4.4.2) based on new simulation paths S^l, $l \in \{n_1 + 1, \ldots, n_1 + n_2\}$ and the coefficients a_k^i from the in-sample valuation.

Sparse Grids: Choice of Basis Functions

The described Monte Carlo algorithm works well in case of a single factor model. But, in case of a moving window call protection, we have to deal with additional dimensions. In some special cases, the moving window constraint can be reformulated as a set of one-dimensional problems, where the number of elements grows exponentially with the size of the window [55]. This reformulation is only useful for moving windows, with relatively small length, e.g. a condition is feasible where the underlying has to stay 5 out of 15 days above a trigger level before a call can be issued. However, call protection requiring 20 out of 30 days over a trigger level is not feasible. The 20 out of 30 protection is fairly typical of real convertibles.

Since the value of the convertible will depend on the daily observations in the window of historic samples, this represents a high dimensional problem (i.e. 30 dimensional). An American type option pricing problem with so many dimensions is challenging. The procedure we are following is presented in Section 1.2.2 as well as in [40] and proves to be useful for moving window type pricing problems.

A key issue in our numerical algorithm revolves around the choice of the basis functions b_k in Equation (4.22). As described in the previous section we will use a linear combination of these basis functions to express an estimate for the current option value, which includes dependence on all relevant input parameters. Thus, we need one additional dimension in the basis for each observation in the window of historic samples. In the case of a window with M samples, we need, in principle, a basis of dimension M.

As in Chapter 3, where we evaluated a moving window Asian option, we choose a sparse basis B_L^{sparse} (see Equation (1.14)),

$$B_L^{\text{sparse}}(x_1, \ldots, x_M) := \bigcup_{\sum \ell_i = L} B_{\beta(\ell)}^{\text{full}}(x_1, \ldots, x_M),$$

which is smooth everywhere.

Implementation Details

In our implementation, we perform the regressions required by Equation (4.22) on sparse polynomial basis functions as presented in the previous paragraphs. We use sparse levels L from 0 to 3 which are sufficient for our purposes. But, we do not perform the regressions on S directly. Similar to the MWAO option pricing, we use scaled values of S such that for each path j, we compute $x^j = (\gamma_1(S_{t_i}^j), \ldots, \gamma_M(S_{t_{i-M}}^j))$, with linear transformation function

$$\gamma_j(S_{t_i}^j) := \frac{S_{t_i}^j - \min(\mathbf{S}_{t_i})}{\max(\mathbf{S}_{t_i}) - \min(\mathbf{S}_{t_i})},$$

such that $x^j \in [0,1]^M$ lie in a unit cube. Since sparse polynomial basis functions are used, this creates matrices with better conditions numbers then without the transformation.

The regression itself is performed solving the linear least-squares problem of Equation (4.22) implicitly via QR-decompositions.

In case of a call notice period, we use the PDE method for the computation of the convertible bond value V^{called,t_i} after a call. The problem that, in general, the required value $V^{\text{called},t_i}(S^j_{t_i}, t_i)$ of the Monte Carlo paths S^j does not lie on a grid point of the PDE is solved by a cubic-spline interpolation of the PDE values.

In the following, all reported values for a valuation by our Monte Carlo methods are out-of-sample values V^{out}.

4.5 Case Study

The base case data is presented in Table 4.1 and all following examples are computed using this data. The data is consistent with the data used by other authors [12, 112].

Any variation to the base case data will be explicitly noted: some of the parameters will be varied so that the effect on the model price of optimal decisions can be analyzed. We will denote a computed approximation for the value of a convertible with an asset value S_t by $\mathbf{V}(S_t)$. Varied parameters will be denoted by a "|" sign, e.g.

$$\mathbf{V}(S_t|T_n = 0, p(S_t) = 0)$$

denotes the approximation for the value of a convertible with no notice period and no default.

We assume that even in the presence of a hard call protection during the initial lifetime of the bond, a call notice can only be given after this protection period. Note that a call period of $\mathcal{T}_{\text{called}} = \{t|t > (2.0 \text{ years} + T_n)\}$ means that the first notice of a call can be issued at time $t = 2.0$.

Table 4.1 *Base case data of a convertible bond with AFV default model and a call notice period.*

General features			
conversion ratio κ	1	**Putability**	
face value F	100	putable at time $\mathcal{T}_{\text{put}}$	$\{t\|t = 3.0 \text{ years}\}$
coupon payment c_i	4, (8% p.a.)	clean put price B_{pl}	105
coupon times $\mathcal{T}_{\text{coupon}}$	$\{0.5, 1.0, \ldots, 5.0\}$	put price $B_p(t = 3)$	$B_{pl} + c_3 = 109$
maturity t_T	5.0 years		
risk-free rate r	5% p.a.		
volatility σ	20% p.a.		
dividends D_i	0,	**Default model**	
		hazard rate $p(S_t)$	$p_0 \left(\frac{S_t}{S_{t_0}}\right)^{\alpha}$
Callability		spread p_0	2% p.a.
notice period T_n	1/12 years	α	-1.2
call period $\mathcal{T}_{\text{call}}$	$\{t\|t > (2.0 \text{ years} + T_n)\}$	S_{t_0}	100
clean call price B_{cl}	110	recovery rate R	0
first call notice	$t > 2.0$	jump factor η	1
call trigger $B_{c,\text{trigger}}$	0		

4.5.1 Convergence Analysis - PDE

In Table 4.2 the values are displayed for a convertible using the base case data in Table 4.1. Crank-Nickolson time stepping is used. To mitigate numerical oscillations, the method presented in [99] is used. This method uses two implicit time-steps after each non-smooth solution and proceeds with Crank-Nickolson time stepping. The reason for this is that the implicit time-steps smooth out the non-smooth initial conditions, which can then be used by but Crank-Nickolson time stepping. The Crank-Nickolson method has a better convergence than the implicit time stepping for smooth initial conditions, such that the combination of both has a better convergence than the implicit method alone [99, 50].

Table 4.2 shows a numerical convergence analysis. At each refinement, we double the number of nodes in the S grid and the number of time steps. The number of substeps used to determine $V^{\text{called},t}$ (inner time stepping in loop in pseudo code, Listing 1) is also shown. From the column "difference", we can estimate the order of the convergence. A method with first order convergence has the property that the absolute difference halves from one refinement to the next, while the absolute difference of a method with second order convergence is reduced to a quarter.

For both methods, the case with a finite call notice period and the case where the notice period is zero, the convergence seems to be about first order. This contrasts with the smooth quadratic convergence reported in [50] for simple American options. We conjecture that the effect of discrete coupon payments and accrued interest may cause some difficulties on obtaining smooth convergence. However, this is not a problem of practical concern, since we can obtain results which are clearly correct to five digits, which is much more accuracy than would be required in any real situation.

Each time step of the algorithm in Listing 1 requires about (#substeps+1) times the work required for a convertible bond with no notice period. In all cases, the constraint $V^{\text{called},t}$ is solved on a grid with the same spacing as that for V. From Table 4.2, we see that a grid with 3200 nodes has an absolute error of less than 0.01. All results of PDE values in subsequent sections are reported using a 3200 node grid.

Table 4.2 *Convergence results for a convertible with data in Table 4.1 in the AFV model with a one months and without a call notice period. Substeps refers to the number of time steps used to determine $V^{\text{called},t}$, at each discrete time.*

The AFV model - no call notice $\mathbf{V}(S_t|T_n = 0)$

grid for V $S \times t$	$V(_{S\,=\,100,\,t\,=\,0})$	difference
50×50	122.3672	
100×100	122.3713	0.0041
200×200	122.3851	0.0138
400×400	122.3692	-0.0159
800×800	122.3660	-0.0032
1600×1600	122.3653	-0.0007
3200×3200	122.3649	-0.0004

The AFV model - call notice $\mathbf{V}(S_t)$ ($T_n = \frac{1}{12}$)

grid for V, $V^{\text{called},t}$ $S \times t \times$ substeps	$V(_{S\,=\,100,\,t\,=\,0})$	difference
$50 \times 50 \times 1$	123.0799	
$100 \times 100 \times 2$	122.9873	-0.0926
$200 \times 200 \times 4$	122.9365	-0.0508
$400 \times 400 \times 7$	122.9073	-0.0292
$800 \times 800 \times 14$	122.8928	-0.0145
$1600 \times 1600 \times 27$	122.8853	-0.0075
$3200 \times 3200 \times 54$	122.8814	-0.0039

4.5.2 Convergence Analysis - Monte Carlo

For the Monte Carlo valuation, we use the algorithm discussed above. The number of in-sample paths n_1 and out-of-sample paths n_2 are equal, i.e. $n_1 = n_2$. The sparse basis functions are used even in the case of a single dimension. In one dimension, the sparse basis functions become a full basis with 2^L basis polynomials. We use a level of $L = 3$ for up to $n_1 + n_2 = 10^5$ asset paths and a level of $L = 2$ for $n_1 + n_2 = 10^6$ paths simulations. The reported values of the Monte Carlo procedure are always the out-of-sample values.

In Table 4.3, we list average values we computed using the Monte Carlo method. The reported average $\overline{\mathbf{V}^{\text{MC}}}$ is based on as many Monte Carlo valuations as required to compute the prices with an accuracy of ± 0.02 with 95% confidence (cp. Table 1.2). The reported standard deviations are the standard deviations of the set of Monte Carlo values.

We can observe in Table 4.3 that the Monte Carlo simulations appear to converge to the value computed by the PDE-method. This is the case for the convertible without notice period as well as for the convertible with notice period. Note that we cannot expect precise convergence to the PDE values since we are taking daily MC steps leading to errors due to finite sized MC time stepping.

As mentioned before, there are mainly two sources of time stepping errors of the Monte Carlo

Table 4.3 *Convergence results of Monte Carlo simulations for the AFV models with and without a call notice period. Values for convertibles with data in Table 4.1 are presented.* $\overline{\mathbf{V}^{MC}}$ *denotes the average of at least 20 simulations and std*$(\mathbf{V}^{MC})$ *denotes the standard deviation of these simulations. The real mean values lie in: reported value* ± 0.02*, with a probability of* 95%*.*

no notice period
$\mathbf{V}^{\text{PDE}}(S_{t_0} = 100 | T_n = 0) = 122.36$

# asset paths	$\overline{\mathbf{V}^{\text{MC}}}$	std$(\mathbf{V}^{\text{MC}})$
10^4	122.18	0.31
10^5	122.37	0.11
10^6	122.39	0.05

one month notice
$\mathbf{V}^{\text{PDE}}(S_{t_0} = 100) = 122.88$

# asset paths	$\overline{\mathbf{V}^{\text{MC}}}$	std$(\mathbf{V}^{\text{MC}})$
10^4	122.67	0.30
10^5	122.88	0.10
10^6	122.90	0.04

method compared to the PDE method: the discrete call of the Monte Carlo method and the discretization (4.25) of the asset price process with variable default intensity. Table 4.4 shows the effect of different time steps. The upper table presents the isolated effect due to the variable default intensity $p(S)$. The effect introduces an error $O(\Delta t)$, weekly timesteps (5/250) create only an error of about 0.01. A little larger error is introduced by fewer call times which is presented in the lower table. The order of the error is again $O(\Delta t)$, but daily timesteps are required for an error of about 0.01.

However, in spite of this, the MC method appears to be converging to a solution with at least four digit accuracy, which is sufficient for many practical purposes.

Table 4.4 *Convergence results of Monte Carlo simulations for different time steps. Values for convertibles with data in Table 4.1 are presented with different time intervals of the Monte Carlo simulation. The convertible evaluated on the upper table is only convertible at maturity, allows neither a put nor a call. The values on the lower table are computed using daily timesteps ($t_{i+1} - t_i = \frac{1}{250}$) for the underlying process but perform call or conversion fewer times (see # time between calls) than daily. The real mean values lie in the interval of the reported value ± 0.02, with a probability of 95%.*

variable default intensity time steps
conversion only at maturity T, no call,
no put, no coupons, 10^6 paths
$\mathbf{V}^{\text{PDE}}(S_{t_0} = 100) = 104.18$

time step $(t_{i+1} - t_i)$	$\overline{\mathbf{V}^{\text{MC}}}$
$625/250$	106.01
$125/250$	104.43
$25/250$	104.23
$5/250$	104.19
$1/250$	104.20

variable time between call/conversion
$\mathbf{V}^{\text{PDE}}(S_{t_0} = 100) = 122.36$
10^6 paths, sampling of paths: $t_{i+1} - t_i = 1/250$

# time between calls	$\overline{\mathbf{V}^{\text{MC}}}$
$16/250$	122.51
$8/250$	122.46
$4/250$	122.43
$2/250$	122.41
$1/250$	122.39

4.5.3 Properties of Different Call Strategies

The call strategy of the issuer is an important factor for the value of the convertible bond. Early papers [65, 24] derive the optimal call strategy which an issuer should follow without a notice period. The optimal call strategy for a continuously callable convertible without a notice period is to call if the conversion value rises to the effective call price [65]. Or, as an Equation:

$$\kappa S^* = B_c \tag{4.27}$$

with B_c being the effective call price (including accrued interest) and S^* the stock price at which it is optimal for the issuer to call. Note that in case of no default, the sufficient condition for the optimality is that the coupon rate has to be less than the risk-free rate[2].

[2] See [65], Theorem III

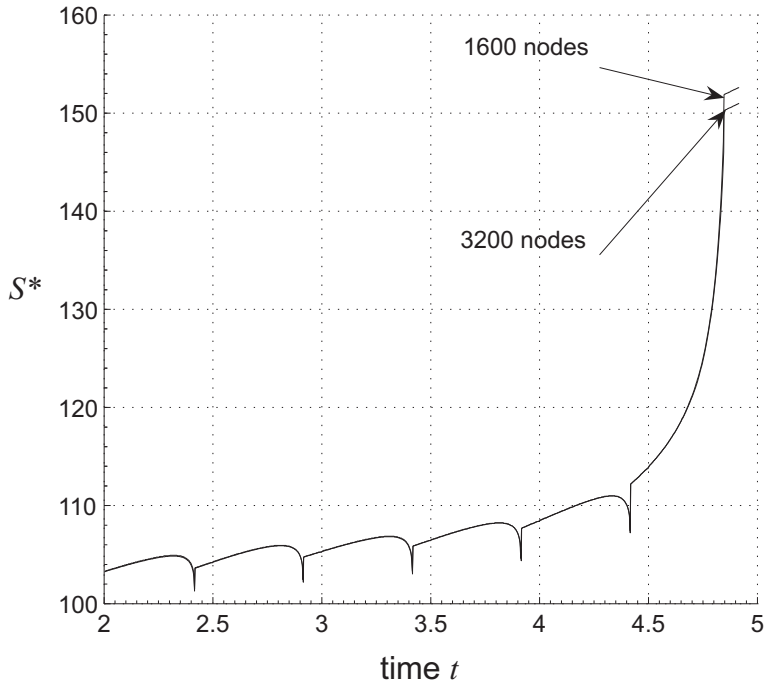

Figure 4.2: *A convertible using the data from Table 4.1. The level of stock price S^* for the optimal call strategy versus time t approximated by the AFV model using a PDE solver. The plot shows the solution on a coarse grid (1600 nodes in S, 1600 time steps, 27 substeps) versus a fine grid (3200 nodes in S, 3200 time steps, 54 substeps): Except close to maturity, both estimates lie virtually on the same line.*

Under some simplistic assumptions, these results are extended in [30] to include notice periods. But, the simplification does not lead to realistic approximations for optimal call strategies of traded securities.

A more sophisticated model that takes more of the complex features of the convertible bond into account follows from the discretization of the PDE in the AFV model. For each node V^j at time t_i of the discretization we check if

$$V^j(t_i) = (V^{\text{called},t})^j(t_i), \tag{4.28}$$

i.e. we see if the maximum constraint in Equation (4.16) is active. At time step t_i, let $V^j(S^j, t_i)$ be the node with S^j the smallest value in $\mathcal{S}$ that results in an active constraint. Then S^j gives a good approximation for the optimal stock price level: $S^*(t_i) \approx S^j(t_i)$. This method is simply a nearest neighbor interpolation.

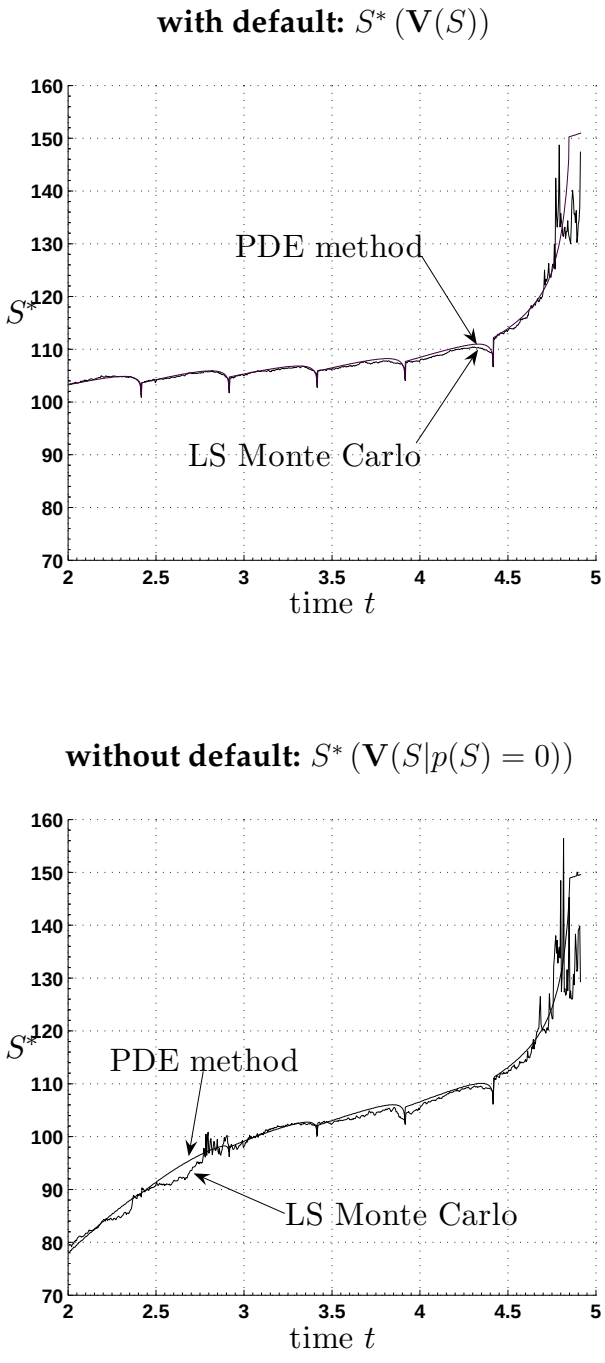

Figure 4.3: *The optimal stock price S^* for a call of a convertible using the data from Table 4.1 is presented on the top. On the bottom, the stock price S^* is presented for a non-defaultable convertible: $p = 0$. The level of stock price S^* for the optimal call strategy versus time t is estimated by the AFV model using the PDE solver and the Monte Carlo solver.*

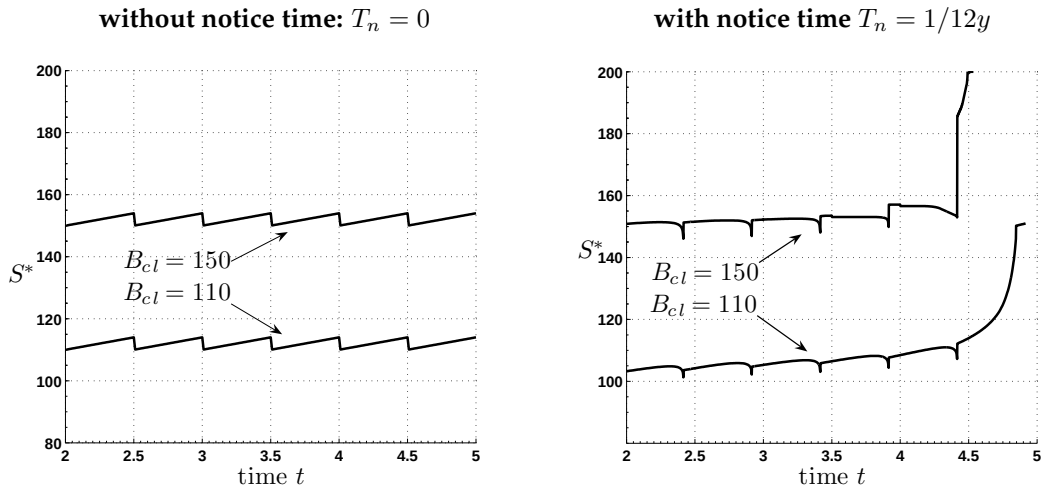

Figure 4.4: *The optimal stock price S^* for a call of a convertible using the data from Table 4.1 except a call notice time of zero is presented on the left. On the right, the stock price S^* is presented with a notice time of one month. The level of stock price S^* for the optimal call strategy over time t approximated by the AFV model using the PDE solver.*

The error of approximating the optimal strategy by the PDE method is presented in Figure 4.2. In this figure, S^* is approximated by a PDE solution on a coarse grid and on a fine grid. The difference between the two approximations is less than 0.02 for $t < 4.5$. The calculation of S^* is less accurate for $t > 4.5$ because the grid is still coarse in both discretizations (a structured mesh is used). Furthermore, the gradients of the convertible bond value and the constraint are very close for $S \gg B_c$:

$$\frac{\partial V}{\partial S} \approx \frac{\partial V^{\text{called},t}}{\partial S} \approx 1. \tag{4.29}$$

A similar estimation procedure for the stock price S^* above which it is optimal to call, is possible by the Monte Carlo algorithm:

$$S^*(t_i) \approx \min\left(S^j_{t_i} : P^e(S^j_{t_i}, t_i) \geq V^{\text{called},t_i}(S^j_{t_i}, \hat{t} = t_i)\right).$$

This leads to similar results as for the PDE. Figure (4.3) presents the estimates for $S^*(t_i)$ computed by a PDE method with 3200 nodes in S and an estimate by a Monte Carlo method with 10^6 asset paths. One can observe that both approaches deliver comparable results, but the PDE method has much less noise in the estimate. Consequently, the PDE method provides a reference point for the other approximations using a sufficiently fine grid. Note that the MC estimate is particularly noisy near expiry, which happens because the holder of the convertible receives about the same cash flows with and without an issuer's call. Consequently, the shapes of $V(S_{t_i})$ and $V^{\text{called},t_i}(S_{t_i})$ are very similar and the numerical procedure starts oscillating.

In Figure 4.4, the left graph depicts the PDE estimate of the optimal stock price level for a call S^* for two convertibles without notice period and different call prices. The value of the optimal

stock price level equals the clean call price plus the accrued interest, as derived in [65]. The right graph in Figure 4.4 presents the optimal stock price level S^* for the same convertible but including call protection by a one month notice. Now, the optimal call strategy is more complex: there is a large drop in S^* just before a coupon payment is within the notice period and a jump as soon as the coupon payment is within the notice period. A closer look at this phenomenon shows that this is a result of the "screw clause": The holder will not receive the accrued interest if he chooses to convert into shares, but he can still receive a coupon if the payment date is within the notice period. An issuer's call just before the coupon falls within the notice period avoids this situation.

An interesting property of the optimal call strategy (Figure 4.4) is that it does not seem to be optimal to call after the last coupon before maturity is paid. This is because the issuer is trying to minimize the value of the convertible. Consequently, the issuer tries to avoid the situation where the holder gets a coupon plus the opportunity to convert into shares. Thus, the value for S^* is relatively low just before a coupon payment takes place. But at maturity, the holder gets either the face value plus the last coupon or κ shares and no coupon. So, there is no need for the issuer to call because the holder cannot get both.

From Figure 4.4, we can see that in the case of a notice period, the optimal S^* at which the issuer should call the convertible is most of the time higher than in case of no notice period with $B_{cl} = 150$. But, with $B_{cl} = 110$, S^* is most of the time lower than in case of no notice period. This is surprising since it means that in general, the delayed call observed in the real market cannot be explained by a call notice period.

Implications of Different Call Strategies

It is interesting to examine the effect of the call policies on the convertible bond value. In Figure 4.5 on the left, we can see the effect of different notice periods on the value of the convertible. These results are all obtained using our accurate PDE method (AFV model). The premium for a notice period varies over the stock price S with a maximum between 85 and 95. As predicted, the premium is larger for a longer notice period. The premium for a typical notice period with 30 days is about 0.55, a significant addition. The reason for this is that the issuers interest it to minimize the value of the convertible and the notice period makes it more difficult of her. Thus, the price of the convertible rises introducing call notice period.

The right graph in Figure 4.5 shows the gain in value holding the notice period fixed at one month notice and varying the clean call prices B_{cl}. A higher call price leads to higher gains in value due to the call notice period.

Another interesting subject is the effect of suboptimal call policies, especially the delayed call phenomenon. Assuming that issuers call their convertibles late, what is the effect on the value? Consider the following strategy. The issuer calls only if it is beneficial for him to call, but he will not call until the stock price level S^* is reached. This seems to be a realistic assumption, because the issuers tend to call their convertible bonds late.

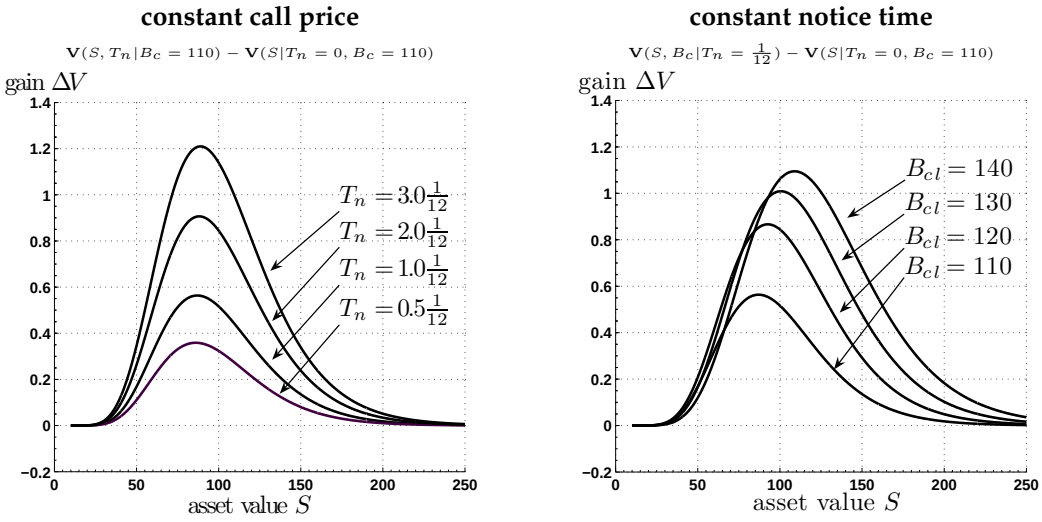

Figure 4.5: *The impact of notice periods on the initial value of the convertible bond. The difference in value compared to a convertible with $T_n = 0$ is shown across the stock price ($t = 0$). The graph on the left shows the effect of different notice periods with constant call price $B_c = 110$, while the graph on the right shows the effect of different call prices and a constant notice period of $T_n = 1/12$. All values are computed using the AFV model with data in Table 4.1.*

This last calling strategy is implemented by altering the model for valuation with notice periods. The Inequalities (4.13)-(4.16) become for $S_t < S^*$

$$\frac{\partial V_t}{\partial t} + \frac{\sigma^2}{2} S_t^2 \frac{\partial^2 V_t}{\partial S_t^2} + (r + p\eta) S_t \frac{\partial V_t}{\partial S_t} - (r + p) V_t + p\kappa S_t (1 - \eta) \geq 0$$

$$V(S_t, t) \geq \max(B_p(S_t, t), \kappa S_t)$$

and for $S_t \geq S^*$

$$\frac{\partial V_t}{\partial t} + \frac{\sigma^2}{2} S_t^2 \frac{\partial^2 V_t}{\partial S_t^2} + (r + p\eta) S_t \frac{\partial V_t}{\partial S_t} - (r + p) V_t + p\kappa S_t (1 - \eta) \geq 0$$

$$V(S_t, t) \geq \max(B_p(S_t, t), \kappa S_t)$$

$$\frac{\partial V_t}{\partial t} + \frac{\sigma^2}{2} S_t^2 \frac{\partial^2 V_t}{\partial S_t^2} + (r + p\eta) S_t \frac{\partial V_t}{\partial S_t} - (r + p) V_t + p\kappa S_t (1 - \eta) \leq 0$$

$$V(S_t, t) \leq V^{\text{called},t}(S_t, t),$$

where at least one of the inequalities holds with equality on the complete solution.

The impact of this new call strategy is presented in Figure 4.6 in the graph on the left. The difference in value compared with the optimal strategy (for a range of $S^* = \{120, 130, 140, 150\}$) for a convertible bond from Table 4.1 is shown over the stock price. This premium rises sharply for higher values S^*. But, for $S^* = 150$, we have a maximum impact on the value of about 7.00.

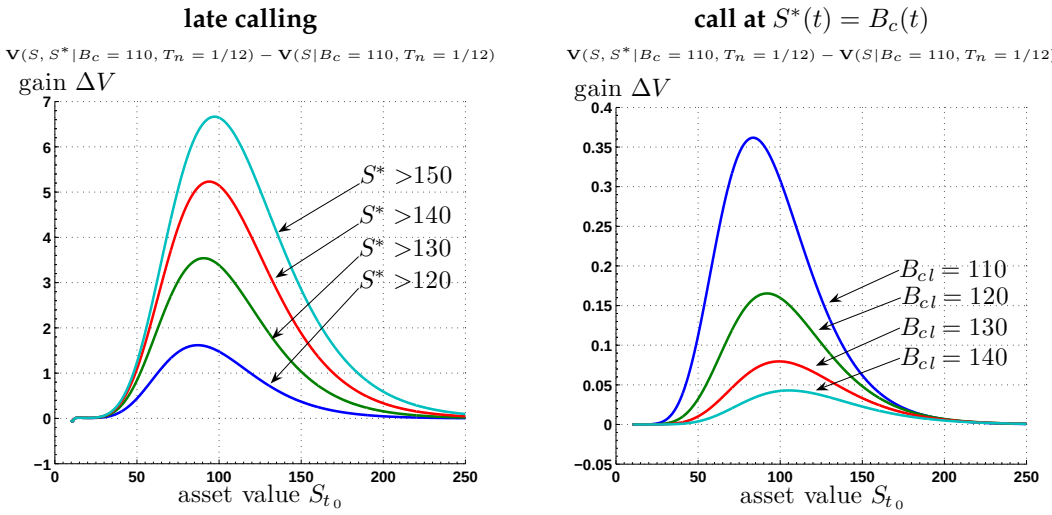

Figure 4.6: *The impact of suboptimal calling. The value of a convertible with data in Table 4.1 and optimal call strategy is compared to suboptimal call strategies. A strategy which calls at higher than optimal values results in an increase of the security value presented in the left graph. The right graph depicts the difference in value of the convertible called as if there were no notice compared to the optimal call strategy.*

That means that the optimality of the issuer's behavior also has a significant impact on the value of a convertible bond.

We now consider a second, somewhat realistic scenario. Suppose the issuer uses an approximate method to determine the optimal call policy which we denote S^*. This strategy can be modeled by replacing Inequalities (4.13)-(4.16) by

$$\frac{\partial V_t}{\partial t} + \frac{\sigma^2}{2} S_t^2 \frac{\partial^2 V_t}{\partial S_t^2} + (r + p\eta) S_t \frac{\partial V_t}{\partial S_t} - (r + p) V_t + p\kappa S_t (1 - \eta) \geq 0$$
$$V(S_t, t) \geq \max(B_p(S_t, t), \kappa S_t)$$

for $S_t < S_t^*$ and by

$$\frac{\partial V_t}{\partial t} + \frac{\sigma^2}{2} S_t^2 \frac{\partial^2 V_t}{\partial S_t^2} + (r + p\eta) S_t \frac{\partial V_t}{\partial S_t} - (r + p) V_t + p\kappa S_t (1 - \eta) \geq 0$$
$$V(S_t, t) \geq \max(B_p(S_t, t), \kappa S_t)$$
$$\frac{\partial V_t}{\partial t} + \frac{\sigma^2}{2} S_t^2 \frac{\partial^2 V_t}{\partial S_t^2} + (r + p\eta) S_t \frac{\partial V_t}{\partial S_t} - (r + p) V_t + p\kappa S_t (1 - \eta) \leq 0$$
$$V(S_t, t) = V^{\text{called},t}(S_t, t) \tag{4.30}$$

for $S_t \geq S_t^*$, where at least one of the inequalities holds with equality on the complete solution. Since this strategy is suboptimal, all values computed using this set of Equations will be larger

than values obtained with the optimal method (Inequalities (4.13)-(4.16)). This makes the resulting premium a good measure of the error of approximating the optimal call strategy.

The graph on the right in Figure 4.6 shows the premium (compared to the optimal strategy) at $t = 0$ due to a call strategy which ignores the call notice period. More specifically, we set $S_t^* = B_c(S_t, t)$, which is optimal for a notice period of zero (cp. Equation (4.27)). One can see that this policy has only a slight effect on the value (varying from 0.04 to 0.36). In other words, computing the optimal value S^* for calling by ignoring the call notice period can be a good approximation of the optimal call (but note that this assumes taking into account that the holder receives an option worth V^{called} upon call notice). This depends on the set of parameters. Using the Ingersoll call policy in case of dividend payments can add significant value.

4.5.4 Moving Window and Call Notice Protection

We now want to assess the effects of a moving window trigger as soft call protection. More precisely, the underlying asset has to stay $M = 20$ out of the last $N = 30$ days above a trigger level before a call notice can be issued. In our example, we set the trigger level to the clean call price, which is often seen in convertible bond contracts. This trigger level constraint will increase the value of the convertible. Furthermore, the issuer must give 30 days notice before the bond is called.

A mathematical formulation of the trigger constraint can be found in Equation (4.12). We use a discrete formulation since discrete observation is more realistic in convertible bond contracts.

As already mentioned, since no similar pricing procedure can be done using the PDE method, we rely on our Monte Carlo estimates.

Influence of Moving Window Constraints

Numerical experiments of the PDE model show that in our example (no dividends), a conversion of the convertible is not optimal for a holder at any time, even in the presence of the default model. Consequently, we will use the approximation that only call stopping times are computed for the Monte Carlo valuations in this section, conversion is ignored. This allows to see the average value of many Monte Carlo estimates as an estimate for an upper bound of the true price.

Table 4.5 shows the results for various choices of basis functions in the Least-Squares Monte Carlo regression. The dimension of the basis refers to the number of observations taken into account in the window of historic observations. The *correct* method requires a 30 dimensional basis. The L in Table 4.5 refers to the level of the sparse basis, as in Equation (1.14), which results in m (# basis) different basis functions.

In Table 4.5 we can see the values for a different number of observations in the moving window. Note: *value* is the average of #*sims* simulations with #*paths* asset paths each. The confidence level is $2\frac{std(\mathbf{V})}{\sqrt{\#sims}}$ where $\mathbf{V}$ is the set of values of the simulations with #*sims* elements, which cor-

responds to over 95% probability that the correct value lies within the reported interval.[3] The presented values are the best values of a large series of computations. It turns out that an increase in the sparseness level L (i.e. the #basis) does not always lead to lower upper bound estimates. This is a result of numerical difficulties (poor condition of the matrix with basis function values) and insufficiently many simulations for a good approximation of the conditional expected continuation value. The rows where the trigger is set to "-", contain the values of the example without a trigger condition. All other rows contain the values for the example with the constraint that the underlying stock has to be 20 out of the last 30 days above the trigger price before a call notice can be issued.

The values in Table 4.5 suggest that the estimation of the continuation value based only on the current asset price is already extremely good. That means, we include the complete trigger condition in order to decide, which paths allow a call notices. But then, we use only the current stock price of these paths for the computation of the expected continuation value. This has obviously a systematic error. But, the approximation is cheap and leads to good estimates: The value is very close (for $B_{cl} = 110$) to the estimate of the full 30 dimensional basis functions 123.69 ± 0.01. This is also true, for a clean call price of $B_{cl} = 140$.

Consequently, it appears that a one dimensional basis function can already provide good estimates for the continuation value of a convertible bond with a 20 out of 30 day moving window trigger price protection. Furthermore, the value added for the moving window call protection, compared with an unprotected convertible (0.79 for $B_{cl} = 110$ and 0.38 for $B_{cl} = 140$) can be a significant effect.

At first sight, it is difficult to understand why such a comparatively poor approximation (only considering a basis using the current asset price) should yield such a good solution, when, at least in theory, the value of the bond depends on the past 30 day history. However, note that in the Monte Carlo simulation, calls cannot be issued along a particular path, unless the asset has been above the trigger for 20 out of the past 30 days. Consequently, it appears that much of the path dependency has already been taken into account, and there is little error introduced when we use only a low-dimensional basis. In our example, the required CPU time for one evaluation of the one dimensional approximation is only one tenth of the time required for the thirty dimensional problem.

[3] See Table 1.2 for details.

Table 4.5 *Average Monte Carlo estimates of value for a convertible callable when S is 20 out of 30 days above $B_{triggger} = B_{cl} = 110$ and $B_{triggger} = B_{cl} = 140$ is presented compared to one without this trigger condition. The data of the convertible is given in Table 4.1, the average Monte Carlo values lie with a probability of about 68% within the reported confidence.*

trigger	B_{cl}	observations in basis	value	confidence	dim	L	#paths n	#sims I	#basis m
-	110	S_t	122.91	+- 0.01	1	3	10^6	30	15
110	110	S_t	**123.69**	+- 0.01	1	3	10^6	20	15
110	110	S_t, S_{t-29}	123.73	+- 0.02	2	2	10^5	250	17
110	110	$S_t, S_{t-5}, ..., S_{t-25}, S_{t-29}$	123.72	+- 0.02	7	1	10^5	250	15
110	110	$S_t, S_{t-5}, ..., S_{t-25}, S_{t-29}$	**123.69**	+- 0.02	7	2	$3 \cdot 10^5$	15	361
110	110	$S_t, S_{t-1}, ..., S_{t-29}$	**123.69**	+- 0.01	30	1	10^6	30	61
-	140	S_t	129.26	+- 0.02	1	2	10^5	250	7
140	140	S_t	**129.64**	+- 0.02	1	2	10^5	250	7
140	140	S_t, S_{t-29}	129.64	+- 0.02	2	2	10^5	250	17
140	140	$S_t, S_{t-5}, ..., S_{t-25}, S_{t-29}$	129.67	+- 0.02	7	1	10^5	250	15

4.6 Summary

Convertible bonds are a popular financial instrument with complex behavior. The notice period which prevents the issuer from an immediate call for conversion has a significant impact on the theoretical value of a convertible bond and the optimal call strategy of the issuer.

In this chapter, we compare both PDE and Monte Carlo approaches to pricing convertible bonds with complex call features. In particular, we examine calls with notice periods, and moving window call protection, whereby calls cannot be issued unless the underlying asset is observed above a trigger level for m out of the last n days.

Various authors have analyzed the delayed call phenomenon. Evidence suggests that issuers wait to call their convertibles until the stock price is well above its optimal level. If we assume such a delayed call, we of course find that the value of the convertible is larger than the convertible without a notice period. For example if the convertible is called 10% above the optimal value, with a notice period of 30 days, the value of the convertible increases by about 1% compared with the optimal strategy. A notice period of 30 days, assuming optimal issuer behavior, adds about 0.5% to the value compared to a bond with no notice period.

Some authors argue that the introduction of a notice period results in a higher stock price level which is optimal for the issuer to call. We find that the call price and the schedule of coupon payments have a significant effect on this stock price level. In general, the optimal stock price is higher than the call price for convertibles with notice periods, but in some cases, it is lower. Just before a coupon payment falls within the notice period, an optimal call by the issuer can be at a considerably lower stock price than the call price.

For the case of convertible bonds with simple notice periods, we find that a Least-Squares Monte Carlo approach gives quite good solutions compared with an accurate PDE solution. These results are consistent with those reported in [84], we extend their results to continuous call and conversion. We find that the PDE approach is computationally much more efficient and it is very expensive to get cent accurate Monte Carlo estimates.

However, it is not feasible to value convertibles with the moving window soft call protection using a PDE method. Our Monte Carlo method is based on least-squares regressions on sparse basis functions. In principle, we need a large dimensional basis set to take into account the path dependency of this contract feature. However, it appears that a very low order approximation (i.e. a basis using only the current asset price) yields a very good solution, compared with a full dimensional sparse basis. This is very fortunate, since the computational cost of using a full dimensional basis (even for the case of a sparse basis) is very high.

Chapter 5

Simulation-Based Hedging and Incomplete Markets

5.1 Overview

The previous chapters demonstrated a few applications of regressions for a complete market which follows the Black-Scholes assumptions. The presented methods can increase the speed of Monte Carlo pricing significantly and even allow to evaluate options for which pricing was not possible before. Applying regressions to incomplete markets, which do not follow the Black-Scholes assumptions, this chapter goes a large step further than the previous ones.

Even though, the axiom that financial markets do not allow arbitrage and the axiom that they are complete lead to various breakthroughs in the previous decades, this chapter lies out of this line of derivatives research by only assuming that a real-world model for the underlying hedge instrument exists. The optimal hedging strategies are computed based on statistical properties of the market and prices are obtained by the computation of all cost components of the derivatives hedge including the cost of risk. This allows to apply Monte Carlo pricing to many more realistic market scenarios than the ones based on the Black-Scholes assumptions.

It turns out that this approach can be seen as a powerful extension to common pricing frameworks such as Least-Squares Monte Carlo. But it implies desirable properties, such as higher accuracy for complete markets than comparable Least-Squares Monte Carlo methods and more realistic prices in incomplete markets.

5.2 Introduction

Many extensions to the famous Black-Scholes model have been published including local volatility surfaces[1], stochastic volatility[2], jumps[3] and transaction costs[4]. These models often still assume that markets are arbitrage-free and in some cases even complete. For the incomplete markets, valuation can be conduced in many different ways, since the corresponding so called equivalent martingale measure is not unique[5] which it is in the complete market case. Solutions to the option pricing problem in incomplete markets include[6] risk minimization [46, 95] (especially variance minimization [48, 47, 104, 33]), utility maximization [68, 54, 91] and martingale methods [51, 75]. In this chapter we will present a simple numerical framework which can provide numerical solutions to many of the complete and incomplete market models. We propose simple properties which we think an option price should satisfy. The new setting we propose is similar to the one presented by Schweizer [104]. His work provides valuable structural results, which correspond to our findings. But, our work extends these structural results to an efficient numerical valuation technique based on Monte Carlo simulation.

We present a method which is especially designed for but not limited to the pricing and hedging of OTC (Over-The-Counter) options which are not liquidly traded in the market. The new method will be based on the view of the hedging issuer of a derivative and overcomes many of the deficiencies of other methods. Additionally, the framework fits seamless into the risk management already existent in banks. The numerical implementation is very general and can be easily extended to all kinds of options and hedging scenarios.

We call the algorithm of this chapter Simulation-Based Hedging. It will compute optimal portfolios explicitly in order to obtain prices which have a foundation on a hedging strategy in the physical or real-world measure. Especially in illiquid markets, where no option prices can be fitted to traded prices, the complete market models suffer from systematic errors. In these situations the Simulation-Based Hedging can rely on econometric models for the underlying which capture the real-world dynamics in order to compute realistic prices.

In order to obtain a versatile method which can be used to price a large multitude of options such as path-dependent, exercisable, or callable options we present a framework which can be seen as an extension to the Least-Squares Monte Carlo by Carrière [32] resp. Longstaff and Schwartz [81]. The underlying principles for the valuation of exotic options can easily be adapted to our algorithm.

[1] Local volatility models were introduced as implied trees by Dupire [41] as well as Derman and Kani [39]. Further references can be found in Wilmott [117, p. 357ff].

[2] Introduced by Hull and White in 1987 [63], stochastic volatility models became especially popular after closed-form solutions were published 1993 by Heston [61].

[3] Option pricing when the underlying jumps was first studied 1976 by Merton [88].

[4] In 1985, option pricing with transaction costs was studied by Leland [79] who introduced proportional transaction costs to the Black-Scholes option pricing framework. A summary of references can be found in [117, p.353].

[5] In an arbitrage-free market, a trader can only earn more than the risk-free rate with a risky investment. These and other concepts are presented by many authors, e.g. Zagst [120], Panjer [92] and Pliska [94].

[6] An overview of the different methods can be found in [90, p.99ff, p.252ff] as well as [34].

Simulation-Based Hedging is similar to mean-variance option pricing in incomplete markets, which corresponds to the maximization of a quadratic utility function [104]. But, the method of this chapter adjusts the prices to account for the remaining risk of the hedged position which delivers prices a bank could directly trade on. A numerical method which is similar to the Simulation-Based Hedging is the Hedged Monte Carlo presented by Potters et. al.[96] respectively Pochart and Bouchaud [95]. Potters' method did not find wide acceptance due to its convergence properties: It is challenging to find the right parameter set which delivers accurate option prices because they compute their prices directly by regressions. Instead, we perform regressions to compute the optimal hedge only. This preserves the convergence to the correct values and allows for a higher accuracy since the option prices do not rely much on the specific set of basis functions used. Other related methods are proposed by Ryabchenko et. al. [101] with a global quadratic optimization as well as Luenberger [82] who unifies a dynamic version of CAPM and Black-Scholes in a continuous time setting.

The remainder of this chapter is structured as follows: the next section will introduce the setting we want to study including a definition of the objective of the trader and the risk management of the bank. A section follows, which presents the optimal solution to the setting. Implementing the optimal solution in a numerical algorithm, a section explains details for the practical pricing and hedging estimation. Then, we discuss possible extensions illustrated using the example of a GARCH process for the return distribution of the option's underlying. Finally, we conclude with a summary of the results.

5.3 Derivation

5.3.1 Basic Requirements for a Pricing Method

Before we proceed with the derivation of a pricing method for options, we want to see what the objectives for a derivatives price are.

The motivation for a new pricing method lies mainly in the observation that a large OTC market exists where positions of exotic options are traded. Some of these exotics are rarely traded so that a market price is not observable. Others might be traded in higher volume, but the ask-bid spread is substantially high. Even the ask price of the different sellers for such an option might be considerably varying because the different market participants use different models and assumptions since the Black-Scholes prices are not sufficient. The models used instead usually try to present a *fair market value* plus some spread the bank wants to earn. But, it is not even clear whether this fair market value covers the cost of the hedging strategy.

The reason why the derivatives prices based on strategies like maximal utility, minimal risk and market price of risk cannot ensure that the price is at least on average greater than the cost of the hedging strategy lies in the treatment of risk. On the one hand, the option is priced by a utility function of the bank - whatever that is - or a market price of risk. On the other hand, the bank has

to pay for the risk of her portfolio by keeping money in a margin account. This margin account will have to cover losses occurring and has to be supported by the bank's own funds, which have to earn the equity return.

In order to develop a pricing tool, which does not suffer from these deficits, we will deduce four fundamental properties we require for the prices at which a hedging derivative's issuer should trade.

Property 5.1 *A method which delivers the price of a derivative also delivers a corresponding hedging strategy, which an issuer can follow.*

Property 5.2 *The price of a derivative covers at least on average all cost components which occur at the issuer.*

Property 5.3 *The hedging strategy reduces the risk of the issuer.*

Property 5.4 *Any realistic, physical market model is allowed in the pricing method.*

The first property is clearly a matter of realism: the issuer of an option desires to replicate the option using a hedging strategy. That means Property 5.1 has to be fulfilled, otherwise the issuer has no information, how her hedging strategy looks like.

Property 5.2 denotes a second principle, which is clearly a requirement of a bank. The average cost of a derivatives trade must be estimated, otherwise one could not control the companies earnings accurately.

The third property seems obvious. The important issue of Property 5.3 is that a hedging strategy should not be designed for maximizing profits or utilities. It should be designed to minimize the required capital in the margin account since the bank's own funds are limited and thus expensive.

The last Property 5.4 is a requirement such that the issuer can tie the derivative's price to as much information as possible. The issuer has specific knowledge about the underlying and it's market and she should be able to make it available to the pricing method.

5.3.2 Hedging and Pricing of a Liquidly Traded Security

Using these Properties 5.1- 5.4, we will deduce pricing procedures which allow the computation of prices one could trade on in a real market. We start with a derivative which is already traded in the market with a small ask-bid spread. From Properties 5.1-5.4 it is immediately clear what the method for the pricing of such a liquidly traded derivative is: the issuer sells the derivative at a price V^{ask} with

$$V^{\text{ask}} = V^{\text{ask, market}} + C$$

where $V^{\text{ask, market}}$ is the market ask price of the option and C is the residual transaction cost plus a spread the issuer wants to earn. Now, the issuer's hedging strategy is to buy the derivative at the market for $V^{\text{ask, market}}$.

However, the method for pricing of an illiquid derivative is not obvious. In the next sections, we will see that pricing and hedging are still possible.

5.3.3 Setting for an Illiquid Market

Based on the requirements of Properties 5.1-5.4, we proceed assembling the parts of a real-world pricing method for illiquid derivatives. However, we will restrict our first setting to a derivative with a value V and a payoff function depending on a single underlying. This is extended in Section 5.4.6 to an example of a hedge with two hedge instruments. The terminal value of the derivative is denoted by $V_{t_T} = P(S_{t_T})$ with payoff P which depends of the asset price at maturity time t_T, only. A generalization at a payoff which depends on $\mathbb{S} := \{S_\tau | \tau \in \mathbb{I}\}$, $\mathbb{I} \subseteq \{t_0, t_1, \ldots, t_T\}$, i.e. the whole paths history is straight forward. We consider an issuer who sells this derivative and holds no other position in the market.

Property 5.1 requires a discrete-time model, because a hedging party can only buy and sell the underlying at discrete times in order to follow a hedging strategy. Note that this does not prevent the computation of the option's payoff $P(S_{t_T})$ because it may only depend on discrete samples S_{t_i} of the underlying. Consequently, we employ a discrete-time market model, where

$$S_{t_{i+1}} = f_{t_{i+1}}(S_{t_i}), \tag{5.1}$$

with the underlying asset S_{t_i} at time t_i and the transition function $f_{t_{i+1}}(S_{t_i})$ which describes the change of S from time t_i to time t_{i+1}. The transition function $f_{t_{i+1}}$ is arbitrary with a continuous probability density which allows to model a large variety of markets. This function can easily be extended to account for state variables such as multiple underlyings or stochastic volatility.

Suppose that the derivative is not actively traded in the market so that the position in the derivative cannot easily be closed by the issuer. That means, the issuer's risk management has to account for the risk over the whole lifetime of the derivative. A hedging strategy reducing the risk involved in holding the short position in the derivative will be initiated and the question arises, how much the issuer should charge for the derivative. It is clear that, the issuer should charge at least all costs occurring in the derivatives trade (Property 5.2), i.e. she should charge the expected cost of the hedge portfolio, the transaction costs plus the capital costs of the margin account.

To determine the expected cost of the hedge portfolio we are going through the hedging process backwards in time, starting at maturity.

In this setting, we are computing the value of a portfolio Π which compensates exactly for the cost occurring in the hedge of the derivative. We will restrict ourselves here to a simple portfolio

$$\Pi_{t_i} = B_{t_i} + \phi_{t_i} S_{t_i} \tag{5.2}$$

with a bank account value B_{t_i} and a position ϕ_{t_i} in the options underlying S_{t_i}.

The exact quantity of money required in the bank account B and thus the portfolio value Π are unknown at the initialization time t_0 of the derivatives trade. Thus, distributions for B_{t_0} and Π_{t_0} will be computed. The exact value of Π and B are known at maturity time t_T of the option, only. This makes the processes Π and B measurable at t_T. However, since the hedging strategy should be feasible in a real environment (Property 5.1), the hedging strategy itself must be known (measurable) at each time step t_i.

Focussing on the bank account B, the issuer has to pay the payoff of the derivative at maturity, i.e. the trader sells the hedge

$$\phi_{t_T} = 0$$

and the bank account compensates for the payoff payed to the option holder,

$$B_{t_T} = V_{t_T}.$$

At this time (maturity), Π is measurable:

$$\Pi_{t_T} = B_{t_T} + \phi_{t_T} S_{t_T} = V_{t_T}.$$

In order to obtain a hedging strategy for the whole life time of the option, we proceed by an induction from t_{i+1} to t_i. The bank account at time t_i should be able to compensate for the money required at time t_{i+1}, i.e.

$$\begin{aligned} e^{r(t_{i+1}-t_i)} B_{t_i} + \phi_{t_i} S_{t_{i+1}} &= B_{t_{i+1}} + \phi_{t_{i+1}} S_{t_{i+1}} && (5.3) \\ \Leftrightarrow B_{t_i} &= e^{-r(t_{i+1}-t_i)} (B_{t_{i+1}} + (\phi_{t_{i+1}} - \phi_{t_i}) S_{t_{i+1}}) && (5.4) \end{aligned}$$

where $(\phi_{t_{i+1}} - \phi_{t_i}) S_{t_{i+1}}$ denotes the profit from the position in the underlying and r is the interest rate the option's issuer receives respectively pays on a bank account. That means, B_{t_i} is only measurable at maturity time $t_i = t_T$. Furthermore, the equation for B describes a self-financing hedging strategy[7]. We can use Equation (5.4) to derive the value of the hedge portfolio

$$\begin{aligned} \Pi_{t_i} &= B_{t_i} + \phi_{t_i} S_{t_i} \\ &= e^{-r(t_{i+1}-t_i)} (B_{t_{i+1}} + (\phi_{t_{i+1}} - \phi_{t_i}) S_{t_{i+1}}) + \phi_{t_i} S_{t_i} \\ &= e^{-r(t_{i+1}-t_i)} \left(\Pi_{t_{i+1}} - \phi_{t_i} \left(S_{t_{i+1}} - e^{r(t_{i+1}-t_i)} S_{t_i} \right) \right), \qquad (5.5) \end{aligned}$$

which is helpful in formulating the numerical evaluation in the next section. Again, Π_{t_i} is only measurable at maturity time $t_i = t_T$. Finally, the expected cost of the hedging strategy is given by

$$V_{t_0} = \mathbb{E}[\Pi_{t_0}]. \qquad (5.6)$$

[7] A self-financing hedging strategy is a trading strategy which does not require external cash-flows hedging the option (see Zagst [120, p.52] for a more formal definition): All money required is available in the bank account. Note that in contrast to Schweizer [104] neither the bank account value nor the portfolio value are measurable at time $t_i < t_T$. Schweizer's framework contains a measurable portfolio, which is mean self-financing. That means the expected cost of hedging the option is met by the portfolio value.

Again, it is important to note that the hedge portfolio in Equation (5.5) is not deterministic at time t_0, because the bank account B_{t_0} denotes the money required to compensate for the hedging costs including the hedging error. Since the exact path of the underlying is not known at any time t_i, $i < T$, the required amount in the bank account is stochastic.

In order to determine, which trading strategy ϕ_{t_i}, $i \in \{0, \dots, T\}$ the issuer should follow, we have to consider the objectives of the issuer.

That means, the crucial part for the determination of the hedge is the choice of the objective function. The objective of the hedging party of a derivatives trade is to minimize the risk incurred in the derivative (Property 5.2). The risk can be measured in many ways which lead to different strategies.[8]

Hedging Strategies

There are several objectives of an issuer, which we have to consider in order to determine a specific hedging strategy. The issuer is usually a bank which constitutes of a derivatives trading department and an investment department. While the investment department actively takes risks, the derivatives trading department should not take risks. That means that the market models of the derivatives department are not made for investment strategies.

Consequently, **the objective of a derivatives hedge should be to minimize a symmetric risk measure**. Most other strategies demand that the drift

$$\mu = \frac{1}{(t_{i+1} - t_i)} \cdot \log \frac{\mathbb{E}\left[f_{t_{i+1}}(S_{t_i})\right]}{S_{t_i}}$$

of the underlying is correctly identified which is extremely difficult as this knowledge is usually not present to the derivatives department. If it were, one would still not consider over investing (or under investing) in an underlying to make a profit on the investment: The investment department already has an optimal portfolio allocation and usually such an over-investment in the derivatives underlying does not result in a better portfolio allocation. Consequently, a strategy is required which does not depend much on the underlying's drift. The traditional option pricing model of Black-Scholes perfectly fulfils this property, since the drift μ does not appear in the pricing equations. In other models it is desirable that the more often a hedge is conducted, the less should be the influence of μ on the option's price. In the later, we will see that variance minimization has this property.

Variance is a very simple risk measure, which corresponds to a quadratic utility of the issuer. Some higher order minimization strategies are also independent of the drift rate μ. However, they are not only harder to evaluate, but one looses the linear superposition of optimal hedges for individual options in a trading book.[9]

[8] It is worth noting that the risk should be measured in the physical measure and not in a risk-neutral way, otherwise the connection to the real-world trader would be lost.

[9] A strategy minimizing the fourth moment of the error and the resulting properties is described by Selmi and Bouchaud [105].

Another property to look at is the optimization procedure itself. Consider an issuer who already followed a risk-minimizing strategy for some time. During that time, he lost some money due to hedging errors. Should this issuer change the hedging portfolio in order to compensate for the errors? The answer is no, because the same arguments as before apply: the derivatives department should not invest in order to make gains on the position in the underlying but in order to minimize future risks. Consequently **only future risk should be hedged, that means a hedge should be local in time not global**[10].

Quadratic Hedging

Before we actually consider the variance minimization strategies of this work, we want to focus on related work in the literature: A setting similar to the setting of this thesis is the quadratic hedging or local variance minimization as described by Föllmer and Schweizer [47] and Schweizer [104].[11] We provide a brief summary of their framework using a notation similar to the previous section. We have an option value V with an underlying asset S. At maturity time t_T, the option has the same value as the payoff $P(S_{t_T})$,

$$V_{t_T} = P(S_{t_T}).$$

For technical reasons, this local variance minimization is defined with a risk-free interest rate of $r = 0$ and with process S being a martingale, i.e. $S_{t_i} = \mathbb{E}[S_{t_{i+1}}|S_{t_i}, t_i]$. Furthermore, given a hedging strategy $\phi = \phi_{t_0}, \ldots, \phi_{t_T}$ we define the function $\tilde{\Pi}_{t_T}$ as

$$\tilde{\Pi}_{t_T} := P(S_{t_T}) - \sum_{i=0}^{T-1} \phi_{t_i}(S_{t_{i+1}} - S_{t_i}),$$

i.e. it has the value of the payoff $P(S_{t_T})$ minus the so called gains process $\sum_{i=0}^{T-1} \phi_{t_i}(S_{t_{i+1}} - S_{t_i})$ from time t_0 until t_T. For $t_i < t_T$, we define

$$\tilde{\Pi}_{t_i} := \mathbb{E}\left[\tilde{\Pi}_{t_T}|S_{t_i}, t_i\right].$$

Consider a market such that the contingent claim $P(S_{t_T})$ is attainable[12]. Then, for the option price process V_{t_i},

$$\begin{aligned} V_{t_i} &:= \mathbb{E}\left[P(S_{t_T}) - \sum_{k=i}^{T-1} \phi_{t_k}(S_{t_{k+1}} - S_{t_k}) \mid S_{t_i}, t_i\right], \\ &= \tilde{\Pi}_{t_i} + \sum_{k=0}^{i-1} \phi_{t_k}(S_{t_{k+1}} - S_{t_k}) \end{aligned}$$

[10] Local in time means that the objective function for the decision on the hedge position at time t_i is not based on previous variable values at time $t_j,\ j < i$. Global in time means that the complete hedging strategy is defined at the initial time t_0.

[11] The local variance minimization is based on the work of Föllmer and Sondermann [48] who use variance minimization in a global setting. Černý and Kallsen [35] extended the work of Föllmer and Schweizer [47] to a use regressions in a global variance minimization, which is also related to the work of this thesis.

[12] A contingent claim $P(S_{t_T})$ is called attainable if there is a feasible hedging strategy, which perfectly replicates the contingent claim. For details see Zagst[120, p.62]

holds, especially $V_{t_0} = \tilde{\Pi}_{t_0}$ holds as well. Minimizing the local quadratic risk defined as the quadratic deviation from $\tilde{\Pi}_{t_{i+1}}$ to $\tilde{\Pi}_{t_i}$ at time t_i the optimal strategy is given by

$$\begin{aligned}
\{\phi_{t_i}, V_{t_i}\} &= \arg\min_{\phi_{t_i}, V_{t_i}} \left(\mathbb{E}\left[(\tilde{\Pi}_{t_{i+1}} - \tilde{\Pi}_{t_i})^2 \mid S_{t_i}, t_i\right]\right) \\
&= \arg\min_{\phi_{t_i}, V_{t_i}} \left(\mathbb{E}\left[\left(\left(V_{t_{i+1}} - \sum_{k=0}^{i} \phi_{t_k}(S_{t_{k+1}} - S_{t_k})\right) - \left(V_{t_i} - \sum_{k=0}^{i-1} \phi_{t_k}(S_{t_{k+1}} - S_{t_k})\right)\right)^2 \mid S_{t_i}, t_i\right]\right) \\
&= \arg\min_{\phi_{t_i}, V_{t_i}} \left(\mathbb{E}\left[\left(V_{t_{i+1}} - V_{t_i} - \phi_{t_i}(S_{t_{i+1}} - S_{t_i})\right)^2 \mid S_{t_i}, t_i\right]\right).
\end{aligned}$$

In this setting, V_{t_i} and ϕ_{t_i} are measurable at time t_i. Thus the solution to the minimization problem is given by

$$\begin{aligned}
\phi_{t_i} &= \frac{\operatorname{cov}\left[V_{t_{i+1}}, S_{t_{i+1}} \mid S_{t_i}, t_i\right]}{\operatorname{var}\left[S_{t_{i+1}} \mid S_{t_i}, t_i\right]}, \\
V_{t_i} &= \mathbb{E}[V_{t_{i+1}} | S_{t_i}, t_i] - \phi_{t_i}\mathbb{E}[S_{t_{i+1}} - S_{t_i} | S_{t_i}, t_i].
\end{aligned}$$

This setting is equivalent to the solution of

$$\{\phi_{t_i}\} = \operatorname*{argmin}_{\phi_{t_i}} \left(\operatorname{var}\left[\left(\phi_{t_i} S_{t_{i+1}} - V_{t_{i+1}}\right) \mid S_{t_i}, t_i\right]\right) \tag{5.7}$$

and setting the option value, such that it is self-financing on average:

$$V_{t_i} = \mathbb{E}[V_{t_{i+1}} | S_{t_i}, t_i] - \phi_{t_i}\mathbb{E}[S_{t_{i+1}} - S_{t_i} | S_{t_i}, t_i].$$

In contrast to the work of this thesis, where no intermediate option prices exist and hedge portfolio values are only measurable at t_T, the work of Föllmer and Schweizer is focused on option prices and portfolio values measurable at time t_i. That means, it is hard to compute global hedging errors and extensions to non-martingale underlyings and non-zero interest-rates are challenging. Furthermore, no efficient numerical methods are known for the computation of these minimum-variance hedges. However, a simple numerical method with a multi-nominal tree for this setting can be found in Černý [33]. A more general Monte Carlo method for the setting is presented by Potters et al [96] which we summarize in Section 5.4.3.

Now, we are proceeding with the new method of this thesis. In the next section, we want to see what are the different possible types of variance minimization and which one should use.

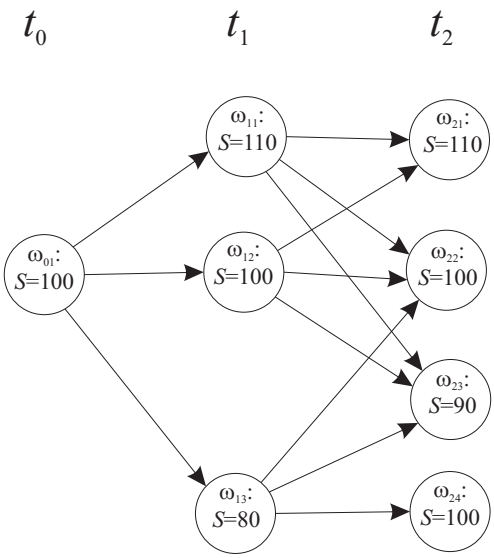

Figure 5.1: *Tree model of stock price S for an incomplete market with states $\omega_{01}, \ldots, \omega_{24}$.*

Variance Minimization Strategies

This section will take a closer look at the different possibilities for a variance minimization of the required portfolio value Π.

More formally, if we look at the value of the hedge portfolio Π_{t_i} required to compensate for the hedging strategy from time t_0 until maturity time t_T, the choices for variance-risk minimization we consider are *Global Hedging*, *Local Hedging*, and something we call *Forward Global Hedging*.

We will describe the different strategies based on the example of hedging a European call option in an incomplete market. Consider a discrete market model of asset price S with states as presented in Figure 5.1 and a risk-free rate of $r = 0$. The option has a payoff V_{t_2} at maturity time t_2 of

$$V_{t_2} = \max(S_{t_2} - 95, 0).$$

Additionally, the hedge portfolio Π_{t_2} is given by

$$\Pi_{t_2} = V_{t_2}.$$

With this information and the information of the transition $\Pi_{t+1} \rightarrow \Pi_t$, we will present the effects of the different variance minimization strategies.

Table 5.1 *Global optimization of a derivatives hedge in a market as in Figure 5.1.*

t_0				t_1				t_2		
S_{t_0}	Π_{t_0}	ϕ_{t_0}	$\mathrm{var}(\Pi_{t_0})$	S_{t_1}	Π_{t_1}	ϕ_{t_1}	$\mathrm{var}(\Pi_{t_1})$	S_{t_2}	V_{t_2}	Π_{t_2}
100	10.29			110	15.00			110	15	15
100	6.74			110	11.45	**0.65**	**3.18**	100	5	5
100	8.20			110	12.91			90	0	0
100	7.50			100	7.50			110	15	15
100	5.00	**0.47**	**2.54**	100	-5.00	**0.75**	**2.08**	100	5	5
100	7.50			100	7.50			90	0	0
100	8.37			80	-1.05			100	5	5
100	6.40			80	-3.02	**0.30**	**2.36**	90	0	0
100	9.42			80	0.00			80	0	0

1.) Global Hedging A globally optimal hedge minimizes the variance for the total hedging error,

$$\{\phi_{t_0}, \ldots, \phi_{t_T}\} = \arg \min_{\phi_{t_0}, \ldots, \phi_{t_T}} \left(\mathrm{var}\left[\Pi_{t_0} \mid S_{t_0}, t_0\right]\right),$$

with

$$\Pi_{t_i} = e^{-r(t_{i+1}-t_i)} \left(\Pi_{t_{i+1}} - \phi_{t_i} \left(S_{i+1} - e^{r(t_{i+1}-t_i)} S_i\right)\right), \quad \Pi_{t_T} = V_{t_T} = P(S_{t_T})$$

i.e. it minimizes the variance $\mathrm{var}(\Pi_{t_0})$ at time t_0. This can create odd artifacts: In some cases the trader would try to loose gains she made in the past and even pay for that. In other cases, she would change her strategy in order to compensate for losses in the past.

In the example of Figure 5.1, we use the parameters $\phi_{t_0}^{\omega_{01}}$, $\phi_{t_1}^{\omega_{11}}$, $\phi_{t_1}^{\omega_{12}}$ and $\phi_{t_1}^{\omega_{13}}$ to minimize the sample variance $\mathrm{var}(\Pi_{t_0})$.

Table 5.1 presents the result of the single-step four-dimensional optimization. The computation itself starts by setting all required values at time t_2. Then, the process proceeds at time t_1, where the underlying value S_{t_2} and the position ϕ_{t_1} in S_{t_1} is used for the computation of Π_{t_0}:

$$\Pi_{t_0} = \left(\Pi_{t_{i+1}} - \phi_{t_i} \left(S_{i+1} - S_i\right)\right), \quad i \in \{0, 1\}$$

since $r = 0$. Then, at time t_0, we compute Π_{t_0} from Π_{t_1}, ϕ_{t_0}, S_{t_1} and S_{t_0}. The values for the 4 different ϕ's are chosen by a simple gradient-based method such that the variance of the 9 samples of Π_{t_0} at time t_0 is minimal.

Table 5.2 *Local optimization of a derivatives hedge in a market as in Figure 5.1.*

t_0					t_1					t_2		
S_{t_0}	V_{t_0}	Π_{t_0}	ϕ_{t_0}	$\text{var}(\Pi_{t_0})$	S_{t_1}	V_{t_1}	Π_{t_1}	ϕ_{t_1}	$\text{var}(\Pi_{t_1})$	S_{t_2}	V_{t_2}	Π_{t_2}
100	8.27	9.35			110	14.17	15.0			90	0	0
100	8.27	9.35			110	14.17	15.0	**0.75**	**2.08**	110	15	15
100	8.27	9.35			110	14.17	12.5			100	5	5
100	8.27	6.67			100	6.67	7.5			110	15	15
100	8.27	6.67	**0.48**	**1.51**	100	6.67	5.0	**0.75**	**2.08**	100	5	5
100	8.27	6.67			100	6.67	7.5			90	0	0
100	8.27	8.81			80	-0.83	-0.0			100	5	5
100	8.27	8.81			80	-0.83	-2.5	**0.25**	**2.08**	90	0	0
100	8.27	8.81			80	-0.83	-0.0			80	0	0

2.) Local Hedging Local hedging minimizes the variance from one time step to the next, which is similar to the quadratic hedging presented in the section about quadratic hedging. This setting is intended for hedging of options for which we can observe a market price V_{t_i}. Since the hedge portfolio $\tilde{\Pi}_{t_i}$ should replicate the option price, we define

$$V_{t_{i+1}} := \mathbb{E}[\tilde{\Pi}_{t_{i+1}} \mid S_{t_i}, t_i], \quad V_{t_T} = P(S_{t_T})$$

i.e. that the expected value of the hedge position equals the option price.

As usual, we define the hedge portfolio as

$$\tilde{\Pi}_{t_i} := B_{t_i} + \phi_{t_i} S_{t_i}.$$

The objective of this local hedge at time t_i is to minimize the discrepancy be between a hedge portfolio $\tilde{\Pi}_{t_{i+1}}$ and the option value $V_{t_{i+1}}$, i.e.

$$\begin{aligned} \{\phi_{t_i}\} &= \arg\min_{\phi_{t_i}} \left(\text{var}\left[\tilde{\Pi}_{t_{i+1}} - V_{t_{i+1}} \mid S_{t_i}, t_i \right] \right), \\ &= \arg\min_{\phi_{t_i}} \left(\text{var}\left[B_{t_{i+1}} + \phi_{t_{i+1}} S_{t_{i+1}} - V_{t_{i+1}} \mid S_{t_i}, t_i \right] \right). \end{aligned}$$

Using the self financing property of Equation (5.4), this equals

$$\begin{aligned} \{\phi_{t_i}\} &= \arg\min_{\phi_{t_i}} \left(\text{var}\left[e^{r(t_{i+1}-t_i)} B_{t_i} + \phi_{t_i} S_{t_{i+1}} - V_{t_{i+1}} \mid S_{t_i}, t_i \right] \right), \\ &= \arg\min_{\phi_{t_i}} \left(\text{var}\left[\phi_{t_i} S_{t_{i+1}} - V_{t_{i+1}} \mid S_{t_i}, t_i \right] \right), \end{aligned}$$

which is essentially the same solution as presented in the quadratic hedging, Equation (5.7).

The result of this step-wise minimization for the example in Figure 5.1 and payoff $V_{t_2} = \max(S_{t_2} - 95, 0)$ is presented in Table 5.2. One can observe that local minimization results in

Table 5.3 *Forward global optimization of a derivatives hedge in a market as in Figure 5.1.*

t_0				t_1				t_2		
S_{t_0}	Π_{t_0}	ϕ_{t_0}	$\text{var}(\Pi_{t_0})$	S_{t_1}	Π_{t_1}	ϕ_{t_1}	$\text{var}(\Pi_{t_1})$	S_{t_2}	V_{t_2}	Π_{t_2}
100	10.18			110	15.00			110	15	15
100	7.68			110	12.50	**0.75**	**2.08**	100	5	5
100	10.18			110	15.00			90	0	0
100	7.50			100	7.50			110	15	15
100	5.00	**0.48**	**3.07**	100	5.00	**0.75**	**2.08**	100	5	5
100	7.50			100	7.50			90	0	0
100	9.64			80	-0.00			100	5	5
100	7.14			80	-2.50	**0.25**	**2.08**	90	0	0
100	9.64			80	-0.00			80	0	0

a different strategy than the global minimization in Table 5.1. However, since the transition from $\tilde{\Pi}_{t_{i+1}} \rightarrow \tilde{\Pi}_{t_i}$ is different than that from $\Pi_{t_{i+1}} \rightarrow \Pi_{t_i}$ in the global setting, the resulting variances cannot be compared directly.

3.) Forward Global Hedging A strategy between the local and the global is one, we call forward global. In a forward global variance minimization, one solves

$$\Pi_{t_i} = e^{-r(t_{i+1}-t_i)}\left(\Pi_{t_{i+1}} - \phi_{t_i}\left(S_{i+1} - e^{r(t_{i+1}-t_i)}S_i\right)\right), \quad \Pi_{t_T} = V_{t_T} = P(S_{t_T})$$

with

$$\{\phi_{t_i}\} = \arg\min_{\phi_{t_i}} \left(\text{var}\left[\Pi_{t_i}\right] \mid S_{t_i}, t_i\right)$$

and thus only minimizes the variance of the hedging actions required in the future and ignores all information from the past $t_0, \ldots, t_{t-1}$. In this setting, in contrast to Local Hedging, the complete future hedging errors can be obtained for further computations.

Table 5.3 presents the Forward Global Minimization. Starting at $\phi_{t_1}^{\omega_{11}}$, $\text{var}(\Pi_{t_0}^{\omega_{11}})$ is minimized, followed by $\phi_{t_1}^{\omega_{12}}$ where $\text{var}(\Pi_{t_0}^{\omega_{12}})$ and $\phi_{t_1}^{\omega_{13}}$ where $\text{var}(\Pi_{t_0}^{\omega_{13}})$ is minimized. Finally, $\phi_{t_0}^{\omega_{01}}$ is used for minimizing $\text{var}(\Pi_{t_0}^{\omega_{01}})$. The resulting strategy is identical to the Local Minimization, and different to the Global Minimization. In fact, the forward global and the local variance minimization strategy coincide in many cases. [13]

Choice of the Variance Minimization Strategy

It is obvious that a forward global risk minimization, not a global risk minimization should be pursued, since the global risk minimization tries to correct errors from the past by investing. The forward global risk minimization only reduces future hedging errors, based on the assumption that the future strategy does the same.

[13] In a similar mean-variance setting with no-arbitrage this has been shown by Černý [33, Theorem 2]. And in an other similar setting, this has been proved by Lamberton et al [76, Proposition 2].

Consequently, we focus on the forward global optimization. Then, the minimization decomposes into a minimization of the risky capital costs for each time-step. That means the optimal fraction ϕ_{t_i} of the hedge instrument S is given by

$$\begin{aligned} \{\phi_{t_i}\} &= \arg\min_{\phi_{t_i}} \left(\text{var}\left[\Pi_{t_i} \mid S_{t_i}, t_i\right]\right) \\ &= \arg\min_{\phi_{t_i}} \left(\text{var}\left[e^{-r(t_{i+1}-t_i)}(\Pi_{t_{i+1}} - (\phi_{t_i}(S_{t_{i+1}} - e^{r(t_{i+1}-t_i)}S_{t_i}))) \mid S_{t_i}, t_i\right]\right) \\ &= \arg\min_{\phi_{t_i}} \left(\text{var}\left[\Pi_{t_{i+1}} - \phi_{t_i} S_{t_{i+1}} \mid S_{t_i}, t_i\right]\right), \end{aligned} \tag{5.8}$$

which is similar to the quadratic hedging in discrete time which is discussed in the literature (cp. Section 5.3.3, *Quadratic Hedging*). We basically reduced all future values of the stochastic variables S and V to one stochastic variable $\Pi_{t_{i+1}}$. Now, the solution to (5.8) can be found by setting the first derivative $\frac{\partial}{\partial \phi_{t_i}}$ to zero and rearranging to ϕ_{t_i},

$$\phi_{t_i} = \frac{\text{cov}[\Pi_{t_{i+1}}, S_{t_{i+1}} \mid S_{t_i}, t_i]}{\text{var}[S_{t_{i+1}} \mid S_{t_i}, t_i]}. \tag{5.9}$$

The values of (5.9) can be computed using the regression method presented in Section 1.2.1.

Determination of the Cost of Risk

The cost of risk for a bank is given by the bank's risk management system: The bank has to deposit a certain amount of capital in a margin account. For simplicity, we assume that this capital is deposited at the initial time t_0 of the trade until maturity of the option t_T.

We propose a risk measure which is called Conditional Value at Risk of the future hedge error at the time of the initiation of the trade t_0:

$$R := \text{CVaR}\left[\Pi_{t_0}\right].$$

Other choices of the risk measurement (e.g. VaR) are easy to accommodate in the model, too. In a more complicated setting, the risk management department computes a marginal CVaR value, which is the CVAR of all hedges minus CVAR of all hedges except the one under consideration.

The risk management has to allocate R units of money in the margin account in order to compensate for unexpected future losses on the hedge. This money is equity capital which has to earn the return on equity, a risky interest rate r_e. That means, the derivative trade has to earn the accrued interest

$$(e^{r_e(t_T - t_0)} - 1)R \tag{5.10}$$

for the risk management, payable at maturity. Consequently, the cost of capital for the risk management is worth

$$C_R = e^{-r(t_T - t_0)}(e^{r_e(t_T - t_0)} - 1)R$$

at the initial time t_0.

The task of the issuer is to sell the derivative as expensive as possible and minimize the capital cost caused by the risk management. Again, the issuer's task is not to earn excess returns from the derivatives hedge because that would result in an investment strategy. But, investment decisions should be made consciously and not based on random derivative sales.

Now, we collected all ingredients for the option valuation: The value of an illiquid option is given by

$$V_{t_0}^{ask} = V_{t_0} + C_R.$$

The unadjusted value $V_{t_0} = \mathbb{E}[\Pi_{t_0}]$ is computed using the variance minimization starting at maturity time, while the margin capital cost is a direct result from the residual randomness in the hedged portfolio, measured by the bank's risk management system.

Now, $V_{t_0}^{ask}$ is the lowest price at which a trader should sell the option: Since she wants to maximize the bank's profit[14], the trader will add an additional spread to the option price, which is an excess profit.

5.3.4 Transaction Costs

In the previous section, we saw how to convert remaining risk of a derivatives hedge into costs which can be included in the derivative's price. In this section, we want access another important feature often omitted in option pricing: transaction costs. In this section we briefly discuss possible choices for the inclusion of transaction costs in Simulation-Based Hedging. In the next Chapter, a Monte Carlo method is presented which is then extended to transaction costs. This way, the complete costs of an option hedge can be compared for different hedging strategies.

Any kind of transaction costs model can be implemented for the Simulation-Based Hedging. But, an optimal solution requires the introduction of an additional state variable: The current position ϕ in the asset. This is computationally expensive and the optimal rehedge also requires a numerical minimization with respect to this state variable, to decide wether to rehedge or not. This is often not feasible due to insufficient computing capacity.

A more practical strategy follows from the current practice in option pricing: a time-based rehedge[15]. Here, one fixes specific dates (often equidistantly) at which the rehedge occurs. This is a strategy which can deliver satisfying results without introducing large computational efforts. The parameters of the strategy can also be optimized in order to find the optimal time interval. The Monte Carlo method in the following Chapter (Section 5.4.6) is based on this time-based rehedge.

As in the case without transaction costs, a rebalancing strategy which minimizes directly $V_{t_0}^{ask}$ including the expected replication cost would effectively be a strategy betting on the drift of the

[14] Maximizing the bank's profit, maximizes the trader's bonus.

[15] Different hedging strategies with transaction costs are discusses in Wilmott [117, p.331], the time-based rehedge for the Black-Scholes framework was analyzed by Leland [79]

underlying. As a result, an over (or under) investment in the underlying, which violates Property 5.3.

The objective, which one minimizes in order to find an optimal rehedge under transaction costs requires some further thoughts: A usual risk minimization would result in a strategy which rehedges every time step. Thus extremely large transaction costs occur. The Simulation-Based Hedging allows a more intuitive procedure: Instead of minimizing some risk, we try to find a strategy which minimizes the capital costs of the margin account plus the transaction costs. Thus, we find a balance between paying transaction costs and taking risk. Later, we will provide examples for this procedure.

5.3.5 American Put Options

Valuing exercisable options within this general framework cannot be based on no-arbitrage. However, this is still realistic because options are usually not exercised at the optimal values computed by no-arbitrage arguments [2]. Consequently, the American put option valuation has to be based on other means: The bank has to prepare for a hostile option holder who exercises at the worst possible time for the issuer. That means the expected conditional value of the hedge portfolio has to be at least as high as the exercise value at all times. This is accomplished by replacing Equation (5.6) with

$$\Pi_{t_i} = \begin{cases} (K - S_{t_i}) & \text{if } e^{-r(t_{i+1}-t_i)} \cdot \mathbb{E}[\Pi_{t_{i+1}} | S_{t_i}, t_i] < K - S_{t_i} \\ B_{t_i} + \phi_{t_i} S_{t_i} & \text{else} \end{cases} .$$

This is easy to compute following the ideas of Least-Squares Monte Carlo as presented in Section 1.2.1.

5.4 Monte Carlo Implementations

Implementing an algorithm for the Simulation-Based Hedging is easy using the techniques presented in Chapter 1, especially the regression method presented in Section 1.2.1 helps to determine the optimal hedges. A simple implementation can be found in Appendix 7.6.

In the next section, we will start with a detailed computation of a simple European put option with Simulation-Based Hedging. Then, brief summaries of Simulation-Based Hedging and the method from Potters [96] follow. A theoretical analysis of the convergence properties of the Least-Squared Monte-Carlo and Simulation-Based hedging follows, which is confirmed by numerical experiments.

5.4.1 Simple Example

After the theoretical considerations in the previous sections, we now focus on a simple example of the Simulation-based Hedging. Consider a European put option with data in Table 5.4. We

Table 5.4 *Data of a European put option.*

General features	
strike price K	100
risk-free rate r	5% p.a.
volatility σ	40% p.a.
drift rate μ	5% p.a.
maturity time t_T	1 year
terminal value $P(S_{t_T}, t_T)$	$\max(K - S_{t_T}, 0)$

evaluate this option with 10 asset paths using the Simulation-Based Hedging. As in the previous Chapters, we simulate the underlying's paths using Equation (1.28)

$$S_{t_i}^j = S_{t_{i-1}}^j e^{(r-\frac{1}{2}\sigma^2)(t_i - t_{i-1}) + \sigma\sqrt{t_i - t_{i-1}}\theta_{i,j}}$$

and the data in Table 5.4. With $S_{t_0} = 100$ and random numbers $\theta_{i,j}$, $i = 1\ldots, 10$, $j = 1, 2$, we get 10 asset paths:

j	$S_{t_0}^j$	$S_{t_1}^j$	$S_{t_2}^j$
1	100	81.6340	61.7521
2	100	120.7011	86.3528
3	100	89.1249	84.7387
4	100	118.4613	87.9178
5	100	104.5817	118.0245
6	100	81.0836	86.1567
7	100	58.9702	40.8700
8	100	101.6986	72.1828
9	100	65.8471	65.5679
10	100	119.6538	133.2725

Computing the option's payoff

$$V_{t_T}^j = V_{t_2}^j = \max(100 - S_{t_2}^j, 0)$$

we get

$$\mathbf{V}_{t_2} = \begin{pmatrix} 38.2479 \\ 13.6472 \\ 15.2613 \\ 12.0822 \\ 0 \\ 13.8433 \\ 59.1300 \\ 27.8172 \\ 34.4321 \\ 0 \end{pmatrix}.$$

For the portfolio $\Pi_{t_T} = B_{t_T} = V_{t_2}$ holds. Pathwise, we obtain

$$\mathbf{\Pi}_{t_2} = \begin{pmatrix} 38.2479 \\ 13.6472 \\ 15.2613 \\ 12.0822 \\ 0.0000 \\ 13.8433 \\ 59.1300 \\ 27.8172 \\ 34.4321 \\ 0.0000 \end{pmatrix}$$

as scenario values.

Now, all values at time $t_T = t_2$ are computed and we can proceed with the backwards time stepping to the previous time step $t_{T-1} = t_1$. At this time, we are computing the optimal hedge for all possible states based on the sample for the next time step.

We saw in the previous section that optimal hedge is given by

$$\phi^j_{t_1}(S^j_{t_1}) = \frac{\mathrm{cov}(S_{t_2}, \Pi_{t_2} | S^j_{t_1}, t_1)}{\mathrm{var}(S_{t_2} | S^j_{t_1}, t_1)}.$$

Following Section 1.2.1, we approximate $\phi^j_{t_1}(S^j_{t_1})$ by

$$\phi^j_{t_1}(S^j_{t_1}) \approx \sum_{j=1}^{m} \tilde{a}_j b_j(S^j_{t_1}),$$

with $\tilde{a}_j$ defined as (cp. Equation (1.7))

$$\begin{aligned} \tilde{\mathbf{a}} &= \left(\mathbf{B}_{\hat{S}}(S_{t_1})^T \mathbf{B}_{\hat{S}}(S_{t_1})\right)^{-1} \mathbf{B}_{\hat{S}}(S_{t_1})^T \hat{\mathbf{\Pi}} \\ &= \left(\mathbf{B}_{\hat{S}}(S_{t_1})\right)^{\dagger} \hat{\mathbf{\Pi}} \end{aligned} \tag{5.11}$$

with $\hat{S}^j = S^j_{t_2} - \mathbb{E}[S_{t_2} | S_{t_1} = S^j_{t_1}]$ and $\hat{\Pi}^j = \Pi^j_{t_2} - \mathbb{E}[\Pi_{t_2} | S_{t_1} = S^j_{t_1}]$. $\hat{\mathbf{S}}$ and $\hat{\mathbf{\Pi}}$ can be computed from the local basis approximation[16] $\mathbb{E}[S_{t_2} | S_{t_1} = S^j_{t_1}]$ of S_{t_2} respectively from the local basis approximation $\mathbb{E}[\Pi_{t_2} | S_{t_1} = S^j_{t_1}]$ of Π_{t_2}, i.e.

$$\hat{\mathbf{\Pi}} := \mathbf{\Pi}_{t_2} - \mathbf{B}(S_{t_1}) \cdot \left(\mathbf{B}(S_{t_1})^{\dagger} \mathbf{\Pi}_{t_2}\right)$$

and

$$\hat{\mathbf{S}} := \mathbf{S}_{t_2} - \mathbf{B}(S_{t_1}) \cdot \left(\mathbf{B}(S_{t_1})^{\dagger} \mathbf{S}_{t_2}\right)$$

with pseudo inverse $\mathbf{B}(S_{t_1})^{\dagger}$ of $\mathbf{B}(S_{t_1})$ (see Definition 1.6). In this Equation, we have to chose the appropriate basis functions $b_j(S_{t_1})$ for

$$\mathbf{B}(S_{t_1}) := \begin{pmatrix} b_1(S^1_{t_1}) & \cdots & b_m(S^1_{t_1}) \\ \vdots & \ddots & \vdots \\ b_1(S^{10}_{t_1}) & \cdots & b_m(S^{10}_{t_1}) \end{pmatrix}..$$

[16] The local basis approximation is defined in Theorem 1.4 and Lemma 1.5

Since we only have 10 samples to do the regression, we choose a polynomial basis with $m = 3$ basis functions, such that

$$\mathbf{B} := \mathbf{B}(S_{t_1}) = (1 \quad S^j_{t_1} \quad (S^j_{t_1})^2)\Big|_{j=1,\dots,10},$$

i.e.

$$\mathbf{B} = \begin{pmatrix} 1 & 81.6340 & 6664.1164 \\ 1 & 120.7011 & 14568.7494 \\ 1 & 89.1249 & 7943.2463 \\ 1 & 118.4613 & 14033.0766 \\ 1 & 104.5817 & 10937.3404 \\ 1 & 81.0836 & 6574.5581 \\ 1 & 58.9702 & 3477.4886 \\ 1 & 101.6986 & 10342.6003 \\ 1 & 65.8471 & 4335.8452 \\ 1 & 119.6538 & 14317.0342 \end{pmatrix}.$$

This results in

$$\hat{\mathbf{\Pi}} = \begin{pmatrix} 13.2697 \\ 4.5509 \\ -3.5234 \\ 3.1732 \\ -10.7258 \\ -11.6489 \\ 6.3162 \\ 16.0707 \\ -8.4905 \\ -8.9921 \end{pmatrix}, \quad \hat{\mathbf{S}} = \begin{pmatrix} -14.284 \\ -17.0855 \\ 1.1383 \\ -14.7551 \\ 22.2798 \\ 10.7196 \\ -5.605 \\ -21.6508 \\ 9.0629 \\ 30.1799 \end{pmatrix},$$

which we use in

$$\mathbf{B}_S := \mathbf{B}_S(S^1_{t_1}, \dots, S^{10}_{t_1}) := \begin{pmatrix} b_1(S^1_{t_1})\hat{S}^1 & \cdots & b_m(S^1_{t_1})\hat{S}^1 \\ \vdots & \ddots & \vdots \\ b_1(S^{10}_{t_1})\hat{S}^{10} & \cdots & b_m(S^{10}_{t_1})\hat{S}^{10} \end{pmatrix}.$$

Then, the numerical values are

$$\mathbf{B_S} = (\hat{S}^j \quad S^j_{t_1} \cdot \hat{S}^j \quad (S^j_{t_1})^2 \cdot \hat{S}^j)\Big|_{j=1,\dots,10},$$

i.e.

$$\mathbf{B_S} = \begin{pmatrix} \text{-14.284} & \text{-1166.0613} & \text{-95190.2909} \\ \text{-17.0855} & \text{-2062.243} & \text{-248914.949} \\ 1.1383 & 101.4493 & 9041.6545 \\ \text{-14.7551} & \text{-1747.9124} & \text{-207059.949} \\ 22.2798 & 2330.0611 & 243681.841 \\ 10.7196 & 869.1857 & 70476.7514 \\ \text{-5.605} & \text{-330.5282} & \text{-19491.3279} \\ \text{-21.6508} & \text{-2201.8519} & \text{-223925.199} \\ 9.0629 & 596.7638 & 39295.1894 \\ 30.1799 & 3611.1368 & 432086.279 \end{pmatrix}.$$

Now, the estimate for the hedge position

$$\phi_{t_1} = -\mathbf{B} \cdot ((\mathbf{B_S})^{\dagger} \cdot \hat{\mathbf{\Pi}})$$

evaluates to

$$\phi_{t_1} = \begin{pmatrix} -0.9347 \\ -0.3008 \\ -0.781 \\ -0.3259 \\ -0.5121 \\ -0.9466 \\ -1.4926 \\ -0.5573 \\ -1.3086 \\ -0.3124 \end{pmatrix}.$$

This result is used to compute the portfolio value $\mathbf{\Pi}_{t_1}$. Following Equation (5.5), we obtain

$$\Pi^j_{t_1} = e^{-0.05 \cdot 0.5} \cdot (\Pi^j_{t_2} - (\phi^j_{t_1}(S^j_{t_2} - e^{0.05 \cdot 0.5} S^j_{t_1}))),$$

which is in this case

$$\mathbf{\Pi}_{t_1} = \begin{pmatrix} 17.2941 \\ 2.3366 \\ 9.8246 \\ 1.1209 \\ 5.3918 \\ 16.2901 \\ 29.1472 \\ 9.6868 \\ 31.0981 \\ 3.2265 \end{pmatrix}.$$

This completes the steps required at time t_1. We proceed with time t_0, where we have to compute the next variance optimal hedge. At time t_0, the computation of ϕ_{t_0} is easier than at time t_1 since all state variables are equal: $S^j_{t_0} = 100 \; \forall j \in \{1, \ldots, 10\}$. Consequently, we can compute ϕ_{t_0} without conditioning at S_{t_0}:

$$\phi_{t_0} = \frac{\text{cov}(\Pi_{t_1}, S_{t_1})}{\text{var}(S_{t_1})},$$

which is computed using the sample covariance and the sample variance of $S^j_{t_1}$ and $\Pi^j_{t_1}$, $j \in \{1, \ldots, 10\}$. The result is

$$\phi_{t_0} = \begin{pmatrix} -0.4633 \\ -0.4633 \\ -0.4633 \\ -0.4633 \\ -0.4633 \\ -0.4633 \\ -0.4633 \\ -0.4633 \\ -0.4633 \\ -0.4633 \end{pmatrix}.$$

Using this in the Equation for Π_{t_0}:

$$\Pi^j_{t_0} = e^{-0.05\cdot 0.5}(\Pi^j_{t_1} - (\phi^j_{t_0}(S^j_{t_1} - e^{0.05\cdot 0.5}S^j_{t_0}))),$$

we obtain

$$\mathbf{\Pi}_{t_0} = \begin{pmatrix} 7.4243 \\ 10.4891 \\ 3.5241 \\ 8.2913 \\ 6.1851 \\ 6.1964 \\ 8.7438 \\ 9.0712 \\ 13.7539 \\ 10.8838 \end{pmatrix},$$

which means

$$V_{t_0} = \frac{1}{10}\sum_{i=1}^{10}\Pi^j_{t_0} = 8.45629.$$

This can be compared with the traditional Monte Carlo estimate, which is

$$\begin{aligned} V_{t_0} &\approx e^{-rt_T}\frac{1}{10}\sum_{i=1}^{10}V^j_{t_2} \\ &\approx e^{-0.05\cdot 1}\cdot 21.4461 \\ &\approx 20.4002, \end{aligned}$$

while the true Black-Scholes price is 13.1459. It is easy to observe that the sample standard deviation std$(\mathbf{V}_{t_2}) = 17.58$ is a lot larger than the sample standard deviation std$(\mathbf{\Pi}_{t_0}) = 2.8720$, which is a clear indication that the Simulation-Based Hedging can provide better estimates than traditional Monte Carlo methods.

5.4.2 Simulation-Based Hedging in a Black-Scholes Market (European Options)

The previous section presented the computations of Simulation-Based Hedging for a simple setting in detail. In order to provide an overview of this method, this section summarizes the important steps of the Simulation-Based Hedging framework for Black-Scholes markets:

1. Simulation of asset $\mathbf{S}_{t_i} = (S^1_{t_i}, S^2_{t_i}, \ldots, S^n_{t_i})$ in physical measure,
   e.g. $S^j_{t_{i+1}} = S^j_{t_i} \exp\left((\mu - \frac{1}{2}\sigma^2)(t_{i+1} - t_i) + \sigma\sqrt{(t_{i+1} - t_i)}\theta_{i,j}\right)$

2. Determine payoff $P(S^j)$ and

$$\begin{aligned} \phi^j_{t_T} &= 0 \\ \Pi^j_{t_T} &= P(S^j_{t_T}) \end{aligned}$$

   of paths S^j.

3. For all j and for $i = T - 1$ down to 0 repeat

   (a) Perform least-squares regressions to determine $\phi^j_{t_i}$ according to

$$\phi^j_{t_i} = \frac{\text{cov}[\Pi_{t_{i+1}}, S_{t_{i+1}} \mid S_{t_i}, t_i]}{\text{var}[S_{t_{i+1}} \mid S_{t_i}, t_i]} \approx \sum_k \alpha_k b_k(S^j_{t_i})$$

   (b) Update portfolio value $\Pi^j_{t_i}$

$$\Pi^j_{t_i} = e^{-r(t_{i+1} - t_i)}\left(\Pi^j_{t_{i+1}} - \phi_{t_i}\left(S^j_{t_{i+1}} - e^{r(t_{i+1} - t_i)} S^j_{t_i}\right)\right)$$

4. Finally, the option value is given by $V_{t_0} \approx \frac{1}{n}\sum_{j=1}^n \Pi^j_{t_0}$ as an in-sample value.[17].

[17] V_{t_0} is computed as an in-sample value, only. This is because an out-of-sample value similar to the out-of-sample value of Equation (1.36) has no benefit for the Simulation-Based Hedging: The convergence of the out-of-sample value to the theoretical price is slower than for the in-sample case. Furthermore, there is no useful average bound as in the case of Least-Squares Monte Carlo, where the out-of-sample price is on average a lower bound on the true value.

5.4.3 Hedged Monte Carlo (Potters et. al. [96]) in a Black-Scholes Market (European Options)

For comparison, we present the Hedged Monte Carlo method of Potters et. al. [96].[18] Instead of a Forward Global Hedge as in Simulation-Based Hedging, this method minimizes the variance using some kind of Local Hedging.

1. Simulation of asset $\mathbf{S}_{t_i} = (S^1_{t_i}, S^2_{t_i}, \ldots, S^n_{t_i})$ in physical measure,
   e.g. $S^j_{t_{i+1}} = S^j_{t_i} \exp\left((\mu - \frac{1}{2}\sigma^2)(t_{i+1} - t_i) + \sigma\sqrt{(t_{i+1} - t_i)}\theta_{i,j}\right)$

2. Determine payoff $P(S^j)$ and

$$\begin{aligned} \phi^j_{t_T} &= 0 \\ V^j_{t_T} &= P(S^j_{t_T}) \end{aligned}$$

   of paths S^j.

3. For all j and for $i = T - 1$ down to 0 repeat

   (a) Perform least-squares regressions to determine $\phi^j_{t_i}$ according to

$$\phi^j_{t_i} = \frac{\text{cov}[V_{t_{i+1}}, S_{t_{i+1}} \mid S_{t_i}, t_i]}{\text{var}[S_{t_{i+1}} \mid S_{t_i}, t_i]} \approx \sum_k \alpha_k b_k(S^j_{t_i})$$

   (b) Update option value $V^j_{t_i}$

$$V^j_{t_i} = e^{-r(t_{i+1} - t_i)} \mathbb{E}\left[V^j_{t_{i+1}} - \phi^j_{t_i}\left(S^j_{i+1} - e^{r(t_{i+1} - t_i)} S^j_i\right) \mid S_{t_i}, t_i\right]$$

4. Finally, the option value is given by $V_{t_0} = \frac{1}{n}\sum_{j=1}^n V^j_{t_0}$ as an in-sample value.

[18] Note that there is no rigorous derivation of the setting or the resulting solution in Potters et. al. [96].

5.4.4 Simulation-Based Hedging in a Black-Scholes Market (American Put Option)

In this section, we summarize the implementation for Simulation-Based Hedging of American put options in a Black-Scholes markets. That means, we extend the algorithm presented in Section 5.4.2 by early exercise opportunities where the option holder exercises in a way which is the worst possible one for the issuer. We already discussed this early exercise behavior in Section 5.3.5. Note that additional to the regressions for estimating the optimal hedge position $\phi^j_{t_i}$, regressions are required for estimating the conditional expected continuation value. The regression are performed on the two possibly different sets of basis functions b_j, $j = 1, \ldots, m_1$ resp. $\hat{b}_j$, $j = 1, \ldots, m_2$.

1. Simulation of asset $\mathbf{S}_{t_i} = (S^1_{t_i}, S^2_{t_i}, \ldots, S^n_{t_i})$ in physical measure,
   e.g. $S^j_{t_{i+1}} = S^j_{t_i} \exp\left((\mu - \frac{1}{2}\sigma^2)(t_{i+1} - t_i) + \sigma\sqrt{(t_{i+1} - t_i)}\theta_{i,j}\right)$

2. Determine payoff $P(S^j)$ and

$$\begin{aligned} \phi^j_{t_T} &= 0 \\ \Pi^j_{t_T} &= P(S^j_{t_T}) \end{aligned}$$

   of paths S^j.

3. For all j and for $i = T - 1$ down to 0 repeat

   (a) Perform least-squares regressions to determine $\phi^j_{t_i}$ according to

$$\phi^j_{t_i} = \frac{\text{cov}[\Pi_{t_{i+1}}, S_{t_{i+1}} \mid S_{t_i}, t_i]}{\text{var}[S_{t_{i+1}} \mid S_{t_i}, t_i]} \approx \sum_{k=1}^{m_1} a_k b_k(S^j_{t_i})$$

   (b) Perform least-squares regression to determine

$$\mathbb{E}[\Pi_{t_{i+1}} | S_{t_i}, t_i] \approx \sum_{k=1}^{m_2} \tilde{a}_k \tilde{b}_k(S^j_{t_i})$$

   (c) Update portfolio value $\Pi^j_{t_i}$

$$\tilde{\Pi}^j_{t_i} = e^{-r(t_{i+1} - t_i)} \left(\Pi^j_{t_{i+1}} - \phi_{t_i}\left(S^j_{i+1} - e^{r(t_{i+1} - t_i)} S^j_i\right)\right)$$

$$\Pi^j_{t_i} = \begin{cases} (K - S^j_{t_i}) & \text{if } e^{-r(t_{i+1} - t_i)} \cdot \mathbb{E}[\Pi^j_{t_{i+1}} | S^j_{t_i}, t_i] < K - S^j_{t_i} \\ \tilde{\Pi}^j_{t_i} & \text{else} \end{cases}$$

   .

4. Finally, the option value is given by $V^j_{t_0} \approx \frac{1}{n} \sum_{j=1}^n \Pi^j_{t_0}$.

5.4.5 Remarks on the Computational Efficiency

This new method based on the computation of optimal hedges and the computation of their cost is not only more realistic than pricing in the Black-Scholes framework. Using the assumptions of the Black-Scholes prices, the new framework is much more efficient than regular Monte Carlo methods: Especially the comparison with Least-Squares Monte Carlo shows that the new method requires dramatically less computations. We will show that by estimating the order of convergence for both methods.

First, we take a look at Least-Squares Monte Carlo. This convergence analysis is similar to the analysis of plain Monte Carlo in Section 1.4.2. We use a few assumptions which allow a brief exposition, because a detailed analysis goes beyond the scope of this work.[19] However, the result is also valid for much more general assumptions which is confirmed by the numerical examples presented later in this chapter.

We assume that the regression leads to perfect estimates of the conditional expectation function $P^e(S_{t_i}, t_i) = \mathbb{E}_Q[V(S, t_{i+1})|S_{t_i}, t_i]$ in Equation (1.34). We also assume that a constant proportion m of the n samples is exercised, the rest reaches maturity time. And we assume that all paths are exercised for $P(S^*)$ which is constant for all t at asset price S^*, additionally, that the interest rate r is zero. Consequently, the expected variance of a single Monte Carlo samples is given by

$$\begin{aligned}\sigma^2_{\mathrm{LS}} &= \operatorname{var}\left[\frac{n-m}{n}V(S,t_T) + \frac{m}{n}P(S^*)\right] \\ &= \left(\frac{n-m}{n}\right)^2 \operatorname{var}[V(S,t_T)].\end{aligned}$$

That means a Monte Carlo estimate, which is the mean value of the samples has a variance of $\frac{\sigma^2_{\mathrm{LS}}}{n}$. Consequently, the standard deviation is in Landau-O notation:

$$\mathrm{O}\left(\frac{1}{\sqrt{n}}\right)$$

(cp. Equation (1.24), Section 1.4.2). This result is independent of the number of time steps made in the Least-Squares Monte Carlo.

The situation is different if we look at the Simulation-Based Hedging. Again, we assume that the regressions lead to perfect estimates of the conditional expectation function. That means, the Simulation-Based Hedging samples deviate only by the hedging error due to finite time-stepping. Furthermore, we assume that the underlying asset follows the Black-Scholes assumptions (i.e. $\mu = r = 0$). In order to obtain a good estimate of the hedging error H_{t_i} between time t_i and t_{i+1}, we look at the total hedging error until timestep t_i, which is

$$\Pi_{t_i} - \mathbb{E}[\Pi_{t_i}|S_{t_i}, t_i].$$

[19] See e.g. Stentoft [108] for a more detailed convergence analysis.

If we now define the value function $V(S_{t_i}, t_i) := \mathbb{E}[\Pi_{t_i}|S_{t_i}, t_i]$, we get the hedging error H_{t_i} as

$$\begin{aligned} H_{t_i} &:= (\Pi_{t_{i+1}} - V(S_{t_{i+1}}, t_{i+1})) - (\Pi_{t_i} - V(S_{t_i}, t_i)) \\ &= \left(\Pi_{t_{i+1}} - \Pi_{t_i}\right) - \left(V(S_{t_{i+1}}, t_{i+1}) - V(S_{t_i}, t_i)\right). \end{aligned} \tag{5.12}$$

From Equation (5.5) we know that with $r = 0$

$$\Pi_{t_{i+1}} - \Pi_{t_i} = \phi_{t_i}\left(S_{t_{i+1}} - S_{t_i}\right)$$

holds. That means together with (5.12)

$$H_{t_i} = \left(\phi_{t_i}\left(S_{t_{i+1}} - S_{t_i}\right)\right) - \left(V(S_{t_{i+1}}, t_{i+1}) - V(S_{t_i}, t_i)\right).$$

With the Taylor expansion

$$\begin{aligned} V(S_{t_{i+1}}, t_{i+1}) - V(S_{t_i}, t_i) &= \frac{\partial V(S_i, t_i)}{\partial S}\left(S_{t_{i+1}} - S_{t_i}\right) + \frac{\partial V(S_i, t_i)}{\partial t}(t_{i+1} - t_i) \\ &\quad + \frac{1}{2}\frac{\partial^2 V(S_i, t_i)}{\partial S^2}\left(S_{t_{i+1}} - S_{t_i}\right)^2 + \mathrm{O}\left((t_{i+1} - t_i)^2 + \left(S_{t_{i+1}} - S_{t_i}\right)^3\right) \end{aligned}$$

and the approximation[20]

$$\phi_{t_i} = \frac{\mathrm{cov}[\Pi_{t_{i+1}}, S_{t_{i+1}}|S_{t_i}, t_i]}{\mathrm{var}[S_{t_{i+1}}|S_{t_i}, t_i]} \approx \frac{\partial V(S_i, t_i)}{\partial S}$$

we obtain

$$-H_{t_i} \approx \frac{1}{2}\frac{\partial^2 V(S_i, t_i)}{\partial S^2}\left(S_{t_{i+1}} - S_{t_i}\right)^2 + \frac{\partial V(S_i, t_i)}{\partial t}(t_{i+1} - t_i) + \text{rest}.$$

where rest $\in \mathrm{O}((t_{i+1} - t_i)^2 + \left(S_{t_{i+1}} - S_{t_i}\right)^3)$. Now, the variance hedging error of the portfolio is given by

$$\begin{aligned} \mathrm{var}[H_{t_i}|S_{t_i}, t_i] &\approx \mathrm{var}\left[\frac{1}{2}\frac{\partial^2 V(S_i, t_i)}{\partial S^2}\left(S_{t_{i+1}} - S_{t_i}\right)^2 + \frac{\partial V(S_i, t_i)}{\partial t}(t_{i+1} - t_i) + \text{rest} \mid S_{t_i}, t_i\right] \\ &= \mathrm{var}\left[\frac{1}{2}\frac{\partial^2 V(S_i, t_i)}{\partial S^2}\left(S_{t_{i+1}} - S_{t_i}\right)^2 + \text{rest} \mid S_{t_i}, t_i\right], \end{aligned}$$

since $\frac{\partial V(S_i, t_i)}{\partial t}(t_{i+1} - t_i)$ is a constant at time t_i and does not contribute to the variance of the one-step error. Assuming that $\left\|\frac{1}{2}\frac{\partial^2 V(S_i, t_i)}{\partial S^2}\right\|_\infty = c < \infty$ and that $\|\text{rest}\|_\infty \approx 0$ and using the Black-Scholes assumptions for the asset price process with $r = 0$, we obtain

$$\begin{aligned} \mathrm{var}[H_{t_i} \mid S_{t_i}, t_i] &\leq \mathrm{var}\left[c \cdot (S_{t_{i+1}} - S_{t_i})^2 \mid S_{t_i}, t_i\right] \\ &\approx \mathrm{var}\left[c \cdot \left(S_{t_i}\left(e^{(-0.5\sigma^2)(t_{i+1} - t_i) + \sigma\theta\sqrt{t_{i+1} - t_i}} - 1\right)\right)^2 \mid S_{t_i}, t_i\right] \\ &\approx \mathrm{var}\left[c \cdot (S_{t_i})^2\left((-0.5\sigma^2)(t_{i+1} - t_i) + \sigma\theta\sqrt{t_{i+1} - t_i}\right)^2 \mid S_{t_i}, t_i\right] \\ &= (S_{t_i})^4 c^2 \mathrm{var}\left[(-0.5\sigma^2)^2(t_{i+1} - t_i)^2 + \sigma^2\theta^2(t_{i+1} - t_i) \mid S_{t_i}, t_i\right] \\ &\in \mathrm{O}\left((t_{i+1} - t_i)^2\right) \end{aligned}$$

[20] This can be seen from the fact that with zero interest-rate in a one-period Black-Scholes market, both expressions minimize the variance of the hedge portfolio. Further discussions can be found in [33], Section 3.2.

with $\theta \sim N(0,1)$ holds, which is the variance of a single time step.

Since there are $T := t_T/(t_{i+1} - t_i)$ time steps in each paths (assuming equal time steps), the total variance of the set of Simulation-Based Hedging paths is given by

$$\operatorname{var}\left[\sum_{i=0}^{T-1} H_{t_i}\right].$$

Assuming furthermore, that the hedging errors are uncorrelated and $\operatorname{var}[H_{t_{i^*}}|S_{t_{i^*}}, t_{i^*}]$ denotes the largest hedging error, i.e. $\operatorname{var}[H_{t_{i^*}}|S_{t_{i^*}}, t_{i^*}] \geq \operatorname{var}[H_{t_i}|S_{t_i}, t_i]\ \forall i = 0, \ldots, T-1$, then

$$\begin{aligned}\operatorname{var}\left[\sum_{i=0}^{T-1} H_{t_i}\right] &\leq T\operatorname{var}\left[H_{t_{i^*}}\right] \\ &\in \mathrm{O}\left(\frac{t_T(t_{t_{i^*}+1} - t_{t_{i^*}})^2}{t_{t_{i^*}+1} - t_{t_{i^*}}}\right) \\ &\in \mathrm{O}\left(t_{t_{i^*}+1} - t_{t_{i^*}}\right).\end{aligned}$$

That means, the variance of the mean of n sample paths of hedged portfolios is in $\mathrm{O}(\frac{1}{nT})$ and the standard deviation is in

$$\mathrm{O}\left(\frac{1}{\sqrt{nT}}\right).$$

These results show that the Least-Squares Monte Carlo method requires about T-times as many asset paths simulations as the Simulation-Based Hedging method for a computation with the same accuracy. Using equally many asset paths, both methods require the same order of numerical operations: The regressions of the Simulation-Based Hedging require about triple[21] the computations of the Least-Square Monte Carlo, which is irrelevant in Landau-O notation. In reality, where many time steps are used, Simulation-Based Hedging is an order of magnitude quicker than Least-Squares Monte Carlo for pricing American put options.

5.4.6 Numerical Experiments

After we theoretically confirmed the high speed of the Simulation Based Hedging and went through a simple example, we now want to obtain more insight into the practical properties of this new method. A few numerical experiments will demonstrate the efficiency and other effects compared with regular Least-Squares Monte Carlo.

Black-Scholes: American Put Options

First, we focus on an American option as in Table 5.5. Its Black-Scholes price is given by $V = 13.66761$ (all digit correct, computed by a PDE method). In order to compare the continuous time PDE value with the discrete American (thus Bermudan) option values of the Monte Carlo methods, we first compare the computed option values for different numbers of time steps. Note

[21] Simulation-Based Hedging requires three regressions at each time-step t_i for estimating $\hat{\Pi}_{t_i} = \mathbb{E}[\Pi_{t_{i+1}}|S_{t_i}, t_i]$, $\hat{S}_{t_i} = \mathbb{E}[S_{t_{i+1}}|S_{t_i}, t_i]$ and ϕ_{t_i}, while Least-Squares Monte Carlo requires a regression for estimating $\mathbb{E}[V_{t_{i+1}}|S_{t_i}, t_i]$, only.

Table 5.5 *Base case data of an American put option.*

General features	
strike price K	100
risk-free rate r	5% p.a.
volatility σ	40% p.a.
drift rate μ	5% p.a.
dividends D_i	none
maturity time t_T	1 year
terminal value $P(S, t_T)$	$\max(K - S_{t_T}, 0)$
exercise price at $t_j < t_T$	$\max(K - S_{t_j}, 0)$

that we are computing Black-Scholes prices, i.e. we are not assigning a margin capital cost to the remaining hedging error.

A summary of these result is presented in Table 5.6, which contains the values for the Least-Squares Monte Carlo, the Simulation-Based Hedging and the Hedged Monte Carlo[22]. Both, the Least-Squares Monte Carlo and the Simulation-Based Hedging seem to converge to the reference value: 128 time-steps are required for cent accurate values (± 0.01). On the other hand, our implementation of the Hedged Monte Carlo does not converge to the reference value, which is mainly due to the nature of the Hedged Monte Carlo. The main difference of Hedged Monte Carlo to Simulation-Based Hedging is that in Hedged Monte Carlo, the regression is performed on a function that is interpreted as option value V, which is unique in each state. Instead, in Simulation-Based Hedging, we perform regressions on the amount Π required for a self-financing hedge, which is stochastic in each state. Consequently, the Hedged Monte Carlo has a large interpolation error in the expected value V at each time-step, which increases with the number of time-steps. This is not the case for Simulation-Based Hedging. In Simulation-Based Hedging, the regression (or interpolation) error influences the hedging strategy ϕ only, the expected value of Π is not affected.

Another experiment (Table 5.7) shows that if the drift rate is changed, the option value with few time-steps can be much different, but with an increasing number of time steps, the value still converges to the Black-Scholes price. This is expected since the drift of the underlying is irrelevant for the Black-Scholes price. But, one can see that the option price depends on the underlying market such that even daily hedging (256 time steps) introduces a difference of 0.02 (drift $\mu = 30\%$) compared with the Black-Scholes price. This is not a model or numerical error, a real-world trader would obtain the same results for this market with a daily hedge rebalancing.[23]

A different aspect is presented in Figure 5.2: One can see that the Simulation-Based Hedging is indeed a lot more accurate due to a higher convergence. While the standard deviation of the sample of Least-Squares Monte Carlo valuations does not change with increasing time steps, it reduces quickly with the Simulation-Based Hedging valuations. Again, this is expected as we

[22] As presented in [96]

[23] For details of different hedging strategies we refer the reader to [117], p. 326.

Table 5.6 *Valuation results for an American option as in Table 5.5, comparing the values of Least-Squares Monte Carlo, Potters' method (Hedged Monte Carlo) and the Simulation-Based Hedging. All values are computed using the same set of 1,000,000 asset paths and 1 to 256 time-steps of the sample. The PDE reference value is 13.66761.*

# time steps	$V^{\text{Simulation-Based Hedging}}$	$V^{\text{Least-Squares Monte Carlo}}$	$V^{\text{Hedged Monte Carlo}}$
256	13.6605	13.6584	13.8073
128	13.6593	13.6605	13.8016
64	13.6533	13.6569	13.7787
32	13.6431	13.6476	13.7384
16	13.6268	13.6317	13.7071
8	13.5906	13.5782	13.6283
4	13.5126	13.4927	13.5257
2	13.3926	13.3574	13.2526
1	13.1474	13.1288	12.4762

Table 5.7 *Valuation results for an American option as in Table 5.5, comparing the values of Simulation-Based Hedging with different drift rates μ. All values are computed using the same set of 1,000,000 asset paths. The PDE reference value is 13.66761.*

Simulation-Based Hedging with different drift rates

# time steps	$V : \mu = 5\%$	$V : \mu = 30\%$	$V : \mu = -15\%$
256	13.6605	13.6395	13.6588
128	13.6593	13.6205	13.6540
64	13.6533	13.5764	13.6518
32	13.6431	13.4908	13.6352
16	13.6268	13.3285	13.6062
8	13.5906	12.9984	13.5357
4	13.5126	12.3494	13.4267
2	13.3926	11.1942	13.1903
1	13.1474	9.3091	12.7706

theoretically saw the higher convergence of the Simulation-Based Hedging in Section 5.4.5.

Black-Scholes + Transaction Cost

After the verification in the previous section that the proposed Simulation-Based Hedging method yields to the correct values, we will extend the portfolio strategy by introducing a proportional transaction cost factor κ :

$$B^j_{t_i} = e^{-r(t_{i+1}-t_i)} \left(B^j_{t_{i+1}} + (\phi^j_{t_{i+1}} - \phi^j_{t_i}) S^j_{t_{i+1}} + \kappa \left| (\phi^j_{t_{i+1}} - \phi^j_{t_i}) S^j_{t_{i+1}} \right| \right).$$

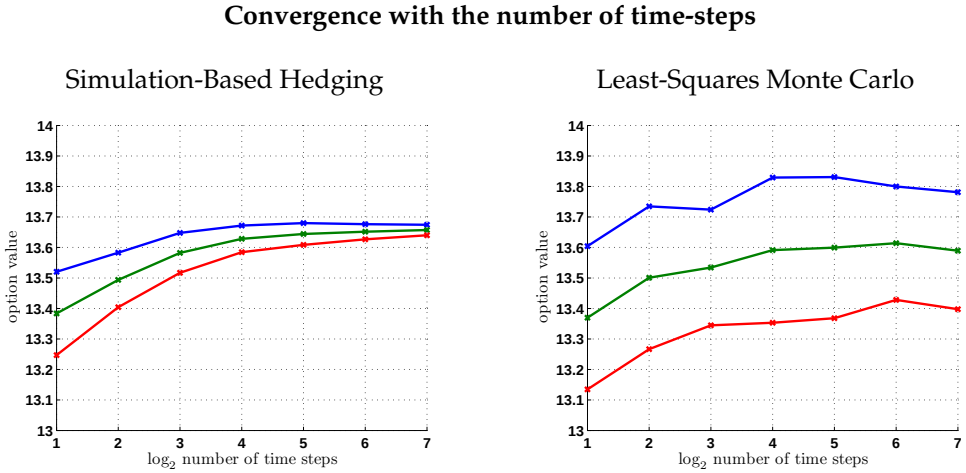

Figure 5.2: *Valuation results for an American option as in Table 5.5, comparing the values of Least-Squares Monte Carlo and the Simulation-Based Hedging. All values are mean values of 100 valuations, computed using a set of* 10^4 *asset paths and 1 to 128 time steps. The presented interval is the mean value of the valuations* $\pm$ *the standard deviation of the valuations. The PDE reference value is 13.66761.*

The resulting optimization problem is

$$\begin{aligned}
\{\phi_{t_i}\} &= \arg\min_{\phi_{t_i}} \left(\mathrm{var}\left[\Pi_{t_i} \mid S_{t_i}, t_i\right]\right) \\
&= \arg\min_{\phi_{t_i}} \left(\mathrm{var}\left[e^{-r(t_{i+1}-t_i)}\left(\Pi_{t_{i+1}} - \phi_{t_i}(S_{t_{i+1}} - e^{r(t_{i+1}-t_i)}S_{t_i}) + \kappa\left|(\phi_{t_{i+1}} - \phi_{t_i})S^j_{t_{i+1}}\right|\right) \mid S_{t_i}, t_i\right]\right) \\
&= \arg\min_{\phi_{t_i}} \left(\mathrm{var}\left[\Pi_{t_{i+1}} - \left(\phi_{t_i} - \kappa\left|\phi_{t_{i+1}} - \phi_{t_i}\right|\right) S_{t_{i+1}} \mid S_{t_i}, t_i\right]\right).
\end{aligned}$$

The corresponding solution is similar to the one in the case without transaction costs (Equation (5.9)), i.e. the solution is,

$$\phi_{t_i} - \kappa\left|\phi_{t_{i+1}} - \phi_{t_i}\right| = \frac{\mathrm{cov}[\Pi_{t_{i+1}}, S_{t_{i+1}} \mid S_{t_i}, t_i]}{\mathrm{var}[S_{t_{i+1}} \mid S_{t_i}, t_i]},$$

which is for $\phi_{t_{i+1}} > \phi_{t_i}$:

$$\phi_{t_i} = \frac{1}{1+\kappa}\left(\frac{\mathrm{cov}[\Pi_{t_{i+1}}, S_{t_{i+1}} \mid S_{t_i}, t_i]}{\mathrm{var}[S_{t_{i+1}} \mid S_{t_i}, t_i]} + \phi_{t_{i+1}}\kappa\right),$$

and for $\phi_{t_{i+1}} < \phi_{t_i}$:

$$\phi_{t_i} = \frac{1}{1-\kappa}\left(\frac{\mathrm{cov}[\Pi_{t_{i+1}}, S_{t_{i+1}} \mid S_{t_i}, t_i]}{\mathrm{var}[S_{t_{i+1}} \mid S_{t_i}, t_i]} - \phi_{t_{i+1}}\kappa\right).$$

Interestingly, for

$$\frac{1}{1-\kappa}\left(\frac{\mathrm{cov}[\Pi_{t_{i+1}}, S_{t_{i+1}} \mid S_{t_i}, t_i]}{\mathrm{var}[S_{t_{i+1}} \mid S_{t_i}, t_i]} - \phi_{t_{i+1}}\kappa\right) < \phi_{t_i} < \frac{1}{1+\kappa}\left(\frac{\mathrm{cov}[\Pi_{t_{i+1}}, S_{t_{i+1}} \mid S_{t_i}, t_i]}{\mathrm{var}[S_{t_{i+1}} \mid S_{t_i}, t_i]} + \phi_{t_{i+1}}\kappa\right),$$

the solution does not exist. We propose that in this situation $\phi_{t_i} = \phi_{t_{i+1}}$ holds, which means that one does not change the hedge.

Table 5.8 *Base case data of a European put option. The Black-Scholes option price is* $V_{BS} = 13.1459$

General features	
strike price K	100
risk-free rate r	5% p.a.
volatility σ	40% p.a.
drift rate μ	5% p.a.
dividends D_i	none
maturity time T	1 year
terminal value $P(S,T)$	$\max(K - S_{t_T}, 0)$
exercise price at $t < T$	0

Table 5.9 *Valuation results for a European option as in Table 5.8 with proportional transaction costs (κ) in a risk-neutral setting, i.e. $\mu = r$, 8 time steps are used for rehedging. The table provides values for comparing the not admissible strategy ϕ with the approximate strategy $\hat{\phi}$. All values are computed such that the 95% confidence interval is at most ± 0.005.*

	ϕ		$\hat{\phi}$	
κ	V, ϕ	$\text{std}(\Pi_{t_0})$	$V, \hat{\phi}$	$\text{std}(\Pi_{t_0})$
0.001	13.238	8.0476	13.267	8.052
0.020	14.847	8.2429	15.581	8.433

Now, it is important to note that the hedging strategy ϕ_{t_i} is not measurable at time t_i as in the case without transaction costs. The strategy at time t_i depends on the strategy at time t_{i+1}, which is clearly not available at time t_i. Consequently, we approximate $\phi_{t_{i+1}}$ with a simple estimate, i.e. $\phi_{t_{i+1}} \approx \phi_{t_i}$. This leads to an approximate hedging strategy $\hat{\phi}_{t_i}$,

$$\hat{\phi}_{t_i} = \frac{\text{cov}[\Pi_{t_{i+1}}, S_{t_{i+1}} \mid S_{t_i}, t_i]}{\text{var}[S_{t_{i+1}} \mid S_{t_i}, t_i]}$$

which is the same as in the case without transaction costs. Table 5.9 presents values of the example with data in Table 5.8 for the not admissible strategy ϕ, as well as the admissible approximate solution $\hat{\phi}$. In the case with low transaction costs $\kappa = 0.001$, the option value and the hedging are not much different for both solutions, which suggests that the approximate solution is useful in this case. In the case with large transaction costs, the hedging error is still not much different (about 2% difference). But, the option value is affected by almost 5%, which is significant. However, it is important to emphasize that the optimal strategy ϕ_{t_i} is not admissible and thus super optimal because it is measurable at time t_{i+1}.

Leland [79] already solved the time based hedging problem for the case of a Black-Scholes delta hedge in a risk-neutral setting with constant revision intervals $\Delta t = (t_{i+1} - t_i)$. His result is that one effectively has to price the option with an adjusted volatility

$$\sigma^2_{\text{Leland}} = \sigma^2 \left(1 + \sqrt{\frac{8}{\pi}} \frac{\kappa}{\sigma\sqrt{\Delta t}}\right),$$

Table 5.10 *Valuation results for a European option as in Table 5.8 with proportional transaction costs ($\kappa = 0.001$) in a risk-neutral setting, i.e. $\mu = r$. All values are computed such that the 95% confidence interval is at most ± 0.005.*

n_t	V, $\kappa = 0.001$	V_{Leland}
512	13.864	13.851
256	13.669	13.659
128	13.528	13.521
64	13.426	13.423
32	13.351	13.353
16	13.300	13.303
8	13.267	13.268
4	13.249	13.243
2	13.228	13.226
1	13.219	13.213

such that the option price is given by

$$V_{\text{Leland}} = V_{\text{BS}}(\sigma_{\text{Leland}}) + \|\kappa\theta_{t_0} S_{t_0}\|, \tag{5.13}$$

where $V_{\text{BS}}(\sigma)$ is a function which returns the Black-Scholes price for the option under consideration and θ_{t_0} is the Black-Scholes delta at time t_0. The additive term $\|\kappa\theta_{t_0} S_{t_0}\|$ results from the fact that we assume that the transaction cost for the $\theta_{t_0} \cdot S_{t_0}$ we buy at the initial time have to be paid. This is not the case in the original model of Leland. Furthermore, we assume physical delivery at maturity, which corresponds to the Leland model.

Table 5.10 presents the results for a European option with data in Table 5.8. The Black-Scholes price of the option is 13.1459, the column V presents the Simulation-Based Hedging prices for a risk-neutral drift $\mu = r$ and V_{Leland} presents the price with Leland's adjusted volatility according to Equation (5.13). In most cases the difference between the Leland price and the Simulation-Based Hedging price is small and within the confidence interval of the Monte Carlo estimates. But, for decreasing time intervals, i.e. larger numbers of time steps T, the cost in Simulation-Based Hedging seem to increase a bit faster than in Leland's model. This is mainly due to numerical properties: the estimated representation of the hedging strategy leads to additional oscillations in the hedge portfolio, which result in higher transaction costs.

Simulation-Based Hedging with Transaction Cost and Margin Capital Cost

In the previous paragraphs, we saw how the transaction costs behave in the Simulation-Based Hedging setting. The values missing for a valuation of ask price $V_{t_0}^{ask}$ of an option are the margin capital costs C_R. These capital costs are easy to obtain in the simulation framework since the distribution of the required capital in the portfolio Π_{t_0} is already computed using simulated asset paths in the Monte Carlo algorithm. In this case, we are interested in the conditional value at risk (CVaR). Consequently, we can compute the required margin capital cost C_R by the corresponding

quantile of paths (using Equation (5.10)):

$$\begin{aligned} C_R &= e^{-r(t_T-t_0)}(e^{r_e(t_T-t_0)}-1)CVaR_{95\%}(\Pi_{t_0}), \\ CVaR_\alpha(\Pi_{t_0}) &:= \mathbb{E}[\Pi_{t_0}|\Pi_{t_0} \leq VaR_\alpha(\Pi_{t_0})], \\ VaR_\alpha(\Pi_{t_0}) &:= \sup(x \in \mathbb{R} \mid P(\Pi_{t_0}) \leq x) < \alpha). \end{aligned}$$

Assuming an expected return on equity capital r_e of 20% p.a., we obtain the capital costs presented in Table 5.11 for the hedge of a European option with data in Table 5.8 and proportional transaction costs $\kappa = 0.001$. Comparing the properties of the prices obtained with different time steps, a few facts are obvious: The transaction costs are the lowest with only a single time step and they are increasing with the number of time steps. The reverse is true for the required margin capital costs C_R, i.e. the capital costs are the lower, the more time steps are used for rehedging. This was expected. Now, we can find some optimal strategy, which minimizes the transaction costs + margin capital costs value: The lowest costs are at $T = 64$ time steps, which equals about weekly rehedging. This option value is 13.6638 and has about 0.51 costs associated compared with the Black-Scholes price, which is a significant addition. Note that this costs can be dramatically reduced using an expected drift rate $\mu > r$, non-equally distant hedging intervals or move-based rehedges.[24]

Table 5.11 *Valuation results for a European option as in Table 5.8 with proportional transaction costs ($\kappa = 0.001$) in a risk-neutral setting, i.e. $\mu = r$. All values are computed such that the 95% confidence interval is at most ± 0.005.*

T	V, $\kappa = 0.001$	trans. $V^1 - V_{BS}$	capital cost C_R	trans. + C_R	$V_{t_0}^{ask}$
256	13.6690	0.5231	**0.1245**	0.6476	13.7935
128	13.5280	0.3821	0.1735	0.5556	13.7015
64	13.4260	0.2801	0.2378	**0.5179**	**13.6638**
32	13.3510	0.2051	0.3385	0.5436	13.6895
16	13.3000	0.1541	0.4843	0.6384	13.7843
8	13.2670	0.1211	0.6823	0.8034	13.9493
4	13.2490	0.1031	0.9324	1.0355	14.1814
2	13.2280	0.0821	1.2940	1.3761	14.5220
1	13.2190	**0.0731**	1.7359	1.8090	14.9549

Simulation-Based Hedging with an Econometric Model for the Underlying

This section is not intended to promote a specific model: It is intended to promote the potential of Simulation-Based Hedging. Therefore, we will describe an econometric model which is somehow realistic, but which will not satisfy everybody due to the specific model restrictions. At the end of this section, we will briefly summarize the method's abilities and possible extensions.

[24] For details of different hedging strategies, we refer the reader to Wilmott [117] the the references therein.

As an econometric model, we use a GARCH(1,1) volatilty model as proposed by Bollerslev [18], with

$$\begin{aligned} r &= \text{const.} \\ R_{t_i} &= a + \epsilon_{t_{i-1}}, \quad \epsilon_{t_i} \sim N(0, \sigma_{t_i}^2) \\ \sigma_{t_i}^2 &= \alpha + \beta\sigma_{t_{i-1}}^2 + \gamma\epsilon_{t_{i-1}}^2 \\ S_{t_{i+1}} &= e^{R_{t_i}} S_{t_i}. \end{aligned}$$

Using an algorithm for parameter estimation as in MATLAB, we can create parameters from Data (German DAX, daily from 01.01.2003 to 06.11.2006):

$$\begin{aligned} r &= 3.9\% \\ R_{t_i} &= 9.28 \cdot 10^{-4} + \epsilon_{t_{i-1}}, \quad \epsilon_{t_i} \sim N(0, \sigma_{t_i}^2) \\ \sigma_{t_i}^2 &= 1.69 \cdot 10^{-6} + 0.91\sigma_{t_{i-1}}^2 + 0.079\epsilon_{t_{i-1}}^2 \\ S_{t_{i+1}} &= e^{R_{t_i}} S_{t_i}. \end{aligned}$$

In order to complicate the setting a little further, we are going to evaluate an option using two hedge instruments. The data in Table 5.12 contains the data of a one year barrier option V which we would like to sell. The hedge will consist of a two-year option V_c on the same underlying plus the underlying S itself. Consequently, we need a model for the volatility of the hedge. Assuming that V_c were liquidly traded, Figure 5.3 suggests that the implied volatility of V_c can be approximated by

$$\sigma_{\text{implied},t_i} = 0.29 \cdot 0.306 + 0.71 \cdot \sigma_{t_i} \tag{5.14}$$

with GARCH volatility σ_{t_i} at time t_i.

In this setting with two hedge instruments, we set up the portfolio similar to Equation (5.2) as

$$\Pi_{t_i} = B_{t_i} + \phi_{t_i} S_{t_i} + \psi_{t_i} V_{c,t_i}$$

with an option which is tradable at price V_{c,t_i}. The self-financing condition leads to

$$\begin{aligned} e^{r(t_{i+1}-t_i)} B_{t_i} + \phi_{t_i} S_{t_{i+1}} + \psi_{t_i} V_{c,t_{i+1}} &= B_{t_{i+1}} + \phi_{t_{i+1}} S_{t_{i+1}} + \psi_{t_{i+1}} V_{c,t_{i+1}} \\ \Leftrightarrow B_{t_i} &= e^{-r(t_{i+1}-t_i)}(B_{t_{i+1}} + (\phi_{t_{i+1}} - \phi_{t_i})S_{t_{i+1}} + (\psi_{t_{i+1}} - \psi_{t_i})V_{c,t_{i+1}}) \end{aligned}$$

for the bank account value B_{t_i}. From the objective to minimize the variance of Π_{t_i} at each time-step t_i, we obtain the optimal solution by

$$\begin{aligned} \{\phi_{t_i}, \psi_{t_i}\} = \arg\min_{\phi_{t_i}, \psi_{t_i}} \Big(\text{var} \Big[& e^{-r(t_{i+1}-t_i)} (\Pi_{t_{i+1}} - \phi_{t_i}(S_{t_{i+1}} - e^{r(t_{i+1}-t_i)} S_{t_i}) \\ & - \psi_{t_i}(V_{c,t_{i+1}} - e^{r(t_{i+1}-t_i)} V_{c,t_i})) \mid S_{t_i}, t_i \Big] \Big), \end{aligned}$$

Table 5.12 *The data for the hedge of an up-and-out barrier option with two hedge instruments in a market with GARCH volatility.*

Up-and-Out Barrier Option

Barrier B	7800
Strike K	6500
Payoff $P(S)$	$\frac{1}{100}\max(S_{t_T} - K)$
Maturity Time T	1.00 years
Kock-out observation	12:00h daily

Hedge Instruments: S **and** V_c

Underlying S	
current asset price S_{t_0}	6500
current volatility σ_{t_0}	18%
transaction cost κ_S	10 basis points (0.001)
Call option	
Strike K_c	6500
Payoff $P_c(S)$	$\max(S_{t_T} - K)$
Maturity Time T	2.00 years
transaction cost κ_V	200 basis points (0.02)

using a regression set of basis functions $\{b_1^\phi, b_1^\psi, b_2^\phi, b_2^\psi, \ldots\}$ and the optimal strategy

$$\phi(\mathbf{x}^i) \approx \sum_j \tilde{a}_j^\phi b_j^\phi(\mathbf{x}^i)$$

and

$$\psi(\mathbf{x}^i) \approx \sum_j \tilde{a}_j^\psi b_j^\psi(\mathbf{x}^i)$$

where the state variable is

$$\mathbf{x}^i := (\sigma_{t_i}^j, S_{t_i}^j)$$

thus two dimensional. We solve this regression using thin-plate splines as presented in Chapter 1. The numerical results are presented in Table 5.13. These results are only correct about ± 0.10 since the Monte Carlo evaluation of such a barrier option is challenging. The transaction costs (row *t.-cost*) is the difference between one evaluation with transaction costs and a second evaluation on the same set of Monte Carlo paths without transaction costs. One can observe that the transaction costs rise dramatically the more often the portfolio is rehedged, while the cost of capital C_R stays constant for 1 to 16 time steps and rises for more than 32. Now, different strategies can be pursued to do something optimal. Looking at the accumulated costs (transaction costs + margin requirements), the optimal rehedging frequency seems to be within 0 and 3 times during the one year

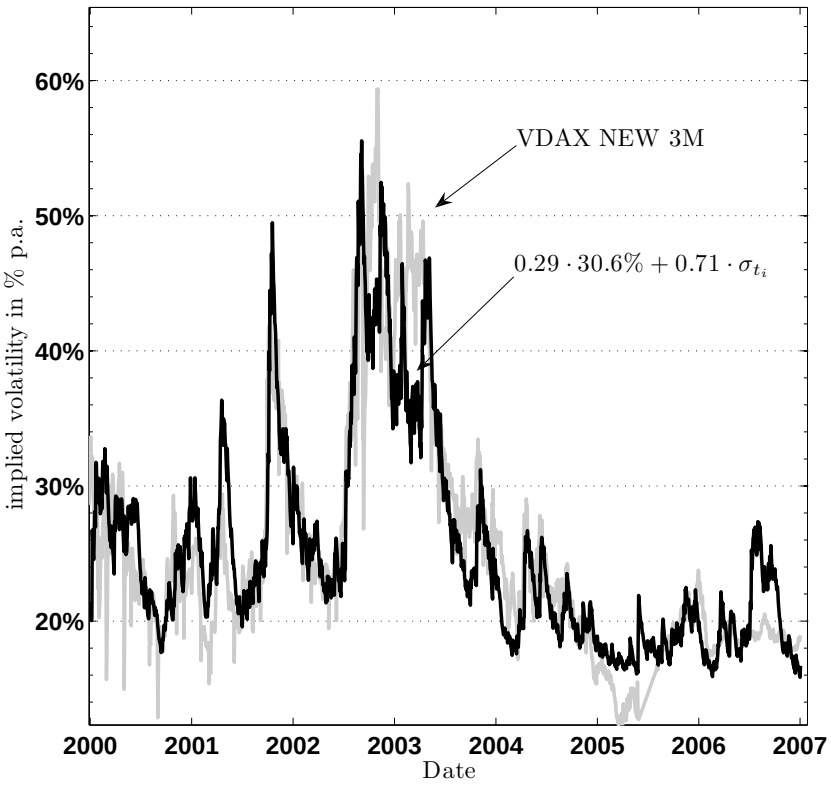

Figure 5.3: *Comparison of at the money implied volatilities (VDAX) and estimated daily volatilities σ_{t_i} for 2000-2007.*

until maturity (1 to 4 time steps). However, one could sell the option for less when one rehedges 8 or 16 times per year.

Table 5.13 *Costs and prices computed using Simulation-Based Hedging for the up-and-out barrier call with data in Table 5.12 in a market with GARCH volatility.*

T	V	t.-cost	C_R	t.-cost + C_R	ψ	ϕ	$V_{t_0}^{ask}$
256	0.77	9.04	1.71	10.75	1.55	-0.90	11.52
128	0.77	3.19	0.84	4.03	0.92	-0.69	4.80
64	0.83	1.28	0.59	1.87	0.75	-0.43	2.69
32	0.88	0.58	0.47	1.05	0.50	-0.28	1.93
16	1.00	0.18	0.44	0.61	0.13	-0.05	1.61
8	1.04	0.14	0.43	0.57	0.03	0.02	1.61
4	1.18	0.05	0.42	0.47	0.05	0.00	1.65
2	1.21	0.17	0.41	0.58	0.05	-0.01	1.79
1	1.48	-0.06	0.42	0.36	0.03	-0.01	1.84

5.5 Summary

The question, which this chapter tried to answer was: How should a pricing and hedging strategy look like, especially for the seller of an exotic OTC-contract? We found that the price of the option should cover all cost components of the bank. These costs consist of the cost of the hedge plus the cost of the risk involved. While the cost of the hedge is the expected capital required for the hedge including transaction costs, the cost of the risk is the required margin capital for the residual risk of the hedge. The costs of the risk are minimized using a dynamic hedging strategy, which minimizes the variance of the hedged portfolio.

In short, this chapter presented a new an versatile Monte Carlo method called Simulation-Based Hedging, capable of pricing derivatives based on optimal hedges. The method can be used in very general market settings including econometric market models such as GARCH. Theoretical considerations and numerical experiments confirm that the new method is capable of efficient pricing and hedging in incomplete markets, while being faster than regular Least-Squares Monte Carlo in complete markets.

Chapter 6

Conclusions

Summary of Results

This thesis provides three new ideas for the application of Least-Squares regressions to the pricing of financial derivatives. All of them contribute to low variance Monte Carlo pricing, each in a separate context.

The first idea is called Feature Extraction and parts of the first idea were previously presented by Grau [55]. But, the idea was extended and the theoretical correctness has been shown. The idea is to accelerate the pricing of path dependent options in complete markets by separating the pricing algorithm into two parts: One which estimates a conditional expected payoff functions, and another which uses this function in a numerical integration to determine the option value. As a result, the required computational effort for the pricing of a delayed barrier option could be reduced dramatically. The method is new and basically combines the ideas of Monte Carlo pricing as presented by Boyle [21] with quadrature pricing as presented by Andricopoulos et. al. [9].

The second idea is the utilization of sparse basis functions for the Least-Squares Monte Carlo method. The Least-Squares Monte Carlo method introduced by Carrière [32] as well as Longstaff and Schwartz [81] is the state of the art method for pricing exercisable options within a simulation framework. So far, the complexity of the options' payoff in Least-Squares Monte Carlo was very limited since the regression in the algorithm could only handle low dimensionality. The sparse basis functions which are similar to sparse grids allow regressions on relatively high-dimensional functions and thus allow the valuation of much more complex derivatives than before. The successful application of the Least-Squares Monte Carlo to a Moving Window Asian option demonstrates the ability of this powerful method. No other practical and convergent approach has been presented yet. In a second application, the complex rights of holders and issuers of convertible bonds are implemented for a numerical valuation of a convertible bond. Although a so called moving window soft call protection is common in convertible bond contracts, this thesis is the first to evaluate this kind of constraints correctly. This work is the first to present a combination

of Least-Squares Monte Carlo [32, 81] with the idea of sparse grids [107, 29].

The last application of regression methods leads to the most powerful method of this thesis: A method is presented, which can evaluate options much quicker than comparable methods as Least-Squares Monte Carlo [32, 81] or Hedged Monte-Carlo [96, 95] while dropping the complete market assumptions. For the pricing of American Put options, the new method is an order of magnitude faster than the state of the art Least-Squares Monte Carlo. This remarkable result is obtained by the direct computation of optimal hedging portfolios. Therefore, we call this new method Simulation-Based Hedging. The option price in an incomplete market is then given by the cost of the hedge plus the cost of the required margin capital for the remaining risk. Since the provided prices are based on a risk minimizing strategy similar to the variance minimization by Schweizer [104], which a trader can follow in real world, this approach can be a benchmark for all issuers in the market.

All together, this thesis has shown that Least-Squares regression is a useful tool in a wide area of derivatives pricing.

Future Work

There are a few topics open for further research. The main question about the Feature Extraction presented in Chapter 2 is, how it could be extended to the pricing of American options. Besides, the careful analysis of the proposed error splitting (Equation (2.5)) would provide more insight to the efficiency of the Feature Extraction.

Even though the sparse basis functions presented in Chapter 3 and 4 create significant speed-ups of the evaluation of exercisable path-dependent options, the required computations are still expensive and further work has to be conducted in order to obtain a fast valuation procedure. Our findings suggest that further research about approximate exercise and call strategies can lead to such a fast valuation procedure.

The Simulation-Based Hedging presented in Chapter 5 could be extended a joint minimization of transaction costs and costs of risk (costs of the margin account) in an efficient procedure. Furthermore, different trading strategies under transaction costs than the presented time-based rehedging should be analyzed. Finally, another open task is the transfer of Simulation-Based Hedging to other numerical procedures, i.e. lattice or PDE methods in order to obtain even better convergence.

Chapter 7

Appendix

7.1 Important Symbols

Symbol	Explanation
α	the level of confidence
β	a regression coefficient
i, j	index variables
m	number of basis functions
n	number of observations
$\mathbf{B} = (\mathbf{b}_1 \dots \mathbf{b}_m)$	regression basis
$\mathbf{b}_j = (b(x_1) \dots b(x_n))^T$	single basis function of the observations $x_1, \dots, x_n$
$x = (x_1 \dots x_n)^T$	vector of random observations
$y = (y_1 \dots y_n)^T$	vector of random observations
$f(x_i)$	some function of the random observation, e.g. a payoff
z	state $\in \mathbb{R}^s$
s	dimension of the state space
S	underlying asset price process
t	time
T	number of time steps
ϕ	number of assets in a portfolio
θ	random number, drawn from a standard normal distribution
f	some function
$P(S, t_T)$	payoff at maturity time
Prob(X)	probability of event X
$\mathbb{E}[X]$	expected value of X
var(X)	variance of X
cov(X, Y)	covariance of X and Y
std(X)	standard deviation of X
$\|\cdot\|_2$	Euclidian vector norm
κ	conditioning number (stability of a problem)

7.2 Notes for the Proof of Theorem 1.4

In this section, we present the proof for Theorem 1.4.

The following lemmas and definitions are restating the properties of usual linear algebra. For details and proofs we refer the reader to [85], Chapters 6 and 7.

Lemma 7.1 *Let $\mathcal{B}$ be a possibly infinite dimensional Euclidian vector space and let $\mathcal{B}^m$ be a linear subspace of $\mathcal{B}$ with dimension m. Then, for every $b \in \mathcal{B}$, there exists a unique decomposition such that*

$$b = \tilde{b} + \epsilon$$

with $\tilde{b} \in \mathcal{B}^m$ and $\epsilon \perp \mathcal{B}^m$.

Proof See [85], Satz 6.10. □

Definition 7.2 *Let $\mathcal{B}$ be a possibly infinite dimensional Euclidian vector space and let $\mathcal{B}^m$ be a linear subspace of $\mathcal{B}$ with dimension m. Furthermore, let $b \in \mathcal{B}$ and $b = \tilde{b} + \epsilon$ be the decomposition from Lemma 7.1. Then, $\tilde{b}$ is called orthogonal projection from b on $\mathcal{B}^m$. The mapping $P : \mathcal{B} \to \mathcal{B}^m$, which assigns to each $b \in \mathcal{B}$ its orthogonal projection $\tilde{b} \in \mathcal{B}^m \subset \mathcal{B}$ is called orthogonal projection from $\mathcal{B}$ onto $\mathcal{B}^m$.*

Lemma 7.3 *Using any orthonormal basis $b_1, \ldots, b_m$ of $\mathcal{B}^m$, the orthogonal projection of b onto $\mathcal{B}^m$ is given by*

$$P(b) = \sum_{j=1}^{m} < b, b_j > b_j$$

with a scalar product $< \cdot, \cdot >$.

Proof See [85], Bemerkung 6.12. □

Lemma 7.4 *Let $b_1, \ldots, b_m$ be an arbitrary basis of $\mathcal{B}^m$. Then $P(b)$ has a unique representation $P(b) = \sum_{j=1}^{m} a_j b_j$, where $a_1, \ldots, a_m$ are determined by the solution to*

$$\begin{pmatrix} < b_1, b_1 > & \cdots & < b_1, b_m > \\ \vdots & \ddots & \vdots \\ < b_m, b_1 > & \cdots & < b_m, b_m > \end{pmatrix} \begin{pmatrix} a_1 \\ \vdots \\ a_m \end{pmatrix} = \begin{pmatrix} < b, b_1 > \\ \vdots \\ < b, b_m > \end{pmatrix},$$

which is unique.

Proof See [85], Bemerkung 6.14 □

Lemma 7.5 *Let $\mathcal{B}$ be a possibly infinite dimensional Euclidian vector space and let $\mathcal{B}^m$ be a linear subspace of $\mathcal{B}$ with dimension m and P is the orthogonal projection onto $\mathcal{B}^m$. Then, for all $b \in \mathcal{B}$*

$$\|b - P(b)\| < \|b - \tilde{b}\| \ \ \forall \tilde{b} \in \mathcal{B}^m, \tilde{b} \neq P(b)$$

holds, i.e. $P(b)$ is the best approximation of b in $\mathcal{B}^m$ and the approximation problem has a solution which is unique for the norm $\|b\| := \sqrt{< b, b >}$ induced by the inner product $< \cdot, \cdot >$.

Proof See [85], Satz 6.16 □

Finally, after describing the above setting, we can start with the proof to Theorem 1.4.

Proof of Theorem 1.4: If we now look at the local basis approximation of a non-noisy observation sample $(\mathbf{X}, \mathbf{y})$ with $y_i = f(\mathbf{x}^i)$, and $f(\mathbf{x}) = \sum_{j=1}^{\infty} a_j b_j(\mathbf{x})$ and basis functions $b_1 \ldots, b_m$. We can use Theorem 1.1, to get

$$\begin{aligned} \lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} f(\mathbf{x}^i) b_j(\mathbf{x}^i) &= \int_{\mathbb{D}} f(\mathbf{x}) b_j(\mathbf{x}) p(\mathbf{x}) \, \mathrm{d}\mathbf{x} \\ &= < f, b >_r \end{aligned}$$

with random vectors $\mathbf{x}^i \in \mathbb{R}^s$, $i = 1, \ldots, n$ which are distributed according to the probability density function $p(\mathbf{x})$. Lemma 7.5 provides that if we chose a basis $b_1, \ldots, b_m \in \mathcal{B}^m$, we can approximate $f \in \mathcal{B}, \mathcal{B}^m \subset \mathcal{B}$ by

$$f(\mathbf{x}) \approx \sum_{j=1}^{m} \tilde{a}_j^n b_j(\mathbf{x})$$

using the projection given by Lemma 7.4. Since the projection is unique, the coefficients $\tilde{a}_j^n, j = 1, \ldots, m$ are unique. Since the basis functions $b_1, \ldots, b_m$ form a basis of $\mathcal{B}^m$, the coefficients $\tilde{a}_j^n$ of the approximation are equal to the coefficients $\tilde{a}_j^n = a_j$ of the represented function $f(\mathbf{x}) = \sum_{j=1}^{\infty} a_j b_j(\mathbf{x})$.

If we now introduce independent noise into the observations such that the sample is given by $(\mathbf{X}, \mathbf{y})$ with $y_i = f(\mathbf{x}^i) + \epsilon_i$, $\epsilon_i, \mathbf{x}^i$ independent, $i = 1, \ldots, n$ and $f(\mathbf{x}) = \sum_{j=1}^{\infty} a_j b_j(\mathbf{x})$, $a_j = \tilde{a}_j^n$ still

holds for $n \to \infty$ since the discrete version of the scalar product of Equation (1.1) is

$$\begin{aligned}
< b, y > &= \lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} b(\mathbf{x}^i) y_i \\
&= \lim_{n\to\infty} \left(\frac{1}{n} \sum_{i=1}^{n} b(\mathbf{x}^i) f(\mathbf{x}^i) + \frac{1}{n} \sum_{i=1}^{n} b(\mathbf{x}^i) \epsilon_i \right) \\
&\overset{\text{Theorem 1.1}}{=} \int_{\mathbb{D}} b(\mathbf{x}) f(\mathbf{x}) p(\mathbf{x}) \, \mathrm{d}\mathbf{x} + \mathbb{E}[b(\mathbf{x})\epsilon] \\
&\overset{\text{independence of } \mathbf{x},\ \epsilon}{=} \int_{\mathbb{D}} b(\mathbf{x}) f(\mathbf{x}) p(\mathbf{x}) \, \mathrm{d}\mathbf{x} + \mathbb{E}[b(\mathbf{x})] \cdot \mathbb{E}[\epsilon_i] \\
&= \int_{\mathbb{D}} b(\mathbf{x}) f(\mathbf{x}) p(\mathbf{x}) \, \mathrm{d}\mathbf{x}.
\end{aligned}$$

If we now, take a closer look at the explicit representation of the determination of the coefficient vector $\mathbf{a}$:

$$\begin{aligned}
\tilde{\mathbf{a}}^n &= (\mathbf{B}(\mathbf{X})^T \mathbf{B}(\mathbf{X}))^{-1} \mathbf{B}(\mathbf{X})^T \mathbf{y} \\
&= \begin{pmatrix} \sum_{i=1}^{n} b_1(\mathbf{x}^i) b_1(\mathbf{x}^i) & \cdots & \sum_{i=1}^{n} b_1(\mathbf{x}^i) b_m(\mathbf{x}^i) \\ \vdots & \ddots & \vdots \\ \sum_{i=1}^{n} b_m(\mathbf{x}^i) b_1(\mathbf{x}^i) & \cdots & \sum_{i=1}^{n} b_m(\mathbf{x}^i) b_m(\mathbf{x}^i) \end{pmatrix}^{-1} \begin{pmatrix} \sum_{i=1}^{n} b_1(\mathbf{x}^i) y_i \\ \vdots \\ \sum_{i=1}^{n} b_m(\mathbf{x}^i) y_i \end{pmatrix} \\
&= \begin{pmatrix} \frac{1}{n} \sum_{i=1}^{n} b_1(\mathbf{x}^i) b_1(\mathbf{x}^i) & \cdots & \frac{1}{n} \sum_{i=1}^{n} b_1(\mathbf{x}^i) b_m(\mathbf{x}^i) \\ \vdots & \ddots & \vdots \\ \frac{1}{n} \sum_{i=1}^{n} b_m(\mathbf{x}^i) b_1(\mathbf{x}^i) & \cdots & \frac{1}{n} \sum_{i=1}^{n} b_m(\mathbf{x}^i) b_m(\mathbf{x}^i) \end{pmatrix}^{-1} \begin{pmatrix} \frac{1}{n} \sum_{i=1}^{n} b_1(\mathbf{x}^i) y_i \\ \vdots \\ \frac{1}{n} \sum_{i=1}^{n} b_m(\mathbf{x}^i) y_i \end{pmatrix}
\end{aligned}$$

Thus

$$\begin{aligned}
\lim_{n\to\infty} \tilde{\mathbf{a}}^n &= \begin{pmatrix} < b_1, b_1 >_r & \cdots & < b_1, b_m >_r \\ \vdots & \ddots & \vdots \\ < b_m, b_1 >_r & \cdots & < b_m, b_m >_r \end{pmatrix}^{-1} \begin{pmatrix} < f, b_1 >_r \\ \vdots \\ < f, b_m >_r \end{pmatrix} \\
&= \begin{pmatrix} a_1 \\ \vdots \\ a_m \end{pmatrix}
\end{aligned}$$

□

7.3 Proof of Equation (1.8)

Proof Starting with

$$\|\mathbf{a}\|_2^2 = \sum_{i=1}^{m} \frac{c_i^2}{\sigma_i^2} \geq \frac{1}{\sigma_1^2} \sum_{i=1}^{m} c_i^2$$

and $\mathbf{U} = (\mathbf{u}^1, \ldots, \mathbf{u}^m)$ we get

$$\sum_{i=1}^{m} (\mathbf{u}^i)^T \mathbf{y} \mathbf{u}^i = \sum_{i=1}^{m} c_i \mathbf{u}^i = \mathbf{U}^T \mathbf{c}.$$

Furthermore,

$$\begin{aligned} \left\| \sum_{i=1}^{m} (\mathbf{u}^i)^T \mathbf{y} \mathbf{u}^i \right\|_2^2 &= \left\| \mathbf{U}^T \mathbf{c} \right\|_2^2 = \left(\mathbf{U}^T \mathbf{c} \right)^T \left(\mathbf{U}^T \mathbf{c} \right) = \mathbf{c}^T \mathbf{U} \mathbf{U}^T \mathbf{c} = \mathbf{c}^T \mathbf{I} \mathbf{c} = \mathbf{c}^T \mathbf{c} \\ &= \sum_{i=1}^{m} c_i^2 \end{aligned}$$

with identity matrix $\mathbf{I}$ holds. That means

$$\|\mathbf{a}\|_2^2 \geq \frac{1}{\sigma_1^2} \left\| \sum_{i=1}^{m} (\mathbf{u}^i)^T \mathbf{y} \mathbf{u}^i \right\|_2^2$$

□

7.4 Proof of Equation Set (4.4)-(4.7)

In this section we are following the arguments of Ayache et al [12] in order to proof Equations (4.4)-(4.7).

The Equation set (4.4)-(4.7) describes the risk-neutral dynamics of a convertible bond. First, we want to restate the partial differential inequality for the convertible value V

$$\frac{\partial V}{\partial t} + \frac{\sigma^2}{2} S^2 \frac{\partial^2 V}{\partial S^2} + (r + p\eta) S \frac{\partial V}{\partial S} - (r + p)V + p\kappa S(1 - \eta) \geq 0 \tag{7.1}$$

$$V(S, t) \geq \max(B_p(S, t), \kappa S) \tag{7.2}$$

$$\frac{\partial V}{\partial t} + \frac{\sigma^2}{2} S^2 \frac{\partial^2 V}{\partial S^2} + (r + p\eta) S \frac{\partial V}{\partial S} - (r + p)V + p\kappa S(1 - \eta) \leq 0 \tag{7.3}$$

$$V(S, t) \leq \max(B_c(S, t), \kappa S), \tag{7.4}$$

where either one of (4.4)-(4.5) or (4.6)-(4.7) hold, and one of the inequalities holds with equality at each point in the solution domain.

Note that we leave out the indices for the time t of the stochastic processes S and V in this section to make the equations more readable.

The main difference to the derivation of the Black-Scholes Equation in Section 1.3 lies in the possibility of default of the company. In this case, one has to model what happens to the holder of the convertible.

Consider an instantaneous probability $p(S, t)$ of default in the time interval $[t, t + \mathrm{d}t]$ conditional on no default in $[t_0, t]$. In general, we assume that the asset drops to $(1 - \eta)S$ upon default and the present value of holding the convertible until liquidation of the company is F giving the convertible a value of

$$\max(\kappa(1 - \eta)S, F).$$

For simplicity, we assume that F is zero. In the following, we are considering a hedged portfolio

$$\Pi = V(S, t) - \phi_1 S - \phi_2 L$$

with defaultable bond L of the same issuer as V and zero recovery rate upon default. Furthermore, let $\mathrm{d}L = rL\ \mathrm{d}t$, $F = 0 = R$ and $p := p(S, t)$ hold.

Then, we need to determine the dynamics of portfolio Π due to changes of the underlyings. Consequently, there are two main cases:

No default with probability $1 - p(S,t)$ **d**t**.** The arguments from Section 1.3 apply since the portfolio is hedged against small changes in the underlying asset S. That means

$$\delta\Pi = \mathrm{d}V - \phi_1 \,\mathrm{d}S - \phi_2 \,\mathrm{d}L,$$

which is consistent with Equation (1.16).

Default with probability $p(S,t)$ **d**t**.** The change $\mathrm{d}\Pi$ in the portfolio is given by the assets held in the portfolio and the value of the bond as well as the underlying after default:

$$\delta\Pi = \kappa S(1-\eta) - V - \phi_1(-\eta S) - \phi_2 \cdot 0.$$

Unifying both cases and computing the dynamics of Π from Itô's Lemma [67] we obtain

$$\begin{aligned}
\mathrm{d}\Pi &= (1 - p\,\mathrm{d}t)\cdot(\,\mathrm{d}V - \phi_1\,\mathrm{d}S - \phi_2\,\mathrm{d}L) + p\,\mathrm{d}t\cdot(\kappa S(1-\eta) - V - \phi_1(-\eta S) - \phi_2\cdot 0)\\
&= (1 - p\,\mathrm{d}t)\cdot\left(\left[\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2\frac{\partial^2 V}{\partial S^2}\right]\mathrm{d}t + \frac{\partial V}{\partial S}\,\mathrm{d}S - \phi_1\,\mathrm{d}S - \phi_2\,\mathrm{d}L\right)\\
&\quad + p\,\mathrm{d}t\cdot(\kappa S(1-\eta) - V + \phi_1(\eta S)). \qquad (7.5)
\end{aligned}$$

When choosing $\phi_1 = \frac{\partial V}{\partial S}$, Equation (7.5) becomes

$$\begin{aligned}
\mathrm{d}\Pi &= \left(\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2\frac{\partial^2 V}{\partial S^2}\right)\mathrm{d}t - (1 - p\,\mathrm{d}t)\cdot\phi_2\,\mathrm{d}L\\
&\quad + p\,\mathrm{d}t\cdot\left(\kappa S(1-\eta) - V + \frac{\partial V}{\partial S}(\eta S) - \left[\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2\frac{\partial^2 V}{\partial S^2}\right]\mathrm{d}t\right),
\end{aligned}$$

which should be risk-free assuming that the probability of default p is given in the risk-neutral world. Consequently, using $\mathrm{d}L = rL\,\mathrm{d}t$, we obtain

$$\begin{aligned}
\mathrm{d}\Pi &= r\Pi\,\mathrm{d}t = r(V - \frac{\partial V}{\partial S}S - \phi_2 L)\,\mathrm{d}t\\
\Leftrightarrow 0 &= \frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2\frac{\partial^2 V}{\partial S^2} + (r+p)\eta S\frac{\partial V}{\partial S} - (r+p)V\\
&\quad + p\kappa S(1-\eta) - p\cdot\left(\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2\frac{\partial^2 V}{\partial S^2} - rL\phi_2\right).
\end{aligned}$$

If we now choose $\phi_2 = \frac{1}{rL}\left(\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2\frac{\partial^2 V}{\partial S^2}\right)$ we finally obtain the governing PDE of the convertible, which is

$$\frac{\partial V}{\partial t} + \frac{\sigma^2}{2}S^2\frac{\partial^2 V}{\partial S^2} + (r + p\eta)S\frac{\partial V}{\partial S} - (r+p)V + p\kappa S(1-\eta) = 0. \qquad (7.6)$$

Since the holder can convert into shares worth κS or return the convertible to the issuer for the put price B_p and the issuer can call the convertible for B_c, the following boundary constraints

have to hold during the lifetime of the security:

$$\begin{aligned} V(S,t) &\geq \kappa S \\ V(S,t) &\geq B_p \\ V(S,t) &\leq B_c. \end{aligned}$$

Imposing these constraint on the value of the convertible, described by Equation (7.6), leads to the Equation set (4.4)-(4.7).

7.5 Feature Extraction in Octave/MATLAB

```
function start(n)
    % start(n) is a test function for the Asian option pricer with
    % Feature Extraction value(t_T, T, S0, K, r, sigma). n is the
    % number of valuations performed.
    for i = 1:n
        [V1(i) V2(i) V3(i) V4(i) V5(i)] = value(0.5, 125, 100, 100, 0.05, 0.25);
    end

    disp(sprintf('V1: mean=%g std=%g', mean(V1), std(V1)/sqrt(n)));
    disp(sprintf('V2: mean=%g std=%g', mean(V2), std(V2)/sqrt(n)));
    disp(sprintf('V3: mean=%g std=%g', mean(V3), std(V3)/sqrt(n)));
    disp(sprintf('V4: mean=%g std=%g', mean(V4), std(V4)/sqrt(n)));
    disp(sprintf('V5: mean=%g std=%g', mean(V5), std(V5)/sqrt(n)));
    disp('The correlation is:')
    corrcoef([V1 V2 V3 V4 V5])
    disp('The covariance is:')
    cov([V1 V2 V3 V4 V5])
end

function B = B_spline(S, S_min, S_max)
% return basis B of cubic spline according to Section 1.2.2
    m = 4;
    x = linspace(S_min,S_max,m+2);
    x = x(2:end-1);
    B = [ones(size(S)) S S.^2 S.^3 max(repmat(S,1,m)-repmat(x,size(S,1),1),0).^3];
end

function f = B_approx(x, a, S_min, S_max)
% return value of spline at x
% 'a' are the basis function coefficients
% 'S_min' and 'S_max' are the boundary values of the spline
    f = B_spline(x', S_min, S_max)*a;
end
```

% please turn page

```
function [V0_nn V0_fn V0_np V0_fp V0_sim]=value(t_T,T,S0,K,r,sigma)
    % [V0_nn V0_fn V0_np V0_fp V0_sim] = value(t_T, T, S0, K, r, sigma)
    % Computes the value of an Asian option with maturity time t_T, T observations,
    % initial stock price S0, strike K risk-free rate r and volatility sigma.

    % simulate stock S and payoff P
    S = S0 * ones(1000,1);
    dt = t_T/T;
    I = 0;
    for i=1:T
        S  = S .* exp( (r-0.5*sigma^2)*dt + sqrt(dt)*sigma*randn(size(S)));
        if (i<T)
            I = I + S/(T-1);
        end
    end
    P = max(I-K,0);

    % define probability density function p(x), _est stands for estimated;
    % _nest for not estimated
    p_nest = @(x) 1./(x*sigma*sqrt(2*pi*t_T)).*...
                exp(-(log(S0./x)+(r-0.5*sigma^2)*t_T).^2/(2*sigma^2*t_T));
    p_est  = @(x)(ksdensity((S), (x)));

    % perform regression to compute f_tilde := E(P|S)
    a_est = B_spline(S, min(S), max(S))\(P);

    % for the case with no estimation use precalculated highly-accurate values
    S_nest =   1.0e+002 *[   0.35049011165481375 2.476959609710143];
    a_nest =   1.0e+002 *[ -2.271270933408031;  0.095967077029140; -0.001335089683737
                           0.000006124458780; -0.000006609485300; -0.000000109620663
                           0.000002091568479; -0.000007210515689];

    f_est  = @(x)B_approx(x,a_est, min(S),max(S) );
    f_nest = @(x)B_approx(x,a_nest, min(S_nest),max(S_nest) );

    % V_t0 computed as integral
    V0_nn = exp(-r*t_T) * quad(@(x)f_nest(x).*p_nest(x)',0,10*S0,1E-5);
    V0_fn = exp(-r*t_T) * quad(@(x)f_est(x).*p_nest(x)',0,10*S0,1E-5);
    V0_np = exp(-r*t_T) * quad(@(x)f_nest(x).*p_est(x)',0,10*S0,1E-5);
    V0_fp = exp(-r*t_T) * quad(@(x)f_est(x).*p_est(x)',0,10*S0,1E-5);

    % V_t0 computed using the simulations
    V0_sim = exp(-r*t_T) * mean(P);
end
```

7.6 Simulation-Based Hedging in Octave/MATLAB

```
function [ V, delta] = value( S0, K, r, mu, sigma, T, timesteps, number_paths)
% Computes the value V and the optimal hedge delta
% of an American put option.
%
% [ V delta] = value( S0, K, r, mu, sigma, T, timesteps, number_paths)
% Example: [ V delta] = value( 100, 100, 0.05, 0.05, 0.4, 1, 10, 10000)

% Simulate the asset values 1:number_paths/2 starting in [0.5*S0,1.5*S0]
% the other half (number_paths/2+1:number_paths) of the assets start at S0

dt = T/timesteps;
S = zeros(number_paths,timesteps+1);
S(:,1) = [linspace(0.5*S0,1.5*S0,number_paths/2) S0*ones(1,number_paths/2)]';
for i = 2:timesteps+1
    S(:,i) = S(:,i-1).*exp((mu-0.5*sigma*sigma)*dt + sigma*sqrt(dt).* ...
             randn(number_paths,1));
end

% Determine values at maturity time T
V = max(K-S(:,end),0);
B = zeros(number_paths,timesteps+1);
delta = zeros(number_paths,timesteps+1);
B(:,end) = -V;
delta_tmp = zeros(number_paths,1);

% Determine the other values using a dynamic program
for i = timesteps:-1:1

   % Determine the expected portfolio value Pi
   Pi_j = -B(:,i+1)-delta(:,i+1).*S(:,i+1);
   E_Pi  = regress( S(:,i+1), Pi_j); % E[ Pi(t_{i+1}) | S(t_{i+1})]

   % Compare with Exercise Value
   Payoff = K - S(:,i+1);
   ex_index = (E_Pi < Payoff) & (Payoff > 0);

   % Adjust bank accounts of exercised paths
   if sum(ex_index)>0
       B(ex_index,i+1) = -(Payoff(ex_index))- delta(ex_index,i+1).*S(ex_index,i+1);
   end

   % Compute cor/var, then update delta and bank account B
   E_S  = exp(mu*dt)*S(:,i);        % E[ S(t_{i+1}) | S(t_i)]
   E_Pi  = regress( S(:,i), Pi_j); % E[ Pi(t_{i+1}) | S(t_i)]
   delta(:,i) = -regress_cov( S(:,i), S(:,i+1)-E_S, Pi_j-E_Pi);
   B(:,i) = exp(-r*dt) * (B(:,i+1) + (delta(:,i+1)-delta(:,i)).*S(:,i+1));
end
```

```
% adjust bank account for initial delta position
B(:,1) = B(:,1)+  delta(:,1).*S(:,1);

% return the option parameters for the paths starting at S0
V=-mean(B(number_paths/2:end,1));
delta = -mean(delta(number_paths/2:end,1));
% END of value function



function [ E_y ] = regress( x_hat, y);
% Perform a regression on a polynomial basis
% to compute conditional expectation E[y|x]

% scale x domain onto [0,1]
x = (x_hat-min(x_hat))/(max(x_hat)-min(x_hat));

% compute polynomial basis up to degree n
n=6;
B = [];
for i=0:n
    B = [B x.^i];
end
n = round(length(x)/2);

% perform the regression
E_y = B * (B(1:n,:)\y(1:n));
% END of regress



function [ Cov ] = regress_cov( x_hat, co, y);
% Perform a regression on a polynomial basis for cov/var parameter,
% which is the conditional regression coefficient cov(co,y|x_hat)/var(y|x_hat)

% scale x domain onto [0,1]
x = (x_hat-min(x_hat))/(max(x_hat)-min(x_hat));

% compute polynomial basis up to degree n
n=6;
B = [];
B_plain=[];
for i = 0:n
    B = [B co.*x.^i];
    B_plain = [B_plain x.^i];
end
n = round(length(x)/2);
Cov =B_plain*(B\ y);
% END of regress_cov
```

Bibliography

[1] F. S. Acton. *Numerical Methods that Work*, page 106. The Mathematical Association of America, August 1997.

[2] H. Ahn and P. Wilmott. On trading American options. Technical report, Mathematical Finance Group at the University of Oxford (OCIAM), 1997.

[3] D. E. Allen, G. MacDonald, K. D. Walsh, and D. M. Walsh. Using regression techniques to estimate futures hedge ratios, some results from alternative approaches applied to Australian 10 year treasury bond futures. Edith Cowan Finance & Business Economics Working Paper, September 2001.

[4] Z. A. Altintig and A. Butler. Are they still late? The effect of notice period on calls of convertible bonds. *Journal of Corporate Finance*, 11:337–350, 2002.

[5] M. Ammann, A. Kind, and C. Wilde. Are convertible bonds underpriced? An analysis of the French market. *Journal of Banking and Finance*, 27:635–653, 2003.

[6] M. Ammann, A. Kind, and C. Wilde. Simulation-based pricing of convertible bonds. Working paper, University of St. Gallen, Switzerland, 2005.

[7] L. Andersen and D. Buffum. Calibration and implementation of convertible bond models. *Journal of Computational Finance*, 7(2):1–34, 2003/04.

[8] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. D. Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK User's Guide*, chapter Linear Least Squares (LLS) Problems. SIAM, Philadelphia, 3rd edition, 1999.

[9] A. D. Andricopoulos, M. Widdicks, P. W. Duck, and D. P. Newton. Universal option valution using quadrature methods. *Journal of Financial Economics*, 67:447–471, 2003.

[10] P. Asquith. Convertible bonds are not called late. *The Journal of Finance*, 50(4):1275–1289, September 1995.

[11] E. Ayache, P. A. Forsyth, and K. R. Vetzal. Next generation models for convertible bonds with credit risk. *Wilmott Magazine*, pages 68–77, December 2002.

[12] E. Ayache, P. A. Forsyth, and K. R. Vetzal. The valuation of convertible bonds with credit risk. *Journal of Derivatives*, 11:9–29, Fall 2003.

[13] V. Bathelmann, E. Novak, and K. Ritter. High dimensional polynomial interpolation on sparse grids. *Advances in Compuational Mathematics*, 12:273–288, 2000.

[14] C. Bender, A. Kolodko, and J. Schoenmakers. Iterating snowballs and related path dependent callables in a multi-factor libor model. WIAS Preprint, ISSN 0946-8633, 2005.

[15] S. J. Berridge and J. H. Schumacher. Pricing high-dimensional American options using local consistency conditions. Technical Report 2004-19, CentER Discussion Paper, 2004.

[16] R. Bilger. Valuing American-Asian options using the Longstaff-Schwartz algorithm. Msc thesis in computational finance, Oxford University, 2003.

[17] F. Black and M. Scholes. The pricing of options and corporate liabilities. *Journal of Political Economy*, 81:637–659, 1973.

[18] T. Bollerslev. Generalized autoregressive condtional heteroscedasticity. *Journal of Econometrics*, 31:307–327, 1986.

[19] T. Bonk. A new algorithm for multi-dimensional adaptic numerical quadrature. In W. Hackbush, editor, *Proceedings of the 9th GAMM-Seminar, Kiel 1993*, Braunschweig, January 1994. Vieweg.

[20] A. W. Bowman and A. Azzalini. *Applied Smoothing Techniques for Data Analysis: The Kernel Approach with S-Plus Illustrations*. Oxford Statistical Science Series, 1997.

[21] P. Boyle. Options: A Monte Carlo approach. *Journal of Financial Economics*, 4(3):323–338, 1977.

[22] P. Boyle. A lattice framework for option pricing with two state variables. *Journal of Financial and Quantitative Analysis*, 23(1):1–12, March 1988.

[23] P. Boyle, M. Broadie, and P. Glasserman. Monte carlo methods for security pricing. *Journal of Economic Dynamics and Control*, 21(8/9):1276–1321, 1997.

[24] M. J. Brennan and E. S. Schwartz. Convertible bonds: Valuation and optimal strategies for call and conversion. *Journal of Finance*, 32:1699–1715, 1977.

[25] M. J. Brennan and E. S. Schwartz. Analysing convertible bonds. *Journal of Financial and Quantitative Analysis*, 15:907–929, 1980.

[26] M. J. Brennan and E. S. Schwartz. The case for convertibles. *Chase Financial Quarterly*, 1(3):27–46, 1982.

[27] M. J. Buchan. *Convertible bond pricing: Theory and evidence*. PhD thesis, Harvard University, 1997.

[28] H. J. Bungartz. Higher order finite elements on sparse grids. *Electronic Transactions on Numerical Analysis*, 6:63–77, December 1997.

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

[30] A. W. Butler. Revisiting optimal call policy for convertible bonds. *Financial Analyst Journal*, 58(1):50–55, 2002.

[31] H. Cardot. Conditional functional principal components analysis. *Scandinavian Journal of Statistics*, (OnlineEarly Articles):–, 2006.

[32] J. F. Carrière. Valuation of the early-exercise price for options using simulations and nonparametric regression. *Insurance: Mathematics and Economics*, 19:19–30, 1996.

[33] A. Cerny. Dynamic programming and mean-variance hedging in discrete time. Technical report, Cass Business School Research Paper, October 2003.

[34] A. Cerny. *Mathematical Techniques in Finance: Tools for Incomplete Markets*. Princeton University Press, January 2004.

[35] A. Cerny and J. Kallsen. Hedging by sequential regression revisited. Working paper, City University London and TU München, 2007.

[36] J. Cox, S. Ross, and M. Rubenstein. Option pricing: A simplifyed approach. *Journal of Financial Economics*, 7:229–263, 1979.

[37] M. G. Cox. The numerical evaluation of b-splines. *IMA Journal of Applied Mathematics*, 10(2):134–149, 1972.

[38] C. de Boor. *A practical guide to splines*. Springer, 1985.

[39] E. Derman and I. Kani. The volatility smile and its implied tree. Quantitative strategies research notes, Goldman Sachs, January 1994.

[40] S. Dirnstorfer, A. J. Grau, and R. Zagst. Moving window Asian options: Sparse grids and Least-Squares Monte Carlo. submitted, December 2005.

[41] B. Dupire. Pricing with a smile. *Risk magazine*, 7(1):18–20, 1994.

[42] L. Ederington. The hedging performance of the new futures markets. *Journal of Finance*, 34(1):157ff, Mar 1979.

[43] L. Fahrmeir, R. Knstler, and I. Pigeot. *Statistik*. Springer, 2004.

[44] J. Fan and Q. Yao. Efficient estimation of conditional variance functions in stochastic regression. Technical report, Department of Statistics, UCLA, 1998.

[45] W. Feller. *An Introduction to Probability Theory and Its Applications, Vol. 2, 3rd ed.*, chapter The Berry-Essen Theorem., pages 542–546. Wiley, 1971.

[46] H. Föllmer and P. Leukert. Quantile hedging. *Finance and Stochastics*, 3:251–273, 1999.

[47] H. Föllmer and M. Schweizer. Hedging by sequential regression: an introduction to the mathematics of option trading. *ASTIN Bulletin*, 18:147 – 160, 1989.

[48] H. Föllmer and D. Sondermann. *Contributions to Mathematical Economics*, volume 34, chapter Hedging of Non-Redundant Contingent Claims, pages 205–223. North-Holland, 1986.

[49] P. A. Forsyth. Lecture notes for semiar on computational finance. Faculty of Mathematics, University of Waterloo, Mai 2001.

[50] P. A. Forsyth and K. R. Vetzal. Quadratic convergence of a penalty method for valuing American options. *SIAM Journal on Scientific Computing*, 23:2096–2123, 2002.

[51] M. Frittelli. The minimal entropy martingale measure and the valuation problem in incomplete markets. *Mathematical Finance*, 10(1):39–52, 2000.

[52] J. Garcke, M. Griebel, and M. Thess. Data mining with sparse grids. *Computing*, 67(3):225–253, October 2001.

[53] P. Glasserman. *Monte Carlo Methods in Financial Engineering*. Springer, Berlin, 2003.

[54] T. Gol and J. Kallsen. Optimal portfolios for logarithmic utility. *Stochastic Processes and their Applications*, 89(1):31–48, 2000.

[55] A. J. Grau. Soft call constraints in convertible bonds and a pricing framework for path dependent options. Master's thesis, University of Waterloo, Canada, 2003.

[56] A. J. Grau, P. A. Forsyth, and K. R. Vetzal. Convertible bonds with call notice periods. In *Proceedings to IASTED International Conference on Financial Engineering and Applications*, Banff, Canada, July 2003.

[57] A. J. Grau, P. A. Forsyth, and K. R. Vetzal. PDE and monte carlo methods for pricing convertible bonds with soft call constraints. Technical report, University of Waterloo, Canada, 2006. http://www.cs.uwaterloo.ca/ paforsyt/.

[58] D. Greiner, A. Kalay, and H. K. Kato. The market for callable-convertible bonds: Evidence from Japan. *Pacific-Basin Finance Journal*, 10:1–27, 2002.

[59] K. Hallatschek. Fouriertransformation auf dünnen Gitttern mit hierarchischen Basen. *Numer. Math.*, 63:83–97, 1992.

[60] W. Härdle. *Applied Nonparametric Regression*. Cambridge University Press, 1992.

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

[62] N. J. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM, Philadelphia, 2nd edition, 2002.

[63] J. Hull and A. White. The pricing of options on assets with stochastic volatilities. *Journal of Finance*, 42(2):281–300, June 87.

[64] J. C. Hull. *Options, Futures and Other Derivatives*, chapter 21, pages 520–524. Prentice Hall, New Jersey, 6th edition, 2006.

[65] J. Ingersoll. A contingent-claims valuation of convertible securities. *Journal of Financial Economics*, 4:289–322, 1977.

[66] J. Ingersoll. An examination of corporate call policies on convertible securities. *Journal of Finance*, 32:463–478, 1977.

[67] K. Ito. On stochastic differential equations. *Memoirs, American Mathematical Society*, 4:1–51, 1951.

[68] J. Kallsen. A utility maximization approach to hedging in incomplete markets. *Mathematical Methods of Operations Research*, 50:321–338, 1999.

[69] J. Kallsen and C. Kühn. *Exotic Option Pricing and Advanced Lévy Models*, chapter Convertible bonds: financial derivatives of game type. Wiley, Chichester, 2005.

[70] C.-H. Kao and Y.-D. Lyuu. Pricing of moving average-type options with applications. *Journal of Futures Markets*, 23(5):415–440, 2003.

[71] A. Kemna and A. Vorst. A pricing method for options based on average asset values. *Journal of Banking and Finance*, 14:113–129, 1990.

[72] C. Koziol. *Valuation of Convertible Bonds when Investors Act Strategically*, volume 110 of *Beiträge zur betriebswirtschaftlichen Forschung*. Dt. Univ.-Verl., Wiesbaden, Germany, 2004.

[73] C. Koziol. Optimal exercise strategies for corporate warrants. *Quantitative Finance*, 6(1):37–54, February 2006.

[74] C. Koziol and W. Bühler. Valuation of convertible bonds with sequential conversion. *Schmalenbach Business Review (sbr)*, 54:302–334, 2002.

[75] C. Kühn. *Shocks and Choices - an Analysis of Incomplete Market Models*. PhD thesis, TU-München, Germany, Fakultät für Mathematik, November 2002.

[76] D. Lamberton, H. Pham, and M. Schweizer. Local risk-minimization under transaction costs. *Mathematics of Operations Research*, 23:585–612, 1998.

[77] K. W. Lau and Y. K. Kwok. Pricing algorithms for options with exotic path dependence. *Journal of Derivatives*, pages 28–38, Fall 2001.

[78] K. W. Lau and Y. K. Kwok. Optimal calling policies in convertible bonds. *Proceedings of International Conference on Computational Intelligence for Financial Engineering*, March 2003.

[79] H. E. Leland. Option pricing and replication with transactions costs. *Journal of Finance*, 40(5):1283–1301, 1985.

[80] J. Li, Y. C. Hon, and C. S. Chen. Numerical comparisons of two meshless methods using radial basis functions. *Engineering Analysis with Boundary Elements*, 26(3):205–225, March 2002.

[81] F. A. Longstaff and E. S. Schwartz. Valuing American options by simulation - a simple least-squares approach. *The Review of Financial Studies*, 14(1):113–147, 2001.

[82] D. G. Luenberger. Pricing a nontradable asset and its derivatives. *Journal of Optimization Theory and Applications*, 121(3):465–487, June 2004.

[83] S. L. Lummer and M. W. Riepe. Convertible bonds as an asset class: 1957-1992. *Journal of Fixed Income*, 3(2):47–57, September 1993.

[84] D. Lvov, A. Yigitbasioglu, and N. E. Bachir. Pricing convertible bonds by simulation. Working Paper, ISMA Center, the University of Reading, 2004.

[85] W. Mackens and H. Voss. *Mathematik I.* HECO-Verlag, Alsdorf, 1993.

[86] C. D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. The MIT Press, 1999.

[87] R. Merton. Theory of rational option pricing. *Bell Journal of Economics and Management Science*, 4(1):141–183, 1973.

[88] R. C. Merton. Option princing when the underlying stock returns are discontinuous. *Journal of Financial Economics*, 3:125–144, 1976.

[89] Y. Muromachi. The growing recognition of credit risk in corporate and financial bond markets. Technical Report Paper # 126, Financial Research Group, NLI Research Institute, 1-1-1 Yurakucho, Chiyoda-ku, Tokyo 100-0006, Japan, 1999.

[90] M. Musiela and M. Rutkowski. *Martingale Methods in Financial Modelling*. Springer, 1997.

[91] C. R. Niethammer. On convergence to the exponential utility problem with jumps. *Stochastic Analysis & Applications*, forthcoming.

[92] H. H. Panjer, P. P. Boyle, S. H. Cox, H. U. Gerber, H. H. Mueller, H. W. Pedersen, S. R. Pliska, M. Sherris, E. S. Shiu, and K. S. T. (Author). *Financial Economics: With Applications to Investments, Insurance and Pensions*. The Actuarial Foundation, 1998.

[93] U. Pettersson, E. Larsson, G. Marcusson, and J. Persson. Improved radial basis function methods for multi-dimensional option pricing. Technical report 2006-028, Uppsala University, 2006.

[94] S. R. Pliska. *Introduction to Mathematical Finance: Discrete Time Models*. Wiley, July 1997.

[95] B. Pochart and J.-P. Bouchaud. Option pricing and hedging with minimum local expected shortfall. *Quantitative Finance*, 4:607–618, 2004.

[96] M. Potters, J.-P. Bouchaud, and D. Sestovic. Hedged monte carlo: Low variance derivative pricing with objective probabilities. *Physica A*, 289:517–525, 2001.

[97] M. J. D. Powell. The uniform convergence of thin plate spline interpolation in two dimensions. *Numerische Mathematik*, 68(1):107–128, 1994.

[98] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in FORTRAN: The Art of Scientific Computing*, chapter Richardson Extrapolation and the Bulirsch-Stoer Method., pages 718–725. Cambridge University Press, Cambridge, England, 2nd edition, 1992.

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

[100] C. Reisinger. *Numerische Methoden für hochdimensionale parabolische Gleichungen am Beispiel von Optionspreisaufgaben*. Dissertation, Ruprecht-Karls-Universität, Heidelberg, June 2004.

[101] V. Ryabchenko, S. Sarykalin, and S. Uryasev. Pricing european options by numerical replication: Quadratic programming with constraints. *Asia-Pacific Financial Markets*, 11(3):301–333, 2004.

[102] H. Scheid, editor. *Schüler Duden: Die Mathematik*, chapter Regression, page 340ff. Dudenverlag, 3rd. edition, 1991.

[103] E. S. Schwartz. Valuation of warrants implementing a new approach. *Journal of Financial Economics*, 4:79–93, 1977.

[104] M. Schweizer. Variance-optimal hedging in discrete time. *Mathematics of Operations Research*, 20:1–32, 1995.

[105] F. Selmi and J. Bouchaud. Alternative large risks hedging strategies for options. *Wilmott Magazine*, March:64–67, 2005.

[106] R. Shao and B. Roe. The design and pricing of fixed- and moving window contracts: An application of Asian-basket option pricing methods to the hog-finishing sector. *The Journal of Futures Markets*, 23(11):1047–1073, 2003.

[107] S. A. Smolyak. Quadrature and interpolation formulas for tensor products of certain classes of functions. *Dokl. Akad. Nauk SSSR*, 148:1042–1043, 1963.

[108] L. Stentoft. Convergence of the least squares Monte-Carlo approach to American option valuation. School of Economics & Management, University of Aarhus, September 2003.

[109] C. J. Stone. Optimal rates of convergence for nonparametric regression. *Annals of Statistics*, 10(4):1040–1053, 1982.

[110] M. H. Stone. The generalized Weierstrass approximation theorem. *Mathematics Magazine*, 21(4):167–184, 1948.

[111] W. Törnig and P. Spellucci. *Numerische Mathematik für Ingenieure und Physiker*. Springer-Verlag, 2nd edition, 1990.

[112] K. Tsiveriotis and C. Fernandes. Valuing convertible bonds with credit risk. *Journal of Fixed Income*, 8(2):95–102, September 1998.

[113] E. van de Heiligenberg, W. Klijnstra, and M. Lundin. The optimal portfolio choice in today's markets. Technical report, Fortis Investment Management, 2002.

[114] H. Voss. Grundlagen der numerischen Mathematik. http://www.tu-harburg.de/mat/LEHRE/material/grnummath.pdf, 2004.

[115] H. Wahba. *Spline Model for Observational Data*. Society for Industrial and Applied Mathematics, Philadelphia and Pennsylvania, 1990.

[116] H. Werner and R. Schaback. *Praktische Mathematik II*, page pp. 150 151. Springer Verlag, 1972.

[117] P. Wilmott. *Paul Wilmott on Quantitative Finance*. John Wiley & Sons Ltd., West Sussex, England, 2000.

[118] S. N. Wood. Thin plate regression splines. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 65(1):95, February 2003.

[119] A. Yigitbasioglu and C. Alexander. Pricing and hedging convertible bonds: delayed calls and uncertain volatility. *International Journal of Theoretical & Applied Finance*, 9(3):415–437, 2004. International Journal of Theoretical & Applied Finance.

[120] R. Zagst. *Interest Rate Management*. Springer Verlag, Heidelberg, 2002.

[121] C. Zenger. Sparse grids. In W. Hackbusch, editor, *Notes on Numerical Fluid Mechanics*, volume 31, Braunschweig, 1991, 1990. Vieweg.

[122] R. Zvan, P. A. Forsyth, and K. R. Vetzal. Robust numerical methods for PDE models of Asian options. *Journal of Computational Finance*, 1:39–78, 1998.

[123] R. Zvan, P. A. Forsyth, and K. R. Vetzal. Discrete Asian barrier options. *Journal of Computational Finance*, 3:41–68, 1999.

[124] R. Zvan, P. A. Forsyth, and K. R. Vetzal. A finite volume approach for contingent claims valuation. *IMA Journal of Numerical Analysis*, 21:703–731, 2001.