UNIVERSITÄT BAYREUTH
MATHEMATISCHES INSTITUT

Financial Option Valuation by Monte Carlo Simulation

von

Andrea Kölz

Datum: September 28, 2005

Aufgabenstellung / Betreuung:

Prof. Dr. L. Grüne

Contents

1 Introduction **1**

2 Options **5**

2.1 Financial Derivatives 5

2.2 Options 6

2.2.1 Influences on Option Values 8

2.2.2 Payoff Functions 11

2.2.3 Put-Call Parity 12

2.2.4 Upper and Lower Bounds 13

2.2.5 Hedging Strategies 17

2.2.6 Exotic Options 19

3 Essentials of Stochastics **21**

3.1 Random Variables 23

3.1.1 Integration and moments 25

3.1.2 Distributions 27

3.1.3 Multiple Random Variables 29

3.1.4 Confidence Intervals 32

3.2 Convergence and approximation 33

4 Asset Price Model **37**

4.1 The Wiener Process 37

4.1.1 Approximation of a Wiener process 37

4.1.2 Some Properties of the Wiener process 38

4.2 Stochastic Integral 43

4.3 SDE model for asset price movements 45

4.3.1 Ito Lemma 48

4.3.2 Solution of Geometric Brownian Motion 48

4.3.3 Options on dividend-paying assets 49

5 Random Number Generators **51**
5.1 Uniformly distributed random numbers . . . 51
5.1.1 Linear congruential generators . . . 52
5.1.2 Fibonacci Generators . . . 52
5.2 Sequences of Numbers with Low Discrepancy . . . 56
5.3 Normally distributed random numbers . . . 59
5.3.1 Inversion . . . 60
5.3.2 Transformations . . . 63
5.3.3 Comparison of methods . . . 66

6 Monte Carlo Method **69**
6.1 Monte Carlo Method for option pricing . . . 70
6.2 Convergence of the Monte Carlo Method . . . 74
6.3 Variance reduction . . . 76
6.4 Quasi-Monte Carlo . . . 78
6.5 Volatility models . . . 80
6.5.1 Historical volatility . . . 82
6.5.2 Implied volatility . . . 83
6.5.3 Stochastic Volatility . . . 83
6.6 Results . . . 86

7 American Options **91**
7.1 Tilley's Bundling Algorithm . . . 95
7.2 Bounded Recursive Stochastic Simulation (BRSS) . . . 98
7.3 Broadie and Glasserman algorithm . . . 101
7.4 Comparison of methods . . . 112

8 Summary and Conclusions **117**

A Black-Scholes **119**
A.1 The Black Scholes Analysis . . . 119
A.2 Boundary and final conditions . . . 120
A.3 The Black-Scholes formulae . . . 121

B The Euler-Maruyama algorithm **125**

C MATLAB codes **127**
C.1 MATLAB code for Random number Generators . . . 127
C.1.1 Congruential . . . 127
C.1.2 Fibonacci . . . 127
C.1.3 Lagged Fibonacci . . . 128
C.1.4 Boxmuller . . . 128

C.1.5 Moro 129
C.1.6 Halton sequence 131
C.2 MATLAB codes used in Chapters 4 and 6 131
C.2.1 Black Scholes 131
C.2.2 Wiener Process 132
C.2.3 Asset Price Movement 132
C.2.4 Monte Carlo simulation for evaluating a European call option 133
C.2.5 Stochastic volatility 136
C.2.6 Historical volatility 138
C.3 MATLAB code for Broadie and Glasserman algorithm 138
C.3.1 Broadie and Glasserman algorithm 138
C.3.2 Low estimator tree 139
C.3.3 High estimator tree 141

Bibliography **143**

List of Tables

5.1 Coefficients for Moro's inversion method 63
5.2 Comparison: Running time of Moro's and the Boxmuller method . . . 68

6.1 MC simulation for $S_0 = 120$ 73
6.2 MC simulation for $S_0 = 100$ 74
6.3 MC simulation for $S_0 = 80$ 75
6.4 MC simulation with AV for $S_0 = 120$ 79
6.5 Quasi Monte Carlo simulation for $S_0 = 120$ 82
6.6 Different numbers of simulations for the same level of accuracy . . . 86
6.7 Different levels of accuracy for a constant number of simulations . . . 88

7.1 AO without dividends for $S_0 = 80$ 109
7.2 AO without dividends for $S_0 = 100$ 109
7.3 AO without dividends for $S_0 = 120$ 110
7.4 Comparison of different start values for S_0 111
7.5 Values for an American type call option with a continuous dividend rate 113

List of Figures

2.1 Payoff diagrams of a European call (left) and a European put (right). . 12
2.2 Qualitative curves of the prices of European and American options. . . 17
2.3 Payoff diagrams of a long straddle (left) and a long strangle (right) . . 18
2.4 Payoff function of a butterfly spread 19

3.1 Graph of the density function of the Standard Gaussian Distribution . 28

4.1 Zooming in on a path of a Wiener Process 39
4.2 Illustrative example: 3 different paths of the Wiener Process 40
4.3 Real asset price movement compared to simulated ones 46

5.1 1500 pairs of points (U_{i-1}, U_i) computed with the congruential generator 53
5.2 1500 pairs of points (U_{i-1}, U_i) computed with the fibonacci generator 54
5.3 1500 pairs of points (U_{i-1}, U_i) computed with the lagged fibonacci generator 55
5.4 1500 pairs of points (U_{i-1}, U_i) computed with the MATLAB function 'rand' 57
5.5 The first 1500 points of the Halton sequence in dimensions 1 (base $p_1 = 2$) and 2 (base $p_2 = 3$) 60
5.6 The first 1500 points of the Halton sequence in dimensions 27 (base $p_1 = 103$) and 28 (base $p_2 = 107$) 61
5.7 The first 10000 numbers of the Halton sequence uniformly distributed and transformed to a normal distribution by the Boxmuller method and the Moro algorithm. 66
5.8 10000 numbers computed with the MATLAB function 'rand' uniformly distributed and transformed to a normal distribution by the Boxmuller method and the Moro algorithm. 66
5.9 10000 numbers computed with the linear congruential generator uniformly distributed and transformed to a normal distribution by the Boxmuller method and the Moro algorithm. 67

5.10 10000 numbers computed with the lagged fibonacci generator uniformly distributed and transformed to a normal distribution by the Boxmuller method and the Moro algorithm. 67
5.11 10000 numbers computed with the MATLAB function 'randn' 67

6.1 Confidence interval for a European call option with $S_0 = 120$ 74
6.2 Confidence interval for a European call with $S_0 = 120$ and AV 78
6.3 Comparison of the relative error with Boxmuller and Moro 80
6.4 Confidence interval for a European call with $S_0 = 120$ (QMC) 81
6.5 Confidence interval for a European call with $S_0 = 120$ (QMC AV) . . 81
6.6 Mean-reverting volatility 85
6.7 Option value with constant and mean-reverting volatility 85
6.8 Asset price movement with constant and mean-reverting volatility . . 86
6.9 Relative error of the Monte Carlo method for a European call with $S_0 = 120$ 87
6.10 Relative error of the Monte Carlo method for a European call with $S_0 = 100$ 88
6.11 Relative error of the Monte Carlo method for a European call with $S_0 = 80$ 89

7.1 Simulated tree for $b = 3$ 102
7.2 The high estimator Θ 102
7.3 The simple low estimate 105
7.4 The low estimator θ 105
7.5 Confidence interval for an American call with $S_0 = 80$ (b=10) 110
7.6 Confidence interval for an American call with $S_0 = 80$ (b=25) 111
7.7 Confidence interval for an American call with $S_0 = 80$ (b=50) 112
7.8 Relative error of the BRSS Method for an American call with $S_0 = 80$ 113

Chapter 1

Introduction

The variety of methods for pricing financial options is very big. Depending on the complexity of the restricting assumptions that are made and on the type of option that has to be evaluated, analytical approaches, partial differential equations or computational methods can be used. In practice, the complexity of most options requires approximation schemes for determining the price of the option. In this thesis we want to examine Monte Carlo simulation as a method for evaluating such options. Since it is very flexible and applicable to a large range of different problems it is also referred to in plural and the term Monte Carlo methods can be found in literature a lot of times. Monte Carlo simulation methods are a powerful and widely used tool for pricing path dependent or higher dimensional European type options since the approach is straightforward, easy to understand and not hard to implement.
In case the option under consideration is American, however, simulation approaches for option pricing do not seem to be practical at all. In fact, the pricing problem for American options had been considered to be computationally intractable for a long time. Only during the last decade, several simulation methods for pricing American type options have been developed. The first approaches were very limited in their flexibility but better methods have been invented since then and Monte Carlo simulation is no longer considered to be unsuitable for the problem of pricing American options today.

The objective of this thesis is the valuation of financial options of both European and American types by Monte Carlo simulation. In what follows the respective algorithms are all implemented in MATLAB.
Although simulation methods are not actually needed for the simplest kinds of European options which can be priced faster using other methods, we will illustrate the concept of the Monte Carlo method for option pricing on the example of the simplest kinds of European options, the so called plain vanilla ones.
Subsequently we will make the same restrictions when evaluating American options.

Before we can price options using Monte Carlo simulation, some groundwork has to be done.

We will start with taking a look at the concept of an option as a special case of a financial derivative in Chapter 2. Different factors that have an influence on the price price will be listed before first boundaries on option values can be derived. The chapter then concludes with a short overview of different option types that exist on the market after we discussed what options are used for.

In Chapter 3 some stochastic definitions and theorems are introduced that will be needed for deriving a mathematical model of asset price dynamics in later chapters.

Having thus covered basics of option theory and stochastics, we can then derive a mathematical model for the price movement of the underlying asset in Chapter 4.

Examining several properties of the Wiener process as a special case of a general stochastic process will lead us to the concept of a stochastic integral or Ito integral. With this important tool we will then be able to construct the stochastic differential equation(SDE) for modeling the dynamics of asset prices, called Geometric Brownian Motion. A solution to this SDE for the simple one-dimensional case as well as an extension of the model for options on dividend paying assets will be given at the end of that chapter.

Due to the randomness of the asset price movement, it is of importance to have access to a large quantity of random numbers when evaluating options via Monte Carlo simulation. This is why we will deal with different methods for generating random numbers with certain properties in Chapter 5. Although Monte Carlo methods work with normally distributed deviates, we will start out with presenting different methods for generating uniformly distributed pseudo-random numbers. Since randomness is not as essential for the Monte Carlo method as an equidistant distribution, we will then introduce low discrepancy sequences as an alternative to pseudo-random numbers. Those so called quasi-random numbers fill the unit square as uniformly as possible.

Subsequently we will examine two different classes of methods that generate normally distributed numbers out of numbers with a uniform distribution. Those are the ones of interest for Monte Carlo simulation methods.

At this point we will have covered the last piece of background information that is needed and we can work with Monte Carlo simulation methods in Chapter 6. After the general principle of the method is explained, we will illustrate the Monte Carlo Method for option valuation on the example of a European call option in MATLAB.

This simple example will show clearly that the convergence of Monte Carlo simulation is very slow. We therefore will examine the convergence behavior of the method and see that it is not only proportionally to the square root of the number of simulation

runs but also to the variance of the random variable in question. Consequently, we will explore antithetic variates as an example of variance reduction methods. Applying this improved approach to the same example as before yields significantly better results than crude Monte Carlo.
Another approach to be investigated is that of using quasi-random numbers instead of pseudo-random ones. As another example will show, this so called Quasi-Monte-Carlo method outperforms Monte Carlo simulation with antithetic variates by a large degree.

In Chapter 7 we will then concern ourselves with the valuation of American type options. At first we will give an overview of the differences to evaluating European options and explain why difficulties have to be faced when using forward simulation methods for a problem that requires a backward algorithm. Having this done we will introduce three different approaches of solving this problem and give a computational example of the Broadie and Glasserman algorithm before we compare the three methods.

To conclude the thesis, we will sum up the results we derived in the previous chapters as well as give a short overview of different topics that could not be explored in this thesis.

Chapter 2

Options

In this chapter we are going to discuss the following questions

- What are options?
- What influences the value of an option?
- What are the upper and lower bounds on the fair value of an option?
- What are options used for?
- What kinds of options are there on the market?

in a very basic way. Therefore, we present some fundamental terms which are needed when talking about options. After financial derivatives are introduced in general, we will define the most basic kinds of options, as well as give a short overview of the influencing factors on the value of an option. Once this is done we can derive upper and lower bounds on option values, as well as give a brief sketch of hedging strategies. We will then conclude this chapter by introducing Exotic options before we talk about the Monte Carlo simulation in the following chapters.
For books covering the basics of option theory we refer to [12], [13], [23] and [24], all of which have had a huge influence on the presentation of this chapter.

2.1 Financial Derivatives

In general, financial derivatives are contracts that agree to sell or buy an asset for a certain price at a certain future point of time or during a specified time interval. They are called derivatives because the value of such a contract depends on the value of the underlying asset. Thereby the term asset describes any financial object whose value is known at present but is likely to change in the future. The most common examples of assets would be shares in a company, bonds, commodities or currencies. Financial derivatives are instruments that provide investors with different possibilities for composing their portfolio in order to meet their individual needs. Depending on how they are used, they can reduce risk, or they can increase it. Thus financial derivatives can be instruments for either speculation or hedging. We will deal with different ways of hedging later in this chapter.

Financial derivatives can roughly be categorized in three groups

- **Forwards and Futures**
  A forward contract is an agreement to buy or sell an asset on a specified future date for a price agreed upon today.
  A futures contract is basically a standardized forward. Its value is determined on a daily basis, and thus futures - unlike forwards - can be traded on the market.
- **Swaps**
  A swap is an agreement between two parties to exchange a sequence of cash flows in the future according to some formula.
- **Options**
  An option is a contract that gives its holder the right but not the obligation to buy or sell an asset at or before a given date in the future.

The buyer of a financial derivative is said to hold the long position, while the seller or writer holds the short position. These two terms sum up the possible positions that can be taken in futures, forwards or swaps contracts. Options, however, provide four possible positions:

long call: the right to buy

long put: the right to sell

short call: the obligation to sell in case of exercise

short put: the obligation to buy in case of exercise

2.2 Options

The main difference between options and other forms of financial derivatives is their non-obligatory nature for the holder. He can choose whether or not to exercise his right, whereas the writer has a potential obligation: he must sell (or buy) the underlying asset unless the holder decides to let his option expire worthless. A forward, for example, does not provide its holder with such a right of choice. It costs nothing to enter into a forward contract. The value of such an agreement is determined by the fixed

price which is to be payed for the asset at expiry. However, entering long into an option contract comes at a certain price, called premium. The holder attains a right with no obligation, thus leaving the writer of the option with an asymmetric higher risk. The premium, which the holder pays to the writer when issuing the option, is supposed to compensate the writer for those possible future liabilities, and it also limits the holders potential loss as we will see later on.
In what follows, we will concentrate on the standpoint of the holder. This focus reduces the possible positions down to the long-call and the long-put.
To review, a call option is a contract that gives its holder the right to buy a certain amount of the underlying asset from the writer of the option at or up to the maturity date, and a put option provides its owner with the right to sell the underlying. Thereby, the holder of a call expects the price of the asset to rise, while the buyer of a put option expects the price of the asset to drop.
The simplest form of a financial option, a plain-vanilla European option is defined as follows:

Definition 2.1.

(i) A European call option gives the holder the right, but not the obligation, to buy an underlying asset at a specified expiry date T, for a specified strike price K.

(ii) A European put option gives the holder the right, but not the obligation, to sell an underlying asset at a specified expiry date T, for a specified strike price K.

Remark 2.2. *Other terms used for the expiry date T are expiration date, exercise date or maturity date. Sometimes the term exercise price is used instead of strike price when talking about K.*

A European option can only be exercised at its maturity date T. Until this date, the holder has only two possibilities: he can either keep the option, or he can sell it. Another form of an option which leaves its holder a lot more freedom concerning the exercise date is the American option. Its holder has the possibility to buy or sell the underlying asset at any given time up to the maturity date of the option:

Definition 2.3.

(i) An American call option with strike price K and expiry date T gives the holder the right, but not the obligation, to buy an asset for the strike price K at any time $t_0 \leq t \leq T$.

(ii) An American put option with strike price K and expiry date T gives the holder the right, but not the obligation, to sell an asset for the strike price K at any time $t_0 \leq t \leq T$.

Remark 2.4. *Generally, the letter V will denote the value of an option. However, if distinction makes it necessary we will use the letters C or P for a call or a put option respectively. The indices A and E will mark American and European style options.*

Since the possibilities provided by an American option include those of a European option, the value of an American option can never drop below the value of a European option with the same maturity and strike price.

Remark 2.5. *Later on, we will show that for an American call option early exercise is never profitable which is why the value of an American call equals the value of a European call. However, the same is not true for an American put.*

Mathematically, American options are more complicated than European ones. In addition to the value, one also has to determine the best time to exercise an American option. This problem will be tackled later on in this thesis.
The two forms of options introduced above are the most basic and widespread ones. The classification into European or American options only concerns the terms of exercise and has no geographical meaning. Both types are traded on markets all over the world. Today, most options on stock are American style ones.

Remark 2.6. *When talking about financial options, there are some expressions used to describe the state of an option.*

- in the money
  An option is called in the money if an immediate exercise would be profitable for the holder. Accordingly, a call is in the money if the asset value S is significantly higher than the strike price K, and a put is in the money if $S < K$.

- at the money
  This term is used if there is no difference relating to payoff in exercising the option or letting it expire worthless, i.e. if the strike price K and the underlying value are about equal, $S = K$.

- out of the money
  An option is called out of the money if exercise would mean a loss in comparison to letting it expire worthless. Consequently, a call is out of the money if the asset value S is well below the strike price K, and a put is out of the money if $S > K$.

2.2.1 Influences on Option Values

The most basic definitions and terms concerning financial options have been introduced in the previous section. We will now define the market model that we are working with in order to examine some of the characteristics of option values. The financial market is very complex, which is why it is necessary to make generalizations when approximating reality in mathematical models. Although the idealizations we are making in our model are often criticized since they are not very realistic, the resulting Market model is still widely used:

Assumptions 2.7. [Market model]

- The market is frictionless.
  This means that
  - *There are no transaction costs.*
  - *The costs for buying and selling an asset are the same.*
  - *There is no difference in the constant risk-free interest rate that applies to any amount of money deposited in or borrowed from a bank.*
  - *The price is not influenced by individual trading.*
  - *Any information is immediately accessible to all parties.*
  - *It is possible to buy or sell any amount of asset at any place and any time $0 \leq t \leq T$. This implies that the amount of assets may take any real number, and that we use a continuous market model.*
- The no-arbitrage-principle is valid.
  The no-arbitrage-principle states that arbitrage, i.e. the opportunity to make instantaneous, risk-free profit does not exist on a frictionless market. That is to say, the greatest risk-free return one can make on any investment is the same as the return if the same amount of money were placed in a bank.
  This assumption is justified by the forces of the market. Suppose there was a possibility of putting together a portfolio which guarantied risk-free profit that exceeded the return of a bank deposit. This possibility of arbitrage would cause sensible investors to withdraw their money from the bank and even to borrow money for further investment in the portfolio. Such a situation cannot exist for long. As more and more people will try to lock in the portfolio, the increase in demand will cause its price to rise, thereby restoring parity.
- The asset price follows a geometric Brownian motion.
  We will learn more about that in Chapter 4.
- The risk free interest rate r and the volatility σ are constant for $0 \leq t \leq T$.
  An asset's volatility represents its likeliness to change its value. It can be seen in the jaggedness of the graph of the asset price against time.
- No dividends are paid.

- Short selling is possible.

Remark 2.8.
Of course there is no such thing as a frictionless market, and in practice arbitrage may exist in smaller forms. However, those opportunities are not significant, and the concept of the no-arbitrage principle is widely accepted for the modeling of financial markets. Thus it is justified to say that if one wants a greater return then one has to take a greater risk.
Arbitrage is a powerful tool in option theory. It can be used to compare risky financial investments with investments that are free of risk, but it is also used in proofs.
The assumptions of r and σ being constant and of no dividends being paid are very restrictive and they do not mirror the reality of the market. Still, we shall work with this simplified model to introduce the concepts of the Monte Carlo simulation. Later on, we can loosen up or even drop those assumptions.

Having defined the conditions for our market model, we can now examine the different parameters that have an influence on the value of an option:

Remark 2.9. [Influences that effect option value]
The value of an option depends on certain parameters. All in all, there are 5 major variables that effect the premium of a call or put. We want to investigate how the price of an option reacts to changes in those parameters:

1. The strike price K
   The higher the strike price is, the more the holder of a call option has to pay on exercise, which decreases the profit and thus the value of the option V. Therefore, raising the strike price causes a fall of the value of the call and respectively a rise in the value of the put.

2. The value of the underlying asset S
   A higher value of the underlying asset increases the chances of the call option to be exercised and thus show a profit. The value of the call increases. Hence a rise in the price of the underlying asset causes the price of the call option to increase while the price of the put option decreases.

3. The time left until the maturity date T
   The closer the option comes to its expiry date T, the less likely it is for the underlying asset's price to change at a significant degree. This lowers the chances of the underlying asset to change its value in a positive way for the holder of the option. Consequently, a longer period of time left to expiry increases the price of the call as well as the put option.

4. The volatility σ
   If the volatility σ of the underlying asset is high, a beneficial movement of the asset becomes more likely. This means that a higher volatility causes both a higher call price and a higher put price.

5. The risk free interest rate r
   While the holder of an option has to pay the premium at the time the contract is closed, he will have to wait for his expected profit until the option's maturity date. This is why the concept of discounting has to be applied here. Suppose the holder invested the money $M(0)$ he has to pay for the premium in a risk free bond instead. In this case, he would expect to receive more money when his bond contract expires, namely $M(t) = M(0)e^{rt}$.
   A rise in the risk free interest rate r causes the discounted strike price Ke^{-rT} to fall, and on account of this the value of the call will rise. The opposite is the case with the put option; a higher interest rate causes a lower put price.

2.2.2 Payoff Functions

While it is quite difficult to calculate the fair value V of a European option for a time $0 \leq t < T$ before the option's expiry date, it is fairly easy to evaluate its value at its maturity date $t = T$. For a European call, there are three possible cases:

1. $S > K$
   The price of the underlying asset is higher than the strike price of the option. Since there are no transaction costs, the holder will make a profit of $V(S,T) = S - K > 0$ (also called cash-settlement) per unit in exercising his option. Thus it is reasonable for him to make use of his right to buy the underlying asset for the strike price K.

2. $S < K$
   The price of the underlying asset is below the strike price. This means that if he chooses to exercise the option, the holder would have to pay a price above the market price. Since this would represent a loss, it is not likely for the holder to do so. He will let the option expire and thus become worthless, $V(S,T) = 0$.

3. $S = K$
   The price of the underlying asset equals the strike price of the option. It does not make any difference whether or not the holder uses his option to buy a unit of the asset, which also renders the option worthless, $V(S,T) = 0$.

We just proved that the value of a call option at expiry date T is

$$V(S,T) = \max\{S_T - K, 0\} =: (S_T - K)^+. \tag{2.1}$$

This function is called the payoff function, and is depicted in Figure 2.1. To evaluate the profit, the premium has to be subtracted from the payoff.

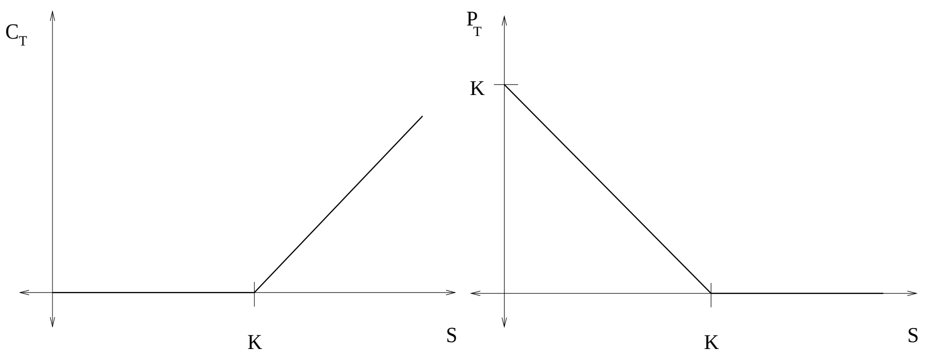


Figure 2.1: Payoff diagrams of a European call (left) and a European put (right).

The same reasoning can be applied to a European put option in order to derive its payoff function. Again, we take a look at the different possible situations:

1. $S \geq K$
   The price of the underlying asset equals or is higher than the strike price of the option. This means that exercising would not be of any advantage for the holder. Therefore, it is reasonable for him to let his option expire worthless, $V(S,T) = 0$.

2. $S < K$
   The price of the underlying asset is below the strike price. The holder of the put will make a profit of $V(A,t) = K - S > 0$ per unit in exercising his option. It therefore makes sense for him to make use of his right to sell the underlying asset for the strike price K.

This leads us to the payoff function of a European put

$$V(S,T) = \max\{K - S_T, 0\} =: (K - S_T)^+ \tag{2.2}$$

which is depicted in Figure 2.1.

2.2.3 Put-Call Parity

So far we have looked at the value of a European put and a European call as two totally different things. However, it can be shown that there exists a relationship between the two, the so called Put-Call-Parity:

Theorem 2.10. [Put-Call-Parity (for European options)]*:*
Vanilla European put and call options satisfy the following equation:

$$S_t + V_P - V_C = Ke^{-r(T-t)} \tag{2.3}$$

Proof. (By arguments of arbitrage)

Take the following portfolio: Buy one share of the underlying asset S and one European Put V_P with strike price K and maturity T, and sell a European Call V_C with the same strike price K and maturity T. Our portfolio now has the value

$$\pi = S + V_P - V_C$$

The value of π at time T is

$$\pi_T = S_T + (K - S_T)^+ - (S_T - K)^+ = \begin{cases} S_T + K - S_T - 0 = K & \text{in case } K > S_T, \\ S_T + 0 - (S_T - K) = K & \text{in case } K < S_T, \\ S_T + 0 + 0 = K & \text{in case } K = S_T. \end{cases}$$

Taking into account that the no-arbitrage-principle is valid, the value of π at the time t must have been $Ke^{-r(T-t)}$, which equals K at time T. □

Remark 2.11. *The put-call parity makes it possible to calculate the value of an European call option if the value of an European put option is known and vice versa. In addition to that, it is also an important tool for deriving upper and lower bounds on options, which will be done in the next section.*
The Put-Call-Parity for options on dividend-paying assets is given by the following formula:

$$P_t - C_t = Ke^{-r(T-t)} - Se^{-\delta(T-t)} \tag{2.4}$$

2.2.4 Upper and Lower Bounds

By definition, the holder of an option is not obligated to exercise his right to buy or sell the underlying asset. In the worst case, he has to let it expire worthless, which puts the lower limit of the value of an option at 0. Consequently, the highest possible loss for the holder is limited by the premium. Other than this simple barrier, the value of an option also satisfies other bounds which are known a priori.

Theorem 2.12. *European options satisfy the following bounds at all times* $0 \leq t \leq T$*:*

1.

$$(S_t - Ke^{-r(T-t)})^+ \leq C_t \leq S_t \tag{2.5}$$

2.
$$(Ke^{-r(T-t)} - S_t)^+ \leq P_t \leq Ke^{-r(T-t)} \tag{2.6}$$

Proof.

1. (a) $C_t \geq 0$: This bound is trivial since an option is a non-obligatory contract, and thus cannot have a negative value.

   (b) $C_t \leq S_t$: The strike price is never negative, and therefore the call option cannot be more valuable than the underlying asset.

   (c) $S_t - Ke^{-r(T-t)} \leq C_t$: Suppose that $S_t - Ke^{-r(T-t)} > C_t$ was true at some time $0 \leq t \leq T$.
   In this case, sell an underlying S, buy a call C and invest $Ke^{-r(T-t)}$ in a risk-free bank account or zero-bond. The cash-flow in this case would be $S_t - C_t - Ke^{-r(T-t)} > 0$. At expiry date T there are two possible cases:

      i. $S_T \leq K$ The call is worthless and thus the total value of the portfolio amounts to $-S_T + Ke^{-r(T-t)}e^{r(T-t)} = K - S_{(T-t)} \geq 0$

      ii. $S_T > K$ The call's value is $S_T - K$, which leads to a total value of the portfolio of $-S_T + S_T - K + K = 0$

   In both cases, the value at expiry date T is non-negative and thus leaves the purchaser of the portfolio with the instant risk-free profit of $S_t - C_t - Ke^{-r(T-t)} > 0$ at the time of purchase t. This violates the no-arbitrage principle, which is why our assumption has to be wrong.

2. The bounds for the put option can be derived directly from those for the call option and the put-call-parity.

□

Similar bounds can be derived for American options:

Theorem 2.13. *American options satisfy the following bounds at all times* $0 \leq t \leq T$*:*

1.
$$(S_t - Ke^{-r(T-t)})^+ \leq C_t \leq S_t \tag{2.7}$$

2.
$$C_A(S_t, t) = C_E(S_t, t) \tag{2.8}$$

3.
$$(Ke^{-r(T-t)} \leq S_t + P_A(S_t, t) - C_A(S_t, t) \leq K \tag{2.9}$$

(This inequality can be seen as the put-call parity for American options.)

4.

$$(Ke^{-r(T-t)} - S_t)^+ \leq P_A(S_t, t) \leq K \tag{2.10}$$

5.

$$P_A(S_t, t) \geq (K - S_t)^+. \tag{2.11}$$

Proof.

1. It is easy to see why this inequality applies to an American as well as to a European call. In the proof of Theorem 2.12, the opportunity of arbitrage was present at time t. If it is possible to make even higher profit than in the case of an European call by early exercise, then do so. If not, wait until maturity. Either way, the owner of the portfolio will make an instant risk-free profit of at least $S_t + C_t - Ke^{-r(T-t)} > 0$.

2. Suppose we make use of our right to exercise the American call at a date $0 \leq t < T$ before maturity. Since this is only reasonable if $S_t > K$, we gain an amount of $S_t - K$. Applying Theorem 2.13(1) we get

   $$C_A(S_t, t) \geq (S_t - Ke^{-r(T-t)})^+ = S_t - Ke^{-r(T-t)} > S_t - K.$$

   This shows that an early exercise is not advisable, since the value of the call exceeds the profit that an early exercise would yield. If we wait until maturity, however, there is no difference between a European and an American option.

3. (a) $(Ke^{-r(T-t)} \leq S_t + P_A(S_t, t) - C_A(S_t, t)$:
   Recalling Theorem 2.13(2) and the fact that an American put is at least as valuable as a European one, we get

   $$C_A - P_A \leq C_E - P_E$$

   Applying the put-call-parity for European options leads to

   $$C_E - P_E = S_t - Ke^{-r(T-t)} \text{ for } 0 \leq t \leq T,$$

   and thus

   $$Ke^{-r(T-t)} \leq S_T - P_A - C_A$$

   holds.

   (b) $S_t + P_A(S_t, t) - C_A(S_t, t) \leq K$:
   Let us take a look at the following portfolio: Suppose there was a time $0 \leq t \leq T$ at which $S_t - K > C_A(S_t, t) - P_A(S_t, t)$. Sell a put $P_A(t)$ and an underlying S_t, buy a call $C_A(t)$ and invest K in some bank account. This amounts to a cash flow of $P_A - C_A + S - K > 0$. Again there are two possibilities at the time of exercise $\widetilde{T} \leq T$:

i. $S_{\widetilde{T}} \leq K$: Theorem 2.13(1) states that the value of the call has to be non-negative at $t = \widetilde{T}$, and thus the portfolio is worth at least $-(K - S_{\widetilde{T}}) - S_{\widetilde{T}} + Ke^{r(\widetilde{T}-t)} = Ke^{r(\widetilde{T}-t)} - K \geq 0.$

ii. $S_{\widetilde{T}} > K$: Again Theorem 2.13(1) can be employed for estimating a lower bound on the value of the call of $S_{\widetilde{T}} - K$. Therefore, the value of the portfolio amounts to at least $S_{\widetilde{T}} - K - S_{\widetilde{T}} + Ke^{r(\widetilde{T}-t)} = Ke^{r(\widetilde{T}-t)} - K \geq 0$

This means that we make an instant profit at time t without running a risk, since the portfolio will have a non-negative value at exercise time $\widetilde{T}$. Thus we have found an opportunity of arbitrage which contradicts our model of the market.

4. Recalling Theorem 2.13 (3), we get

$$Ke^{-r(T-t)} - S_t + C_A \leq P_A \leq K - S_t + C_A$$

(a) $P_A \geq (Ke^{-r(T-t)} - S_t)^+$: Applying Theorem 2.13(1) and Theorem 2.12(1) to the left hand side of this inequality delivers

$$\begin{aligned} P_A \geq Ke^{-r(T-t)} - S_t + C_A = Ke^{-r(T-t)} - S_t + C_E \geq \\ \geq Ke^{-r(T-t)} - S_t + (S_t - Ke^{-r(T-t)})^+ = (Ke^{-r(T-t)} - S_t)^+ \end{aligned}$$

(b) The same proceeding applied to the right hand side of the inequality shows that:

$$P_A \leq K - S_t + C_A = K - S_t + C_E \leq K - S_t + S_t = K$$

5. Again we have to differentiate between two cases:

(a) $K \leq S_t$: Since the value of an option can never be negative, this is trivial.

(b) $K > S$: Suppose there was a time t at which $P_A(S_t, t) < (K - S_t)^+$. This would mean that it was possible to make an instant risk-free profit of $K - S_t - P_A(S_t, t)$ by buying a put option and exercising it on the spot.

$\square$

Figure 2.2 gives a rough outline of the bounds we just derived.

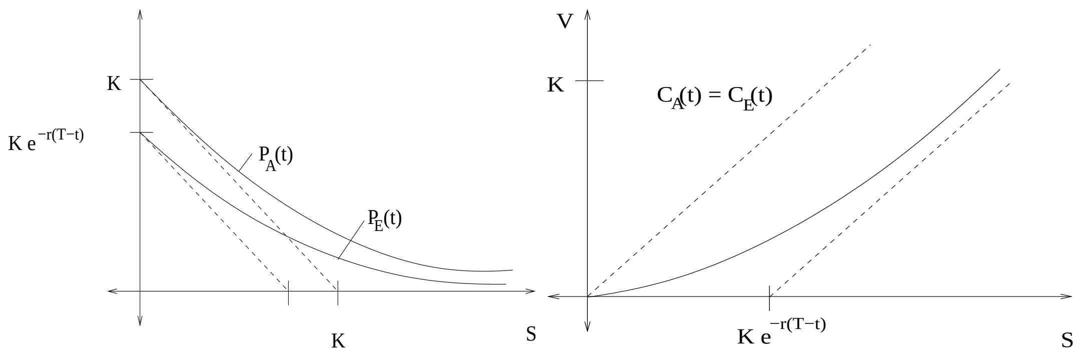


Figure 2.2: Qualitative curves of the prices of European and American options.

2.2.5 Hedging Strategies

As mentioned above, options can be used either for speculating or for reducing risk, which is called hedging. While a call or put option reflects the expectation of the value of the underlying asset to rise or to drop respectively, the combination of both makes it possible to model some more complex expectations on the underlying's behavior. There are a lot of different ways of combining financial options, and it would exceed the frame of this thesis to present them all. We shall, however, present three main groups of hedging strategies and give an example to each of them.

1. **Straddles**
   The simplest form of a straddle is represented by the purchase (long straddle) or selling (short straddle) of both a call and a put on the same underlying asset with the same strike price K and expiry date T.

   Example 2.14. *Buy a put and a call with the same maturity date T and strike price K. The value of this portfolio at time T is denoted by*

   $$\pi_T = P_T + C_T = (K - S_T)^+ + (S_T - K)^+ = \begin{cases} S_T - K & S_T > K, \\ K - S_T & S_T \leq K. \end{cases}$$

   Figure 2.3 shows that the purchase of a long straddle is reasonable if one expects major changes of the value of the underlying. Only a significant difference between the strike price K and the value of the asset S_T makes this portfolio profitable. Thereby it does not matter whether the value of the underlying asset drops or rises.

2. **Strangles / Combinations**
   Both strategies contain either buying or selling a call and a put with the same maturity date but with different exercise prices.
   Thereby, the portfolio is called a combination if the options are out-of-the money at the time of initiating the contracts. Respectively, the portfolio is called a strangle if both options are in-the-money when the contracts are sealed.

Example 2.15. *Buy a call with maturity T and strike price K_2 and buy a put with the same maturity T but with a strike price $K_1 < K_2$. The two options form a portfolio with the value*

$$\pi_T = (K_1 - S_T)^+ + (S_T - K_2)^+$$

which can be also be seen in Figure 2.3.
Similarly to the straddle, investors in a strangle expect the value of the underlying to change significantly. The opportunities of drawing a profit with a strangle are smaller, however, which makes that portfolio less expensive.

3. **Spreads**
Generally speaking, a spread is the combination of two options of the same type, i.e either two calls or two puts, on the same underlying with different maturity dates or different strike prices.

Example 2.16. *Imagine the following portfolio: one long call K_1, one long put K_2, one short call K and one short put K. All four options have the same maturity date T. The value of this portfolio at time T is*

$$\pi_T = C_1 + P_1 - C_2 - P_2 = (S_T - K_1)^+ + (K_2 - S_T)^+ - (S_T - K)^+ - (K - S_T)^+$$

(compare Figure 2.4)
The buyer of a butterfly spread expects the value of the underlying asset to stick closely to the exercise price K

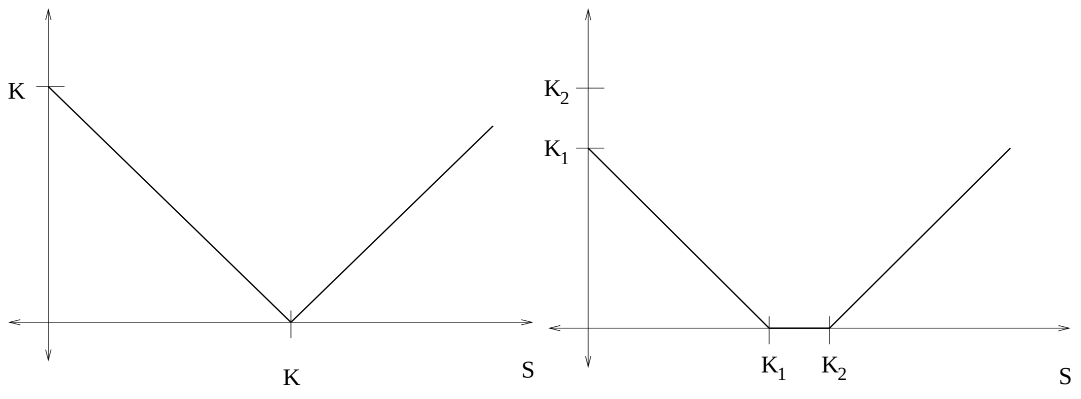


Figure 2.3: Payoff diagrams of a long straddle (left) and a long strangle (right)

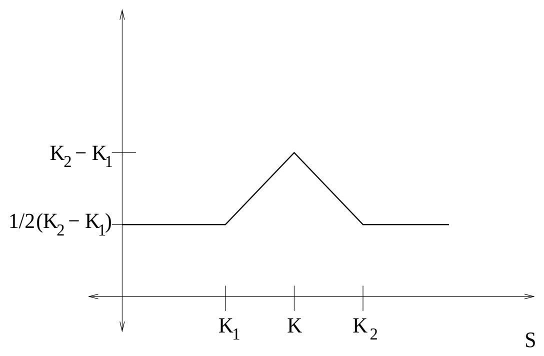


Figure 2.4: Payoff function of a butterfly spread

2.2.6 Exotic Options

We have now covered the basics on the most simple versions of American or European options, the so called vanilla ones. There are, however, a lot of other varieties of options on the market. The payoff of those so called exotic option is different from the vanilla call or put. Exotic options can roughly be split into two categories. The term path-dependent is used if the payoff is dependent on the history of an asset price before maturity, otherwise they are called path-independent options. There is a huge variety of exotic options on the market, and we shall only give a brief sketch of the two most common ones for each category here.

Path-independent options

- **Binary Options**
  The option becomes worthless if the value of the underlying is above (a cash-or-nothing put) or below (a cash-or-nothing call) a predetermined barrier K. In case the option is not worthless, the fixed payoff B is independent of the underlying asset value. A binary call for example has the payoff function

$$C_T = \begin{cases} B & S_T > K, \\ 0 & S_T \leq K. \end{cases} \tag{2.12}$$

  It is easy to see why binary options are path-independent. The underlying asset value is only needed to determine whether or not there is a payoff. The history of the asset is of no importance at all.

- **Compound Options**
  This is simply an option on an option. It gives the holder the right to buy or sell another option with maturity $T' > T$ at strike price K. The four basic types of compound options are call-on-call, call-on-put, put-on-put and put-on-call. The payoff function of the latter would be

$$P_T = (K - C_T)^+ \tag{2.13}$$

Again, the value of the underlying before maturity has no influence on the payoff.

Path-dependent options

- **Barrier Options**
  These options either come into existence or become worthless if the underlying crosses a certain barrier B before maturity. The four basic types are

  - up-and-in (the option is activated if the barrier is crossed from below)
  - down-and-in (the option is activated if the barrier is crossed from above)
  - up-and-out (the option becomes worthless if the barrier is crossed from below)
  - down-and-out (the option becomes worthless if the barrier is crossed from above)

  If the option is active at the time of exercise, the payoff function is that of a European option. The payoff function of a down-and-out-call for example would be

$$C_T = \begin{cases} (S_T - K)^+ & \min\limits_{0\leq t\leq T} S_t \geq B, \\ 0 & \min\limits_{0\leq t\leq T} S_t < B. \end{cases} \tag{2.14}$$

  Since the value of the asset can cross the barrier at any time up to maturity, the payoff is dependent on the underlying's behavior not only at exercise. That is why barrier options belong to the group of path-dependent options.

- **Asian Options**
  Whereas the focus of barrier options lies on extreme values of the asset, the payoff of Asian options depends on some average of the asset price over the lifetime of the contract. There are many different kinds of Asian options, depending on what kind of average is used for the calculation of the payoff. The European average rate call uses the continuous average, for example, which results in the payoff function

$$C_T = \left(\frac{1}{T}\int_0^T S_\tau d\tau - K\right)^+ \tag{2.15}$$

  The dependence on the asset behavior previous to maturity is obvious.

There are a lot more different kinds of exotic options on the market, and their exercise mode can either be European or American style. All the types mentioned above are single-asset options, i.e. there is only one underlying asset. Options on multiple underlying assets would be called multi-asset options.

Chapter 3

Essentials of Stochastics

This chapter introduces some basic tools and notations of statistics and probability theory. They set the groundwork for stochastic processes and stochastic differential equations (SDEs) which are very important for deriving a model for asset price dynamics and which we will deal with in Chapter 4.
Let us begin with the concept of the probability space:

Definition 3.1. *A probability space is an ordered triple $(\Omega, \mathcal{A}, P)$ where*

- *Ω is an arbitrary non-empty sample space of sample points ω,*
- *$\mathcal{A}$ is a σ-algebra whose elements, the so called events $A \in \mathcal{A}$, are subsets of Ω, and*
- *P is a probability measure on $\mathcal{A}$ assigning a probability $P(A) \in [0, 1]$ to each event $A \in \mathcal{A}$.*

Thereby, the definition of the probability measure P is given by

Definition 3.2. *A probability measure P on a measurable space $(\Omega, \mathcal{A})$ is a non-negative valued set function on $\mathcal{A}$ satisfying*

$$P(\emptyset) = 0, \tag{3.1}$$

$$P(\bigcup_{n=1}^{\infty} A_n) = \sum_{n=1}^{\infty} P(A_n) \tag{3.2}$$

for any sequence $A_1, A_2, ..., A_n, ... \in \mathcal{A}$ with $A_i \cap A_j = \emptyset$, $i \neq j$, $i, j = 1, 2, 3, ...$, and for which

$$P(\Omega) = 1 \tag{3.3}$$

holds.

The σ-algebra mentioned in Definition 3.1 which describes $\mathcal{A}$ in a probability space is defined as follows:

Definition 3.3. *A collection A of subsets of Ω is called σ-algebra if*

$$\Omega \in \mathcal{A} \tag{3.4}$$

$$A \in \mathcal{A} \Rightarrow A^c \in \mathcal{A} \tag{3.5}$$

$$A_1, A_2, ..., A_n, ... \in \mathcal{A} \Rightarrow \bigcup_{n=1}^{\infty} A_n \in \mathcal{A} \tag{3.6}$$

We can see from this definition that there is a certain freedom in the choice of the subsets of Ω. Basically, $\mathcal{A}$ has to be closed under the set operations of complementation and countable unions and it has to contain the sample space Ω and the empty set $\emptyset$. The choice of subsets of a sample space depends on the events that are to be modeled.

Remark 3.4. *In our financial scenario, $\mathcal{A}$ is made up in a way such that it represents all events that can be observed on the market.*

A collection $\mathcal{C}$ of subsets of Ω does not necessarily have to be a σ-algebra. Therefore, we need to find a σ-algebra containing $\mathcal{C}$. $P(\Omega)$ would be a simple solution for this problem, but we want to find another σ-algebra $\mathcal{A}(\mathcal{C})$ containing $\mathcal{C}$, namely the smallest one, called the σ-algebra generated by $\mathcal{C}$. In the special case of $\Omega = \mathbb{R}^n$ being the n-dimensional Euklidian space with

$$|x-y| = \left(\sum_{k=1}^{n} |x_k - y_k|^2\right)^{1/2},$$

we choose our σ-algebra to be the σ-algebra $\mathcal{B}$ of Borel subsets generated by the n-dimensional intervals $a_k \leq x_k \leq b_k, k = 1, 2, ..., n$. If Ω is a subset of $\mathbb{R}^n$, we shall choose $A = \mathcal{B}^n(\Omega) = \{A = B \cap \Omega : B \in \mathcal{B}^n\}$.
We conclude this introduction by listing some important terms used when talking about events:

- Nonempty subsets $A \in \mathcal{A}$ with probability $P(A) = 0$ are called null events. $P(\emptyset) = 0$, is not a null event, however.

- The sample space Ω for which $P(\Omega) = 1$ holds is called the sure event.

- All other events $A \in \mathcal{A}$ with $P(A) = 1$ are said to occur almost surely (a.s.) or with probability 1 (w.p.1).

Remark 3.5. *Note that there is a difference between the sure event $P(\Omega) = 1$ and events $B \in \mathbb{R}$ for which $P(B) = 1$ that are called almost sure events.*
While the sure event will occur with absolute certainty, this is not true for the almost sure event, although the probability for both events to happen is 1.

3.1 Random Variables

The concept of the probability space introduced above sets the ground for examining probabilistic experiments. However, operating on it can be quite difficult since it is not possible to work with real-valued numbers. This is why we will introduce the concept of the random variable X here. The name may be misleading, X is not an actual variable, it is rather a real-valued function $X : \Omega \to \mathbb{R}$. A random variable provides us with information about the outcome of a probabilistic experiment, assigning each element of Ω with a real valued number. Thus we have the advantage of being able to use all techniques of analysis when dealing with probabilities. The exact definition of a random variable is given below.

Definition 3.6. *A real-valued function X on Ω is called random variable if the sets*

$$\{X \leq x\} := \{\omega \in \Omega : X(\omega) \leq x\} = X^{-1}((-\infty, x])$$

are measurable for all $x \in \mathbb{R}$. That is, if $\{X \leq x\} \in \mathcal{A}$ for each $x \in \mathbb{R}$.

Remark 3.7. *$X(\omega)$ is called a realization of X and can be seen as the result of the random experiment that has been evaluated by X.*

The behavior of the realizations of $X(\omega)$ is best described by the probability distribution P_X of X, which indicates for a set $B \subset \mathbb{R}$ the probability with which a realization $X(\omega)$ lies in B:

Definition 3.8. *The function P_X defined for all $B \in \mathcal{B}$ by*

$$P_X(B) = P(\{\omega \in \Omega : X(\omega) \in B\}) \tag{3.7}$$

is called the distribution of the random variable X.

The ordered triple $(\mathbb{R}, \mathcal{B}, P_X)$ is a probability space containing all of the essential information on the random variable X. However, P_X is a set function and we therefore loose all the advantages we gained by introducing the concept of random variables. This disadvantage can be overcome by introducing a formula for P_X, the distribution function F_X. Thereby it not necessary to know the exact function $X : \Omega \to \mathbb{R}$ or the probability measure P on Ω, although they do, of course, exist. P.E. Kloeden and E. Platen gave the following definition of the distribution in their book [15]:

Definition 3.9. *Let $X : \Omega \to \mathbb{R}$ be a random variable on a probability space $(\Omega, \mathcal{A}, P)$. The distribution function $F_X : \mathbb{R} \to [0, 1]$ of X is defined by the probability P that $X \leq x$,*

$$F_X(x) = P_X(X \leq x) = P(\omega \in \Omega : X(\omega) \leq x) \tag{3.8}$$

This function indicates the probability with which X takes values smaller than a fixed point $x \in \mathbb{R}$ and satisfies the following properties:

- F_X is nondecreasing, right-continuous and satisfies the limits

$$\lim_{x\to+\infty} F_X(x) = 1 \text{ and } \lim_{x\to-\infty} F_X(x) = 0$$

- In case X is a discrete random variable, that is if it only takes a finite or countably infinite number of distinct values, [22] states that the distribution function is given by a step function with jumps ω_i with the height of p_i, $i \in \mathbb{N}_0$:

$$F = \begin{cases} 0: & x < x_0, \\ \sum_{i=0}^{n} p_i: & x_n \leq x < x_{n+1}. \end{cases}$$

  where p_i is the probability for X to take the value x_1. F_X is a step-function with steps the height of p_n at $x = x_n$.

- If X is a continuous random variable, i.e. a random variable taking all possible values in $\mathbb{R}$ with $P(\{\omega \in \Omega : X(\omega) = x\}) = 0$ for all $x \in \mathbb{R}$, its distribution function F is differentiable w.p.1, and the deviation is called density function.

Definition 3.10. *Let $X : \Omega \to \mathbb{R}$ be a random variable on a probability space $(\Omega, \mathcal{A}, P)$ with distribution function F_X. Then the function $f : \mathbb{R} \to \mathbb{R}$ with*

$$F_X(x) = \int_{-\infty}^{x} f(s)ds$$

is called density function if it exists. In case f is continuous, $F' = f$ holds for each $x \in \mathbb{R}$.

This function can be seen as the equivalent to the probability $P(x)$ for sample points in the discrete case in the sense that the distribution function can be represented by

$$F_X(a) = \int_{-\infty}^{a} f(x)dx.$$

A way of graphically approximating the density function is described in [11]:
Divide the set of real numbers into equidistant intervals $I_j = [jh, (j+1)h], j \in \mathbb{Z}, h > 0$ and to count how many times n_j the realizations $X_1, X_2, ..., X_N$ lie in the interval n_j. Plotting the amounts $\frac{n_j}{N}$ as a histogram then results in an approximation of the graph of the density function f.

3.1.1 Integration and moments

The distributions of random variables are important for deriving more information on X, for example the average value or the way in which the values are spread about this average value. These values are called the mean or expected value of X, notated as $E[X]$ and the variance of X written as $Var[X]$ respectively. Let us start with $E[X]$. It can be seen as the average of all possible realizations $X[\omega]$ of X with respect to their likelihood of occurrence. The Law of Large Numbers states that the mean can be approximated by

$$E[X] := \lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} X(\omega_i). \tag{3.9}$$

For a discrete random variable the mean can be calculated by employing its density function:

$$E[X] := \sum_{x_i \in X(\Omega)} x_i f_i(x) \tag{3.10}$$

For random variables with absolutely continuous distributions, the expectation has to be expressed in terms of integrals:

$$E[X] := \int_{-\infty}^{\infty} x f(x) dx. \tag{3.11}$$

Obviously, this integral is not a Riemann integral. Since we still want to work with that integral, we need to introduce the definition of the integral due to Lebesgue here.

Definition 3.11. *Let $\phi = \sum\limits_{i=1}^{m} a_i I_A$, $\bigcup\limits_{i=1}^{m}$, $A_i \cap A_j = \emptyset, i \neq j$, $A_i, A_j \in \mathcal{A}$ be a so called simple function or step function on a probability space $(\Omega, \mathcal{A}, P)$, where I_A is the indicator function of a set $A \subset \Omega$:*

$$I_A(\omega) = \begin{cases} 1 & \textit{if}\ \omega \in A, \\ 0 & \textit{if}\ \omega \notin A. \end{cases}$$

The Lebesgue integral of such a function ϕ is then defined as

$$\int_\Omega \phi(\omega) dP(\omega) = \int_\Omega \phi dP = \sum_{i=1}^{m} a_i P(A_i).$$

The definition of the Lebesgue integral of a real-valued measurable function $F : \Omega \to \mathbb{R}$ is then built up step by step from this definition by first extending it on positive-valued measurable function and then dividing it into its positive and negative parts.

Remark 3.12. *When looking at a particular random variable, one often denotes the expected value by μ: $E[X] = \mu$. Some other interesting properties of $E[X]$ are listed below:*

- *For integrable functions $g : \mathbb{R} \to \mathbb{R}$*

$$E[g(X)] = \int_{-\infty}^{\infty} g(x)f(x)dx$$

  holds.

- *The expected value is linear in X, which means that for two random variables X, Y (discrete or continuous) the equality $E[\alpha X + \beta Y] = \alpha E[X] + \beta E[Y]$ is true for $\alpha, \beta \in \mathbb{R}$.*

The second quantity we want to examine is the variance of X, $Var[X]$. Its square root, the so called standard deviation, indicates the average distance the individual realizations of X take from the mean. The variance of a random variable X is defined by

$$Var[X] = E[(X - E[X])^2]. \tag{3.12}$$

Remark 3.13. *Another expression often used for the variance would be $\sigma^2 = Var(X)$, where σ is the standard deviation of X. We want to list some interesting properties of the variance:*

- $Var[X] = E[X^2] - E^2[X]$
- $Var[\alpha X + \beta] = \alpha^2 Var[X]$.

Note that unlike the expectation, the variance operator is not linear.

Theorem 3.14. *Let X be a nonnegative random variable. Then the Markov inequality*

$$P(\{\omega : X(\omega) \geq a\}) \leq \frac{1}{a}E[X] \tag{3.13}$$

holds for all $a > 0$.

For a proof we refer to N. Schmitz's book [22] p 60.

Remark 3.15. *As a special case of the Markov inequality we get the Chebyshev inequality:*
Let X be a random variable with expectation $a = E[X]$ and variance $\sigma^2 = Var[X]$. Then

$$P(\{\omega : |X - a| \geq \epsilon) \leq \frac{\sigma^2}{\epsilon^2} \tag{3.14}$$

for all $\epsilon > 0$.

Remark 3.16. *We will need the Chebyshev inequality later when we talk about confidence intervals as well as for proving the convergence behavior of the Monte Carlo method.*

Another inequality that will be needed later is the following one:

Theorem 3.17. *Let $(\Omega, \mathcal{A}, P)$ be a probability space and $J \subset \mathbb{R}$ some interval. Then the Jensen Inequality*

$$\int g \circ X dP \leq g\left(\int X dP\right) \tag{3.15}$$

holds for a random variable $X : \Omega \to J$ and a concave function $g : J \to \mathbb{R}$ such that $g \circ X$ are integrable.

3.1.2 Distributions

Recalling Definition 3.9, we will now examine some different ways in which random variables can be distributed:

- **Uniform Distribution**
  A random variable whose probability of being in a given subinterval of $[a, b]$ is proportional to the length of $[a, b]$ is said to be uniformly distributed on that subinterval $[a, b]$. Such a random variable is denoted by $X \sim U(a, b)$ and its density function is

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

  The mean of the uniform distribution is given by

  $$E[X] = \int_a^b \frac{x}{b-a} dx = \frac{b^2 - a^2}{2(b-a)} = \frac{b+a}{2}, \tag{3.16}$$

  and its variance is

  $$Var[X] = E[X^2] - E^2[X] = \int_a^b \frac{x^2}{b-a} dx - \left(\frac{a+b}{2}\right) = \frac{(b-a)^2}{12} \tag{3.17}$$

  The most common form is the uniform distribution over the interval $[0, 1]$ with $E[X] = 1/2$ and $Var[X] = 1/12$

- **Gaussian Distribution / Normal Distribution**
  A random variable denoted by $X \sim N(\mu, \sigma^2)$ is normally distributed with expectation μ and variance σ^2. Its density function is given by

  $$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}.$$

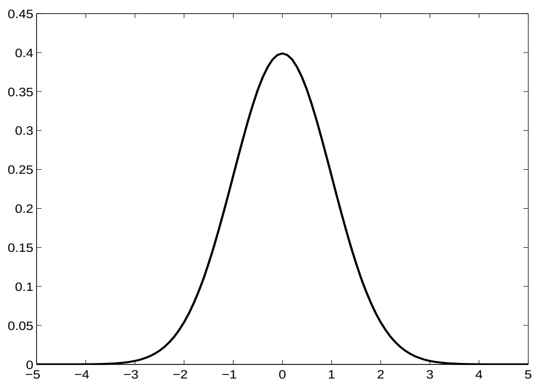


Figure 3.1: Graph of the density function of the Standard Gaussian Distribution

Unlike a normally distributed random variable, a Gaussian random variable may take any real value. The graph of this function is bell-shaped and centers around the mean value $x = \mu$. It is stretched or compressed according to the magnitude of the variance σ^2.

Remark 3.18. *There is no closed-form expression for the distribution function of the normal distribution, but it is possible to approximate it numerically.*

- **Standard Gaussian Distribution / Standard Normal Distribution**
  This is the simplified version of a normally distributed random variable X where the expectation $\mu = 0$ and the variance $\sigma^2 = 1$. Standard Gaussian random variables $X \sim N(0, 1)$ are also denoted by Z and just like the Gaussian distribution, their density function
$$f(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2}$$
  has a bell-shaped graph which is symmetric about $x = \mu = 0$ as shown in Figure 3.1.

- **Lognormal Distribution**
  A random variable X is lognormally distributed if $\ln X$ is normally distributed. It is denoted by $\Lambda(\mu, \sigma^2)$ and its density function is given by
$$f(x) = \begin{cases} \frac{1}{x\sqrt{2\pi\sigma^2}} e^{\left(-\frac{(\ln(x)-\mu)^2}{2\sigma^2}\right)} & x > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{3.18}$$
  The relationships between the parameters of a lognormal and a normal distribution are given by

$$\begin{aligned} E[X] &= e^{\mu+\sigma^2/2} && (3.19)\\ E[\ln X] &= \mu && (3.20)\\ Var[X] &= e^{2\mu+\sigma^2}(e^{\sigma^2}-1) && (3.21)\\ Var[\ln X] &= \sigma^2 && (3.22) \end{aligned}$$

Remark 3.19. *If X is a Gaussian random variable with parameters μ and σ^2 then $\alpha X + \beta$ is normally distributed with parameters $\alpha\mu + \beta$ and $\alpha^2 + \sigma^2$. It is therefore possible to transform a normally distributed random variable into a standard normally distributed one:*
For a random variable $X \sim N(\mu, \sigma)$

$$Z = \frac{X - \mu}{\sigma}$$

satisfies the standard Gaussian distribution and the transformation

$$X = \sigma Z + \mu$$

converts a $N(0, 1)$ distributed random variable into a Gaussian random variable.

3.1.3 Multiple Random Variables

So far we have only looked at one random variable at a time. There are, however, cases in which there might be a large number of random variables defined on the same probability space. An example for that would be the components of a vector-valued random variable $X : \Omega \to \mathbb{R}^2$, a so called random vector. It is possible to define a distribution function for such random variables which is then called the joint distribution function:

Definition 3.20. *Let $X_1, X_2, ..., X_n$ be random variables on the same probability space. Then the function $F_{X_1,X_2,...X_n} : \mathbb{R}^n \to [0, 1]$ defined by*

$$F_{X_1,X_2,...,X_n}(x_1, x_2, ..., x_n) = P(\{\omega \in \Omega : X_i \leq x_i, i = 1, 2, ..., n\}) \tag{3.23}$$

is called joint distribution function.

The joint distribution function for n random variables satisfies certain properties similar to those of the distribution function for a single random variable. We will illustrate them here for the case $n = 2$:

- $\lim_{x_i \to -\infty} F_{X_1X_2}(x_1, x_2) = 0$ and $\lim_{x_i \to \infty} F_{X_1X_2}(x_1, x_2) = 1$

- $F_{X_1X_2}$ is nondecreasing and right-continuous in x_1 and x_2.
- $F_{X_1X_2}(x_1,x_2) = F_{X_2X_1}(x_2,x_1)$

  and
- for $i_1 = 1$ or 2 and $i_2 = 1$ or 2 the marginal distribution $F_{X_{i_1}}$ satisfies

  $F_{X_{i_1}} = \lim_{x_{i_2}\to\infty} F_{X_{i_1}X_{i_2}}(x_{i_1}x_{i_2})$.

Remark 3.21. *The request that the random variables should all be defined on the same probability space is not as restrictive as it might seem at first glance. It is possible to modify them such that they have a common probability space.*

For continuous random variables the function F is absolutely continuous and therefore almost everywhere differentiable. This means that for the density function $f : \mathbb{R}^n \to \mathbb{R}^+$

$$\frac{\partial^n F}{\partial x_1 \partial x_2 ... \partial x_n} = f(x_1, x_2, ..., x_n)$$

holds almost everywhere.
In case the density function is given by

$$f(x_1,x_2) = \frac{\sqrt{detC}}{2\pi} e^{\left(-\frac{1}{2}\sum\limits_{i,j=1}^{2} c^{i,j}(x_i-\mu_i)(x_j-\mu_j)\right)}$$

for some 2×2 positive definite and symmetric matrix $C = [c^{i,j}]$ with determinant $\det C$, the random variables are called jointly Gaussian with mean vector $\mu = (\mu_1, \mu_2) \in \mathbb{R}^2$ and covariance matrix C^{-1}. Thereby, the ij-th component of C^{-1} is $E[(X_i - \mu_i)(X_j - \mu_j)]$ for $i, j = 1, 2$.
Another important concept needed when working with multiple random variables on one probability space is the so called stochastic independence.

Definition 3.22. *Let $(\Sigma, \mathcal{A}, P)$ be a probability space. Two random variables X and Y are called (stochastically) independent if the two events $(X \leq x)$ and $(Y \leq y)$are independent, that is if*

$$F(x,y) = P(X \leq x, Y \leq y) = P(X \leq x)P(Y \leq y) = F_X(x)F_Y(y).$$

Or more generally, the random variables $X_1, X_2, ..., X_n$ are called (stochastically) independent if

$$P(\{X_1 \leq x_1\} \cap \{X_2 \leq x_2 \cap ... \cap \{X_n \leq x_n\}) = P(\{X_1 \leq x_1\})...P(\{X_n \leq x_n\}).$$

Definition 3.23. *Two random variables on the same probability space are thus called stochastically independent if the realization of one random variable X does not contain any information about the realization of Y and vice versa.*

For independent random variables X and Y the equations

$$\begin{aligned} E(XY) &= E(X)E(Y) \text{ and} \quad (3.24)\\ Var(X+Y) &= Var(X)+Var(Y) \quad (3.25)\end{aligned}$$

hold. The concept of independence will be important in the following, when we will deal with sequences of random variables that are independent and identically distributed, short i.i.d. One major property of a set of i.i.d random variables $\{X_1, X_2, ..., X_m\}$ is the fact that they are pairwise independent and therefore

$$E[X_iX_j] = E[X_i]E[X_j], \text{ for } i \neq j. \quad (3.26)$$

Not all random variables are independent however, so it will be useful to have a measure of the degree of independence, called covariance:

Definition 3.24. *Let X and Y be two random variables for which $E[X]$, $E[Y]$ and $E[XY]$ exist. Then*

$$Cov[X,Y] := E[(X - E[X])(Y - E[Y])]) = E[XY] - E[X]E[Y]$$

is called the covariance of X and Y.

Remark 3.25. *If X and Y are stochastically independent, their covariance is zero. The opposite is not always true however, thus the covariance is thus just a measure of dependence and not of independence. Other useful properties of the covariance are*

$$\begin{aligned} Cov[X,X] &= Var[X]\\ Cov[X,Y] &= Cov[Y,X]\\ Cov[\alpha X,Y] &= \alpha Cov[X,Y]\\ Cov[X,Y+Z] &= Cov[X,Y]+Cov[X,Z]\\ Var(X \pm Y) &= Var[X]+Var[Y] \pm 2Cov[X,Y]. \quad (3.27)\end{aligned}$$

Therefore

$$Cov[X,Y] \leq \frac{1}{2}(Var[X]+Var[Y]) \quad (3.28)$$

can be directly derived from Equation (3.27) and the positivity of variances.

If $Cov[X,Y] > 0$ then X tends to be large if Y is large and it tends to be small when Y is.

The last property implies that the value of the covariance depends on the magnitude of X and Y. We want to introduce some more terminology in this context:

Definition 3.26. *Let X and Y be two random variables for which $E[X]$, $E[Y]$ and $Cov[X,Y]$ exist.*
We then call X and Y

- *positively correlated if $Cov(X,Y) > 0$,*
- *negatively correlated if $Cov(X,Y) < 0$ and*
- *uncorrelated of $Cov(X,Y) = 0$.*

Another important tool when working with dependence is conditional expectation. With this operator we can investigate how the distribution of X can be influenced by an event like $(Y = y_j)$.
The conditional expectation is defined as

$$\begin{aligned} E[X|Y = y_j] &= \sum_i x_i P\{X = x_i | Y = y_j\} = \\ &= \frac{\sum_i x_i P\{X = x_i | Y = y_j\}}{P\{Y = y_j\}} \end{aligned} \tag{3.29}$$

or

$$E[X|Y = y_j] = \frac{\int_{-\infty}^{\infty} x f(x,y) dx}{\int_{-\infty}^{\infty} f(x,y) dx}. \tag{3.30}$$

for discrete and continuous random variables respectively.
Similarly conditional variance is defined by

$$Var[X|Y] = E[(X - E[X|Y])^2 | Y]. \tag{3.31}$$

3.1.4 Confidence Intervals

When the expectation is approximated by the sample mean, it is useful to have an indicator of the accuracy of this approximation. This is where the confidence interval comes into play.
We say that $[a, b]$ is a 95%-confidence interval for X if

$$P(a \leq X \leq b) = 0.95. \tag{3.32}$$

We are interested in a standard normal random variable X which has a distribution that is symmetric around 0. We therefore can rewrite Equation (3.32) as

$$P(|X| \leq b) = 0.95 \tag{3.33}$$

with $a = -b$.
Recalling Definitions 3.9 and 3.10, this probability can be expressed as the distribution function of the random variable X:

$$P(|X| \leq b) = \int_{-b}^{b} f(x)dx. \tag{3.34}$$

Remark 3.27. *As we already mentioned in Remark 3.18 there is no closed-form expression for the distribution function of the standard normal distribution. It is therefore necessary to use numerical methods for computing the inverse of the distribution function in order to derive a value for b.*

Remark 3.28. *A 95% confidence interval for $X \sim N(0,1)$ is given by*

$$P(|X| \leq 1.96) = 0.95. \tag{3.35}$$

Remark 3.29. *Remark 3.19 shows that in case X is a normal random variable $X \sim N(\mu, \sigma^2)$, Y is a standard normal random variable with $Y = \frac{X-\mu}{\sigma}$. Applying this to Equation (3.35), we get*

$$P(\mu - 1.96\sigma \leq X \leq \mu + 1.96\sigma) = 0.95. \tag{3.36}$$

Hence $[\mu - 1.96\sigma, \mu + 1.96\sigma]$ is a 95% confidence interval for a standard normally distributed random variable X.

3.2 Convergence and approximation

In this section, we want to introduce some important terminology concerning convergence of sequences of random variables.
Let us start with the two fundamental theorems of probability, the Law of Large Numbers which we already mentioned when talking about expectation, and the Central Limit Theorem (For a proof of both theorems we refer to N. Schmitz's book [22]):

Theorem 3.30. *A sequence of random variables X_n is said to satisfy the strong law of large numbers if*

$$\frac{1}{n}\sum_{i=1}^{n}(X_i - E[X_i]) \tag{3.37}$$

converges to zero w.p.1 $n \to \infty$, that is to say

$$P(\lim_{n\to\infty} \frac{S_n}{n} = \mu) = 1 \; with S_n = \sum_{i=1}^{n} X_i. \tag{3.38}$$

Remark 3.31. *The Law of Large Numbers says that computing the average of large enough number of random variables gives a good approximation of their expectation, as already stated in Equation (3.9).*

Theorem 3.32. *Let $(X_n)_{n\in\mathbb{N}}$ be a sequence of independent and identically distributed (i.i.d.) random variables on $(\Sigma, \mathcal{A}, P)$ with variance $\sigma^2 = E(X_n - \mu)^2 > 0$ and expectation $\mu = E(X_n)$. Then for S_n defined as in Theorem 3.30*

$$\lim_{n\to\infty} P(\frac{S_n - n\mu}{\sigma\sqrt{n}} \leq x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} e^{-\frac{1}{2}z^2} dz$$

for all $x \in \mathbb{R}$.

Remark 3.33. *This theorem is known as Central Limit Theorem. It says that the sequence of normalized random variables*

$$Z_n = \frac{(S_n - n\mu)}{\sigma\sqrt{n}}$$

with $E(Z_n) = 0$ and $Var(Z_n) = 1$ converges in distribution to a standard Gaussian random variable Z. This property will be of importance in later chapters.

Both theorems make statements about the convergence behavior of a sequence of random variables. We now want to give a proper definition of such a sequence:

Definition 3.34. *A collection of random variables on a common probability space $(\Omega, \mathcal{A}, P)$ is called stochastic process $X = \{X_t, t \in [0, \infty)\}$. The index t is usually interpreted as time. For a fixed sample point $\omega \in \Omega$, the function $t \mapsto X_t(\omega)$ is called the sample path or realization of the process X associated with ω.*

Remark 3.35. *A particular kind of a stochastic process, the Wiener process will be of importance in our mathematical model that describes the dynamics of the price of an asset. It will be necessary to simulate such a stochastic process on a computer, which is why we are interested in examining the quality of approximations of stochastic processes.*

There exists a large number of different kinds of convergence, and we will only introduce the most common and important ones. In general, there are two kinds of approximations:

- A strong approximation is an approximation of paths. Thereby it is possible that not all paths are approximated in the best possible way, but the error is at least on average small. This can be seen as the generalized version of a deterministic approximation.

- There are cases, however, when one is not interested in the trajectories (paths) of X, but it suffices to know its moments, like the mean or variance of X. This information is provided by the weak convergence, which is numerically cheaper.

Definition 3.36. *Let $\widetilde{X_i}$ be a sequence of stochastic processes.*

(i) A strong approximation of X at time T with respect to the function $g : \mathbb{R} \to \mathbb{R}$ satisfies

$$\lim_{i\to\infty} E[|g(X) - g(\widetilde{X_i})|] = 0.$$

(ii) A weak approximation of X with respect to g fulfills

$$\lim_{i\to\infty} |E[g(X)] - E[g(\widetilde{X_i})]| = 0.$$

As mentioned before, strong approximations are numerically more expensive than weaker ones.

The strong approximation is the one that is of interest in this thesis. We therefore want to take a closer look at that kind of convergence now. Two terms describing strong convergence are widely used, which is why we want to introduce them here:

- Convergence with probability one ($w.p.1$) / almost sure convergence:
  A sequence of random variables X_N is said to converge against the random variable X with probability one if

  $$P(\{\omega \in \Omega : \lim_{n\to\infty} |X_n(\omega) - X(\omega)| = 0\}) = 1 \tag{3.39}$$

  holds.

- Mean-square convergence:
  A sequence of random variables X_N is said to converge in the square mean to the random variable X if $E(X_n^2) < \infty$ for $n = 1, 2, ..., E(X^2) < \infty$ and

  $$\lim_{n\to\infty} E(|X_N - X|^2) = 0. \tag{3.40}$$

For further reading on probability theory and statistics we recommend the books [15] and [22].

Having defined all necessary tools, we are now able to mathematically model the option-pricing problem.

Chapter 4

Asset Price Model

In the previous chapter we introduced the concept of a stochastic process which are the solutions to stochastic differential equations (SDEs). We now want to take a look at a special stochastic process which will play an important role in the SDE that describes the price movement of an asset, the Wiener process. After giving a definition of this special stochastic process, we will take a closer look at some of its properties before we introduce some important tools of stochastic calculus so that we can derive a mathematical model for asset price movements.
Among others we want to mention the books [6], [12], [21] and [23] for references on this topic.

4.1 The Wiener Process

Definition 4.1. *A real-valued time continuous stochastic process $\{W_t\}_{t\geq 0}$ is called a standard Wiener process (or Brownian motion) if for some real constant σ*

1. $W_0 = 0\ w.p.1.$
2. *The increments $W_{t_1} - W_{t_0}$ are standard Gaussian random variables with mean $E[W_{t_1} - W_{t_0}] = 0$ and variance $Var[W_{t_1} - W_{t_0}] = t_1 - t_0$, that is that is $W_t \sim N(0, t)$ for $t_1 \geq t_0 \geq 0$.*
3. *The increments $W_{t_1} - W_{t_0}$ and $W_{s_1} - W_{s_0}$ are independent random variables for $s_1 \geq s_0 \geq t_1 \geq t_0 \geq 0$.*

4.1.1 Approximation of a Wiener process

In this section we want to approximate the Wiener process on a discrete grid. Let therefore $t_n = n\Delta t, n = 1, ...N$ be discrete time instances in a constant time interval $T = \{t_0, t_1, ..., t_N\}$.

Algorithm 4.2. ***Approximation of the Wiener process***

Given:

- *Step length* $h = \Delta t$
- *Grid* $T = \{t_0, t_1, ..., t_N\}$ *with* $t_n = n\Delta t, n = 1, ...N$

An approximation of the Wiener process is then given by the recursion

$$\begin{aligned} W_{t_0} &= 0, \\ W_{t_{i+1}} &= W_{t_i} + \Delta W_i \end{aligned} \tag{4.1}$$

for $i = 0, ..., N-1$ *and* $N(0, \Delta t)$*-distributed random variables* ΔW_i *.*

Remark 4.3. *For a standard normally distributed random variable* $Z \sim N(0,1)$ *we have* $Z\sqrt{\Delta t} \sim N(0, \Delta t)$*. We can therefore rewrite Equation (4.1) as*

$$\Delta W_n = Z\sqrt{\Delta t} \text{ for } Z \sim N(0,1) \text{ for each } n. \tag{4.2}$$

Remark 4.4. *The MATLAB code for such an approximation of a Wiener process is given in Appendix C.2.2. It was used for plotting Figures 4.1 and 4.2.*

Remark 4.5. *Since the approximated Wiener Process satisfies the conditions on the exact Wiener process at every mesh point, the algorithm delivers a strong approximation of the Wiener process.*

4.1.2 Some Properties of the Wiener process

An interesting fact about the Wiener process is that its paths always look the same no matter how close you zoom in on them.

Remark 4.6. *Note that the Gaussian distribution is only defined for* $\sigma > 0$.
Let us now take a closer look at the definition of the Wiener process and the implications of the three conditions:

The first one is just a convention, if we want a Wiener process to start at x *instead of zero, it can be obtained as* $\{x + W_t\}_{t \geq 0}$.
It follows from the first and second condition that W_t *itself is a standard Gaussian random variable with* $E[W_t] = 0$ *and* $Var[W_t] = t$*. From this we see that the dispersion of the sample paths gets bigger and bigger with growing time* t *while the mean always remains zero. A path of a Wiener process will thus eventually take any and*

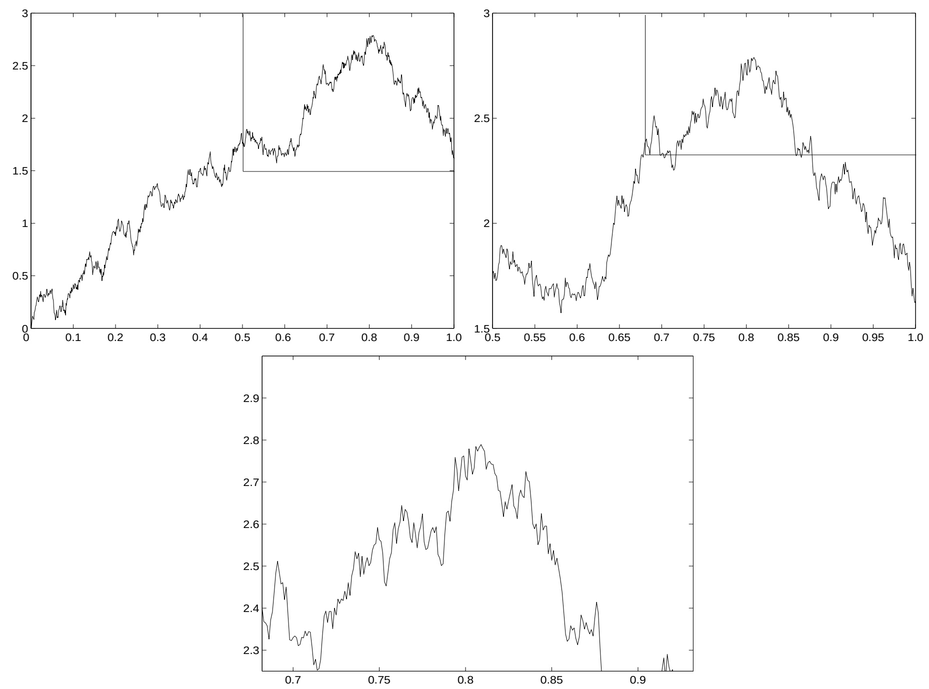


Figure 4.1: Zooming in on a path of a Wiener Process

every real value no matter if positive or negative, and it will then return to zero again with probability one.
The second condition also shows that $\tilde{W}_t = W_{t+t_0} - W_{t_0}$ is a Wiener process. A new path of a Wiener process can thus be obtained by cutting off the end part of a path of a Wiener process and moving it to t_0.
Those properties can be observed in Figure 4.2 where three different paths of a Wiener process are illustrated.
Another property which can not be seen as easily in the plot is stated in condition (3). It says that knowing a part of the path in the interval $[t_1, t_2]$ is of no help for predicting the path in another partial interval $[s_1, s_2]$ with $s_1 \geq t_2$ if the information on the initial values $W(s_1)$ and $W(t_1)$ is not available. The probability for a path of a Wiener process to move up or down is thus the same independently of its previous behavior.

This last condition says that only the present value of the Wiener process is needed to determine its future value. For general stochastic processes this is called the Markov property and can mathematically be expressed as

$$P(X_{t_{n+1}} \in B \mid X_{t_1} = x_1, X_{t_2} = x_2, ..., X_{t_n} = x_n) \quad = \tag{4.3}$$

$$= \quad P(X_{t_{n+1}} \in B \mid X_{t_n} = x_n, X_2 = x_{i_2}, ..., X_n = x_{i_n}) \tag{4.4}$$

for all Borel subsets B of $\mathbb{R}$, time instants $0 < t_1 < t_2 < ... < t_n < t_{n+1}$ and all states

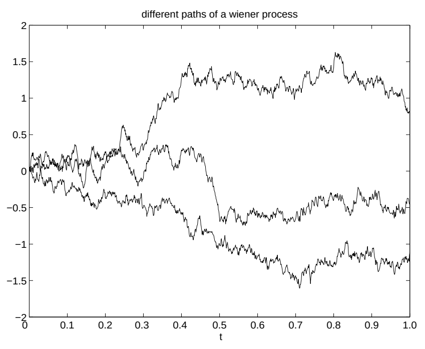


Figure 4.2: Illustrative example: 3 different paths of the Wiener Process

$x_1, x_2, ..., x_n \in \mathbb{R}$ for which the conditional probabilities are defined.
In the discrete-time case, such a Markov process is also called a discrete-time Markov chain.

Remark 4.7. *The Markov property says that the probability that $X_{n+1} \in B$ given that the history of the process is known up to time n equals the probability that $X_{n+1} \in B$ given only the value of X_n. Hence a Markov process has no memory.*

With the help of filtration and adaption we can now introduce another concept which is important for our model of the asset price movement, namely that of a martingale:

Definition 4.8. *A filtration on a probability space $(\Omega, \mathcal{A}, P)$ is an increasing family $((A)_t)_{t\geq 0}$ of sub-σ-algebras of (A), that is for each t there is a sub-σ-algebra $(A)_t$ and $(A)_s \subset (A)_t$ if $s < t$. A probability space $(\Omega, \mathcal{A}, P)$ endowed with a filtration $((A)_t)_{t\geq 0}$ is said to be a filtrated space.*

Remark 4.9. *A filtration $(A)_t$ can be seen as the collection of possible events up to time t.*

Definition 4.10. *A stochastic process X on the probability space $(\Omega, \mathcal{A}, P)$ is adapted to the filtration $\mathcal{A}_t$ if X_t is $\mathcal{A}_t$-measurable for each t.*

Remark 4.11. *Any process X is adapted to its natural filtration $A_t^0 = \sigma(X_s, s \leq t)$ which is at the same time the minimal filtration to which X is adapted.
The σ-algebra A_t includes all events which may occur before or at time t. Thereby, the natural filtration A_t^0 is a collection of exactly those events that have occurred up until time t.*

Remark 4.12. *For a one dimensional Wiener process $\{(W_t, A_t)\}_{t\geq 0}$ with given filtration A_t Condition 3 in the definition of the Wiener process 4.1 is also often written as*

$$W_t - W_s \text{ is independent of } A_s \text{ for } 0 \leq s \leq t. \tag{4.5}$$

Now we are able to define a martingale:

Definition 4.13. *Let $(\Omega, \mathcal{A}, P)$ be a probability space, with filtration $((A)_t)_{t\geq 0}$ to which the stochastic process $\{Xt; t \in [t_0, T]\}$ is adapted.*

(i) The stochastic process is then called a martingale if

$$E[X_t \mid A_s] = X_s \text{ w.p.}1 \tag{4.6}$$

for all $s, t \in [t_0, T]$ where $s \leq t$.

(ii) Is the equality sign in equation (4.6) is replaced by a "$\leq$" or a "$\geq$", the stochastic process $\{Xt; t \in [t_0, T]\}$ is called a supermartingale, or a submartingale respectively.

Remark 4.14. *Although a lot of martingales do also have the markov property, not all martingales are Markov processes and not all Markov processes are martingales.*

Remark 4.15. *Martingales are the equivalents to a fair game in a gambling context. They state that the expected winnings of the next game conditioned by what is known of what had been won in previous games up to the present is exactly the winnings of the present game.*

Theorem 4.16. *The one dimensional Wiener process W_t is a martingale.*

Proof. It follows from Equation (4.5) that

$$\begin{aligned} E[W_t|A_s] &= E[W_t - W_s + W_s|A_s] = E[W_t - W_s|A_s] + W_s = \\ &= E[W_t - W_s] + W_s = W_s \end{aligned}$$

for $s < t$ w.p.1 as can be seen in [16]. □

Another concept that should be mentioned in connection with filtrations is that of a stopping time. It will be useful in Chapter 7 for evaluating American type options:

Definition 4.17. *Let Ω be a sample space equipped with a filtration $\{F_n\}_{n\geq 0}$, then the random variable $\tau : \Omega \to Z_+$ with the property that*

$$\{\tau \leq n\} \in F_n, \text{for all } n \geq 0 \tag{4.7}$$

is called a stopping time or optional time.

This means that the information whether or not $\tau \leq n$ is based on the knowledge available at time n, there is no need to look into the future.

Remark 4.18. *A stopping time denotes the moments at which a process is stopped and preserved in its present state. Condition (4.7) states that one has to be able to decide at time τ whether the process should be stopped or not.*

Since we are interested in using the Wiener process as the random component of a stochastic differential equation, we would like to emphasize a very important property of the Wiener process:

Theorem 4.19. *Almost all realizations of Wiener processes W_T are nowhere differentiable.*

Proof. In accordance with [22] we will first show that

$$D_1 := \Big\{\omega \in \Omega : \exists t' \in [0;1) \text{ with } \frac{dW}{dt} \text{ exists in } t'\Big\}$$

is a P-null set:
Let therefore

$$A(j,k) := \Big\{\omega \in \Omega : \exists t \in [0;1) : |W(t+h) - W(t)| \leq jh \, \forall h \in [0;\frac{1}{k}]\Big\}$$

for $j,k \in \mathbb{N}$. Then

$$\bigcup_{j\in\mathbb{N}} \bigcup_{k\in\mathbb{N}} A(j,k) =$$
$$= \Big\{\omega \in \Omega : \exists t' \in [0;1) : -\infty < \liminf_{h\downarrow 0} \tfrac{1}{h}(W(t'+h) - W(t'))$$
$$\leq \limsup_{h\downarrow 0} \tfrac{1}{h}(W(t'+h) - W(t')) < \infty\Big\} \supset D_1$$

holds. It suffices now to show that there exists a set $C(j,k) \in \mathcal{A}$ for every $A(j,k)$ with $C(j,k) \supset A(j,k)$ and $P(C(j,k)) = 0$. In that case $\bigcup\limits_{j\in\mathbb{N}} \bigcup\limits_{k\in\mathbb{N}} C(j,k)$ is a P - null set which encloses D_1.
Let $A(j,k)$ be fixed and define

$$B(i,n) := \left\{ \omega \in \Omega : \begin{array}{c} |W(\frac{i+1}{n}) - W(\frac{i}{n})| \leq \frac{8j}{n} \\ |W(\frac{i+2}{n}) - W(\frac{i+1}{n})| \leq \frac{8j}{n} \\ |W(\frac{i+3}{n}) - W(\frac{i+2}{n})| \leq \frac{8j}{n} \end{array} \right\}$$

for $i,n \in \mathbb{N}$. We will show now that $A(j,k) \subset \bigcup\limits_{i=1}^{n} B(i,n)$ holds for all $n \geq 4k$. Let there be a $t \in [0;1)$ for $\omega \in \Omega$ with $|W(t+h) - W(t)| \leq jh$ for all $h \in [0;\frac{1}{k}]$. Let i be chosen such that $\frac{(i-1)}{n} \leq t \leq \frac{i}{n}$. Thus

$$|W(\frac{i+\nu+1}{n}) - W(\frac{i+\nu}{n})| \leq |W(\frac{i+\nu+1}{n}) - W(t)| + |W(\frac{i+\nu}{n}) - W(t)| \leq j\frac{2\cdot 4}{n}$$

follows for $\nu = 0,1,2$, since $0 < \frac{i+\nu}{n} - t \leq \frac{i+\nu+1}{n} - t \leq \frac{4}{n} \leq \frac{1}{k}$. Therefore $\omega \in B(i,n)$ and we get

$$A(j,k) \subset \bigcup_{i=1}^{n} B(i,n) \text{ for all } n \geq 4k. \tag{4.8}$$

Remember that the paths of a Wiener process follow a normal distribution. Therefore with

$$\int_{-c}^{c} \frac{1}{\sqrt{2\pi\sigma^2 t}} e^{\frac{-x^2}{2\sigma^2 t}} dx \leq \frac{1}{\sqrt{2\pi\sigma^2 t}} \int_{-c}^{c} dx = \frac{2}{\sqrt{2\pi}} \frac{c}{\sigma\sqrt{t}} < \frac{c}{\sigma\sqrt{t}}$$

$t \in [0;1)$ and $c \in \mathbb{R}_+$ we have

$$P(|W_{\frac{1}{n}} - W_0| \leq c) < \frac{c}{\sigma} n^{\frac{1}{2}}.$$

Taking into consideration that the increments of the Wiener process are independent and stationary and denoting $c = \frac{8j}{n}$, we get

$$P(B(i,n)) < \left(\frac{8jn^{\frac{1}{2}}}{n\sigma} \right)^3 = \left(\frac{8j}{\sigma} \right)^3 n^{-\frac{3}{2}}.$$

So if we define

$$C(j,k) := \bigcap_{n=4k}^{\infty} \bigcup_{i=1}^{n} B(i,n),$$

Equation (4.8) delivers $C(j,k) \supset A(j,k)$ on the one hand, and on the other hand we get for all $n \geq 4k$

$$P(C(j,k)) \leq P\left(\bigcup_{i=1}^{n} B(i,n) \right) \leq \sum_{i=1}^{n} P(B(i,n)) < \left(\frac{8j}{\sigma} \right)^3 n^{-0.5},$$

which means $P(C(j,k)) = 0$. This delivers the proof for the interval $[0,1)$.
Analogously it can be shown that for $[0;M), M \in \mathbb{N}$

$$D := \left\{ \omega \in \Omega : \exists t' \in T \text{ with } \frac{dW}{dt} \text{ exists in } t' \right\}$$

is a subset of a P null-set. □

4.2 Stochastic Integral

We saw that w.p.1 the paths of a Wiener process are nowhere differentiable (Theorem 4.19), they are not even of bounded variation. It is therefore not possible to define a stochastic integral, that is an integral with respect to the Wiener process

$$I(f) = \int_{t_0}^{t} f(s) dW_s \tag{4.9}$$

for general stochastic integrands $f(t)$ with the tools of deterministic integration theory. If we rewrite Equation (4.9) as

$$\int_{t_0}^{t} f(s)dW_s = \int_{t_0}^{t} f(s)\frac{dW_s}{ds}ds, \tag{4.10}$$

we see that the integral on the right side cannot be a Riemann- or Lebesgue integral. We do, however, need to find a definition for the integral in Equation (4.9) in order to be able to define stochastic differential equations.

The approach we will use here was first introduced by Kiyosi Ito in the 1940s. His idea was it to find a stepwise definition of a stochastic integral. He starts with f being a constant function, extends the definition over to non-random step-functions to random step functions and finally reaches a stochastic integral for general random functions f.

We only want to give an outline of the first steps of the construction of the Ito stochastic integral here, for a deeper investigation we refer to [15]:

- Let f be a constant function:

  For such a function $f(t) = b$ the integral given by (4.9) is equal to $b\{W_t - W_{t_0}\}$.

- Consider f to be a non-random step function $f(t) = f_j$

  for $t \in [t_j, t_{j+1}), j = 1, 2, ..., N$, where $0 = t_1 < t_2 < ... < t_{N+1} = 1$.

  Then the integral is given by

$$I(f) = \sum_{j=1}^{N} f_j\{W_{t_{j+1}} - W_{t_j}\}. \tag{4.11}$$

  Since the sum of random variables with mean μ is a random variable with mean μ itself, the right hand side of this equation is a random variable with zero mean.

- Extend f to be a random step-function.

  In order to make sure that the integrant is non-anticipative, certain measurability conditions have to be made. Suppose that W_t is $\mathcal{A}_t$-measurable for each $t \geq 0$. Then we can construct a random step function

$$f_t = f_j \text{ for } t \in [t_j, t_{j+1}), j = 1, 2, ..., N, \tag{4.12}$$
$$\text{where } 0 = t_1 < t_2 < ... < t_{N+1} = 1 \text{ and } f_j \text{ is } \mathcal{A}_{t_j}\text{-measurable,}$$

  meaning that it can be observed by events that are detectable at or before time t_j. Such a random step function is also called a simple process. In addition to that let each f_j be mean-square integrable over Ω and therefore $E[f_j^2] < \infty$, $j = 1, 2, ..., n$. We can now conclude that $f_j\{W_{t_{j+1}} - W_{t_j}\}$ has expectation

$$E[f_j\{W_{t_{j+1}} - W_{t_j}\}] = E[f_j E[W_{t_{j+1}} - W_{t_j}|A_{t_j}]] = 0 \tag{4.13}$$

for each $j = 1, 2, ..., N$ since $E[W_{t_{j+1}} - W_{t_j} | A_{t_j}] = 0$ w.p.1. Now we can define the integral $I(f)$ similar to the case of a non-random step function:

$$I(f) = \sum_{j=1}^{N} f_j \{W_{t_{j+1}} - W_{t_j}\} \tag{4.14}$$

w.p.1.

Generalizing the definition of the function f further finally leads to the concept of the Ito-Integral.

Definition 4.20. *The Ito integral of a general function f is defined as*

$$\int_{t_0}^{t} f(s) dW_s := \lim_{n\to\infty} \int_{t_0}^{t} f_n(s) dW_s \tag{4.15}$$

for simple processes f_n defined by Equation (4.12).

Remark 4.21. *There is also another definition of a stochastic integral, namely that of Stranovich, given as*

$$\int_{0}^{t} f(s) \circ dW_s := \lim_{N\to\infty} \sum_{k=0}^{N-1} f_{\frac{1}{2}(t_k + t_{k+1})} (W_{t_{k+1}} - W_{t_k}). \tag{4.16}$$

The advantage of this integral is the fact that the chain rule applies just like in the deterministic case. However, it requires the knowledge of the asset price both at times t and $t+1$ for approximating f_s via the average $\frac{1}{2}(t_k + t_{k+1})$. This is why the stochastic integral according to Ito is preferred for financial applications.

4.3 SDE model for asset price movements

We now want to find a mathematical description for asset price dynamics. In order to be able to do that, we need to introduce the concept of stochastic differential equations:

Definition 4.22. *The equation*

$$dX_t = a(X_t, t)dt + b(X - t, t)dW_t \tag{4.17}$$

with $X_{t_0} = X_0$ is called an Ito stochastic differential equation (Ito SDE). This denotation is short for the integral equation

$$X_t = X_0 + \int_{t_0}^{t} a(x_s, s)ds + \int_{t_0}^{t} b(X_s, s)dW_s. \tag{4.18}$$

Thereby $a(X_t, t)$ is the drift term or drift coefficient, $b(X_t, t)$ is the diffusion term and the solution X_t is called an Ito stochastic process.

Remark 4.23. *The Wiener process is a special case of an Ito process with $a = 0$ and $b = 1$.*

We now have everything we need to discuss one of the most important continuous mathematical model for price movements of assets. It is based on the idea that the change of the asset price dS in the time interval t can be represented by an Ito SDE as defined in Definition 4.17 with drift term $a(X_t, t) = \mu S_t$ and diffusion term $b(X_t, t) = \sigma S_t$. The resulting linear SDE is called Geometric Brownian Motion:

$$dS_t = \mu S_t dt + \sigma S_t dW_t. \tag{4.19}$$

With S_0 as the asset price at time t_0 this is our model for the motion of the asset prices S_t as already mentioned in our Market Model 2.7. Thereby the drift rate μS_t with expected rate of return μ is responsible for the up- and downward movement of the price and the diffusion term σS_t with volatility σ influences the stochastic dispersion, i.e. the variance of the motion.

Figure 4.3 shows the price of a real asset on the left in comparison to three simulated asset price paths on the right:

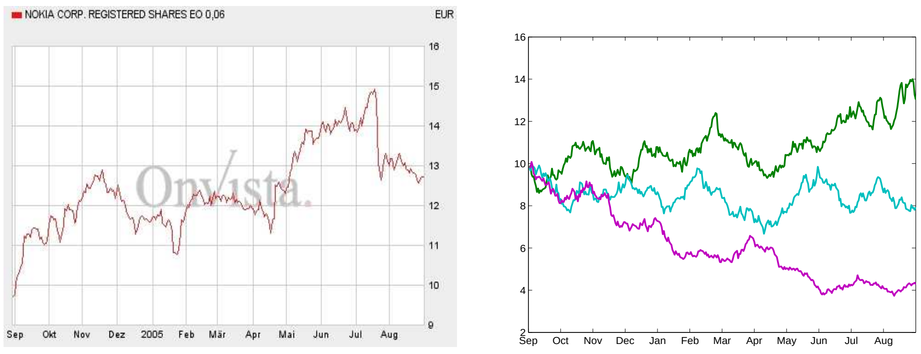


Figure 4.3: Real asset price movement compared to simulated ones

Remark 4.24. *The MATLAB code for such an approximation of the price movement of assets is given in Appendix C.2.3. It was used for computing the three simulated asset price paths on the right hand side ¡of Figure 4.3.¡*

Let us now take a look at how good this model is at approximating the reality of asset price movement.
In the assumptions on our Market Model 2.7 we stated any information is immediately accessible for all parties, and thus it is reasonable to assume that the market responds instantaneously to those external influences. The current asset price therefore reflects all past information. This conclusion known as the weak form of the efficient market

hypothesis is mathematically modeled by the Markov property. Hence we want the stochastic process which influences our asset price movement to be a Markov process. As we already saw in the previous section, the Wiener process satisfies this condition. The second condition we set on the stochastic process in our model for asset price movement is that the motion of the price is not biased, i.e. the chances for the asset price to go up should equal those for it to drop. This is exactly the Martingale property which we already proved for the Wiener process. It therefore is a fair game.
Let us now take a look at the expectation and the variance of the Geometric Brownian Motion. Recalling Equation (4.18) we can rewrite Equation (4.19) as

$$S_t = S_0 + \int_{t_0}^{t} \mu S_s ds + \int_{t_0}^{t} \sigma S_s dW_s. \tag{4.20}$$

This leads to an expectation of

$$\begin{aligned} E[S_t] &= E[S_0] + E[\int_0^t \mu S_s ds] + E[\int_0^t \sigma S_s dW_s] \\ &= E[S_0] + \int_0^t \mu E[S_0] ds, \end{aligned} \tag{4.21}$$

since $E[\int_0^t \sigma S_s dW_s] = 0$ as can be seen in [10].
We see that $g(t) = E[S_s]$ is the solution to the ordinary differential equation

$$\frac{dg(t)}{dt} = \mu g(t) \text{ with } g(0) = E[S_0] = S_0.$$

Therefore we have

$$E[S_t] = S_0 e^{\mu t}. \tag{4.22}$$

The variance can be derived the same way and we get

$$Var[S_t] = S_0^2 e^{2\mu t}(e^{\sigma^2 t} - 1). \tag{4.23}$$

Hence the Geometric Brownian Motion is lognormally distributed with

$$E\left[\frac{lnS_t}{lnS_0}\right] = (\mu - \sigma^2)/2t \tag{4.24}$$

and

$$Var\left[\frac{lnS_t}{lnS_0}\right] = \sigma^2 t. \tag{4.25}$$

We still have to find values for the parameters μ and σ in our model. Since we want to evaluate options under the assumption of a risk-neutral world, the drift parameter μ can be replaced by the risk-free interest rate r. Typical values for r lie between 0.01 and 0.1 (see [13]).

Finding a value for the volatility σ is not as straightforward and we will discuss different volatility models later on in Chapter 6. It is found that σ typically takes values between 0.05 and 0.5. These values are to be understood as per annum since we are measuring time in years.

Remark 4.25. *One advantage of this simple model for asset price dynamics is the fact that one does not need to employ numerical methods for solving this SDE. It is possible to calculate the solution via Ito's Lemma which is the topic of the next section.*

4.3.1 Ito Lemma

A very powerful tool for deriving solutions of SDEs is Ito's Lemma. It can be seen as the stochastic extension to the chain rule in ordinary calculus:

Theorem 4.26. *Let X_t be an Ito process*

$$dX(t) = a(X(t), t)dt + b(X(t), t)dW_t, \tag{4.26}$$

and let $g(x,t)$ be a function with continuous derivatives $\frac{\partial g}{\partial x}$, $\frac{\partial^2 g}{\partial x^2}$, $\frac{\partial g}{\partial t}$.
Then $Y_t := g(X_t, t)$ satisfies the equation

$$\begin{aligned} dY(t) = &\left(\tfrac{\partial g}{\partial x}(t, X(t)) + \tfrac{\partial g}{\partial x}(t, X(t))a(t, X(t)) + \tfrac{1}{2}\tfrac{\partial^2 g}{\partial x^2} b(t, X(t))^2\right) dt \\ &+ \tfrac{\partial g}{\partial x}(t, X(t))b(t, X(t))dW_t, \end{aligned} \tag{4.27}$$

where W is the same Wiener process as in the SDE

A proof to this important theorem can be found in [1].

4.3.2 Solution of Geometric Brownian Motion

Let $Y_t = ln(S_t)$ and accordingly $g(x,t) = ln(x)$. Since we are not interested in the changes of the asset price in between $t_0 = 0$ and t, we can assume dt to be $t - t_0 = t$, which results in $dS = S_t - S_0$. Applying Ito's lemma with the coefficient functions $a = \mu S_t$ and $b = \sigma S_t$ we get

$$\begin{aligned} ln(S_t) - ln(S_0) &= (\frac{1}{S_t}\mu S_t - \frac{1}{2}\frac{1}{S_t^2}\sigma^2 S_t^2)dt + \frac{1}{S_t}\sigma S_t dW_t = \\ &= (\mu - \frac{1}{2}\sigma^2 dt + \sigma S_t dW_t), \end{aligned}$$

and thus

$$ln(S_t) = ln(S_0) + (\mu - \frac{1}{2}\sigma^2 dt + \sigma S_t dW_t). \tag{4.28}$$

This can be written as

$$S_t = S_0 + e^{\mu - \frac{1}{2}\sigma^2 dt + \sigma S_t dW_t}, \tag{4.29}$$

which is the solution to our SDE (4.19).

Remark 4.27. *Since we want to approximate the risk-free price of an option, the parameter μ can be replaced by the risk-free interest rate r. The grounds for this lie in the Black-Scholes model for option pricing (see Appendix A).*

Remark 4.28. *If we are only interested in the asset price at time T it is enough to use Ito's Lemma for solving the SDE modeling the asset price movement. If we are interested in the whole path of the asset price, however, we have to rely on numerical methods for approximating the price path such as the Euler-Maruyama algorithm introduced in Appendix B.*

4.3.3 Options on dividend-paying assets

Starting with the easier of the two additions to our model, we will now explain which changes have to be applied to the equations that model the price movement of the underlying and if necessary to the Monte Carlo algorithm.
In this section, we want to examine the effect of dividend payments of the underlying asset on the option value. Although there are many different types of dividends such as

- deterministic or stochastic
- continuous or discrete

ones, we will only consider the simplest case here, namely that of a continuous and constant dividend yield δ already known at the time the option is issued. Thereby, the payments depend on the price of the underlying, that is they are directly proportional to it. With the dividend payment being set as $\delta S dt$ the dividend yield δ defined as the ratio of the dividend payment to the asset price

$$\delta = \frac{\delta S dt}{S dt} \tag{4.30}$$

is indeed constant and continuous. During a time interval of Δt, the asset pays out a dividend $\delta S dt$. Arbitrage considerations show that this payment must reduce the price of the asset by the same amount of money. Otherwise it was possible to buy the underlying right before the time t when dividends are payed, receive the dividend payment $\delta S \Delta t$ and sell the asset immediately afterwards. This way it would be possible to make an instantaneous risk-free profit of $\delta S \Delta t$ which is ruled out by the no-arbitrage principle. Therefore, SDE (4.19) which describes the asset price movement has to be slightly modified. The new risk neutralized price of the underlying dividend-paying asset S_t now satisfies the stochastic differential equation

$$dS_t = S_t[(r-\delta)dt + \sigma dW_t], \tag{4.31}$$

where W_t is a standard Brownian motion or Wiener process. δ can therefore be seen as a negative interest rate. Under this risk neutral martingale measure S is lognormally distributed with mean $(r-\delta-\frac{\sigma^2}{2})(t_i - t_{i-1})$ and variance $\sigma^2(t_i - t_{i-1})$.
Applying Ito's lemma to the equation above, we get

$$S_t = S_{t-1}e^{(r-\delta-0.5\sigma^2)(t_i-t_{i-1})+\sigma\sqrt{t_i-t_{i-1}}Z}, \tag{4.32}$$

where Z is a standard normal variable.

As we saw, the continuous and constant dividend yield δ affects the model of the asset price movement but it does not complicate the SDE in a way such that it could not be solved using Ito's Lemma anymore. Therefore this additional feature does not have a significant impact on the Monte Carlo algorithm for deriving a fair option value. Equation (4.29) is simply replaced by (4.32), and no additional numerical methods are needed.

Remark 4.29. *This is not the case with more complicated dividend payments however.*

The concept of options on dividend-paying underlyings will be needed again in Chapter 7 when evaluating an American call option.

Remark 4.30. *If we relax the assumption of the volatility being constant, we get a more complicated model represented by a multi-dimensional stochastic process. An illustrative example of this case will be given later on in Chapter 6.*

Chapter 5

Random Number Generators

In order to run simulations of stochastic processes on a computer, it is necessary to have access to a large quantity of random numbers with specified statistical properties. Those numbers can be generated on a digital computer in different ways, but since deterministic algorithms will be used, the outcome can never be completely random. However, the generated numbers are very similar to truly random numbers in most respects, which is why they are often called pseudo-random numbers. In what follows, we will simply use the term random number, although we are talking about pseudo-random numbers. The advantage of using numbers generated by a deterministic algorithm is that the same sequence of numbers can be reproduced if the method used for computing the numbers and the seed is known. It is therefore possible to test programs that are based on random numbers by repeating test runs with the same input of numbers. In the following, we will introduce different methods for generating uniformly distributed random numbers on a digital computer. Having done this, we will introduce the concept of quasi random numbers before we move on to presenting different methods for generating sequences of normally distributed numbers.

The presentation in this chapter is mainly based on the two books [12] and [23].

5.1 Uniformly distributed random numbers

The basis of generating random numbers with certain stochastic properties is the computation of uniformly distributed random numbers. Those can then be transformed into numbers with the required statistical properties. We will introduce two groups of commonly used algorithms for producing uniformly distributed random numbers in this section, linear congruential generators and fibonacci generators.

5.1.1 Linear congruential generators

Starting with the seed $X_0 \in \{0, 1, ..., M-1\}$, a sequence of random numbers is calculated via the following algorithm:

Algorithm 5.1. ***Linear Congruential Generator***

For $i = 1, 2, ...$

$$X_i = (aX_{i-1} + b) \mod M,$$

$$U_i = \frac{X_i}{M},$$

where a and M are positive integers and b is a nonnegative integer.

The modulo function $X = A \mod M$ calculates the remainders when A is divided by M. The first M numbers U_i generated by this algorithm are considered to be uniformly distributed on $[0, 1]$. The quality of the distribution can be improved by choosing a, b and M in a reasonable way.

- One consideration is that M should be sufficiently large. The reason for this is that the X_i are periodic with period $\leq M$, meaning at least two numbers in $\{X_0, ..., X_M\}$ must be equal. A cycle period even smaller than M can be prevented by picking an a relatively prime to M.

- Another consideration is that $a = 1$ is not recommendable because that would settle the generator down to the easily predictable sequence of $X_n = (X_0 + nb) \mod M$. In the special and often used case of $b = 0$, the resulting generator is called a multiplicative generator. Obviously, $X = 0$ must be ruled out here, otherwise the generator would be stuck with $X_i = 0$ for all i.

One major disadvantage of this method is the fact that the resulting pairs of random numbers (U_i, U_{i+1}) are strongly correlated. They are situated on very few hyperplanes in $\mathbb{R}^m$. This can be seen in Figure 5.1 where 1500 random number pairs (U_{i-1}, U_i) are depicted. They were computed using a linear congruential generator with parameters $a = 1229$, $b = 1$ and $M = 2048$.

5.1.2 Fibonacci Generators

Here we want to introduce two different kinds of generators:

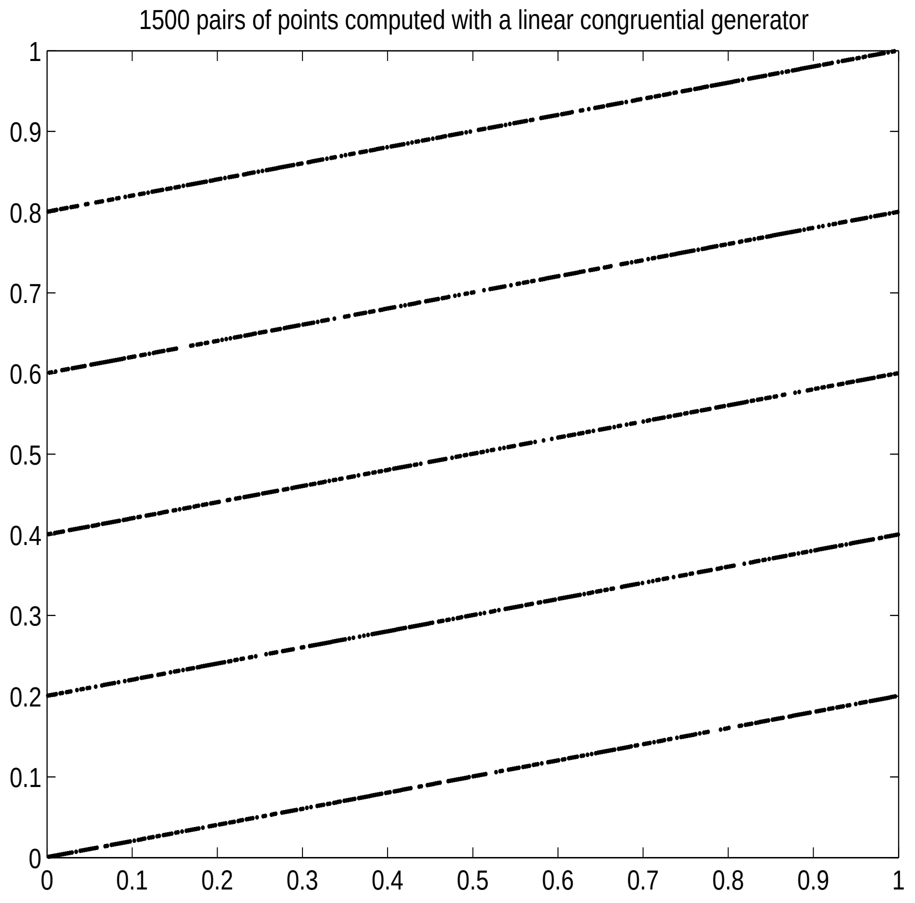


Figure 5.1: 1500 pairs of points (U_{i-1}, U_i) computed with the congruential generator

1. **Fibonacci Generators**

   This method is based on the fibonacci sequence which is recursively defined as

$$\begin{aligned} a_1 &= a_2 := 1; \\ a_n &:= a_{n-1} + a_{n-2} \text{ for } n \geq 3. \end{aligned}$$

The resulting algorithm for generating uniformly distributed random numbers is:

Algorithm 5.2. ***Fibonacci Generator***

For $i = 2, 3, ...$

$$\begin{aligned} X_i &= (X_{i-1} + X_{i-2}) \mod M, \\ U_i &= \frac{X_i}{M}, \end{aligned}$$

where $M, X_0, X_1 \in \mathbb{N}$.

Thereby, the linear congruential method can be used for deriving the initial values X_0 and X_1. However, in most cases the results generated by this method are not very satisfying because the sequence of random numbers is cyclic. This can be seen in Figure 5.2, although 1500 pairs of random numbers have been generated, the picture is relatively empty.

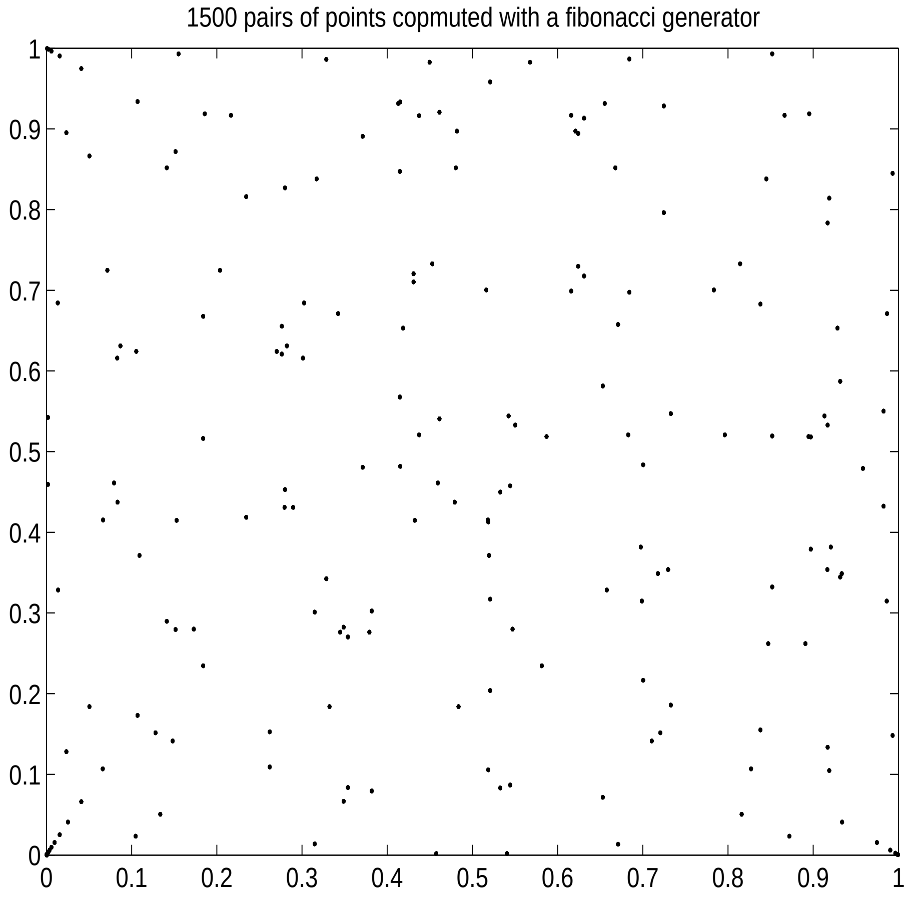


Figure 5.2: 1500 pairs of points (U_{i-1}, U_i) computed with the fibonacci generator

2. **Lagged Fibonacci Generators**
   Similarly to the method described above, lagged fibonacci generators produce random numbers by adding two random numbers which occurred earlier in the sequence. The algorithm would be

Algorithm 5.3. ***Lagged Fibonacci Generator***

For $i \geq \max\{\mu, \nu\}$

$$\begin{aligned} X_i &= (X_{i-\mu} - X_{i-\nu}) \mod M, \\ U_i &= \frac{X_i}{M}. \end{aligned}$$

Again, the initial values $X_1, ..., X_{\max\{\mu,\nu\}}$ can be generated by a linear congruential generator. For many choices of μ and ν, this algorithm produces satisfying results. Another advantage of the lagged fibonacci method is the simplicity of its computation. Pairs of random numbers generated with this method can be seen in Figure5.3

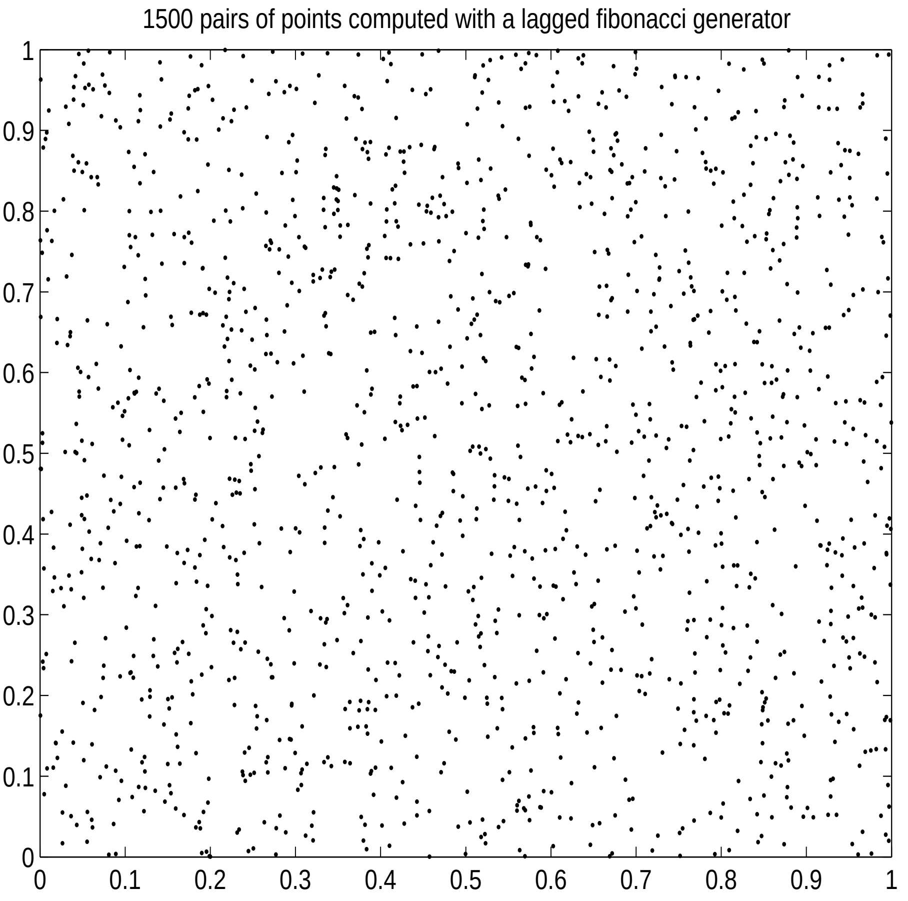


Figure 5.3: 1500 pairs of points (U_{i-1}, U_i) computed with the lagged fibonacci generator

Remark 5.4. *Another generator for uniformly distributed numbers would be Matlab's built in function 'rand'. Since it is a built-in MATLAB routine, is a lot faster than the random number generators mentioned above when implemented as M-files.*

Older versions of MATLAB used a multiplicative congruential generator with parameters

$$\begin{aligned} a &= 7^5 = 16807, \\ b &= 0, \\ M &= 2^{31} - 1 = 2147483647 \end{aligned}$$

which computed a little more than 2 *billion numbers before the sequence repeated itself.*
Versions 5 *and higher of MATLAB use a completely different random number generator which manages without any multiplications or divisions. This quite complicated random number generator is based on an algorithm introduced by George Marsaglia, a professor at Florida State University. Although it has not actually been tested this generator is theoretically able to generate all floating-point numbers between* 2^{-53} *and* $1 - 2^{-53}$*. It therefore has a period length of about* 2^{1429}*. For more details on MATLAB's 'rand' function we refer to C.B. Moler's book [18] Chapter* 9*.*
Pairs of numbers generated with this function can be seen in Figure 5.4

5.2 Sequences of Numbers with Low Discrepancy

When generating random numbers, a very important property of those numbers is their even distribution. In Figures 5.1 - 5.3 it is obvious that some spots in the unit square are more crowded than others. Therefore, it is possible that some areas of the unit cube are not explored at all using pseudo-random numbers. Intuitively, it seems logical that the accuracy of simulation results increases if the random numbers are more evenly distributed. This is what the idea of low discrepancy sequences is based on.
Rather than choosing numbers randomly, the interval $(0, 1)$ is now filled in a deterministic way such that the points are as evenly distributed as possible.

Before we can examine some methods for creating low discrepancy sequences, we have to introduce a measure of the equidistributedness:
Take the unit cube $[0, 1]^m$ on which our random numbers are distributed and let $Q \subseteq [0, 1]^m$ be an m-dimensional rectangle in that cube. Now the idea is that for an evenly distributed point set the percentage of the points being situated within Q should be proportional to the volume of Q, in other words

$$\frac{\#\text{ of } x_i \in Q}{\#\text{ of all points}} \approx \frac{vol(Q)}{vol([0, 1]^m)}$$

for as many rectangles Q as possible. This leads to the definition of discrepancy:

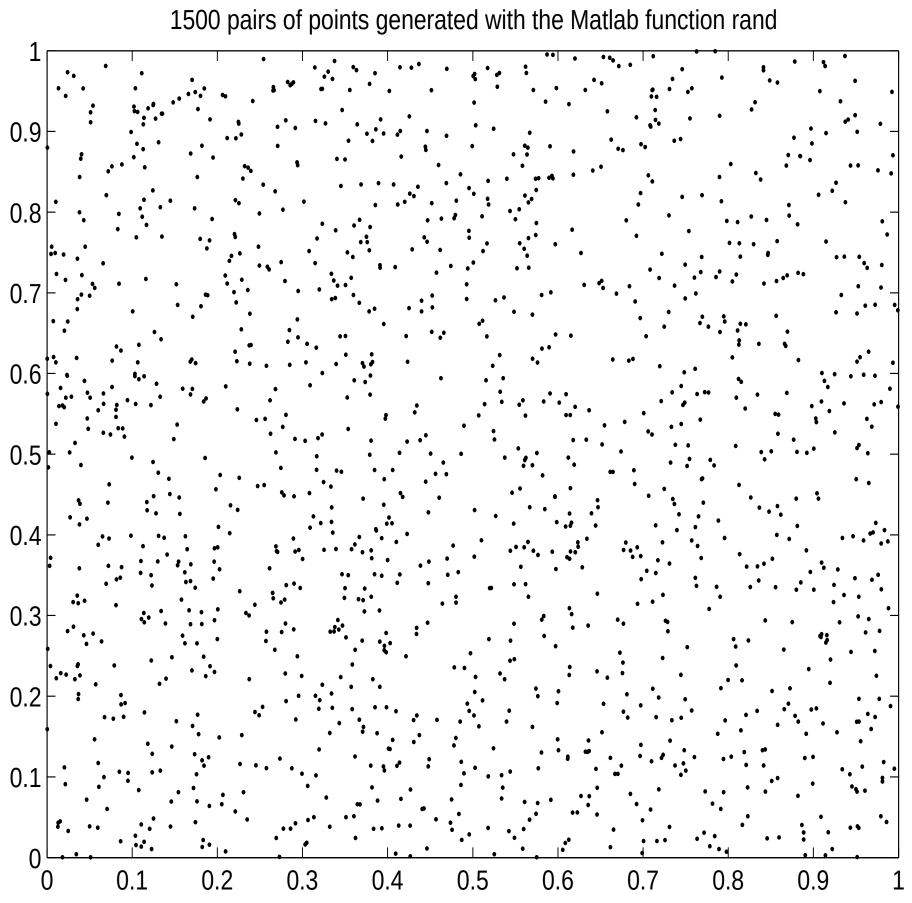


Figure 5.4: 1500 pairs of points (U_{i-1}, U_i) computed with the MATLAB function 'rand'

Definition 5.5. *The discrepancy of the point set* $\{x_1, ..., x_n\}$ *is given by*

$$D_N = \sup_Q \left| \frac{\#\ of\ x_i \in Q}{N} - volQ \right|,$$

where Q *is some rectangle.*

The closer the discrepancy D_N is to zero, the more evenly the points of a sequence are distributed:

Definition 5.6. *The sequence* (y_n) *is said to be evenly distributed in* $[0,1]^m$ *if*

$$\lim_{N\to\infty} D_N = 0,$$

where the index N *refers to the first* N *points of a sequence.*

Definition 5.7. *A sequence of points or numbers* $x_1, x_2, ..., x_N, ... \in \mathbb{R}^m$ *is called a low discrepancy sequence if there is a constant* C_m *such that for all* N

$$D_N \leq C_m \frac{(\log N)^m}{N} \tag{5.1}$$

holds. Thereby, C_m is independent of N.

Remark 5.8. *Just like digitally computed random numbers are called pseudo-random numbers although they are calculated in an entirely deterministic way, numbers of low discrepancy are often called quasi-random numbers.*

We will now examine some examples of sequences of low discrepancy:

Example 5.9. *The point set $\{x_1^N, ..., x_N^N\}$ with*

$$x_i = \frac{2i-1}{2N}, \; i = 1, ..., N \tag{5.2}$$

delivers a sequence of low discrepancy since $D_N^* = \frac{1}{2N}$.

A big drawback of this method is that N has to be fixed. If we increase N for getting more quasi-random numbers, the whole sequence has to be calculated again for each new N. A more efficient way of creating quasi-random numbers would be to calculate the set points dynamically such that they do not have to be recalculated for growing Ns. A sequence that allows us to place its points dynamically would be the van der Corput sequence

$$(x_i) := \frac{1}{2}, \frac{1}{4}, \frac{3}{4}, \frac{1}{8}, \frac{5}{8}, \frac{3}{8}, \frac{7}{8}, \frac{1}{16}, \dots$$

Thereby, the i-th element is derived by reversing the binary digits of the index $i \in \mathbb{N}$ written as a binary number. The index $i = 5$, for example, is given by the binary number $5 = (101)_2 =: (d_2 d_1 d_0)_2$ with $d_i \in [0, 1]$. The 5-th element of the van der Corput sequence can now be derived by reversing these binary digits and putting the radix point in front of the sequence:

$$(.d_0 d_1 d_2)_2 = \frac{d_0}{2} + \frac{d_1}{2^2} + \frac{d_2}{2^3} = \frac{1}{2} + \frac{0}{2^2} + \frac{1}{2^3} = \frac{5}{8}.$$

The numbers of the van der Corput sequence can be defined with the radical-inverse function:

Definition 5.10. *Let*

$$i = (d_j ... d_0) = \sum_{k=0}^{j} d_k b^k$$

be the expansion for $i = 1, 2, ...$ in base b, where b is an integer ≥ 2 and d_k are digits in $\{0, 1, ..., b-1\}$. That is i is given in binary form in case $b = 2$.
Then the radical-inverse function is defined by

$$\phi_b(i) := \sum_{k=0}^{j} d_k b^{-k-1}.$$

This function assigns a rational number $0 < x < 1$ to each index i. The higher the number of digits, the finer the mesh. Each additional digit refines the mesh by a factor of $1/b$. The van der Corput sequence above is obtained by $x_i := \phi_2(i)$.
The radical-inverse function can also be used for computing higher dimensional sequences. Thereby, each dimension m can be seen as one time step in the Monte Carlo simulation. The simplest sequence of points in $[0,1]^m$ would be the Halton Sequence:

Definition 5.11. *Let $p_1, ..., p_m$ be pairwise prime integers. Then the sequence of vectors*

$$x_i := (\phi_{p_1}(i), ..., \phi_{p_m}(i)),\ i = 1, 2, ...$$

is called the Halton sequence.

Halton Sequences are very simple sequences of low discrepancy. In case $m = 2$, the constant is $C_2 = 0.2602$. The first 1500 points of the two-dimensional Halton Sequence for bases $p_1 = 2$ and $p_2 = 3$ can be seen in Figure 5.5.
A comparison between the uniformity of random distributions and the one dimensional Halton sequence can be seen in the histogram of Figure 5.7.

Remark 5.12. *Sequences of higher dimensions are computed by using a different prime base for each new dimension. This leads to longer and longer cycle lengths, meaning it takes a lot longer to fill the grid in the unit cube as the dimensions increase, and thus the speed decreases with each dimension.*

Remark 5.13. *Another drawback of higher dimensions is the fact that neighboring dimensions are highly correlated, which can be seen in Figure 5.6. More sophisticated high dimensional sequences of low discrepancy would be Faure or Sobol sequences. The first dimension is equal to the Halton sequence, but for higher dimensions different methods are used to reduce the drawbacks of the Halton sequence. For more information on those sequences we refer to [20],[2] and [8].*

5.3 Normally distributed random numbers

We mentioned at the beginning of this chapter that random numbers are often required to satisfy certain stochastic properties. Standard Gaussian distributed numbers $Z \sim N(0,1)$ are needed, for example, in order to simulate the Wiener process. In this section we will learn how uniformly distributed numbers can be transformed into Gaussian numbers. The two main approaches are inversion methods and transformation methods.

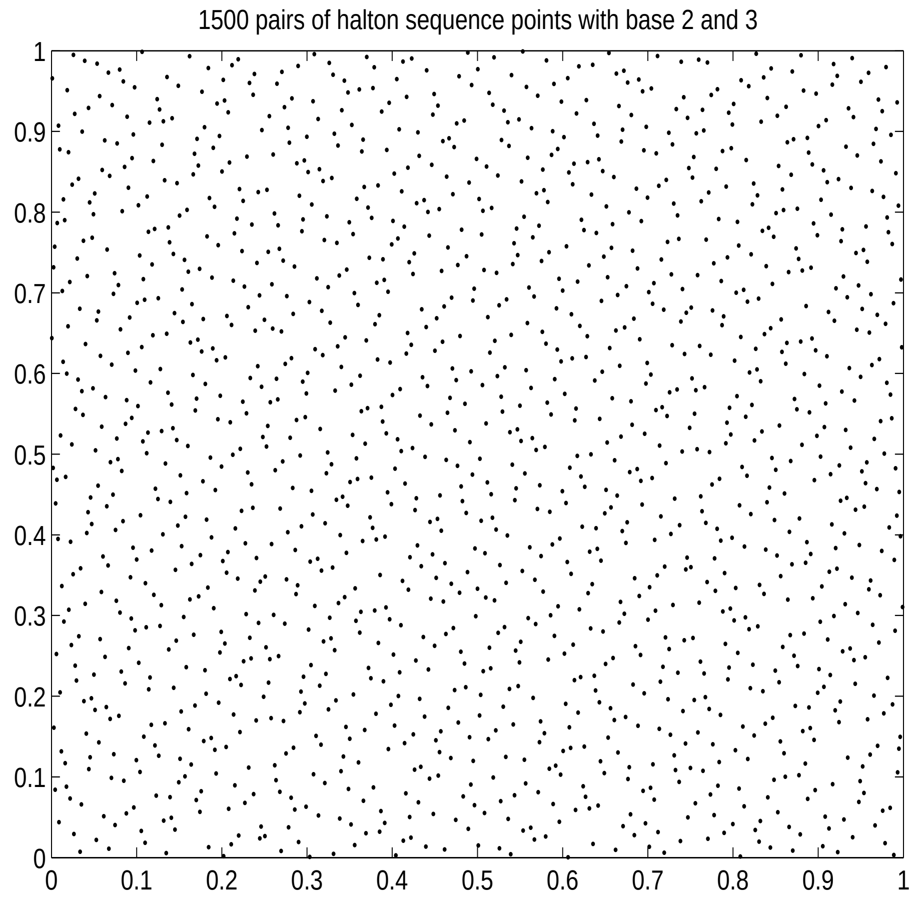


Figure 5.5: The first 1500 points of the Halton sequence in dimensions 1 (base $p_1 = 2$) and 2 (base $p_2 = 3$)

5.3.1 Inversion

This class of methods inverts the distribution function and is based on the following theorem:

Theorem 5.14. *Let $U \sim U[0,1]$ be a uniformly distributed random variable and F be a continuous strictly increasing distribution function. Then $F^{-1}(U)$ is a random variable with distribution function F.*

Proof. Since U is uniformly distributed, $P(U \leq \xi) = \xi$ for all $\xi \in [0,1]$. Consequently

$$P(F^{-1}(U) \leq x) = P(U \leq F(x)) = F(x),$$

which means that $F^{-1}(U)$ is distributed according to F. □

Since there is no closed-form expression for the inverse of the standard Gaussian distribution function F^{-1}, numerical methods are needed for inverting the non-linear equa-

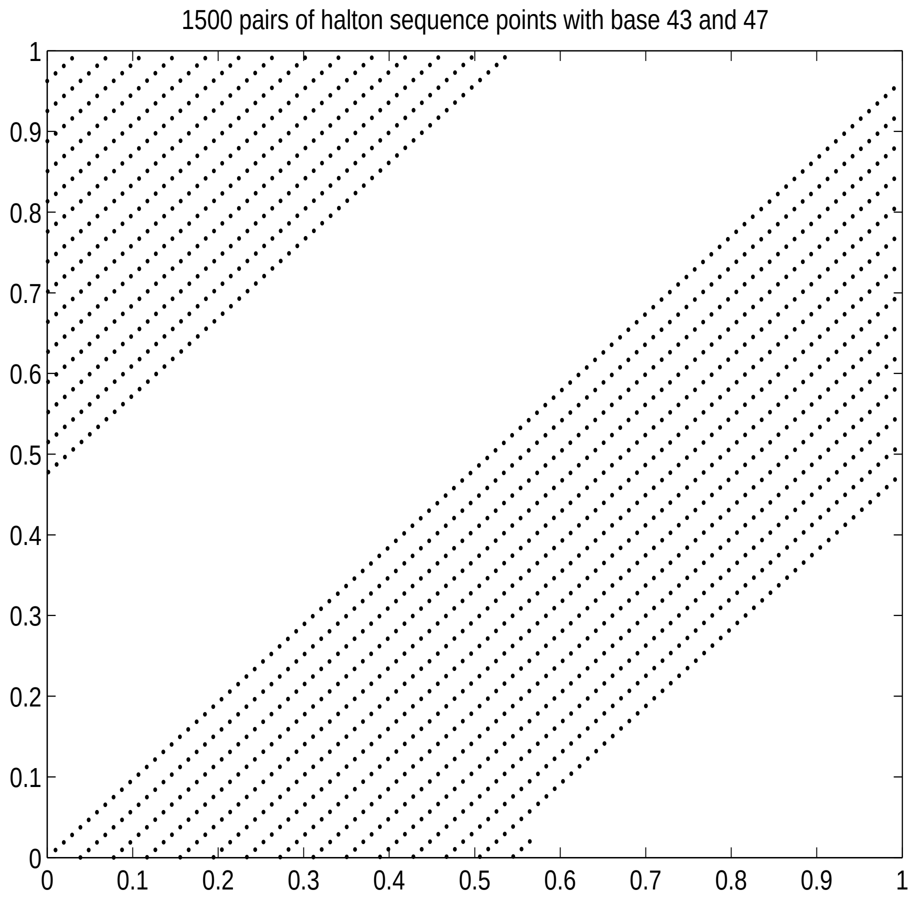


Figure 5.6: The first 1500 points of the Halton sequence in dimensions 27 (base $p_1 = 103$) and 28 (base $p_2 = 107$)

tion

$$F(x) := \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} e^{\left(-\frac{t^2}{2}\right)} dt = u. \tag{5.3}$$

Thereby, a solution x of the equation $F(x) = u$ is iteratively calculated for prescribed u. Methods for evaluating x in such a way would be Newton's method, bisection, or the secant method. However, these methods are poorly conditioned for $u \approx 1$, meaning that small changes in u cause large changes in x. This problem can be avoided by using a suitable function $G(u) \approx F^{-1}(u)$ to approximate the inversion $x = F^{-1}(u)$. It is important to consider that $F^{-1}(u)$ has vertical tangents at $u = 1$ and $u = 0$ when constructing G. An effective method for this approximation would be the rational approximation. This method makes it easy to reproduce this pole behavior since it allows to incorporate point symmetry.

We want to introduce one such method here, the inversion algorithm by Moro:

Moro's inversion formula

Moro's method for inverting the standard normal distribution is divided into two parts. He truncated the interval $0 < u < 1$ to $10^{-12} \leq u \leq -10^{-12}$ in order to derive the two major parts, the main distribution $0.08 < u < 0.92$ and the tails $0.92 \leq u \leq 1 - 10^{-12}$ and $10^{-12} \leq u \leq 0.08$, where the latter one is obtained thanks to the symmetry with respect to $(x, u) = (0, 0.5)$. Now both parts have to be approximated. Let us start with the formula for the middle part

$$(u - 0.5) \frac{\sum\limits_{j=0}^{3} a_j (u - 0.5)^{2j}}{1 + \sum\limits_{j=0}^{3} b_j (u - 0.5)^{2j}}, \tag{5.4}$$

the coefficients can be found in Table 5.1 below.
The tails of the distribution are approximated by a polynomial in $ln(-lnr)$ with $10^{-12} \leq r \leq 0.08$:

Algorithm 5.15. ***Moro's inversion algorithm of the standard normal distribution***

```
Given: $u \sim U(0,1)$

$y := u - 0.5$

case $|y| < 0.42$

$r := y^2$
$x := y \frac{((a_3 r + a_2) r + a_1) r + a_0}{(((b_3 r + b_2) r + b_1) r + b_0) r + 1}$
case $|y| \leq 0.42$
$r := u$ case $y > 0$ set $r := 1 - u$
$r := log(-logr)$
$x := c_0 + r(c_1 + r(c_2 + r(c_3 + r(c_4 + r(c_5 + r(c_6 + r(c_7 + rc_8)))))))$
case $y < 0$
set $x := -x$

Result: $x \sim N(0,1)$
```

$a_0 = 2.50662823884$	$b - 0 = -8.47351093090$	$c - 0 = 0.3374754822726147$
$a_1 = -18.61500062529$	$b_1 = 23.08336743743$	$c_1 = 0.9761690190917186$
$a_2 = 41.39119773534$	$b_2 = -21.06224101826$	$c_2 = 0.1607979714918209$
$a_3 = -25.44106049637$	$b_3 = 3.13082909833$	$c_3 = 0.0276438810333863$
		$c_4 = 0.0038405729373609$
		$c_5 = 0.0003951896511919$
		$c_6 = 0.0000321767881768$
		$c_7 = 0.0000002888167364$
		$c_8 = 0.0000003960315187$

Table 5.1: Coefficients for Moro's inversion method

Remark 5.16. *Moro's inversion method is a simple and fast way for generating normally distributed numbers.*
Note, however, that its conditional nature makes it impossible to work with the sequence of numbers of low discrepancy in matrix form. This turns out to be a big disadvantage when working with MATLAB since it increases the time needed for computing normally distributed sequences of numbers with low discrepancy significantly.

5.3.2 Transformations

The inversion method is not the only way to generate Gaussian distributed numbers. Another approach would be to use transformations between random variables.

Theorem 5.17. *Suppose X is a random variable with distribution function F and density function $f(x)$ on the set $A = \{x \in \mathbb{R}^n : f(x) > 0\}$. Further assume the transformation $h : A \to B = h(A)$ to be invertible (strictly continuous) with continuous inverse h^{-1}. Then $Y := h(X)$ is a random variable with distribution $F(h^{-1}(y))$ and density*

$$y \mapsto f(h^{-1}(y)) \left| det \frac{dh^{-1}(y)}{dy} \right| . \tag{5.5}$$

Proof. We will only give an outline of the proof for the scalar case $n = 1$ here (for a more general proof we refer to [5]):

$P(h(X) \leq y) = P(X \leq h^{-1}(y)) = F(h^{-1}(y))$, thus Y has the distribution $F(h^{-1}(y))$. Since h^{-1} is absolutely continuous, the density of $Y = h(X)$ is equal to the derivative of the distribution function almost everywhere. This assertion is implied by the result we get when evaluating the derivative $\frac{dF(h^{-1}(y)}{dy}$ with the chain rule. □

In case $n = 1$ and $f(x) = 1$ we get the uniform distribution. What we search for is a transformation $y = h(x)$ such that the density in Equation (5.5) is identical to the density of the normal distribution:

$$1\left|\frac{dh^{-1}(y)}{dy}\right| = \frac{1}{\sqrt{2\pi}}e^{\frac{-y^2}{2}} \tag{5.6}$$

This is a common differential equation for h^{-1} which cannot be expressed in closed form. However, it is possible to find a closed-form expression for the case $n = 2$. Applying Theorem 5.17 on $A = [0,1]^2$ and $f(x) = 1$, $x \in A$, we can perform the transformation $y = h(x)$ with

$$h(x) = \begin{pmatrix} \sqrt{-2\ln x_1}\cos(2\pi x_2) \\ \sqrt{-2\ln x_1}\sin(2\pi x_2) \end{pmatrix}, \; x = (x_1, x_2)^T \in A. \tag{5.7}$$

The inverse function his given by

$$h^{-1}(y) = \begin{pmatrix} e^{\frac{|y|^2}{2}} \\ \arctan\frac{(y_2/y_1)}{2\pi} \end{pmatrix}, \; y = (y_1, y_2)^T. \tag{5.8}$$

For the determinant we get

$$\left|det\left(\frac{dh^{-1}}{dy}\right)\right| = \left|det\begin{pmatrix} -y_1x_1 & -y_2x_1 \\ \frac{1}{2\pi}\frac{-y_2/y_1^2}{1+y_2^2/y_1^2} & \frac{1}{2\pi}\frac{1/y_1}{1+y_2^2/y_1^2} \end{pmatrix}\right| = \frac{x_1}{2\pi} = \frac{1}{2\pi}e^{-|y|^2/2}, \tag{5.9}$$

which is the density of the standard normal distribution in $\mathbb{R}^2$. The two components of the vector y are independent. This means that $h(X)$ has a standard normal distribution if X is uniformly distributed on $[0,1]$. The following algorithm for the Boxmuller method is based on this fact.

Algorithm 5.18. ***Boxmuller algorithm***

Step 1
Generate $U_1 \sim U[0,1]$ and $U_2 \sim U[0,1]$ by using methods we introduced in the first section of this chapter.

Step 2
Calculate $\alpha := 2\pi U_2$, $\beta := \sqrt{-2\ln U_1}$

Step 3
Calculate the standard normally distributed numbers

$$Z_1 := \beta\cos\alpha$$

and

$$Z_2 := \beta\sin\alpha.$$

Remark 5.19. *It is important to make sure that the uniformly distributed random numbers do not have any structure because this structure would be transformed as well.*

The computational costs for the Boxmuller method are quite high since the trigonometrical functions have to be evaluated each time we generate a new pair of normally distributed random variables. This can be avoided using a variant introduced by George Marsaglia who also contributed to MATLAB's current random number generator. This method prepares the input of the Boxmuller method via polar coordinates transformation:
Random numbers $U_i \sim [0,1]$ are used for generating $V_i = 2U_i - 1 \sim U[-1,1]$. Pairs of values (V_1, V_2) that fulfill $V_1^2 + V_2^2 < 1$ are then uniformly distributed on the unit disk $D := \{V_1^2 + V_2^2 < 1\}$ with density $f(V_1, V_2) = \frac{1}{\pi}$. We will only accept those points (V_1, V_2) and apply the transformation

$$\begin{pmatrix} V_1 \\ V_2 \end{pmatrix} \mapsto \begin{pmatrix} W_1 \\ W_2 \end{pmatrix} = \begin{pmatrix} V_1^2 + V_2^2 \\ \frac{1}{2\pi} \arctan(\left(\frac{V_2}{V_1}\right)) \end{pmatrix}$$

to generate uniformly distributed points (W_1, W_2) on the unit square $[0,1]^2$.
Using the relations $W_1 = V_1^2 + V_2^2$ and $W_2 = \frac{1}{2\pi} \arctan(\left(\frac{V_2}{V_1}\right)$, we get

$$\cos 2\pi W_2 = \frac{V_1}{\sqrt{V_1^2 + V_2^2}} \text{ and } \sin 2\pi W_2 = \frac{V_2}{\sqrt{V_1^2 + V_2^2}}$$

For those variables trigonometric functions are no longer necessary in the Boxmuller method. The variant of Marsaglia is summed up in the following polar algorithm:

Algorithm 5.20. ***Polar Algorithm***

Step 1
Generate random numbers $U_1, U_2 \sim U(0,1)$.

Step 2
Calculate $V_i := 2U_i - 1 \sim U[-1,1]$ until $W := (V_1^2 + V_2^2 < 1$

Step 3
Calculate the standard normally distributed numbers

$$Z_1 := V_1 \sqrt{-2\ln(W_1)/W_1}$$

and

$$Z_2 := V_2 \sqrt{-2\ln(W_1)/W_1}.$$

Remark 5.21. *The probability for* $V_1^2 + V_2^2 < 1$ *equals the ratio between the volume of the unit disc* $\frac{\pi}{4}$ *and the volume of the unit square* 1, *which is about* 0.785. *This means that* $1 - \frac{\pi}{4} \approx 21\%$ *of all pairs* $(V_1, V_2) \sim U[0,1]$ *have to be rejected. However, Marsaglia's polar method is still more efficient than the Box-Mueller method.*

5.3.3 Comparison of methods

In this section we want to compare the accuracy of the two classes of methods we introduced for generating Gaussian distributed numbers out of uniformly distributed ones. For this we plotted histograms of the distribution of different samples of uniformly distributed numbers and then applied Moro's inversion method and the Boxmuller method on each sequence. The results can be seen in Figures 5.7 - 5.10.
Additionally we want to compare this outcome with the MATLAB function 'randn' which uses by far the shortest computational time.

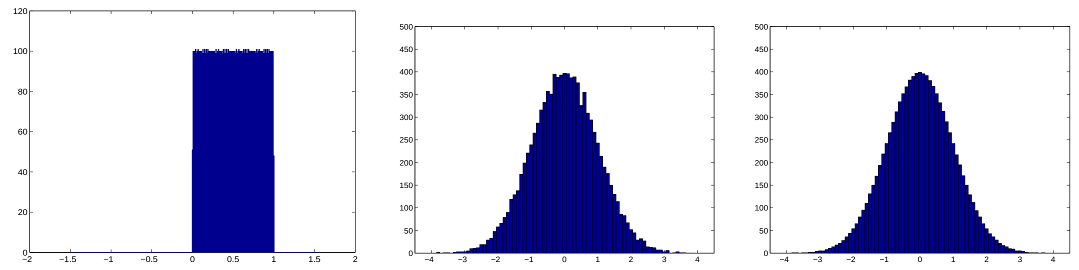


Figure 5.7: The first 10000 numbers of the Halton sequence uniformly distributed and transformed to a normal distribution by the Boxmuller method and the Moro algorithm.

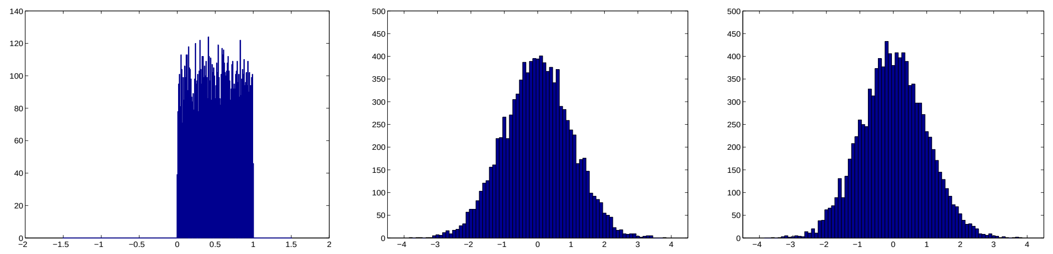


Figure 5.8: 10000 numbers computed with the MATLAB function 'rand' uniformly distributed and transformed to a normal distribution by the Boxmuller method and the Moro algorithm.

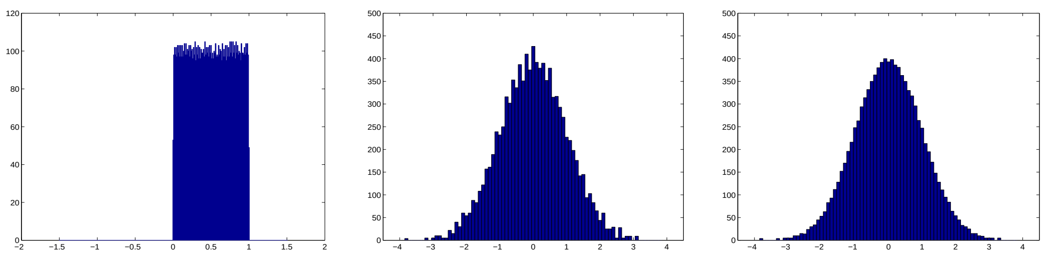


Figure 5.9: 10000 numbers computed with the linear congruential generator uniformly distributed and transformed to a normal distribution by the Boxmuller method and the Moro algorithm.

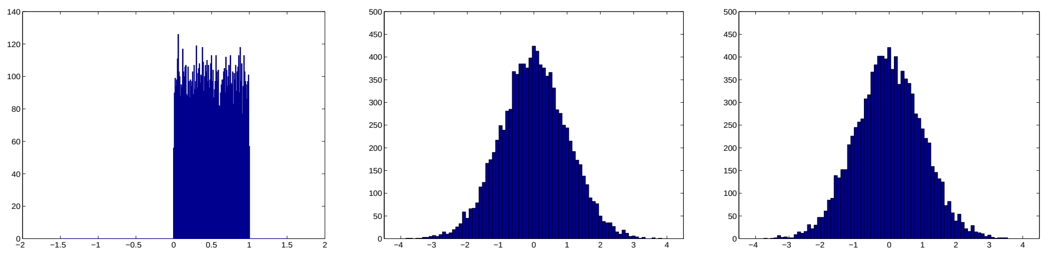


Figure 5.10: 10000 numbers computed with the lagged fibonacci generator uniformly distributed and transformed to a normal distribution by the Boxmuller method and the Moro algorithm.

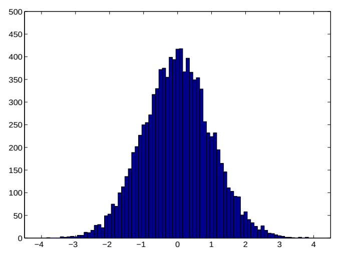


Figure 5.11: 10000 numbers computed with the MATLAB function 'randn'

We see that in case the MATLAB function 'rand' or a lagged fibonacci generator are used as sources for uniformly distributed numbers, the quality of the histograms generated by Moro's method and the Boxmuller algorithm is about the same. However, if a linear congruential generator or the Halton sequence is used, the histograms computed by Moro's inversion method are a lot closer to the graph of the standard normal distribution (Figure 3.1) than the ones computed with help of the Boxmuller algorithm.

Remark 5.22. *Note that although on average Moro's method yields better results than the Boxmuller algorithm, it is not always practical to implement Moro's method in*

MATLAB as already mentioned in Remark 5.16.

The fact that the Boxmuller algorithm works a lot faster than Moro's inversion method is illustrated in Table 5.2, where the time that needed by both methods for generating 30000 normally distributed numbers out of uniformly distributed ones is listed. In order to establish a standard of comparison, we also listed the time that was needed for computing the uniformly distributed numbers which the two methods use as input and which is therefore also included in the computing time taken up by each method.

	computing time for $X \sim U(0,1)$	Moro	Boxmuller
Halton Sequence	0.2524	8.6646	0.4268
MATLAB function 'rand'	0.002	7.1782	0.026

Table 5.2: Comparison: Running time of Moro's and the Boxmuller method

Remark 5.23. *The numbers listed in Table 5.2 were computed on a digital computer with an Intel Celeron M 1.4 GHz processor and* 512 *MB RAM.*
As MATLAB's function 'tic toc' for measuring the running time of its programs works like a stop watch, the results are not very accurate and vary with each run. We therefore took the average over 10 *results for the running time for each program.*

Chapter 6

Monte Carlo Method

The idea of the Monte Carlo method is to get an estimate for a certain value by computing the expected value for a large number of independent simulated samples. This concept is based on the Law of Large numbers which states that the average over a large enough number of samples approximately equals their mean.
We will introduce the concept of the Monte Carlo Method on the example of approximating the unknown expectation $E[X] = m_M$ of some general random variable X. In this simple scenario, the Monte Carlo Method consists of only two steps:

Algorithm 6.1. ***Basic Monte Carlo Method***

Step 1
Simulate a large number M of samples of X by using random number generators to compute independent random variables $X_1, X_2, ..., X_M$ with the same distribution as X.

Step 2
Approximate the mean using Equation (3.9):

$$m_M := \frac{1}{M}\sum_{i=1}^{M} X_i. \tag{6.1}$$

An approximation to $E[X]$ is called an unbiased estimator if it has the same expected value as X.
In addition to the approximation of the expected value, we are also interested in getting a feeling of how close we are to the exact solution, which can be given by a confidence interval $[a, b]$. Thereby, we want the probability for our expected value m_M to be in this confidence interval to be 95%, i.e.

$$P\left(a \leq m_M \leq b\right) = 0.95. \tag{6.2}$$

From the Central Limit Theorem we know that the sum of a large number of i.i.d. random variables will be approximately normal. Let us therefore assume that $\sum_{i=1}^{M} X_i \sim N(M\mu, M\sigma^2)$ and $(m_M - \mu) \sim N(0, \sigma^2/M)$. With Equation (3.36) we get

$$P\left(-1.96\frac{\sigma}{\sqrt{M}} \leq m_M - \mu \leq 1.96\frac{\sigma}{\sqrt{M}}\right) = 0.95. \tag{6.3}$$

Rewriting this as

$$P\left(m - 1.96\frac{\sigma}{\sqrt{M}} \leq \mu \leq m_M + 1.96\frac{\sigma}{\sqrt{M}}\right) = 0.95 \tag{6.4}$$

shows that a 95%-confidence interval for our unknown expectation μ is given by

$$\left[\mu - 1.96\frac{\sigma}{\sqrt{M}}, \mu + 1.96\frac{\sigma}{\sqrt{M}}\right]. \tag{6.5}$$

Remark 6.2. *Throughout this section we assumed the variance σ of our general random variable X to be known. This is a very general assumption, since it is not very likely for the variance to be given while the expectation has to be approximated. In praxis it is therefore often necessary to also compute an approximation of the variance*

$$v_M^2 := \frac{1}{M-1}\sum_{i=1}^{M}(X_i - m_M)^2.$$

The value v_M can then be substituted for σ in Equation (6.5).

6.1 Monte Carlo Method for option pricing

Having introduced the basic principle of the Monte Carlo simulation, we now have all tools we need for evaluating the price of an Option via simulation.
We first want to give a rough outline on how the method works before we will discuss the general algorithm of the Monte Carlo method for option pricing.

Remark 6.3. *The observations made in this chapter are restricted to European call options only. This can be done without loss of generality since the respective results for European put options can be derived by applying the put call parity given in Equation (2.3).*

As we saw in Chapter 2 the value of an option at expiry $\hat{V}_T$ can be calculated via the payoff function (2.1). Discounting that value with the factor $e^{-r(T-t)}$ leads to the option value at time t:

$$V_{S,t} = e^{-r(T-t)} V_{S,T} = e^{-r(T-t)} \max\{S_T - K, 0\}. \tag{6.6}$$

It hence suffices to know the price of the underlying asset at expiry S_T to derive the true value of an option.
The main task in pricing European option lies therefore in approximating the value of the underlying asset at time T.
In Chapter 4 we derived a mathematical model for asset price dynamics, i.e. that of the Geometric Brownian Motion (Equation (4.19)). Solving this SDE provides us with the random variable S_T which denotes the price of the underlying asset at time T.
However, we also saw that this model does not enable us to look into the future. We cannot make any real predictions about the asset price movement, all we have is a stochastic model based on past data.

The idea of the Monte Carlo Method is to overcome this problem by running a large number M of simulations. Thereby the payoff function is calculated for each of the M simulated asset price paths and an approximation of the probable price of the option at time T is then obtained by taking the geometric average of all $V(S,T)$:

$$\hat{V}(S_T, T) = E(V(S_T, T)) = \frac{1}{M} \sum_{i=1}^{M} V(S_i, T) \tag{6.7}$$

Once we know the value of our option at time T, we only need to discount that number in order to obtain the present value of the option:

$$\hat{V}(S, t) = e^{-r(T-t)} E[V(X(T; t, S)T)] \tag{6.8}$$

The algorithm of the basic Monte Carlo Method for option pricing consists of four different steps:

Algorithm 6.4. ***Monte Carlo simulation for option pricing***

Given:

- *Asset price S_t at time t,*
- *SDE which simulates the asset price movement*

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

- *Expiry date T*
- *Risk free interest rate r and*

- Payoff function $V(S_T, T)$

Step 1
Simulate M approximations $W(t, \omega_i)$, $i = 1, 2, ..., M$ of paths of a Wiener process W with the help of a random number generator as introduced in Chapter 5.

Step 2
Compute the solutions $S_i(T), i = 1, ..., M$ of the SDE that simulates the asset price movement for each path of the Wiener processes generated in Step (1).

Step 3
Approximate the mean $E[V(S_T, T)]$ using Equations (2.1) and (3.9):

$$\widetilde{E}[V(S_T, T)] = \frac{1}{M} \sum_{i=1}^{M} V(\widetilde{S}_i, T) \tag{6.9}$$

Step 4
Discount the estimator of Step 3 (6.9) in order to get an approximation of the fair option value $V(S, t)$:

$$\widetilde{V}(S, t) = e^{-rT} \widetilde{E}[V(S_T, T)].$$

Step 5
Compute a 95%-confidence interval

$$\left[E[V(S,T)] - 1.96 \frac{\sqrt{Var[V(S,T)]}}{\sqrt{M}},\ E[V(S,T)] + 1.96 \frac{\sqrt{Var[V(S,T)]}}{\sqrt{M}} \right].$$

Remark 6.5. *Since the payoff function of a vanilla European option is not path dependent, we are only interested in the price of the underlying at time T. It is therefore not necessary to approximate the asset price paths and with them the Wiener process for any time steps between $t = 0$ and $t = T$. Hence Steps 1 and 2 in Algorithm 6.4 can be simplified: It suffices to use Ito's solution for the SDE modeling the Geometric Brownian Motion:*

$$S_T = S_0 e^{((r-\delta)-1/2\sigma^2)T + \sigma\sqrt{T}\xi} \tag{6.10}$$

where ξ is a standard Gaussian random Variable

If the payoff function of the asset is path-dependent, however, or if the model for the price dynamics of the underlying gets more complicated, numerical methods are necessary. An example of such a method would be the Euler-Maruyama algorithm described in Appendix B. An Example of such a case is given in Section 6.5.3 when we introduce a model for stochastic volatility.

Let us now test the accuracy of the method.

Example 6.6. *We want to test the Monte Carlo simulation by approximating the value of a European call option with strike price* $K = 100$*, risk-free interest rate* $r = 0.07$*, volatility* $\sigma = 0.3$*, dividend rate* $\delta = 0.1$*, expiry time* $T = 1$ *and asset price at time* $t_0 = 0$ $S_0 = 120$*.*
Numerical results are listed in Table 6.1.

Remark 6.7. *The Black-Scholes value* $V_{exakt} = 21.2061$ *was used a reference for calculating the relative error of the method. A rough outline of the Black-Scholes Equation for option valuation is given in Appendix A.*

number of simulations n	Monte Carlo option value	95% confidence interval	rel error
100	25.7652	$[19.9381, 31.5922]$	21.50%
200	25.0300	$[20.4475, 29.6126]$	18.03%
500	24.0296	$[21.2438, 26.8155]$	13.31%
$1,000$	22.8681	$[21.0395, 24.6968]$	7.84%
$2,000$	22.2387	$[20.9780, 23.4994]$	4.87%
$5,000$	21.6153	$[20.8333, 22.3974]$	1.93%
$10,000$	21.5945	$[21.0403, 22.1486]$	1.83%
$20,000$	21.4664	$[21.0783, 21.8545]$	1.23%
$30,000$	21.4060	$[21.0921, 21.7198]$	0.94%

Table 6.1: MC simulation for $S_0 = 120$

We see that for $M < 5000$ *the accuracy is quite bad although it improves steadily. The relative error still decreases for a number of simulation runs higher than* 5000*, but with a very slow rate of 'convergence'. This property can also be observed in Figure 6.1 below where a plot for the relative error of the method as well as for the* 95%*-confidence interval is depicted.*
Before we examine the relationship between the number of simulation runs and the accuracy of the Monte Carlo simulation in the next section, we first want to investigate the influence of the initial price of the underlying asset S_0 *on the accuracy of the method. Let us therefore repeat the simulation with* $S_0 = 100$*. Numerical results can be seen in Table 6.2, where the Black-Scholes value* $V_{exakt} = 9.6296$ *was used as a reference.*
We see that the accuracy of the method worsens slightly. However, this can be explained by the fact that the same absolute error leads to a higher relative error as the value we use as a reference gets smaller which is the case here. Table 6.3 lists the results of another Monte Carlo simulation with $S_0 = 80$ *and according Black-Scholes value of* $V_{exakt} = 2.7222$*.*

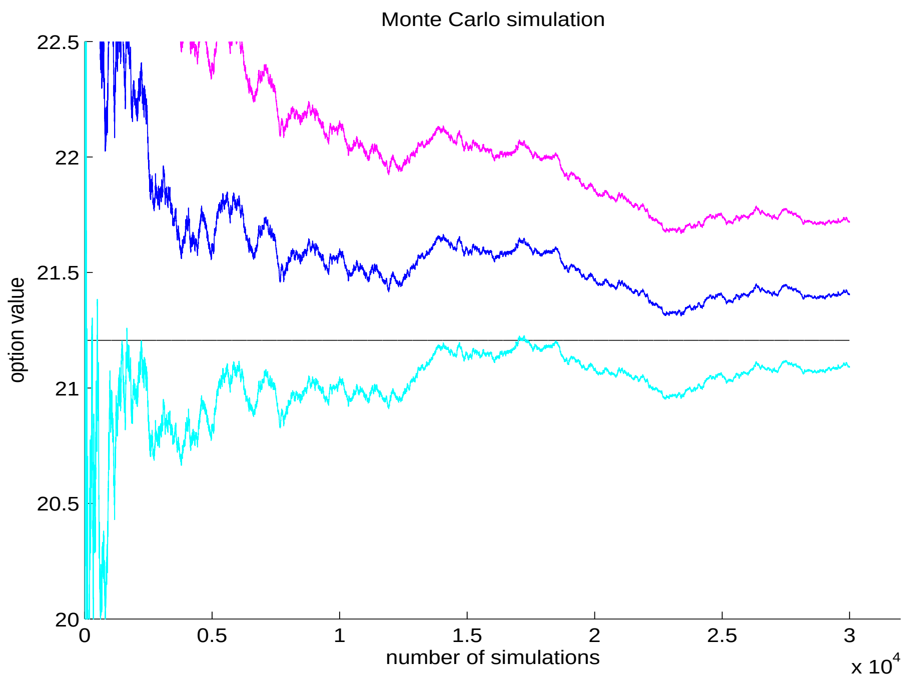


Figure 6.1: Confidence interval for a European call option with $S_0 = 120$

number of simulations n	Monte Carlo option value	95% confidence interval	rel error
100	12.5166	$[8.6316, 16.4016]$	29.98%
200	12.7475	$[9.6009, 15.8941]$	32.38%
500	11.8515	$[9.9436, 13.7594]$	23.07%
$1,000$	10.8294	$[9.5982, 12.0606]$	12.46%
$2,000$	10.4437	$[9.6029, 11.2845]$	8.45%
$5,000$	9.9686	$[9.4481, 10.4892]$	3.52%
$10,000$	9.9496	$[9.5802, 10.3190]$	3.32%
$20,000$	9.8356	$[9.5776, 10.0936]$	2.14%
$30,000$	9.7520	$[9.5439, 9.9600]$	1.27%

Table 6.2: MC simulation for $S_0 = 100$

6.2 Convergence of the Monte Carlo Method

We saw that the number of simulations M has to be very high in order to achieve a satisfying accuracy of the approximated option value. We also saw that the 'rate of convergence' slows down as the number of simulation runs rises. In this section, we want to explain this convergence behavior of the Monte Carlo method.

Let us therefore consider the example of stochastic integration with a Monte Carlo

number of simulations n	Monte Carlo option value	95% confidence interval	rel error
100	3.7218	$[1.8193, 5.6243]$	36.72%
200	4.2835	$[2.5503, 6.0167]$	57.35%
500	3.8468	$[2.7989, 4.8947]$	41.31%
1000	3.2649	$[2.6209, 3.9088]$	19.94%
2000	3.0678	$[2.6390, 3.4965]$	12.70%
5000	2.9156	$[2.6512, 3.1801]$	7.10%
10000	2.9301	$[2.7414, 3.1187]$	7.64%
20000	2.8564	$[2.7254, 2.9874]$	4.93%
30000	2.7878	$[2.6828, 2.8927]$	2.41%

Table 6.3: MC simulation for $S_0 = 80$

approximation of

$$\theta_n = \frac{1}{n}\sum_{k=1}^{n}\phi(X_k) \tag{6.11}$$

and an exact value

$$\theta = \int_\Omega g(x)dx = \int_\Omega \frac{g(x)}{f(x)} f(x)dx = E[\phi(X_k)], \tag{6.12}$$

where $\phi(x) = \frac{g(x)}{f(x)}$ and X_k are independent samples of a random variable with density function f and $Var[\phi(X_k] = \sigma^2$ for all $k = 1, ..., n$.
Since the absolute error given by $|\theta_n - \theta|$ is again a random variable, it is not possible to give a deterministic error bound. Instead we have to work with a confidence interval for the error, which is why we need to calculate the mean and variance of our approximated value:

$$E[\theta_n] = E\left[\frac{1}{n}\sum_{k=1}^{n}\phi(X_k)\right] = \frac{1}{n}\sum_{k=1}^{n}E[\theta(X_k)] = \theta \tag{6.13}$$

and

$$\begin{aligned} Var[\theta_n] &= E\left[\left(\frac{1}{n}\sum_{k=1}^{n}E[\theta(X_k)] - E[\frac{1}{n}\sum_{k=1}^{n}\theta(X_k)]\right)^2\right] = \qquad (6.14) \\ &= \frac{1}{n^2}E\left[\sum_{k=1}^{n}(\phi(X_k) - E[\phi(X_k)])^2\right] = \\ &= \frac{1}{n^2}\sum_{k=1}^{n}Var[\phi(X_k)] = \frac{\sigma^2}{n}. \end{aligned}$$

With the Chebyshev Inequality (3.14) for $\epsilon := \frac{\sigma}{\sqrt{\epsilon n}}$ and $X = \theta_n$ we get

$$P\left(|\theta_n - \theta| \geq \frac{\sigma}{\sqrt{\epsilon n}}\right) \leq \epsilon n$$

which can be written as

$$P\left(|\theta_n - \theta| < \frac{\sigma}{\sqrt{\epsilon n}}\right) \geq 1 - \epsilon n. \tag{6.15}$$

We thus see that the absolute error $|\theta_n - \theta|$ is proportional to $\frac{\sigma}{\sqrt{n}}$. This means that the number of simulations M has to be increased by the factor 100 in order to decrease the error by 10. Improving the accuracy this way is thus connected with high computational costs which is why we want to find an alternative way for decreasing the error in Equation (6.15). The only other parameter that influences the accuracy is the variance σ, which is why a logical approach could be to try to work with a smaller variance.

6.3 Variance reduction

As mentioned above, methods of variance reduction are efficient tools for improving the speed of the Monte Carlo simulation. If we succeed at reducing the variance of V by a factor $C < 1$, we can simulate confidence intervals of the same widths for C times less work. However, we still have to take into consideration the costs for computing this new random variable. In the following, we will introduce a method which is widely applicable, easy to implement and computationally cheap:

Antithetic variates

This method is based on the idea of replacing the set of (i.i.d) normally distributed random variables V_T in the Monte Carlo simulation by another set of normally distributed random variables which have the same mean but a smaller variance. Hereby, we use the fact that if a random variable satisfies $Z \sim N(0,1)$, then $-Z \sim N(0,1)$.
The method works as follows:

Algorithm 6.8. ***Antithetic variates for the Monte Carlo simulation***

Step 1
Compute M values for V_T as described in Algorithm 6.4, using a random variable $Z \sim N(0,1)$.

Step 2
Generate V_T^- the same way, but employ a random variable that satisfies $-Z \sim N(0,1)$

this time instead.

Step 3
Substitute each of the M V_T in Algorithm 6.4 with the antithetic variate

$$V_T^{AV} = \frac{1}{2}(V_T + V_T^-) \tag{6.16}$$

This method is based on the assumption that

$$Var[V_T^{AV}] \leq Var[V_T] \tag{6.17}$$

which we want to prove now.

Proof. Using Equation (3.27), we get

$$\begin{aligned} Var[V_T^{AV}] &= Var[\frac{1}{2}(V_T + V_T^-)] = \frac{1}{4}Var[(V_T + V_T^-)] = & (6.18)\\ &= \frac{1}{4}Var[V_T] + \frac{1}{4}Var[V_T^-] + \frac{1}{2}Cov[V_T, V_T^-]. & (6.19)\end{aligned}$$

With the estimate (3.28), we see that

$$Var[V_T^{AV}] \leq \frac{1}{4}Var[V_T] + \frac{1}{4}Var[V_T^-] + \frac{1}{4}(Var[V_T] + Var[V_T^-]). \tag{6.20}$$

Using $Var[\theta_n] = Var[\theta_n^-]$, we get

$$Var[V_T^{AV}] \leq Var[V_T] \tag{6.21}$$

$\square$

As we see from (6.18), we get best results for negatively correlated V_T and V_T^-. Judging from the method we used for generating V_T and V_T^-, chances are good for their covariance to be negative.
Since the additional computational costs for computing V_T^{AV} are comparatively small in this case, the method of antithetic variates seems to be a good approach for improving the efficiency of the Monte Carlo simulation for option valuation. From Inequality (6.21) we see that in the worst possible case the variance of V_T^{AV} equals the variance of V_T. The highest possible loss in efficiency is therefore limited by the additional costs of the computation of the antithetic variate.

Example 6.9. *We use the same parameters as in Example 6.6 and approximate the option value by Monte Carlo simulation with antithetic variates.*

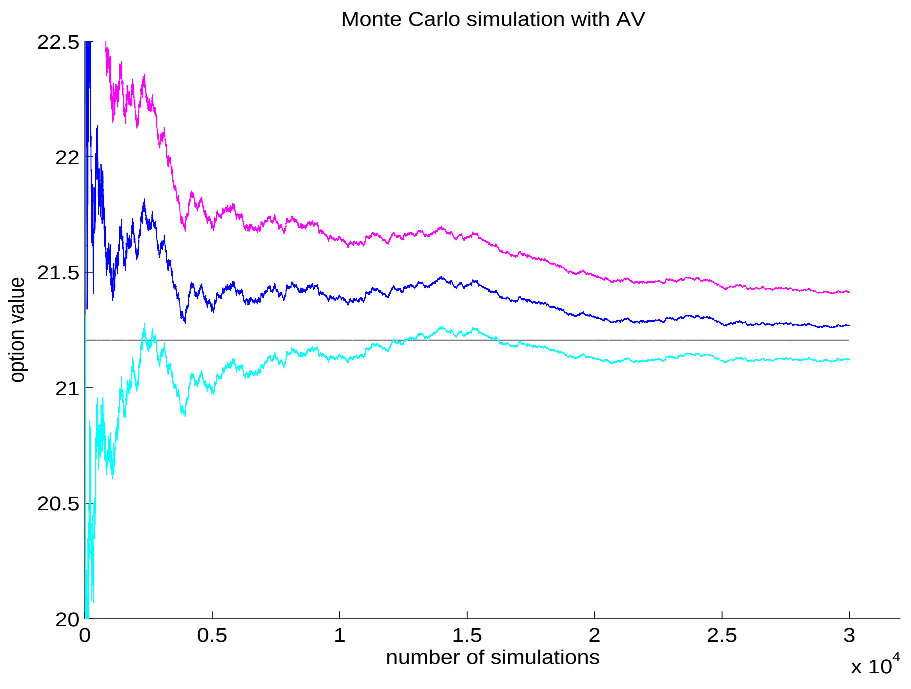


Figure 6.2: Confidence interval for a European call with $S_0 = 120$ and AV

As we saw in Figure 6.2, not only the efficiency of the method improves significantly, but the widths of the confidence interval gets a lot smaller too. This can be explained by looking at the formula for calculating the confidence interval. Its size is directly proportional to the inverse square root of the number of simulation runs and to the standard deviation of the random variable which in our case is the expected payoff.
Using the method of antithetic variates reduces the number of 5000 *simulation runs for getting an acceptable level of accuracy with standard Monte Carlo to* 1000*. Hence using antithetic variates either delivers better results in roughly the same time or reduces the number of simulation runs that are needed for a certain degree of accuracy.*
Numerical values are included in Table 6.4 below.

6.4 Quasi-Monte Carlo

In our estimation of the absolute error of Monte Carlo simulation we saw that the accuracy of the method depends on the number of simulation runs as well as on the variance of our random variables denoting the option value at expiry V_T. What the estimation (6.15) does not reflect, however, is the fact that the efficiency of a Monte Carlo method also depends on the quality of the samples of random numbers.
So far we have worked with pseudo-random numbers, which are to a certain degree very similar to true random numbers. As a consequence, they are not equidistantly distributed, however, because that would stand in contrast to true randomness. This is

number of simulations n	Monte Carlo option value	95% confidence interval	rel error
100	22.1455	$[19.8573, 24.4337]$	4.43%
200	22.6566	$[20.7774, 24.5359]$	6.84%
500	21.9999	$[20.8339, 23.1659]$	3.74%
1000	21.5650	$[20.7544, 22.3756]$	1.69%
2000	21.5967	$[21.0207, 22.1727]$	1.84%
5000	21.3486	$[20.9906, 21.7066]$	0.67%
10000	21.3977	$[21.1429, 21.6525]$	0.90%
20000	21.3052	$[21.1265, 21.4839]$	0.47%
30000	21.2690	$[21.1233, 21.4148]$	0.30%

Table 6.4: MC simulation with AV for $S_0 = 120$

why we introduced so called low discrepancy sequences in Chapter 5, that is a set of quasi-random numbers that are a lot more equidistantly distributed than the numbers that can be computed with pseudo-random number generators.
In this section we want to test the efficiency of the so called Quasi-Monte-Carlo method, i.e. a standard Monte Carlo simulation that uses sequences of quasi-random numbers such as the Halton Sequence.

Since the Monte Carlo simulation works on the basis of standard normally distributed random numbers, we first have to transform the uniformly distributed Halton Sequence such that it has a standard Gaussian distribution. We want to compare the two methods we introduced for transforming the distribution of random variables here, Moro's method and the Boxmuller algorithm.

Remark 6.10. *As we saw before, Moro's method is significantly slower than the Boxmuller algorithm when implemented in MATLAB because the sequence cannot be worked with in matrix form. This is not true, however for other means of implementation such as C++.*
We therefore want to examine the quality of both transformation methods independently of the aspect of computational costs.

Although literature suggests to use Moro's inversion algorithm for that in order to yield optimal results, Figure 6.3 shows that using the Boxmuller algorithm leads to a higher accuracy of the Monte Carlo method.

Remark 6.11. *Since it is also computationally cheaper to work with the Boxmuller method in MATLAB, this is the algorithm we will use in the following to examine the improvement in accuracy yielded by the use of low discrepancy sequences.*

Example 6.12. *Continuing Examples 6.6 and 6.9, we now compute the value of a European call option via Quasi-Monte-Carlo simulation with and without using anti-*

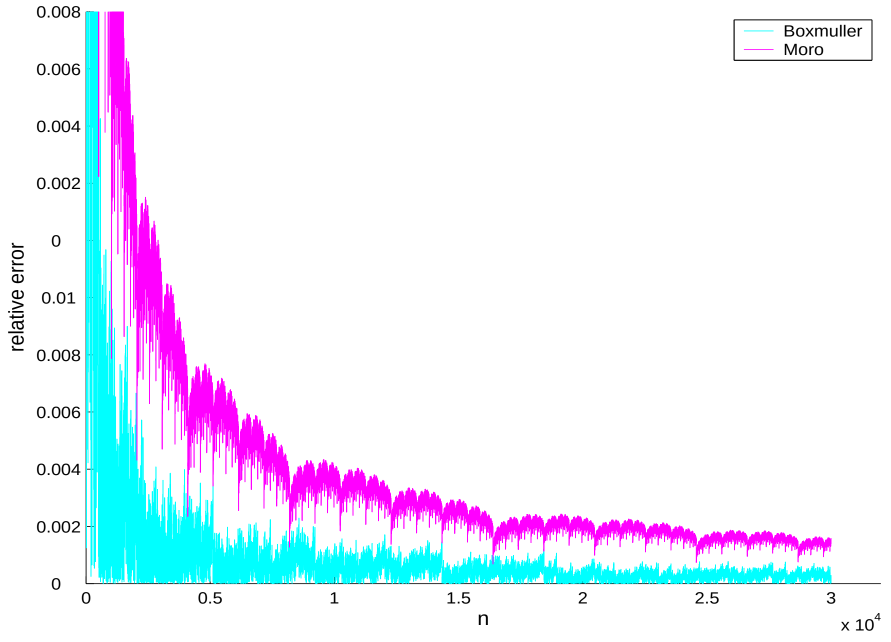


Figure 6.3: Comparison of the relative error with Boxmuller and Moro

thetic variates for $S_0 = 120$.
Figure 6.4 shows that the speed of convergence increases significantly in comparison to simple Monte Carlo simulation shown in Figure 6.1.
We also see that although the the confidence interval can indeed only be shrunk by using variance reduction methods, the accuracy of the interval improves by a large degree. While with simple Monte Carlo approximation the true option value was still situated outside of the 95%*-confidence interval at times, this is not the case with Quasi-Monte-Carlo simulation anymore.*
Table 6.5 gives numerical support for this observation. Again, the true option value derived with the Black-Scholes Equation is given by $V_{exakt} = 21.2061$.

6.5 Volatility models

When introducing the market model in Chapter 2, we mentioned that some of the rather restricting assumptions such as a constant risk-free interest rate or volatility and the fact that the underlying asset does not pay any dividends could be relaxed or dropped later on. This is exactly what we will do in this section. Note that so far it has not been necessary to use simulation methods for deriving the option value, other methods such as Black-Scholes or binomial methods could have done the job with less computational costs. In cases of more complicated models however, those methods can not be applied anymore and this is where simulation methods such as Monte Carlo

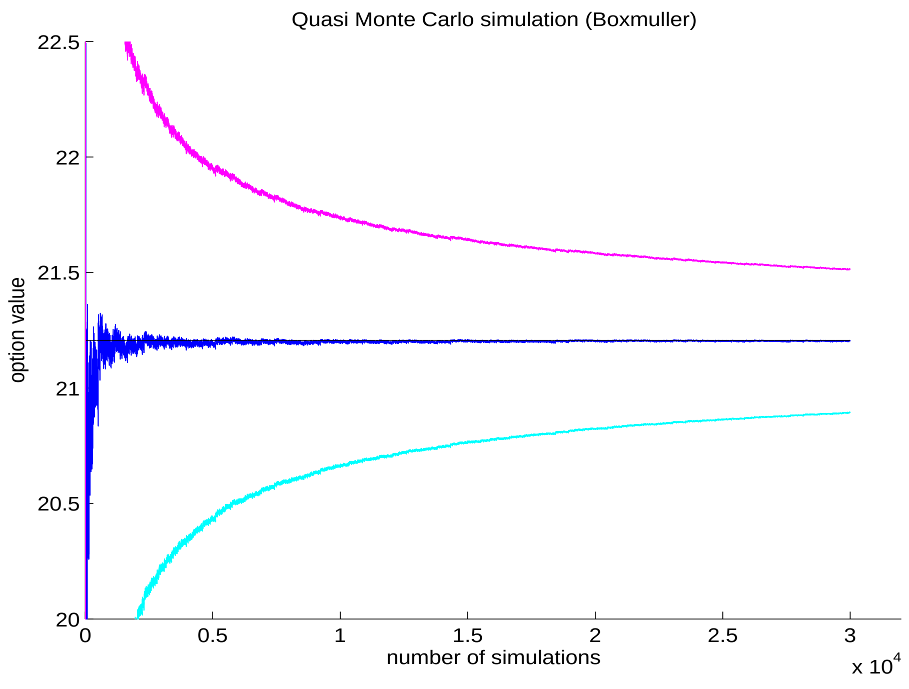


Figure 6.4: Confidence interval for a European call with $S_0 = 120$ (QMC)

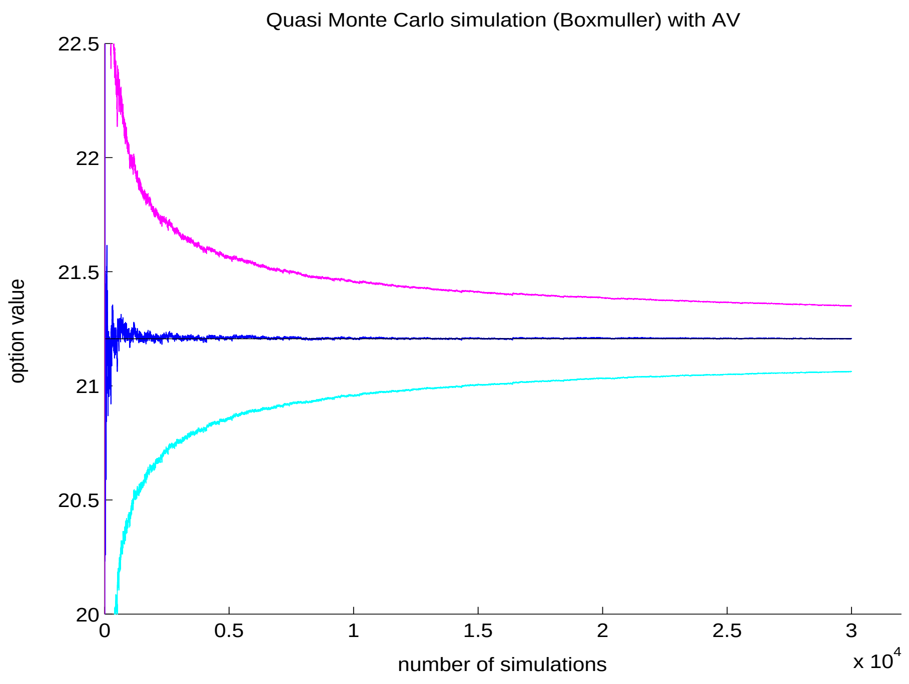


Figure 6.5: Confidence interval for a European call with $S_0 = 120$ (QMC AV)

n	QMC value	95% confidence interval	rel error	QMC value with AV	95% confidence interval	rel error
100	20.6538	$[15.4833, 25.8242]$	2.60%	21.4328	$[19.0106, 23.8551]$	1.07%
200	20.913	$[17.3287, 24.6540]$	1.01%	21.1163	$[19.4527, 22.7798]$	0.42%
500	20.9405	$[18.6195, 23.2615]$	1.25%	21.1191	$[20.0337, 22.2045]$	0.41%
1000	21.1343	$[19.4407, 22.8280]$	0.34%	21.2047	$[20.4134, 21.9960]$	0.01%
2000	21.1719	$[19.9737, 22.3701]$	0.16%	21.2070	$[20.6505, 21.7635]$	0.01%
5000	21.1958	$[20.4354, 21.9522]$	0.06%	21.2084	$[20.8558, 21.5611]$	0.01%
10000	21.1963	$[20.6594, 21.7331]$	0.05%	21.2063	$[20.9571, 21.4555]$	0.001%
20000	21.2036	$[20.8236, 21.5836]$	0.01%	21.2089	$[21.0323, 21.3855]$	0.01%
30000	21.2064	$[20.8961, 21.5168]$	0.001%	21.2073	$[21.0633, 21.3514]$	0.01%

Table 6.5: Quasi Monte Carlo simulation for $S_0 = 120$

Simulation are needed.

As already mentioned, the volatility σ determines the average change in asset price movement of the underlying. It has a huge impact on the simulation of asset prices and thus a good estimate of this parameter has to be available in order to get accurate results when pricing options. We will introduce two models for deriving such an estimate here.

6.5.1 Historical volatility

Since we do not know the future development of the asset price, we have to rely on values known from the past. Assuming that we have access to data on the previous behavior of the underlying, the historical volatility σ_{hist} can be derived as the annualized standard deviation of the logarithmic asset price changes.
Let S_i be the value of the asset at time t_i and N be the average number of official trading days in a year.
The historical volatility is then defined as

$$\sigma_{hist} = \sqrt{N}\left(\frac{1}{n-1}\sum_{i=1}^{n-1}(y_i - \bar{y})^2\right), \tag{6.22}$$

where

$$y_i = \ln S_{i+1} - \ln S_i,\, i = 1, ..., n-1,\, \bar{y} = \frac{1}{n}\sum_{i=1}^{n-1} y_i. \tag{6.23}$$

Unfortunately, volatility is not constant in reality. To emphasize the more recent values it is also possible to weight the respective asset prices higher than the ones that lie further in the past.

Another reason why the definition of historic volatility is not precise is the fact that the values vary with the time series they are derived from.

Example 6.13. *To illustrate how this method works, we will now calculate the historical volatility for the Nokia asset whose price path over the past year is depicted on the left hand side of Figure 4.3.*
Thereby, we take the closing value of the asset at every Monday for a year.
Proceeding as described above leads to the value for the historical volatility of the Nokia asset of $\sigma_{hist} = 0.3679$.

The code for the MATLAB program we used for calculating this value is given in Appendix C.2.6.

We want to introduce another model for deriving a value for volatility here which delivers a unique solution if certain conditions are met. It is called the implied volatility:

6.5.2 Implied volatility

The option value depends on S, K, r, T and σ. Assuming all those parameters except for the volatility σ are known and constant, it is possible to derive σ from the Black-Scholes Equation (see Appendix A) if the option value V^* is known at time $t = 0$. The option value is now a function of σ only, and can thus be written as $V(\sigma)$. For the implied volatility σ^* we have the equality $V(\sigma^*) = V^*$. Unfortunately, this requires solving a nonlinear equation, and hence numerical methods are needed. The bisection method or Newton's method would work here.

6.5.3 Stochastic Volatility

As mentioned in the previous section, volatility is not constant in reality. This is why we will examine the example of a more complicated model now: an asset price movement where the volatility follows an SDE itself.
So far our model for simulating the asset price movement has been relatively simple. Even when we allowed continuous dividends to be payed on the underlying asset, it was still possible to use Ito's Lemma for calculating a solution to the SDE that described the asset price movement. In the case of multiple stochastic deviates, such as a stochastic risk-free interest rate or a stochastic volatility, however, it gets more complicated. The dimension of the problem increases and we need to rely on numerical methods for deriving a solution to the SDE such as the Euler-Maruyama-Scheme which is described in Appendix B.
In such a case, the option value cannot simply be derived by using the Black-Scholes formulae (Appendix) anymore either. This is where simulation approaches such as the Monte Carlo method are very useful.

The new asset price model with stochastic volatility results in a three dimensional stochastic process

$$X(t) = (S_t, \sigma(t), \zeta(t))^T \tag{6.24}$$

defined by the equations

$$dS_t = rS_t dt + \sigma(t) S_t dW_t^1 \tag{6.25}$$

$$d\sigma(t) = -(-\sigma(t) - \zeta(t))dt + \alpha\sigma(t)dW_t^2 \tag{6.26}$$

$$d\zeta(t) = \beta(\sigma(t) - \zeta(t))dt, \tag{6.27}$$

where $\alpha > 0$ and $\beta \geq 0$ are known parameters.

The drift term in (6.26) is negative for $\sigma < \zeta$ and positive for $\sigma > \zeta$, which causes the volatility σ to follow ζ. In order to understand the meaning of that, let us now take a closer look at ζ. This is not a stochastic variable at all, it rather follows an ordinary non-homogenous linear differential equation whose exact solution is given by

$$\zeta(t) = e^{-\beta t}\zeta(0) + \int_0^t e^{-\beta(t-s)}\sigma(s, \omega)ds. \tag{6.28}$$

Hence ζ is nothing else but the mean of σ weighted by $e^{-\beta(t-s)}$, which means that ζ causes a pull of σ to its weighted mean. This effect is called mean reversion.

Example 6.14. *To illustrate this, we will now compute the asset price movement (Figure 6.8) and the value of a European put (Figure 6.7) with constant (green) and stochastic (blue) volatility as well as a realization of the volatility tandem* σ_t, ζ_t *(Figure 6.6) for* $0 \leq t \leq 1, \Delta t = 0.004$ *with parameters* $S_0 = 110, K = 100, \sigma_0 = \sigma_{const} = 0.4, \alpha = 0.3, \beta = 10, r = 0.1$.

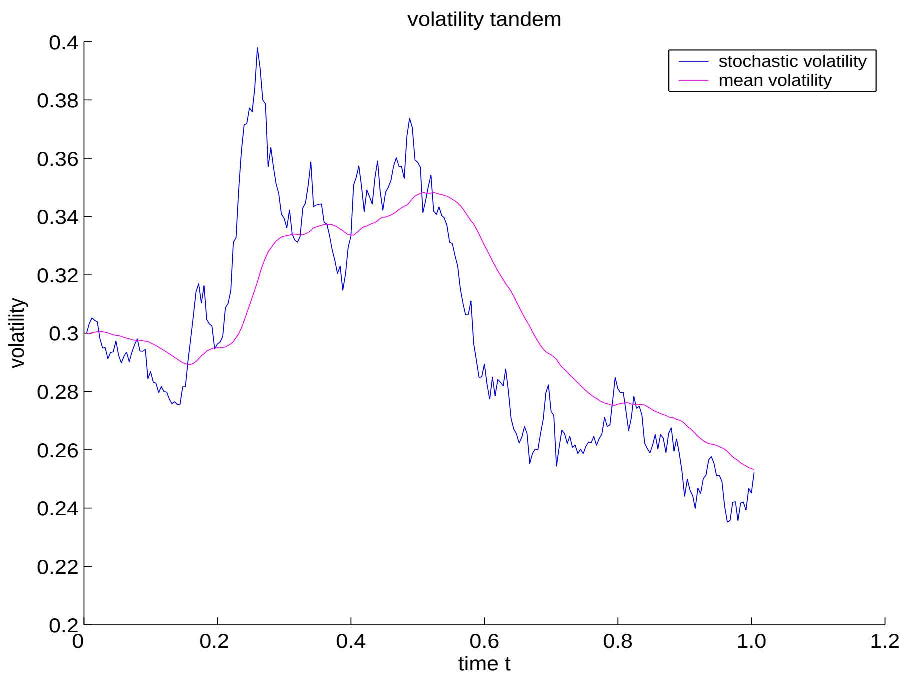


Figure 6.6: Mean-reverting volatility

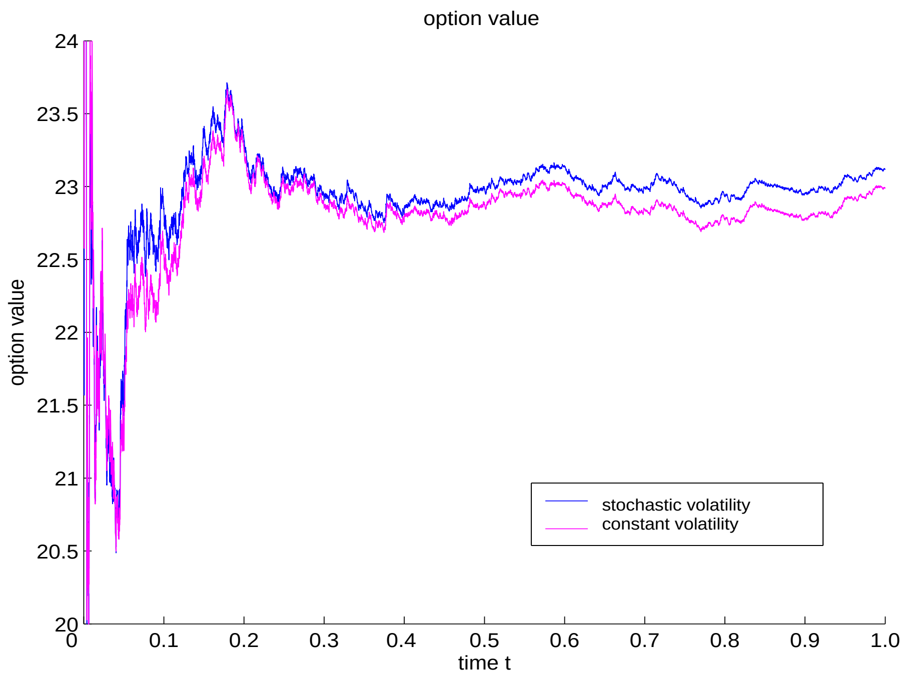


Figure 6.7: Option value with constant and mean-reverting volatility

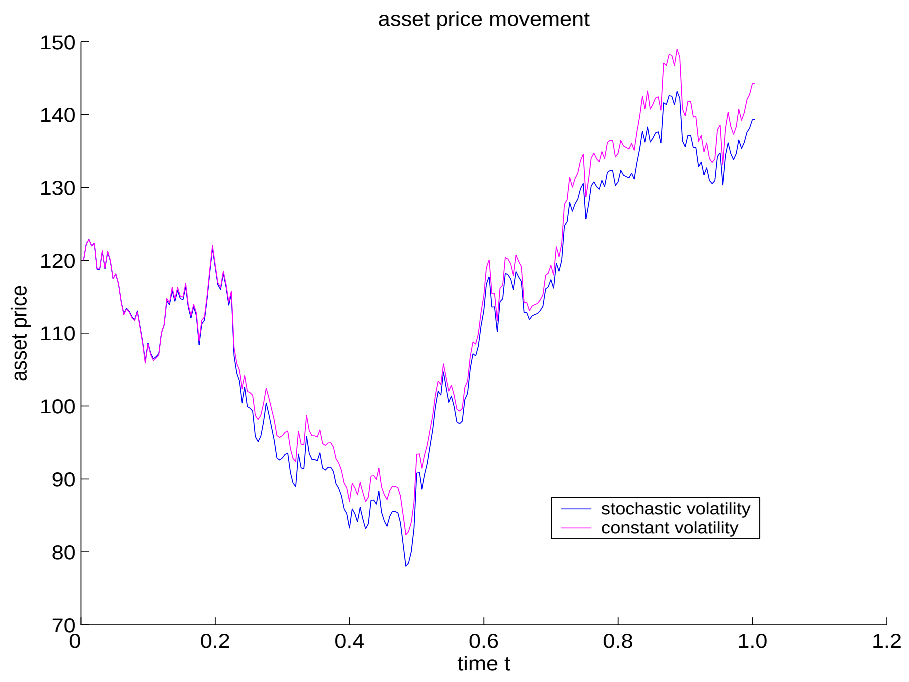


Figure 6.8: Asset price movement with constant and mean-reverting volatility

6.6 Results

Tables 6.6 and 6.7 list the results for different numbers of simulations as seen in Figure 6.9 with parameters $K = 100, r = 0.1, \sigma = 0.4, T = 1, S_0 = 120$.

method	relative error	n
simple Monte Carlo	0.2938%	80000
MC with antithetic variates	0.2970%	30000
QMC (Boxmuller)	0.2791%	750
QMC with AV (Boxmuller)	0.2768%	180

Table 6.6: Different numbers of simulations for the same level of accuracy

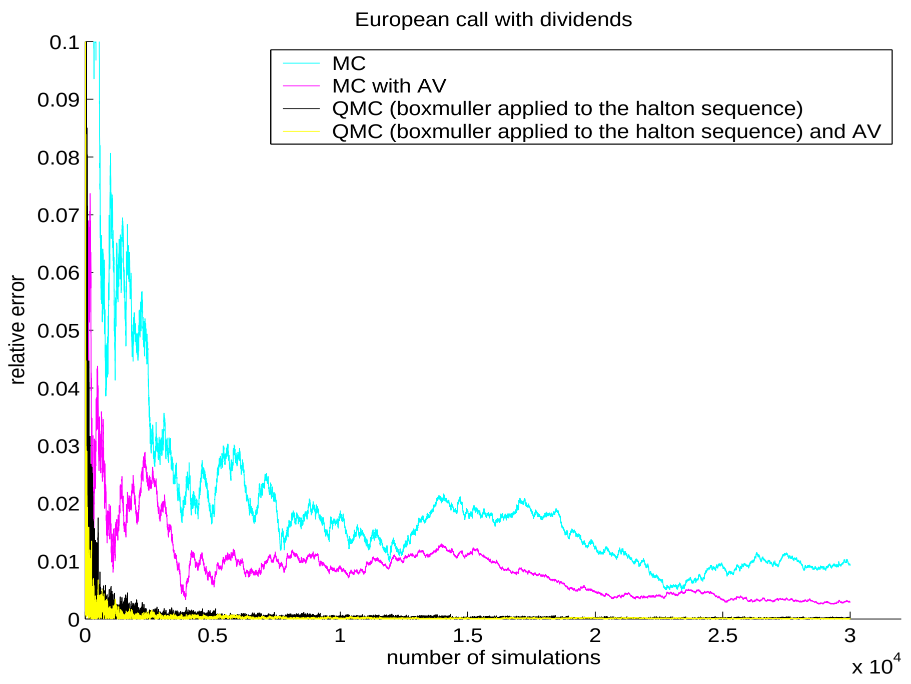


Figure 6.9: Relative error of the Monte Carlo method for a European call with $S_0 = 120$

Remark 6.15. *Note that these numbers are sensitive to the seed that was used for computing the pseudo-random numbers. Also, the degree of the relative error is still unstable when the number of simulation runs is quite small. Therefore, the numbers listed in Table 6.6 are only guiding values.*

We see that using antithetic variates makes it possible to achieve an accuracy level of 30% with almost three times less simulation runs than with crude Monte Carlo simulation.
Quasi-Monte-Carlo methods perform even better, the number of samples needed by this approach are about 100 times less in comparison to simple Monte Carlo simulation.

The values listed in Table 6.7 are more reliable since they are based on a higher number of simulation runs. We see that the level of accuracy reached by Monte Carlo simulation with antithetic variates is more than three times as high as the one yielded without variance reduction methods.
Again QMC simulation outperforms simple Monte Carlo simulation by a large degree, the relative error we get using low discrepancy sequences is about 100 times smaller than using pseudo-random numbers.

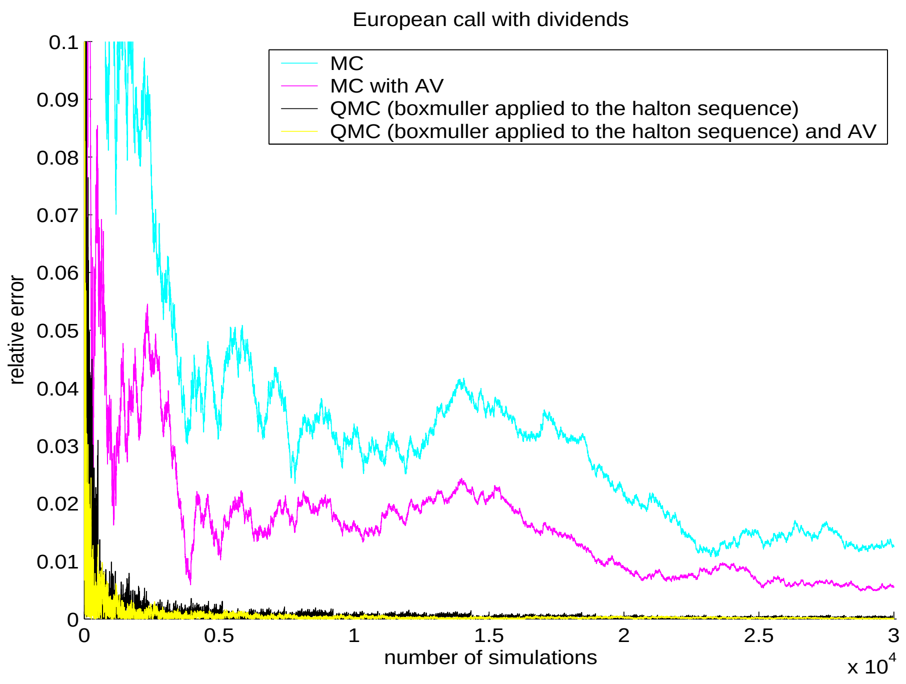


Figure 6.10: Relative error of the Monte Carlo method for a European call with $S_0 = 100$

method	n	rel error
simple Monte Carlo	30000	0.94%
MC with antithetic variates	30000	0.30%
QMC (Boxmuller)	30000	0.001%
QMC with AV (Boxmuller)	30000	0.01%

Table 6.7: Different levels of accuracy for a constant number of simulations

The Monte Carlo Method has many advantages over other methods for option pricing, such as

- The concept is not very complicated is thus easy to understand even for amateurs.

- The algorithm is easy to implement because of the simplicity of the method.

- Different methods for improving the performance of the simulation such as using quasi-random numbers or variance reduction methods are available and can be added to the original algorithm without high additional computing costs.

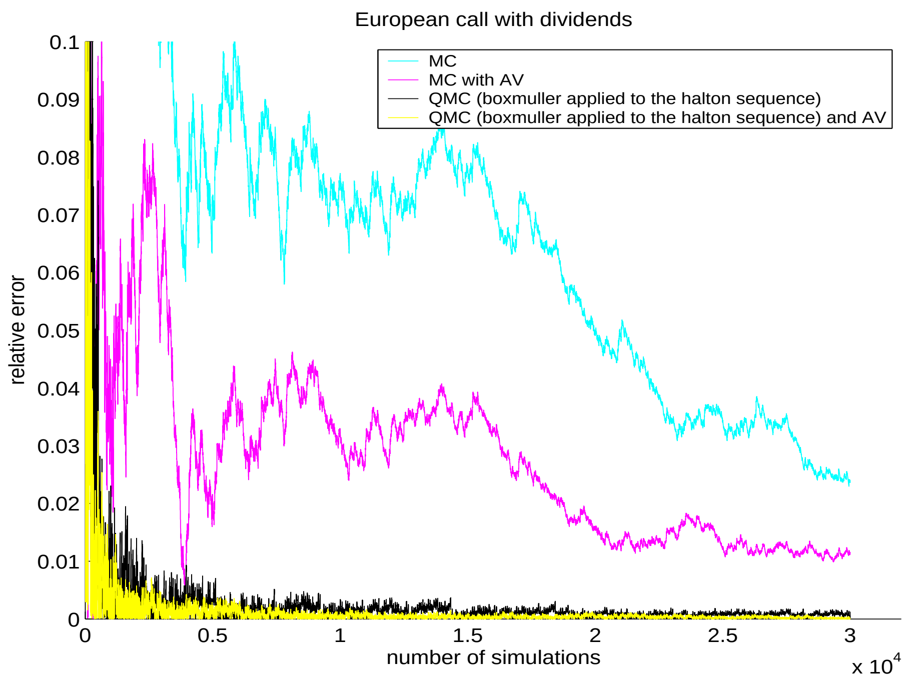


Figure 6.11: Relative error of the Monte Carlo method for a European call with $S_0 = 80$

- It is easy to implement and to modify, which makes it flexible and applicable for a large range of problems.
- Parallel simulations can be run on separate computers in order to reduce the computing time.
- The error convergence rate of the Monte Carlo Method is independent of the dimension of the problem.
- It is possible to estimate the accuracy of the option value by computing confidence intervals

It also has its limitations:

- The convergence rate of $O(\frac{1}{\sqrt{M}})$ is very slow, which is why the Monte Carlo simulation requires a large number of simulations to reach a certain degree of accuracy.
- The results have a statistical nature, therefore only probabilistic error bounds can be given.
- It is not possible to give a termination criterion such as a certain degree of accuracy. The number of simulation runs has to be fixed before the simulation is started.

- The quality of the simulation depends on the quality of the random numbers that are used.
- Although low discrepancy sequences improve the accuracy by a large degree, this fact is not reflected in the width of the confidence interval.
- Standard stochastic simulation procedures can only evaluate European style options.

Chapter 7

American Options

So far we have only evaluated European style options. In this chapter we want to introduce different methods of Monte Carlo simulation for American Options. Recalling Chapter 2, an American option can be exercised at any time up to expiry. It therefore provides its holder with more rights than its European equivalent and thus has a potential higher value. We also saw that for a call option on a non-dividend paying asset the value is the same whether the option is American style or a European style. This is why we can use the European call option value on a non-dividend paying asset as a reference for its American style counterpart and are thus able to get a feeling for the accuracy of the method. Having this done, we can then apply the algorithm to an American call option on an underlying that pays discrete dividends.

Optimal Exercise Boundary

Pricing American type options does not only require an approximation of the value of the option at time zero, one also needs to determine when the option should be exercised. Thereby, we will restrict our observations to so called Bermudan Options, that is options that can only be exercised at certain discrete time instants

$$\tau_i = i\Delta\tau, i = 0, ..., \Gamma, \tau_\Gamma = T \tag{7.1}$$

in the interval $[0, T]$, where τ is a stopping time as defined in Definition 4.17 and T is the expiry date as usual. This restriction is made because the problem for continuous exercise date possibilities can be very complicated and discrete exercise times work best with simulation algorithms which are discrete by nature.
The decision of when to exercise the option is clearly influenced by the price of the underlying: If the option is out-of-the-money (as defined in Remark 2.6) at time τ, then the best strategy would undoubtedly be to hold on to the option. If the option is in-the-money, it seems beneficial to exercise. However, it might also be a good strategy to wait for the payoff to rise even more. In order to solve this dilemma, we will now

take a look at rules for an optimal exercise strategy. In what follows, we will denote the time at which it is best for the option to be exercised out of all stopping times $\tau \leq T$ by τ^*. Over the life of the option, this optimal exercise time τ^* is called the optimal exercise boundary.

This boundary is defined by two values, the intrinsic option value i_τ and the continuation value c_τ:

In the case of a European type option, the decision whether or not to exercise at maturity T could be made with help of the payoff function $V_T = \max(0, S_T - K)$. A more general form of this concept which is also suitable for American type options leads to the so called intrinsic value denoted by

$$i_\tau = \max(0, S_\tau - K) \tag{7.2}$$

for possible exercise times $0 = \tau_0 \leq \tau \leq \tau_\Gamma = T$. Hence the intrinsic value represents the payoff yielded by the decision to exercise immediately. However, the American type option does not have to be exercised at time τ, which is why it also has a so called continuation value c_τ. This value represents the discounted expectation of the future value of the option in case it is held at time τ and the exercise decision is put off until the next possible exercise date $\tau + \Delta\tau$:

$$c_\tau = E(V_{\tau+\Delta\tau})e^{-r\Delta\tau}. \tag{7.3}$$

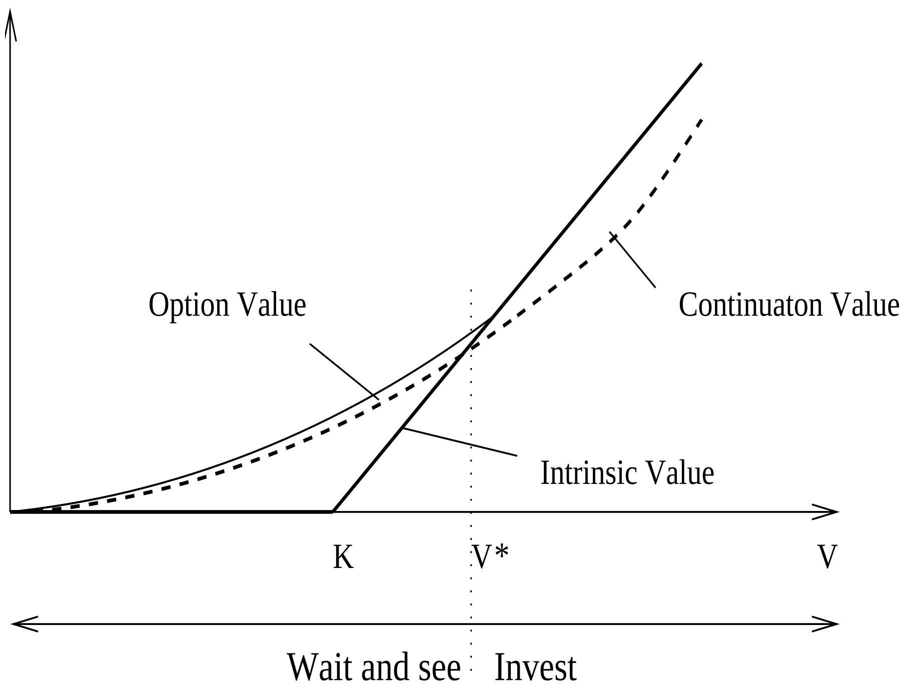
To sum it up, the choice between exercising or holding on to the option and thus delaying exercising results in two different values of the option, depending on which decision is reached. The holder of the option wants to maximize the option value and will thus let his exercise decision depend on what will result in a higher profit. The option value is thus given by the maximum of the intrinsic value i_τ and the continuation value c_τ:

$$V_\tau = \max(i_\tau, c_\tau). \tag{7.4}$$

In other words, the option should be exercised if the intrinsic value equals or exceeds the continuation value. Hence the optimal exercise boundary is situated at the point where the intrinsic value matches the continuation value:

$$i_\tau(S_\tau) = c_\tau(S_\tau). \tag{7.5}$$

Figure 7 shows the graphs of the intrinsic value, the continuation value and the value of an American call option for one potential exercise date τ.
While simulation programs are well-suited for evaluating European options, the additional freedom of choosing the time of exercise seems to make it impossible to apply Monte Carlo methods for pricing American options. The determination of the early



exercise strategy which is not necessary in the European case requires a backward algorithm. Starting at the maturity date, the optimal exercise strategy can be estimated by working backwards in time and applying dynamic programming. Standard simulation programs such as Monte Carlo methods are forward algorithms, however, meaning that paths of state variables are simulated forward in time. The problem of using Monte Carlo simulation for pricing American options lies in the fact that a forward algorithm has to be applied to a problem that requires a backwards procedure to solve.

For a long time, Monte Carlo simulation methods were considered not to be able to deal with the early exercise possibility of American options. Then, in 1993, Tilley introduced a bundling algorithm which estimates the value of an American option on a single asset. Although this method has several major drawbacks such as

- a lack of guarantee for convergence,
- extremely high computational costs and
- low flexibility,

the bundling algorithm proved that it was possible to evaluate American option prices by applying simulation. Several simulation procedures which use different approaches to the problem have been developed since then. We will introduce three different algorithms here, including Tilley's bundling algorithm as the first of them.

The Bias Problem

For evaluating European options, we used the simulation approach of computing the estimation

$$C_E = E[e^{-rT}\max(S_T - K, 0)]. \tag{7.6}$$

The according problem for evaluating American type options would therefore be to calculate

$$C_A = \max_{\tau \le T} E[e^{-r\tau}\max(K - S_\tau, 0)] \tag{7.7}$$

over all stopping times $\tau \le T$. Since we restrict ourselves to a few discrete possible exercise dates, this equality can also be written as

$$C_A = \max_{\tau_i} E[e^{-r\tau}\max(K - S_\tau, 0)], \tag{7.8}$$

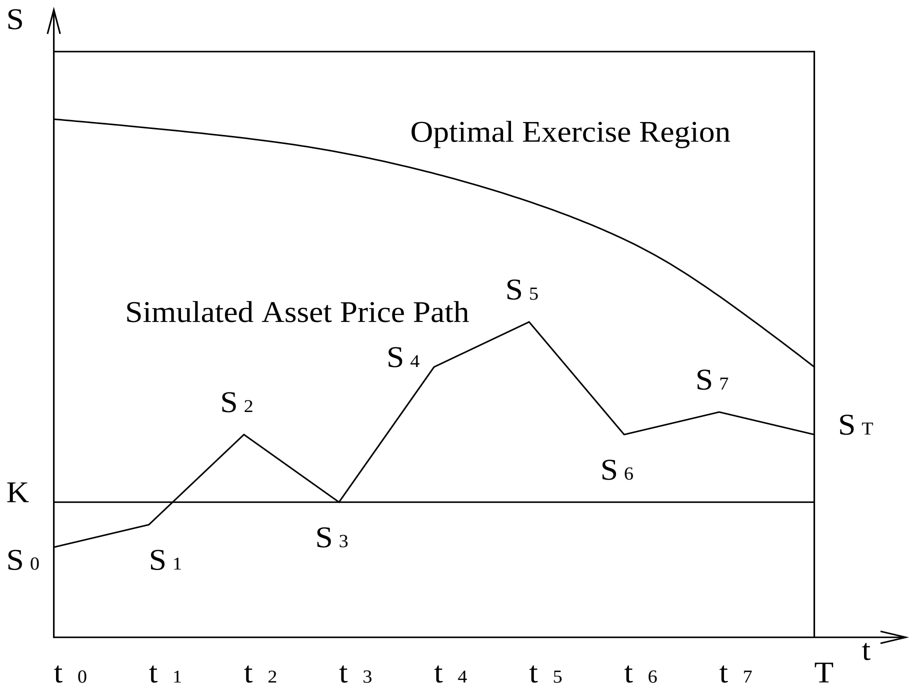
with τ_i defined as in Equation (7.1).
Analogously to the European scenario, the simulation procedure would be to simulate M asset price paths $S^j_{\tau_i}, i = 0, ..., \Gamma, j = 1, ..., M$ as basis for computing M discounted option values V^j which can then be averaged over in order to derive the approximation of the true option value. The problem hereby lies in the question of how to calculate the discounted option value. The optimal exercise date τ^* needs to be determined via simulation. A naive approach would be to do this for each asset price path by calculating

$$C_A = \max_{\tau_i}(e^{-r\tau}\max(S_\tau - K)). \tag{7.9}$$

This path estimate uses perfect foresight, however, since it is based on a single simulated asset price path only. Hence this procedure overestimates the option value as the following inequality shows:

$$C_A = \max_{\tau_i}(e^{-r\tau}\max(S_\tau - K, 0)) \ge e^{-r\tau^*}\max(S_{\tau^* - K}, 0) \tag{7.10}$$

Figure 7 illustrates this problem. The asset price path never crosses the optimal exercise boundary, so according to the exercise strategy discussed above, the option should not be exercised before maturity. This rule is based on an optimal but unknown strategy. In contrast, if the asset price path is known in advance as it is the case in Equation (7.9), we know that the optimal exercise date would be at τ_5, which results in the highest possible profit that also exceeds the profit made by sticking to the optimal exercise strategy. This shows that our approximation of the option value $E[C]$ exceeds the true option value $\hat{C}$, which means that our estimator is biased high. Simulating many paths may improve the situation, but since we are still working with information based on averaged perfect foresight it does not remove this bias problem. It seems unlikely that


S
Optimal Exercise Region
Simulated Asset Price Path
S_0
S_1
S_2
S_3
S_4
S_5
S_6
S_7
S_T
K
t
t_0 t_1 t_2 t_3 t_4 t_5 t_6 t_7 T

it is possible to compute an unbiased simulation estimator when pricing American options via Monte Carlo methods. In fact, Broadie and Glasserman provided a proof for this theorem in their paper [4].

Although it is not possible to find an unbiased estimator, a lot of methods have been developed that try to cope with this problem, three of which we will introduce here.

7.1 Tilley's Bundling Algorithm

As mentioned above, Tilley's bundling algorithm was designed for pricing an American option on a single underlying asset. The procedure is based on stochastic dynamic programming: A backwards induction is applied on bundles of asset price paths in order to derive an estimate for the option's holding value. This way, an estimate of the option value at time $\tau - \Delta\tau$ can be obtained at time τ. Hence by starting at the maturity date T and working backwards from there, an optimal exercise strategy can be obtained. The algorithm works as follows:

Algorithm 7.1. ***Tilley's bundling algorithm***

Step 0 (initialize)
Simulate M asset price paths $S_t(j)$ for $t = 0, ..., T$ and $j = 1, ..., M$ as we did for European options.

Beginning at the last possible early exercise date $\tau - \Delta\tau$, repeat the following steps for each time τ_i as defined in (7.1):

Step 1

Order the asset price paths in descending or ascending order according to the asset price at time τ for a put or a call option respectively.

Divide the ordered price paths into Q bundles: The first P paths make up the first bundle, the second P paths build the second bundle and so forth. Let $B_\tau(j)$ denote the set of paths in the bundle which contains the path indexed with j at time τ.

Step 2

Compute the intrinsic value $i_\tau(j)$ of the option for each path j

$$i_t(j) = \begin{cases} \max((S_\tau(j) - K), 0) & \textit{for a call option}, \\ \max((K - S_\tau(j)), 0) & \textit{for a put option}. \end{cases} \tag{7.11}$$

Step 3

Calculate the continuation value $c_\tau(j)$ of the option:

$$c_\tau(j) = \frac{e^{-r\Delta\tau}}{P} \sum_{\forall k \in B_\tau(j)} V_{\tau+\Delta\tau}(k). \tag{7.12}$$

Thereby $V_\tau(j)$ is defined later on in step 7 and $V_T(j) = i_T(j) \forall j$.

Step 4

Define an indicator variable that decides whether to exercise or to hold the option for each path j:

$$x_t(j) = \begin{cases} 1 & \textit{if } i_\tau(j) > c_\tau(j) \textit{ exercise the option}, \\ 0 & \textit{if } i_\tau(j) \leq c_\tau(j) \textit{ hold the option}. \end{cases} \tag{7.13}$$

Step 5

Find the sequence of 0s and 1s that starts with the first 1 and ends with the last 0 out of all sequences $\{x_\tau(j) : j = 1, ...M\}$. Then determine the start of the first string of 1s which is longer than any of the following strings of 0s. The path index of the first 1 in that exercise-hold boundary is denoted k^*_τ.

Step 6
Define a new exercise-or-hold indicator variable $y_\tau(j)$ based on the just derived boundary:

$$y_\tau(j) = \begin{cases} 1 & \textit{for } k \geq k_\tau^* \textit{ exercise option}, \\ 0 & \textit{for } k < k_\tau^* \textit{ hold option}. \end{cases} \tag{7.14}$$

Step 7
Calculate the current value of the option $V_\tau(j)$ for each path j:

$$V_\tau(j) = \begin{cases} i_\tau(j) & \textit{if } y_\tau(j) = 1, \\ c_\tau(j) & \textit{if } y_\tau(j) = 0. \end{cases} \tag{7.15}$$

Step 8
If $\tau = 0$ go to Step 9,
otherwise let $\tau = \tau - \Delta\tau$ and go back to Step 1.

Step 9
Once the algorithm has arrived at time 0, let $z_\tau(j)$ denote the exercise-or-hold indicator variable:

$$z_t(j) = \begin{cases} 1 & \textit{if } y_\tau(j) = 1 \textit{ and } y_s(j) = 0 \forall s < \tau, \\ 0 & \textit{otherwise}. \end{cases} \tag{7.16}$$

Step 10
Define the option estimator by

$$\frac{1}{M} \sum_{j=1}^{M} \sum_{t=1}^{T} z_\tau(j) e^{-rT} i_\tau(j). \tag{7.17}$$

Remark 7.2. *There are only two possibilities for each path j,:*

- $z_\tau(j) = 0$ *for all τ: The option is never exercised.*
- $z_{\tau^*}(j) = 1$ *and $z_\tau(j) = 0$ for all $\tau \neq \tau^*$: The option is exercised at time τ^**

The accuracy of the estimator given by Equation (7.17) is influenced by two factors, the number of bundles Q and the number of paths per bundle P. If the number of simulated paths M and thus the computational costs stay constant, one has to find the ideal combination of the inversely correlated parameters P and Q.

Tilley's approach is a single pass algorithm, meaning that all simulations are carried out before the actual algorithm is applied. This results in one of the drawbacks mentioned above: the computational costs are very high. A high number of paths has to be simulated in order to receive a satisfying result,it is therefore computationally expensive to have to store all paths and sort all of them at each time step τ.

7.2 Bounded Recursive Stochastic Simulation (BRSS)

The second algorithm we want to introduce is a method that is based on the approach first proposed in Grant et al [9]: a backward-recursive determination of the critical exercise frontier. This is a straightforward and fast way of determining the optimal early-exercise path and thus the option price of an American style call option. Muhoff, Hirschauer and Palmer [19] applied some modifications to this approach and called their method the Bounded Recursive Stochastic Simulation (BRSS). This is the method we want to introduce here:

Algorithm 7.3. ***Bounded Recursive Stochastic Simulation***

Step 1 *Determination of the critical exercise value V_T^*:*

As in every backward-recursive valuation, the starting point of the algorithm is the option value at expiry date T. Since we are at the last possible time of exercise, the value of the option of that point equals value of the payoff function of the corresponding European option:

$$V_T^* = \max(K - S_T^{i_1 \dots i_T}, 0) \tag{7.18}$$

Once we know V_T^* for time T, we can move backwards and calculate the option value at earlier times $T - \Delta\tau, T - 2\Delta\tau, ..., t_0$.

Step 2 *Determination of the critical early-exercise value $V_{T-j\Delta\tau}^*$:*

At each time period , the critical exercise value is the value of the option whose intrinsic value and continuation value are equal. Therefore, we have to compute the continuation value $c_{T-j\Delta\tau}(V_{T-j\Delta\tau})$ via simulation of M asset price paths (Step 2.2) and calculate the intrinsic value $i_{T-j\Delta\tau}(V_{T-j\Delta\tau}), j = 1, ..., \Gamma$ (Step 2.3) each for a set of N different test values for all periods $\tau = T - \Delta\tau, T - 2\Delta\tau, ...t_0$.

For $j = 1, ...\Gamma$ do

***Step 2.1** Definition of test-values:*

Generate an interval for the present period $\tau = T - j\Delta\tau$ using the critical exercise value of the previous period $V^*_{T-(j+1)\Delta\tau}$ as a lower bound and finding an appropriate upper bound. Then divide the resulting interval into $N - 1$ subintervals that are equally sized and sufficiently small for interpolation.

Simulation of asset price values can now be applied to all N endpoints ${}_nV_\tau$ $(n = 1, ..., N)$ of those subintervals. The value we are looking for is situated between the first value for which the intrinsic value exceeds the continuation value and its predecessor.

***Step 2.2** Determination of continuation values for each test-value:*

Generate M paths at each test value ${}_nV_\tau$, starting at the at the lower bound ${}_1V_\tau = V^*_{\tau+\Delta\tau}$. Thereby, the same sequence of random numbers is used, starting at a different number for each starting point and thus reducing computational effort significantly. In this manner, the continuation values ${}^m_nf_\tau$, $m = 1, ..., M$ for all M paths can be calculated as the discounted payoff of the option:

$$ {}^m_nf_\tau = \max(0, {}^m_n V_\kappa - K)e^{-r(\kappa-\tau)}. \tag{7.19}$$

with

$$\kappa = \begin{cases} \tau + \Delta\tau & \textit{if}\, {}^m_nV_{\tau+\Delta\tau} \geq V^*_{\tau+\Delta\tau}, \\ \tau + 2\Delta\tau & \textit{if}\, ({}^m_nV_{\tau+2\Delta\tau} \geq V^*_{\tau+2\Delta\tau}) \wedge ({}^s_nV_{\tau+\Delta\tau} < V^*_{\tau+\Delta\tau}) \\ . & \\ . & \\ T & \textit{otherwise}. \end{cases} \tag{7.20}$$

The estimator for the continuation value ${}_nf_\tau$ is then derived as the mean over all ${}^m_nf_\tau$:

$$ {}_nf_\tau = \frac{1}{M}\sum_{s=1}^{M} {}^m_nf_\tau \tag{7.21}$$

***Step 2.3** Calculation of intrinsic values for each test-value:*

The intrinsic value ${}_ni_\tau$ which is needed for making the decision whether to hold or exercise the option at time τ is given by the now well-known payoff function

$$ {}_ni_\tau = \max(0, {}_n V_\tau - K). \tag{7.22}$$

Step 2.4 *Approximation of the critical early-exercise value:*

The two test-values between which a change in the sign of the difference of continuation value and intrinsic value can be noticed form the new interval within which the critical early-exercise value is situated. This value can be derived via linear interpolation using the intrinsic values $({}_{n'}i_\tau$ *and* ${}_{n''}i_\tau)$ *and continuation values* $({}_{n'}c_\tau$ *and* ${}_{n''}c_\tau)$*, where* n' *and* n'' *denote the respective test-values:*

$$V_\tau^* =_{n''} V_\tau + \frac{{}_{N'}V_\tau -_{n''} V_\tau}{({}_{n'}i_\tau -_{n'} c_\tau) - ({}_{n''}i_\tau -_{n''} c_\tau)} \left[- ({}_{n''}i_\tau -_{n''} c_\tau)\right] \tag{7.23}$$

Step 3 *Control Step:*

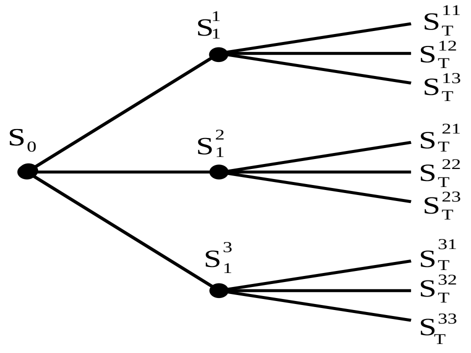
If the interval we chose in step 2.1 *does not contain a test value whose intrinsic value exceeds the corresponding continuation value, step* 2.4 *can not be carried out and thus the upper bound of the initial interval has to be extended before steps* 2.2 *to* 2.4 *are initiated again.*
In case the critical exercise value has already been situated in our initial interval, the approximation described in Step 2 *can now be improved by reducing the length of that interval. This would also shorten the subintervals on which we interpolate in steps* 2.2 *to* 2.4.

Step 4 *Determination of the option value:*

Once the optimal exercise strategy has been determined in the previous steps, standard Monte Carlo simulation can now be applied to retrieve the option value at time $t = 0$. *The (early-)exercise date we derived in the previous steps can thereby be used as the options maturity date.*

Remark 7.4. *The computational costs of the BRSS method are relatively small. If the same sequence of random numbers is used for simulations, only* Γ *simulations and* ΓM *simulation runs are needed for determining the early-exercise strategy. For the last step of determining the option value, another* M *simulation runs are needed, so all in all the method requires a number of simulation runs of* $(\Gamma + 1)M$.

Remark 7.5. *As stated in the introduction of the BRSS method there is one big advantage in comparison to Tilley's method, multiple stochastic variables such as stochastic volatility can be modelled with this approach. Extending the algorithm for such multiple stochastic variables is rather complicated, however. At each time period, we have to determine critical combinations of values that follow different stochastic processes, which turns our early-exercise strategy into a multi-dimensional problem over time.*



7.3 Broadie and Glasserman algorithm

As mentioned before, Broadie and Glasserman's approach makes use of two estimators in order to overcome the bias problem: one which is biased low and one which is biased high. However, both of them are consistent, that is they converge to the true price, and asymptotically unbiased for a sufficient high number of simulation runs. A point estimate and a conservative confidence interval can then be obtained out of those two. The procedure of deriving the estimators is based on simulated random trees rather than sample paths. The branching parameter b represents the number of branches at each node. Starting at S_0, at each node b independent asset price values are computed according to a Geometric Brownian Motion as it was the case with European options. Thereby, the value at each node is derived from the asset price value of the previous node. Thus the random tree can be seen as the array

$$\{S_\tau^{i_j \dots i_\tau} : \tau = 0, 1, ..., T; i_j = 1, ..., b; j = 1, ..., \tau\}, \tag{7.24}$$

as depicted in Figure 7.3:
Each sequence of asset price values $S_0, S_1^{i_1}, S_2^{i_1 i_2}, ..., S_T^{i_1 \dots i_T}$ is a realization of the Markov chain $\{S_\tau : \tau = 0, 1, ..., T\}$. Therefore, two sequences evolve independently of each other once they differ in some i_τ.
Note that unlike lattice methods, the values of the nodes in the tree are not ordered but appear in the order they were computed.

Example 7.6. *An illustration of such a tree for $b = 3$ is given in Figure 7.1. The indices indicate the development of the asset prices. S_T^{11}, S_T^{12} and S_T^{13} are all derived from S_1^1 and thus depend on it. At the same time they are completely independent of S_1^2, which was used for deriving the asset prices S_T^{21}, S_T^{22} and S_T^{23} which are in return not dependent on S_1^1.*

We will now discuss the two estimators for an American call option:

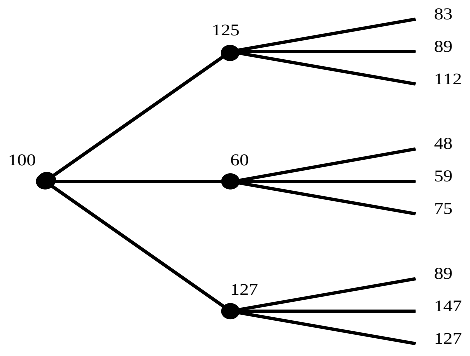


Figure 7.1: Simulated tree for $b = 3$

The high estimator Θ

Our first estimator is defined by a dynamic programming algorithm applied to the simulated tree described above. Starting at the expiry date where the option value is equal to its European counterpart, i.e. its payoff h, the value at each prior time is computed as the maximum of the immediate exercise value or intrinsic value given in Equation (7.2) and the discounted expected option values of the succeeding node, also called continuation value given by Equation (7.3). Working backwards this way we obtain the high estimator Θ at the initial node.

Example 7.7. *A numerical example is depicted in Figure 7.2 below. The parameters are $S_0 = 100, K = 90, T = 1$ and $r = 0$. The choice of a risk-free interest rate equal to zero is made for simplicity reasons because this makes discounting unnecessary. The high estimator for the American call option in this example is $\Theta = 24$.*

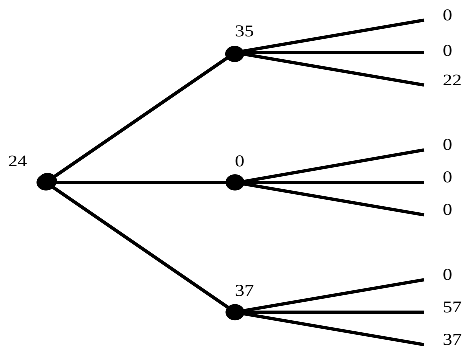


Figure 7.2: The high estimator Θ

The mathematical recursive definition of the high estimator Θ is given by

$$\begin{aligned}\Theta_T^{i_1 \dots i_T} &= \max(S_T^{i_1 \dots i_T} - K, 0) = \\ &= i_T(S_T^{i_1 \dots i_T})\end{aligned} \tag{7.25}$$

$$\begin{aligned}\Theta_\tau^{i_1 \dots i_\tau} &= \max\left\{ \max(S_\tau^{i_1 \dots i_\tau} - K, 0), \frac{1}{b}\sum_{j=1}^{b} e^{-rh}(\Theta_{\tau+1}^{i_1 \dots i_\tau)^j} \right\} \\ &= \max\left\{ i_\tau(S_\tau^{i_1 \dots i_\tau}), \frac{1}{b}\sum_{j=1}^{b} e^{-rh}(\Theta_{\tau+1}^{i_1 \dots i_\tau)^j} \right\}\end{aligned} \tag{7.26}$$

for $\tau = 0, ..., T-1$.

Since Θ gives an estimate of the true option value that is biased high, i.e. $E[\Theta] \geq P_A$, it is called the high estimator. A proof for Θ being biased upwards will be given later on, but we first want to clarify the this fact by intuitive reasoning:
No simulated tree can perfectly reflect the distribution of asset prices. At some nodes the asset prices are too high, and the dynamic programming is likely to decide to hold the call option and thus work with a higher value than the one obtained by the true optimal decision to exercise. Likewise, asset prices are too low at other nodes, which might cause the algorithm to declare it best to exercise at a high profit while the true optimal decision would have been to hold the option.
Either way, the dynamic programming algorithm takes advantage of knowledge of the future to overestimate the option value.

Theorem 7.8. *[High estimator bias]*
The high estimator Θ is biased high, i.e.

$$E[\Theta_0(b)] \geq V_0(S_0) \tag{7.27}$$

for all b.

Proof. It is to show that $E[\Theta_\tau | S_\tau] \geq V_\tau(S_\tau)$ for $\tau = 0, 1, ..., T$. We will do that by backwards induction. It follows from the definition of $\Theta_T = V_T(S_T)$ that $E[\Theta_T | S_T] \geq V_T(S_T)$. Our induction hypothesis is $E[\Theta_{\tau+1} | S_{\tau+1}] \geq V_{\tau+1}(S_{\tau+1})$. Using Jensen's Inequality (3.15) and the definition of Θ_{tau} given in Equation (7.26), we get

$$\begin{aligned}E[\Theta_\tau | S_\tau] &= \max\{i_\tau(S_\tau), E[e^{-r\Delta\tau}\Theta_{\tau+1} | S_{\tau+1}]\} = \\ &= \max\{i_\tau(S_\tau, E[e^{-r\Delta\tau}E[\Theta_{\tau+1} | S_{\tau+1}] | S_\tau]\} \geq \\ &\geq \max\{i_\tau(S_\tau, E[e^{-r\Delta\tau}c_{\tau+1}(S_{\tau+1} | S_\tau]\} = \\ &= \max\{i_\tau(S_\tau), c_\tau(S_\tau)\}.\end{aligned}$$

□

The low estimator θ

In order to overcome the bias problem, Broadie and Glasserman's algorithm uses a low biased estimator in addition to the high biased one. Its calculation is based on the idea of separating the branches at each node in two sets. One is used to decide whether or not to exercise, and in case the option is being held, the other one works as the basis for the estimate of the continuation value.

Example 7.9. *We continue Example 7.7 and use the same parameters and asset price tree (as depicted in Figure 7.1) for deriving a low estimator. The results can be seen in Figure 7.3. At each node the upper branch delivers the continuation value if necessary, and the two bottom branches are used for determining the exercise decision. At time* $\tau_2 = T$ *the node values equal those of the high estimator. Let us now take a look at time* τ_1. *At the top node the decision to exercise the option is reached since the immediate exercise value* $\max(0, 125-90) = 35$ *exceeds the continuation value* $0.5(0+22) = 11$. *The value assigned to this node is* 35. *A different scenario can be observed at the bottom node. Here the immediate exercise value* $\max(0, 127-90) = 37$ *is lower than the continuation value* $0.5(57+37) = 47$, *therefore the exercise decision is delayed until the next possible exercise date. The value assigned to the node is then derived from the upper branch of this node whole value equals zero, so the node value of* 0 *is in fact lower than the immediate exercise value of* 37. *Continuing this calculation for the rest of the tree leads to a low estimate of the American call of* 10.

As the example shows, using only one branch for estimating the continuation value can be quite inaccurate. Therefore it seems reasonable to apply a slight modification to the process of evaluating the low estimator at each node. Instead of just using one branch to determine the continuation value, each of the b branches is taken in turn for that task and finally the b values obtained by that procedure are averaged and thus the estimate of the option value of the node is obtained.

Example 7.10. *We now apply this modification to our example. A depiction of the results is given in Figure 7.4. The calculation of the bottom node value at time* τ_1 *includes all possible scenarios, which is why we will concentrate on that node here.*

- *Branch* 1 *is used to determine the continuation value*
  We already treated this case in the example for the simple version of the low estimator. The decision is reached to continue but the value assigned to the node is 0.

- *Branch* 2 *is used to determine the continuation value*
  Here the optimal strategy is to exercise as the immediate exercise value $\max(0, 127-90) = 37$ *exceeds the'continuation value'* $0.5(0+37) = 18.5$. *The value assigned to the node thus is* 37.

- *Branch* 3 *is used to determine the continuation value*
  Again it is better exercise because the 'continuation value' of $0.5(0+57) = 28.5$ *is not as high as the immediate exercise value of* 37*. The value assigned to the node is thus* 37*.*

Therefore, the final averaged value assigned to the node is 24.7*. Proceeding in an analogous way for the remaining nodes, we see that the low estimator for this American type option price problem is* $\theta = 19.9$*.*

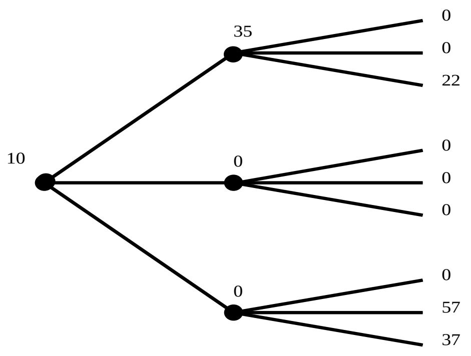


Figure 7.3: The simple low estimate

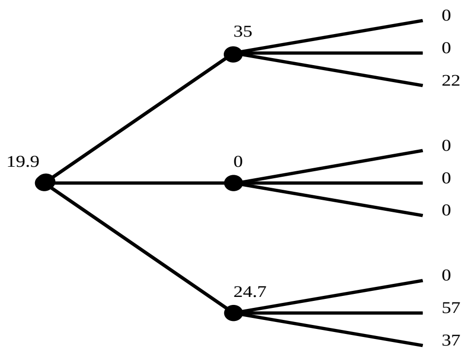


Figure 7.4: The low estimator θ

The formal definition of the low estimator is given as follows:

$$\theta_T^{i_1 \dots i_T} = \max(K - S_T^{i_1 \dots i_T}, 0) \tag{7.28}$$

for the terminal nodes and

$$\theta_t^{i_1 \dots i_t} = \frac{1}{b} \sum_{j=1}^{b} (\eta_t^{i_1 \dots i_t})^j \text{ for } t = 0, ..., T-1. \tag{7.29}$$

Thereby η is defined as

$$\left(\eta_t^{i_1...i_t}\right)^j = \begin{cases} \max(K - S_t^{i_1...i_t}, 0) & \text{if } \max(K - S_T^{i_1...i_T}, 0) \geq \frac{1}{b} \sum_{i=1, i\neq j}^{b} e^{-rh}(\theta_{t+1}^{i_1...i_t})^j, \\ e^{-rh}(\theta_{t+1}^{i_1...i_t})^j & \text{else,} \end{cases} \tag{7.30}$$

for $j = 1, ..., b$. Before a proof for the negative bias of θ is given, we want to give an intuitively reasoning again first. Take a look at the option when it is just about to expire. Here at the maturity date, the exercise decision is based on totally unbiased information. If the correct decision would be induced by this information, the estimator would not be biased. But the probability of arriving at a suboptimal decision based on a finite sample is positive. If this happens, the node value will be an unbiased estimate of the lower value associated with the suboptimal decision. This leads to the expected node value being a weighted average of an unbiased estimate based on the optimal decision and an estimate which is biased low because it is based on a suboptimal decision. Hence on the whole the bias is low.

Theorem 7.11. *[Low-estimator bias]*
The bias of the low estimator is negative, i.e.

$$E[\theta_0(b)] \leq V_0(S_0) \tag{7.31}$$

for all b.

Proof. Again, we use backwards induction to prove the bias theorem. The definition of the low estimator $\theta_T = V_T(S_T)$ (Equation (7.28)) indicates that $E[\theta_T|S_T] \leq V_T(S_T)$. The induction hypothesis is $E[\theta_{\tau+1}|S_{\tau+1}] \leq c_{\tau+1}(S_{\tau+1})$.
It follows from the definition of θ_τ in (7.29) that $E[\theta_\tau|S_\tau] = E[\eta_\tau^j|S_\tau]$ for any $j = 1, ..., b$. Let us define

$$Y_\tau^j(b) = \frac{1}{b-1} \sum_{i=1, i\neq j}^{b} e^{-r\Delta\tau}\theta_{\tau+1}^i. \tag{7.32}$$

Since Y_τ^j is conditionally independent of $\theta_{\tau+1}$ given S_τ, we get

$$\begin{aligned} E[\eta_\tau^j|S_\tau] &= E[i_\tau(S_\tau)I_{\{i_\tau(S_\tau)\geq Y_{\tau+1}^j\}}|S_\tau] + E[e^{-r\Delta\tau}\theta_{\tau+1}^j I_{\{i_\tau(S_\tau)<Y_{\tau+1}^j\}}|S_\tau] = \\ &= i_\tau(S_\tau P[i_\tau(S_\tau) \geq Y_{\tau+1}^j|S_\tau]) + E[e^{-r\Delta\tau}\theta_{\tau+1}^j|S_\tau]P[i_\tau(S_\tau) < Y_{\tau+1}^j|S_\tau] = \\ &= i_\tau(S_\tau)p + E[e^{-r\Delta\tau}\theta_{\tau+1}^j|S_\tau](1-p). \end{aligned}$$

At the same time

$$\begin{aligned} E[\theta_\tau|S_\tau] &= i_\tau(S_\tau)p + E[e^{-r\Delta\tau}E[\theta_{\tau+1}^j|S_{\tau+1}]|S_\tau](1-p) \leq \\ &\leq i_\tau(S_\tau)p + E[e^{-r\Delta\tau}V(S_{\tau+1})|S_\tau](1-p) = \\ &= i_\tau(S_\tau)p + c_\tau(S_\tau(1-p)) \leq \\ &\leq \max\{i_\tau(S_\tau), c_\tau(S_\tau)\} = V_\tau(S_\tau). \end{aligned}$$

□

Remark 7.12. *It is possible to generalize the definition of the low estimator. Instead of using only one branch for determining the continuation value in each iteration of the loop, implement b_1 branches to support the exercise decision and $b_2 = b - b_1$ branches for evaluating the resulting payoff. These alternative estimators are also consistent and biased low.*

Theorems 7.8 and 7.11 showed that the two estimators are indeed biased high and low respectively. This means that the respective expectations are above or below the true value. Theoretically it is still possible for the low estimator to exceed the high estimator. The following theorem shows that this is not the case, however:

Theorem 7.13. *For every realization of the array $\{S_\tau^{i_j \dots i_\tau} : \tau = 0, 1, \dots, \Gamma; i_j = 1, \dots, b; j = 1, \dots, \tau\}$, the high estimator is equal to or higher than the low estimator:*

$$\Theta_\tau^{i_j \dots i_\tau} \geq \theta_\tau^{i_j \dots i_\tau} \tag{7.33}$$

for all $i_j \dots i_\tau$ and all $\tau = 0, 1, \dots, T$ w.p.1.

Proof. Again we use backwards induction starting at $\theta_T = \Theta_T = V_T(S_T)$. The induction hypothesis is $\theta - \tau + 1^j \leq \Theta - \tau + 1^j, j = 1, \dots, b$. Let Y_t^j be defined as in Equation (7.32). In case all $Y_\tau^1, \dots, Y_\tau^b$ are less or equal to the intrinsic value $i_\tau(S_\tau,$ $\eta_\tau^j = i_\tau(S_\tau), j = 1, \dots, b$, and thus $\theta_\tau = i_\tau(S_\tau) \leq \Theta_\tau$. Let us now consider the more complicated case of at least one Y_τ^j being greater than $i_\tau(S_\tau)$. Then

$$\theta_\tau^j = \frac{1}{b} \sum_{j=1}^{b} \eta_\tau^j = \frac{1}{b} \sum_{j=1}^{b} (i_\tau(S_\tau)) I_{\{i_\tau(S_\tau) \geq Y_\tau^j\}} + e^{-r\Delta\tau} \theta_{\tau+1}^j I_{\{i_\tau(S_\tau) \geq Y_\tau^j\}}$$

$$= \left(\frac{1}{b} \sum_{j=1}^{b} I_{\{i_\tau(S_\tau) \geq Y_\tau^j\}} \right) i_\tau(S_\tau) + \left((\frac{1}{b} \sum_{j=1}^{b} I_{\{i_\tau(S_\tau) < Y_\tau^j\}} \right) \frac{\sum_{j=1}^{b} e^{-r\Delta\tau} \theta_{\tau+1}^j I_{\{i_\tau(S_\tau) < Y_\tau^j\}}}{\sum_{j=1}^{b} I_{\{i_\tau(S_\tau) < Y_\tau^j\}}}$$

$$p h_\tau(S_\tau) + (1-p) \frac{\sum_{j=1}^{b} e^{-r\Delta\tau} \theta_{\tau+1}^j I_{\{i_\tau(S_\tau) < Y_\tau^j\}}}{\sum_{j=1}^{b} I_{\{i_\tau(S_\tau) < Y_\tau^j\}}} \tag{7.34}$$

Without loss of generality let us now suppose that $Y_\tau^1, \dots, Y_\tau^k$ are greater than $i_\tau(S_\tau)$ and $Y_\tau^{k+1}, \dots, Y_\tau^b$ are equal to or less than $i_\tau(S_\tau)$. Then the ratio in Equation (7.34) equals $k^{-1} \sum_{j=1}^{k} e^{-r\Delta\tau} \theta_{\tau+1}^j$. For any $i \leq k < j \leq b$, the inequality $Y_\tau^i > Y_\tau^j$ holds and we have $e^{-r\Delta\tau} \theta_{\tau+1}^i \leq e^{-r\Delta\tau} \theta_{\tau+1}^j$. Therefore,

$$\max\{e^{-r\Delta\tau} \theta_{\tau+1}^1, \dots, e^{-r\Delta\tau} \theta_{\tau+1}^k\} \leq \min\{e^{-r\Delta\tau} \theta_{\tau+1}^{k+1}, \dots, e^{-r\Delta\tau} \theta_{\tau+1}^{k+1}\}.$$

This implies

$$\frac{1}{k}\sum_{i=1}^{k} e^{-r\Delta\tau}\theta^i_{\tau+1} \leq \frac{1}{k}\sum_{i=1}^{b} e^{-r\Delta\tau}\theta^i_{\tau+1}.$$

Applying this to (7.34), we get

$$\begin{aligned}\tfrac{1}{b}\sum_{j=1}^{b}\eta^j_\tau &= pi_\tau(S_\tau) + (1-p)\tfrac{1}{k}\sum_{j=1}^{k} e^{-r\Delta\tau}\theta^j_{\tau+1}\\ &\leq pi_\tau(S_\tau) + (1-p)\tfrac{1}{b}\sum_{j=1}^{b} e^{-r\Delta\tau}\theta^j_{\tau+1},\\ &\leq pi_\tau(S_\tau) + (1-p)\tfrac{1}{b}\sum_{i=1}^{b} e^{-r\Delta\tau}\Theta^j_{\tau+1}\\ &\leq \max\left\{i_\tau(S_\tau), \tfrac{1}{b}\sum_{i=1}^{b} e^{-r\Delta\tau}\Theta^i_{\tau+1}\right\} = \Theta_\tau \qquad .\end{aligned}$$

□

Remark 7.14. *A proof for the consistency of both the high and the low estimator can be found in M. Broadie and P. Glasserman's paper [4].*

Numerical results

In contrast to Tilley's bundling algorithm where all asset price paths have to be stored at all times, the storage requirements of Broadie and Glasserman's method are quite low. The algorithm works depth first, meaning that we work branch for branch starting at the terminal nodes until we get to the initial node.
This way, we do not have to compute the whole tree of asset price values all at once. Only the values that are needed for calculating the present nodes estimator value have to be stored. This leads to a total storage cost for the method of $O(nb^d)$.

Although this is a huge improvement in comparison to Tilley's method, the main drawback in Broadie and Glasserman's approach also lies in the computational costs. Since they rise exponentially with the number of possible exercise times, d cannot be chosen to be too large.

Remark 7.15. *Broadie and Glasserman tested their method for different types of American options but all for $d = 4$, i.e. for four possible exercise dates.*

We want to present results for $d = 3$ here. Let us begin with a starting value of the underlying asset of $S_0 = 80$. The other parameters are the same as in the example of the European option: $K = 100, r = 0.07, \sigma = 0.3$ and $T = 1$. As mentioned before, we will take a look at the American Call option on a non-dividend paying asset first because this allows us to use the European counterpart as a control value, which is why we set $\delta = 0$. The Black-Scholes value for those parameters is given by $V_{exakt} = 5.0126$. As Table 7.1 shows, the relative error decreases significantly with higher values for the branching parameter b as well as for more simulation runs n:

b	n	θ	Θ	90% confidence interval	V	rel error
10	50	5.2170	5.6653	[4.4736, 6.4528]	5.4412	8.55%
10	100	5.0295	5.4619	[4.5314, 5.9827]	5.2457	4.65%
10	200	4.9189	5.2973	[4.5975, 5.6311]	5.1081	1.91%
25	50	5.2133	5.3645	[4.8322, 5.7584]	5.2889	5.51%
25	100	4.9107	5.0600	[4.6288, 5.3518]	4.9854	0.43%
25	200	4.9327	5.0592	[4.7024, 5.2973]	4.9959	0.33%
50	50	5.1644	5.2039	[4.8082, 5.5629]	5.1842	3.42%
50	100	5.0182	5.0607	[4.8034, 5.2762]	5.0394	0.53%
50	200	5.0052	5.0443	[4.8462, 5.2040]	5.0248	0.24%

Table 7.1: AO without dividends for $S_0 = 80$

The Figures 7.5, 7.6 and 7.7 show, not only the accuracy of the approximation increases with rising branching parameter b, but the width of the confidence interval decreases as well which can be seen in Figure 7.8 below.
As we already saw in the case of European options, the accuracy of the simulation is not independent of the starting value of the underlying asset. This is why we repeated the simulation runs for $S_0 = 100$ while all the other parameters stay the same. The Black Scholes value for this scenario is given by $V_{exakt} = 15.2105$:

b	n	θ	Θ	90% confidence interval	V	rel error
10	50	15.73000	16.7306	[14.2925, 18.2803]	16.2303	6.70%
10	100	15.2012	16.2143	[14.2383, 17.2360]	15.7077	3.27%
10	200	14.9091	15.9789	[14.2745, 16.6451]	15.4440	1.54%
25	50	15.5769	16.0186	[14.7772, 16.8355]	15.7978	3.86%
25	100	15.0026	15.4354	[14.3968, 16.0563]	15.2190	0.06%
25	200	14.9861	15.4278	[14.5273, 15.9014]	15.2070	0.02%
50	50	15.3149	15.5277	[14.6385, 16.2134]	15.4213	1.39%
50	100	15.1108	15.3055	[14.7003, 15.7257]	15.2102	0.002%
50	200	15.0761	15.2770	[14.7638, 15.5934]	15.1765	0.22%

Table 7.2: AO without dividends for $S_0 = 100$

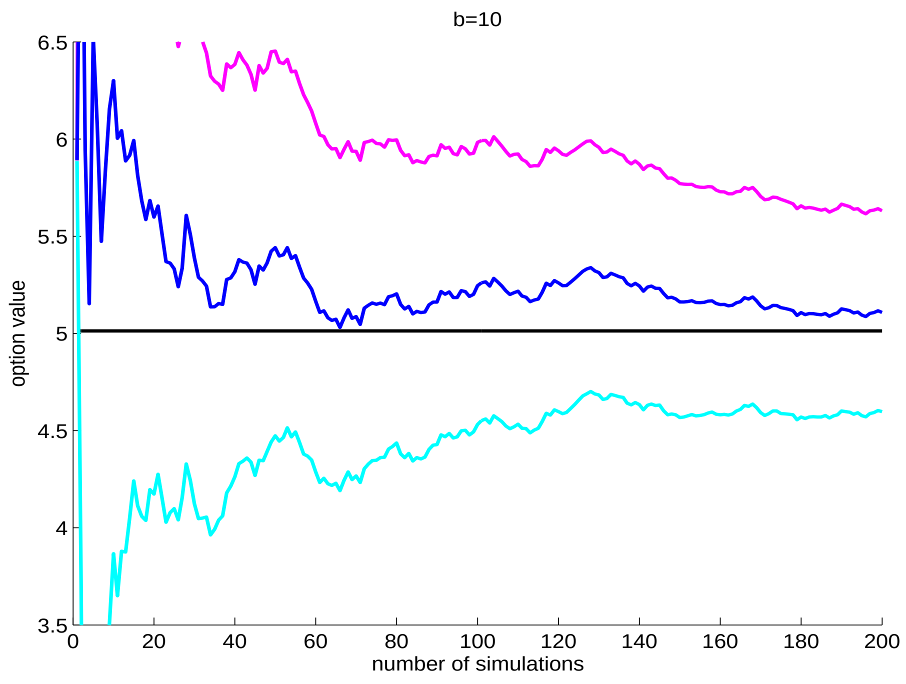


Figure 7.5: Confidence interval for an American call with $S_0 = 80$ (b=10)

The relative error here is on average smaller than in the case of an underlying asset value at time $t = 0$ of $S_0 = 80$. This leads to the educated guess that the accuracy of the method is the better the closer the value of the underlying is to the strike price K at the beginning S_0. Let us check this guess by running the same simulation but this time with $S_0 = 120$. The Black-Scholes value is hereby given by $V_{exakt} = 30.2829$:

b	n	θ	Θ	90% confidence interval	V	rel error
10	50	29.7162	32.8286	[27.3096, 35.0200]	31.2724	3.27%
10	100	28.6130	32.0450	[26.9097, 33.5036]	30.3290	0.15%
10	200	28.3554	31.8528	[27.2010, 32.8116]	30.1041	0.59%
25	50	30.5389	31.6815	[29.2412, 32.8923]	31.1102	2.73%
25	100	29.8021	30.8786	[28.8782, 31.7946]	30.3403	0.19%
25	200	29.7889	30.8752	[29.1019, 31.5586]	30.3321	0.16%
50	50	30.1851	30.7731	[29.2145, 31.7703]	30.4791	0.65%
50	100	29.9505	30.5088	[29.3571, 31.1175]	30.2296	0.12%
50	200	29.9100	30.4708	[29.4527, 30.9378]	30.1904	0.31%

Table 7.3: AO without dividends for $S_0 = 120$

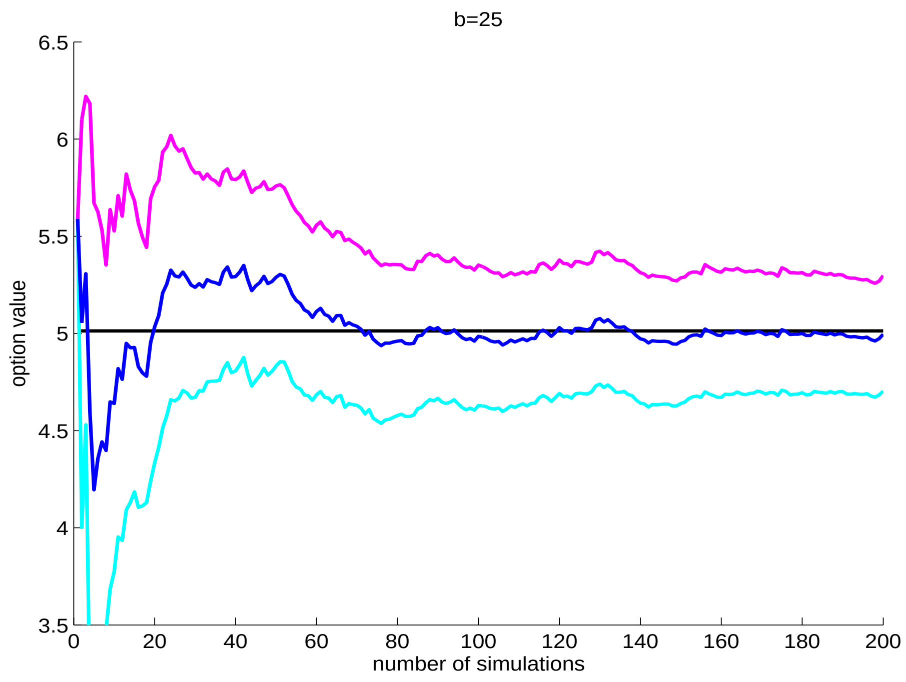


Figure 7.6: Confidence interval for an American call with $S_0 = 80$ (b=25)

Table 7.3 does not support our guess, which is why we will run more simulations for different start values of the underlying asset now:

S_0	θ	Θ	confidence interval	Point estimate	Black-Scholes Value	rel error
70	2.1892	2.2149	$[2.0622, 2.3429]$	2.2021	2.1806	0.986%
80	5.0182	5.0607	$[4.8034, 5.2762]$	5.0394	5.0126	0.535%
90	9.3513	9.4450	$[9.0439, 9.7582]$	9.3982	9.3798	0.535%
100	15.1108	15.3055	$[14.7003, 15.7257]$	15.2102	15.2105	0.002%
110	22.0652	22.4241	$[21.5546, 22.9407]$	22.2447	22.2729	0.127%
120	29.9505	30.5088	$[29.3571, 31.1175]$	30.2296	30.1904	0.118%
130	38.4519	39.2943	$[37.7333, 29.9862]$	38.8731	38.9755	0.263%

Table 7.4: Comparison of different start values for S_0

This shows that on average the relative error is getting smaller with rising values for S_0 at least up to a certain point. As the value of the underlying is situated further away from the strike price $K = 100$, the quality of the approximation worsens again, but not as much as it did on the left side of the strike price.

This can again be explained by the fact that with smaller control values the relative error gets higher in comparison to the absolute error. Taking this into consideration,

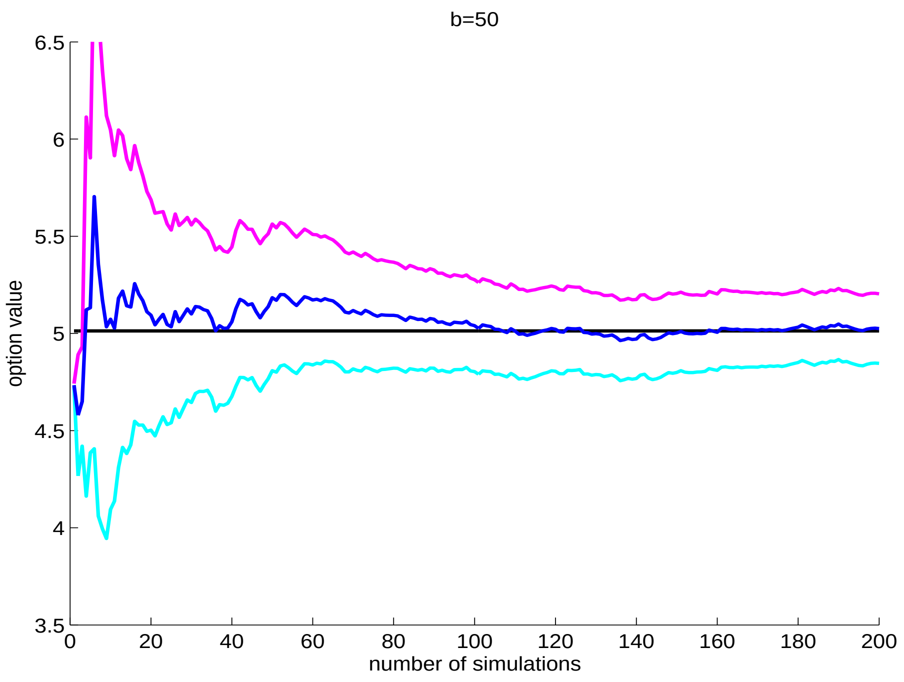


Figure 7.7: Confidence interval for an American call with $S_0 = 80$ (b=50)

we see that the accuracy of the method indeed gets better the closer K is to S_0. We now evaluate an American Call with the same parameters as in the example above but this time with a continuous dividend rate of $\delta = 0.1$.

Remark 7.16. *As mentioned in Chapter 2, the value of an American call option differs from its European counterpart in case dividends are paid. This is why we do not have a control value in form of the European option anymore.*

7.4 Comparison of methods

To conclude this chapter, we want to compare the methods we introduced here. While the three algorithms are simulation methods for pricing American type options they all use different approaches for solving the problem. Consequently each method has its own characteristic features some of which we want to discuss here.

General approach:

- Tilley's bundling algorithm is a single pass method based on sorting a large number of previously simulated asset price paths at each possible exercise date.

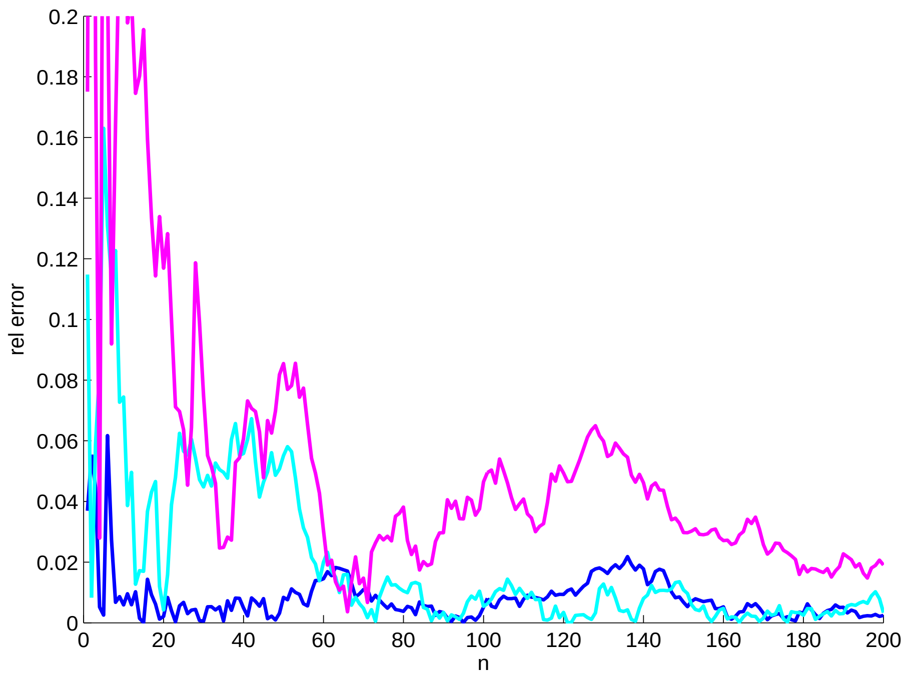


Figure 7.8: Relative error of the BRSS Method for an American call with $S_0 = 80$

S_0	n	θ	Θ	90% confidence interval	V
80	50	2.8435	2.9181	$[2.5976, 3.1745]$	2.8808
80	100	2.7307	2.8057	$[2.5855, 2.9581]$	2.7682
80	200	2.7289	2.8027	$[2.6196, 2.9150]$	2.7658
100	50	9.9665	10.2934	$[9.4292, 10.8472]$	10.1299
100	100	9.7664	10.0948	$[9.4429, 10.4289]$	9.9306
100	200	9.7696	10.0798	$[9.5260, 10.3330]$	9.9247
120	50	21.9226	23.1255	$[20.9341, 23.9062]$	22.5240
120	100	21.2130	22.7857	$[20.4715, 23.2702]$	21.9994
120	200	21.4758	22.8452	$[20.9727, 23.1998]$	22.1605

Table 7.5: Values for an American type call option with a continuous dividend rate

Proceeding backwards from the expiry date of the option an early exercise strategy is obtained by averaging over the continuation values of the prices in each bundle.

- The BRSS approach works with sequential stochastic simulation of asset price paths. Moving backward from the expiry date multiple asset price paths are simulated starting at different test values. Linear interpolation between the two test values that form the boundary of the interval in which the sign of the difference of intrinsic value and continuation value changes then leads to the critical early exercise value. Finally simple Monte Carlo simulation can be applied as in the case of evaluating European options, using the early exercise date as expiry date of the option.

- Broadie and Glasserman's algorithm computes two estimators that are both based on a simulated tree of asset prices. The high estimator is obtained by applying a dynamic programming algorithm to the asset price tree. This algorithm is then slightly modified for computing the low estimator. Only a certain amount of branches is used here to determine the exercise decision, the other ones deliver the continuation value if necessary. This process is repeated for a large number of trees order to obtain a point estimate and a confidence interval for the option value.

Bias:
As already mentioned earlier, it can be shown that an unbiased estimator of the value of an American option does not exist. We are therefore interested in the ways the different algorithms deal with this problem:

- Tilley's estimator is biased high since the algorithm uses the same asset price paths for the exercise policy and the option value.

- According to Boyle et al [2] the estimator computed by the BRSS method tends to be biased low. The reason for this can be seen in the fact that the procedure applies simple Monte Carlo simulation to a suboptimal approximation of the optimal exercise date previously obtained by using repeated simulation runs.

- Broadie and Glasserman's algorithm works with two estimators one of which is biased low while the other one is biased high. Since they are both asymptotically unbiased for a high enough number of simulations, a way of solving the bias problem is obtained.

Computational costs:

- Tilley's bundling algorithm requires all asset paths to be stored at all times. In addition to that the paths have to be sorted in each time step. Since the number of simulated price paths has to be large in order to obtain a satisfactory level of accuracy this results in very high storage and sorting costs.

- Since the BRSS-procedure re-uses one random number sequence by starting at a different number of the same sequence for each simulation of asset price paths the actual Monte Carlo simulation only has to be carried out once. Therefore only Γ simulations and ΓS simulation runs are needed to determine the early-exercise strategy.

- The asset price tree Broadie and Glasserman's method is based on can be explored depth-first, which means that it is possible to wait with computing asset prices until they are actually needed for calculation. This feature leads to minimal storage requirements of only $O(bd)$.

  Also, the computational costs of the method ($O(nb^d)$) are quite low for a small number of possible exercise dates.

Different ways in which the asset price dynamics are simulated:

- The bundling algorithm is based on simulated asset price paths which all have to be computed at the beginning of the algorithm.
- The BRSS approach also uses asset price paths. They do not all have to be computed at the beginning of the algorithm, however.
- Broadie and Glasserman's algorithm works with asset price trees which are explored depths first, meaning that only the values that are necessary for current calculations have to be calculated and stored.

Chapter 8

Summary and Conclusions

The objective of this thesis was the evaluation of financial options by Monte Carlo simulation methods. Thereby, we distinguished two cases of options, European and American style ones. In the case of evaluating simple path-independent European type options, the Monte Carlo simulation approach is straightforward and can be summarized as follows:

- Generate a sequence of uniform pseudo-random or quasi-random numbers that can then be used as input for either an inversion or transformation algorithm in order to get standard normal numbers as discussed in Chapter 5.

- Compute the respective solutions of the SDEs based on those random numbers that simulate the asset price movement according to Chapter 4.

- Use the Law of Large Numbers (Chapter 3) to approximate the expectation of the option value at expiry by averaging over all sample solutions.

- Discount that expectation in order to derive an approximation of the true option value.

Thereby, variance reduction methods such as antithetic variates can be used to improve the accuracy of the method quite efficiently if the original value and the antithetic variate are negatively correlated as could be seen in Chapter 6.
For a discussion of other methods of variance reduction such as control variates, moment matching methods or importance sampling we refer to [2].

We also saw in Chapter 6 that Quasi-Monte-Carlo methods outperform regular Monte Carlo simulation approaches by a large degree. Using the same amount of low discrepancy samples as pseudo-random numbers, the accuracy improves significantly. Accordingly the number of samples and with it the computing time needed to achieve

the same level of accuracy can be reduced radically by working with numbers of the Halton sequence in the one dimensional case.
However, the application of low discrepancy sequences can be problematic in higher dimensional cases. Other sequences superior to the Halton sequence have to be used here to yield satisfactory results. This topic is extensively covered in [20] and [2].

If the option under consideration is American style, the different simulation methods vary widely in their approaches for dealing with the early exercise feature and the bias problem. Out of the many methods available we presented three here an put special emphasis on Broadie and Glasserman's method which avoids the bias problem by working with two estimators that are asymptotically unbiased for a large enough number of simulation runs. This procedure leads to both a fairly accurate point estimate and a confidence interval in relatively short computing time for a small number of exercise opportunities.
For a survey on the results yielded by the use of antithetic variates as well as the application of the method to higher dimensional problems see [4].

Appendix A

Black-Scholes

In this chapter we show how Ito's Lemma can be used for deriving the Black-Scholes Equation, that is an equation for V that does not explicitly depend on W_t and its solutions. We used this solution as a reference for the accuracy of the Monte Carlo simulation in earlier chapters.

A.1 The Black Scholes Analysis

In the following, the assumptions we made in our Market Model in chapter 2 are still valid. This means in particular that the price of the underlying asset follows the Geometric Brownian Motion given in Equation (4.19).

Applying Ito's Lemma to this SDE, we get

$$dV = \left(\mu S \frac{\partial V}{\partial S} + \frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} \right) dt + \sigma S \frac{\partial V}{\partial S} dW. \tag{A.1}$$

In order to eliminate the stochastic parts in Equation (A.1), let us consider a portfolio consisting of a short option with value V and Δ units of the underlying asset with a price of S each. Thereby Δ is chosen in a way such that the portfolio is riskless. With the value of this portfolio being

$$\Pi = -V + \Delta S \tag{A.2}$$

we see that the change in the value during one time-step is given by

$$d\Pi = -dV + \Delta dS. \tag{A.3}$$

Note that Δ is fixed during each time step, which is why the term $d\Delta$ is not necessary. Using Equations (4.19) and (A.1) for the terms dS and dV respectively we get

$$d\Pi = -\left(\mu S \left(\Delta - \frac{\partial V}{\partial S} \right) + \frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} \right) dt + \left(-\frac{\partial V}{\partial S} + \Delta \right) \sigma S dW. \tag{A.4}$$

Following the risk-neutral investment strategy, our goal will now be to hold exactly as many units of the underlying asset as are needed to eliminate the stochastic risk in Equation (A.4). Thus, for

$$\Delta = \frac{\partial V}{\partial S}, \tag{A.5}$$

the change in the value during one time-step in our portfolio is given by the deterministic differential equation

$$d\Pi = \left(\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} \right) dt. \tag{A.6}$$

Let us now take a look at another portfolio where the same amount of money was invested in a sound bank at the risk-free interest rate r:

$$d\Pi = r\Pi dt = \left(rV - rS\frac{\partial V}{\partial S} \right) dt \tag{A.7}$$

Recalling the no-arbitrage principle, the two portfolios given in Equations (A.6) and (A.7) have to be equal, which leads to the standard partial differential equation

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV = 0 \tag{A.8}$$

also known as the Black-Scholes Equation.

A.2 Boundary and final conditions

Note that we have not yet specified what kind of option we are dealing with. Therefore, the Black-Scholes Equation (A.8) applies to any option whose value follows a sufficient smooth function $V(S,t)$.
Since we eliminated the drift term, σ is the only parameter modeling the stochastic behavior in the Black-Scholes Equation. Thus, except for σ, (A.8) is a standard partial differential equation, and therefore has indefinite many solutions. Since we are only interested in one, i.e. the true option value, we need to assign appropriate boundary conditions to the problem which will lead to a unique solution. Fortunately in the case of European options the final condition can be derived quite easily from the option value at expiry date $t = T$.

Let us first consider the case of a European call option with expiry T and strike price K. As we saw in Chapter 2, the value of the call at expiry is given by

$$C(S,T) = \max(S - K, 0), \tag{A.9}$$

which is the final condition for the Black-Scholes Equation. Additionally we know that the right to buy a worthless asset is worthless itself, thus

$$C(0,t) = 0. \tag{A.10}$$

On the other hand, if the value of the underlying asset is very high, it becomes extremely likely for the option to be exercised, which leads to an approximate value of the option of $C(S,t) = Ke^{-r(T-t)}$. However, for very large values of S, the strike price K can be neglected, and we get

$$C(S,t) \sim S \text{ as } S \to \infty. \tag{A.11}$$

The value of a European put option reacts in quite a different way to a high price of the underlying asset. The higher it rises, the less likely it becomes for the put option to be exercised, thus $P(S,t) \to 0$ for $S \to \infty$.
In order to derive the value for $S = 0$, we use the put-call parity introduced in chapter 2:

$$P(0,t) = \big(C(S,t) + Ke^{-r(T-t)} - S\big) \mid_{S=0}= Ke^{-r(T-t)}. \tag{A.12}$$

To sum it up, we get the following boundary conditions

$$V(0,t) = 0, V(S,t) \sim (S \to \infty) \tag{A.13}$$

for a European call and

$$V(0,t) = Ke^{-r(T-t)}, V(S,t) \to 0(S \to \infty) \tag{A.14}$$

for a European put.

A.3 The Black-Scholes formulae

If we restrict ourselves to simple European options with non-stochastic interest rate and volatility, it is possible to solve the Black-Scholes Equation (A.8) explicitly without the need of numerical methods.

Theorem A.1. *For a European Call the solution to the Black Scholes Equation (A.8) with the Final Condition (A.9) and the boundary conditions (A.13) is given by*

$$V(S,t) = S\Phi(d-1) - Ke^{-r(T-t)}\Phi(d_2),\, S > 0,\, 0 \leq t < T, \tag{A.15}$$

where Φ is the standard deviation of the normal distribution

$$\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-s^2/2} ds,\, x \in \mathbb{R}, \tag{A.16}$$

and

$$d_1 = \frac{ln(\frac{S}{K}) + (r + \frac{\sigma^2}{2})(T-t)}{\sigma\sqrt{T-t}} \tag{A.17}$$

and

$$d_2 = \frac{ln\frac{S}{K} + (r - \frac{\sigma^2}{2})(T-t)}{\sigma\sqrt{T-t}} \tag{A.18}$$

For a proof of this theorem we refer to M. Günther, A. Jüngel's book [12], p 57.

Options on dividend-paying assets:

In Chapter 4 we learned that the price of a discrete time dividend-paying asset follows the SDE

$$dS = \sigma S dW + (\mu - D_0)Sdt \tag{A.19}$$

Although the 'dt' term is eliminated when deriving the Black-Scholes Formula, the dividend-paying feature of the asset still has an effect on the value of the portfolio we used for deriving the formula and thus influences the Black-Scholes Equation. The change in value of our portfolio in one period of time is now given by

$$d\Pi = dV - \Delta dS - D - 0S\Delta dt. \tag{A.20}$$

Executing all steps for deriving the Black-Scholes formula that we used in the non-dividend-paying case delivers

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + (r - D_0)S\frac{\partial V}{\partial S} - rV = 0 \tag{A.21}$$

as the Black-Scholes Equation for dividend-paying assets.
For the European call option, the final condition remains the same, and so does the boundary condition at $S = 0$. The only boundary condition that needs to be changed is that at $S \to \infty$. At this limit the option value equals that of the asset, but without the additional income of the dividend, which leads to the boundary condition

$$C(S,t) \sim Se^{-D_0(T-t)}, \text{ as } S \to \infty. \tag{A.22}$$

Thus the value of the European call is given by

$$C(S,t) = e^{-D_0(T-t)}SN(d_{10}) - Ke^{-r(T-t)}N(d_{20}), \tag{A.23}$$

where

$$d_{10} = \frac{log(\frac{S}{K}) + (e - D_0 + \frac{1}{2}\sigma^2)(T-t)}{\sigma\sqrt{T-t}} \tag{A.24}$$

and

$$d_{20} = d_1 - \sigma\sqrt{T-t}. \tag{A.25}$$

The corresponding solution for a European put option on a dividend-paying asset can be derived using the put-call parity introduced in Chapter 2.
For further reading see [12], [23] and [25].

Appendix B

The Euler-Maruyama algorithm

When dealing with more complex options or asset price models that are more complicated than the one-dimensional geometric Brownian motion, it is not possible to solve the SDE(s) describing the asset price movement via Ito's Lemma anymore. This is when numerical methods come into play.
In this section we want to introduce the Euler-Maruyama algorithm as presented in [12]. It was used for modeling asset price paths on the right hand side of Figure 4.3 with the Matlab programm C.2.3 or in Example 6.14 when we dealt with a stochastic volatility. For simplicity reasons, we will fall back on the simple case of dealing with a one-dimensional Geometric Brownian Motion

$$dS_t = rS_t + \sigma S_t dW_t. \tag{B.1}$$

Instead of solving this SDE with help of Ito's Lemma, it can also be approximated by

$$\Delta S_t = rS_t \Delta t + \sigma S_t \Delta W_t. \tag{B.2}$$

Recalling that the Wiener process W_t is given by

$$W - t + \Delta t = W_t + Z\sqrt{\Delta t}, \tag{B.3}$$

where Z is a sample of a standard normal variable, the approximation B.2 can be rewritten as

$$S_{i+1} - S_i = \Delta S_i = rS_i \Delta t + \sigma S_i Z \sqrt{\Delta t}. \tag{B.4}$$

This is the basis of the Euler-Maruyama algorithm:

Algorithm B.1. *Euler-Maruyama Algorithm*

```
- Initialize the step size $h$ $(\Delta t)$, the number of steps $n$;
  $r$, $\sigma$, $S_0$;

- for $i = 0$ to $n$ do
  $S_(i + 1) = S_(i)(1 + rh + \sigma dW_i)$
```

Appendix C

MATLAB codes

The following MATLAB codes are based on the algorithms listed in the books [12] and [23] as well as in the paper [4].

C.1 MATLAB code for Random number Generators

In this section we list all algorithms for computing random numbers that were used throughout the thesis.

C.1.1 Congruential

```
 function [U] = congruential(N);


 a = 1229; b = 1; m = 2048;


 X(1) = 1;
 for i = 2:N+1
    X(i) = mod(a*X(i - 1)+b,m);
    U(i-1) = X(i)/m;
 end
```

C.1.2 Fibonacci

```
 function [U] = fibonacci(M)


m = 2179;


X(1) = 1; X(2)=2; U(1) = X(1)/m; U(2) = X(2)/m;
```

```
 for i = 3:M
    X(i) = mod(X(i-1)+X(i-2),m);
    U(i) = X(i)/m;
end
```

C.1.3 Lagged Fibonacci

```
 function [U] = fibolagged(M)

 m=2048; nu=17; mu=5;
 rand('state',3)
 X = m*rand(1,max(nu,mu));

for i= max(mu, nu)+1:M+max(mu, nu)
   X(i) = mod(X(i-mu)-X(i-nu),m);
   U(i-max(mu, nu)) = X(i)/m;
end
```

C.1.4 Boxmuller

```
 function [Z1] = multiboxmueller(M,U);
%generates M normally distributed numbers out of M uniformly
%distributed ones

%U picks the source for the uniformly distributed random numbers:
%U = 1: rand
%U = 2: congruential
%U = 3: lagged fibonacci
%U = 4: halton sequence with base 2 and 3

if U == 1
    u = rand(1,M);
    u2 = rand(1,M);

elseif U == 2
        ConHelp = congruential(2*M);
        u = ConHelp(1:M);
        u2 = ConHelp(M+1:2*M);

elseif U == 3
        FibHelp = fibolagged(2*M);
        u = FibHelp(1:M);
        u2 = FibHelp(M+1:2*M);

elseif U == 4
        u= corput(M,2);
        u2= corput(M,3);
end

Z1 = sqrt(-2*log(u)).*cos(2*pi*u2);
```

C.1.5 Moro

```
%Moro's inversion algorithm for inverting the standard normal
%distribution function

function [Z1] = Moro(M,U);

a0 = 2.50662823884;
a1 = -18.61500062529;
a2 = 41.39119773534;
a3 = -25.44106049637;

b0 = -8.47351093090;
b1 = 23.08336743743;
b2 = -21.06224101826;
3 = 3.13082909833;

c0 = 0.3374754822726147;
c1 = 0.9761690190917186;
c2 = 0.1607979714918209;
c3 = 0.0276438810333863;
c4 = 0.0038405729373609;
c5 = 0.0003951896511919;
c6 = 0.0000321767881768;
c7 = 0.0000002888167364;
c8 = 0.0000003960315187;

%generates M normally distributed numbers out of M uniformly
%distributed ones

%U picks the source for the uniformly distributed random numbers:
```

```
%U = 1: rand
%U = 2: congruential
%U = 3: lagged fibonacci
%U = 4: halton sequence with base 2

if U == 1
    u = rand(1,M);
elseif U == 2
        u = congruential(M);
elseif U == 3
        u = fibolagged(M);
elseif U == 4
        u = corput(M,2);
end

x = zeros(1,M); y = u - 0.5;
for i= 1:M

   if abs(y(i)) < 0.42
   %main

       r(i) = y(i)^2;
       x(i) = y(i)* (((a3*r(i) + a2)*r(i) + a1)*r(i) +
       a0)/((((b3*r(i) + b2)*r(i) + b1)*r(i) +b0)*r(i)+1);

   elseif abs(y(i)) >= 0.42
   %tails
       r(i)= u(i);
       if y(i) > 0
          r(i) = 1 - u(i);
       end
       r(i) = log(-log(r(i)));
       x(i) = c0 + r(i)*(c1 + r(i)*(c2 + r(i)*(c3 + r(i)*(c4 +
       r(i)*(c5 + r(i)*(c6 + r(i)*(c7 + r(i)*c8)))))));

       if y(i) < 0
          x(i) = -x(i);
       end
   end
end
```

```
 Z1 = x;
```

C.1.6 Halton sequence

```
function x = halton(N,b);

%computes the first N numbers of the halton sequence for base b

m = fix(log(N)/log(b)); % determining the highest power

D = []; n = 1:N;

for i = 0:m
    d = mod(n,b);
    n = (n-d)/b;
    D = [D;d];
end
x = ((1/b).^(1:(m+1)))*D;
```

C.2 MATLAB codes used in Chapters 4 and 6

C.2.1 Black Scholes

These programs provide us with the value V_{exakt} which is needed as a reference for calculating the relative error:

European call (with discrete dividends)

```
function result = divcall (S,t,K,r, sigma, T,delta)
%computes the value of a European call with discrete dividends

d1 = (log(S/K) + (r-delta +0.5*sigma^2)*(T-t))/(sigma*sqrt(T-t)); %log = ln
d2 = d1 - sigma*sqrt(T-t); n1 = 0.5*(1 + erf(d1/sqrt(2)));
n2 = 0.5*(1 + erf(d2/sqrt(2)));
result = S.*exp(-delta*(T-t)).*n1 - K*exp(-r*(T-t))*n2;
```

Remark C.1. *In the case of an option on a non-dividend paying underlying δ can simply be set to equal zero.*

European put (with discrete dividends)

```
function result = divput (S,t,K,r, sigma, T, delta)
```

```
d1 = (log(S/K) + (r-delta +0.5*sigma^2)*(T-t))/(sigma*sqrt(T-t)); %log = ln

d2 = d1 - sigma*sqrt(T-t);

n1 = 0.5*(1 + erf(d1/sqrt(2)));

n2 = 0.5*(1+erf(d2/sqrt(2)));

result = K*exp(-r*(T-t))*(1-n2)-S.*exp(-delta*(T-t)).*(1-n1);
```

C.2.2 Wiener Process

This algorithm was used for computing the plot in figure (4.2):

```
 n = 1000;  % number of time steps
 T = 1; %expiry date
 h = T/n; %delta t
 M = 3; %number of simulation runs or paths
 W = zeros(n,M);

  randn('state',24) Z = randn(n,M);

 for i = 1:n
     W(i+1,:) = W(i,:) + Z(i,:).*sqrt(h);
 end

 figure(1)
 plot(W)
 title('different paths of a wiener process')
 xlabel('time t')
 ylabel('wiener process')
```

C.2.3 Asset Price Movement

The plot for the right hand side of Figure 4.3 was computed with this program. It uses the Euler-Maruyama discretization algorithm introduced in Appendix (B)

```
%plots  M different asset price paths for n timesteps


n= 100; h =1/n;  r=0.1; sigma = 0.4; M=3; S0= 100;

delta = 0;

S = zeros(n+1,M); randn('state',3) S(1,:) = S0;


dW = randn(n,M).*sqrt(h);

for i=1:n

      S(i+1,:) = S(i,:).*(1+(r-delta)*h + sigma*dW(i,:));

end


figure(1)

PH = [1:1:n+1];
```

```
plot(PH,S)

title('different asset price paths')

xlabel('time t')

ylabel('asset price')
```

C.2.4 Monte Carlo simulation for evaluating a European call option

```
%LONG RUNNING TIME!!!

%Plots the relativ error in option price (in comparison to black-scholes)
%for the one dimensional stochastic process solved with Ito's formula for
%regular MC, VR MC, QMC and VR QMC


randn('state',2) K = 100; r = 0.07; sigma = 0.3; delta = 0.1; T = 1;
S0 =120; n = 10; h = T/n;


M = 30000
%Simultaneous computation of 'Wiener Prozesses ' for M paths:

HBM = multiboxmueller(M,4); %halton boxmuller quasi-random matrix
HM = moro(M,4); %halton moro quasi random matrix

dWI = sqrt(T) * randn(1,M); %
dWH = sqrt(T) * HBM; %boxmueller fr halton
dWM = sqrt(T) * HM; % moro fr halton

%simultaneous calculation of the asset price movement with Ito's formula:
%SI = zeros(1,M);
SI(1,:) = S0 *exp(((r-delta) - 1/2 * sigma^2)*T + sigma * dWI);

%simultaneous calculation of the asset price movement with Ito's formula
%for QMC (Halton-Box_Muller):
SH(1,:) = S0 *exp(((r-delta) - 1/2 * sigma^2)*T + sigma * dWH);

%simultaneous calculation of the asset price movement with Ito's formula
%for QMC (Halton-moro):
SM(1,:) = S0*exp(((r-delta) - 1/2 * sigma^2)*T + sigma * dWM);


%simultaneous calculation of the asset price movement with Ito's formula
%for antitetic variates:
SIA(1,:) = S0 *exp(((r-delta) - 1/2 * sigma^2)*T - sigma *
dWI(1,:));

%simultaneous calculation of the asset price movement with Ito's formula
%% for QMC (Halton-Box_Muller) and antitetic variates:
SHA(1,:) = S0 *exp(((r-delta) - 1/2 * sigma^2)*T - sigma *
dWH(1,:));

%simultaneous calculation of the asset price movement with Ito's formula
%% for QMC (Halton-moro) and antitetic variates:
SMA(1,:) = S0 *exp(((r-delta) - 1/2 * sigma^2)*T - sigma *
dWM(1,:));

%Simultaneous calculation of the payoff function (put):
```

```
%payoffI = max(0, K-SI(1,:)); %Ito
%payoffIA = 0.5 * (max(0, K-SI(1,:)) + max(0, K-SIA(1,:))); %AV
%payoffH = max(0, K-SH(1,:)); %LD
%payoffHA = 0.5 * (max(0, K-SH(1,:)) + max(0, K-SHA(1,:))); %AV LD
%payoffM = max(0, K-SM(1,:)); %moro
%payoffMA = 0.5 * (max(0,K-SM(1,:)) + max(0,K-SMA(1,:))); %moro AV

%Simultaneous calculation of the payoff function (call):
payoffI = max(0, SI(1,:)-K); %Ito
payoffIA = 0.5 * (max(0, SI(1,:)-K) + max(0, SIA(1,:)-K)); %AV
payoffH = max(0, SH(1,:)-K); % LD boxmueller
payoffM = max(0, SM(1,:)-K); % LD moro
payoffHA = 0.5 * (max(0, SH(1,:)-K) + max(0, SHA(1,:)-K)); %AV LD boxmueller
payoffMA = 0.5 * (max(0, SM(1,:)-K) + max(0, SMA(1,:)-K)); %AV LD moro

%for i=1:M
%aM(:,1) = mean(payoffI); bM(:,1) =std(payoffI); %MC
%aMIA(:,1) = mean(payoffIA); bMIA(:,1) = std(payoffIA); %QMC
%aMH(:,1) = mean(payoffH); bMH(:,1) = std(payoffH); %AV
%aMM(:,1) = mean(payoffM); bMM(:,1) = std(payoffM); %AV QMC
%aMHA(:,1) = mean(payoffHA); bMA(:,1) = std(payoffHA); %moro
%aMMA(:,1) = mean(payoffMA); bMMA(:,1) = std(payoffMA); %moro AV


for i=1:M
%simultaneous calculation of the estimator for the option value
VI = exp(-r*T)*(cumsum(payoffI)./(1:M)); VI(M); VI2(1,:) =
exp(-r*T).*payoffI(1,:); value = VI(M); confl(1,i) =
mean(VI2(1,1:i)) - 1.96*std(VI2(1,1:i))./sqrt(i); confr(1,i) =
mean(VI2(1,1:i)) + 1.96*std(VI2(1,1:i))./sqrt(i);

VIA = exp(-r*T)*(cumsum(payoffIA)./(1:M)); VIA(M); VIA2(1,:) =
exp(-r*T).*payoffIA(1,:); valueAV = VIA(M); conflA(1,i) =
mean(VIA2(1,1:i)) - 1.96*std(VIA2(1,1:i))./sqrt(i); confrA(1,i) =
mean(VIA2(1,1:i)) + 1.96*std(VIA2(1,1:i))./sqrt(i);

VH = exp(-r*T)*(cumsum(payoffH)./(1:M)); VH(M); VH2(1,:)=
exp(-r*T).*payoffH(1,:); valueBM = VH(M); conflBM(1,i) =
mean(VH2(1,1:i)) - 1.96*std(VH2(1,1:i))./sqrt(i); confrBM(1,i) =
mean(VH2(1,1:i)) + 1.96*std(VH2(1,1:i))./sqrt(i);

VM = exp(-r*T)*(cumsum(payoffM)./(1:M)); VM(M); VM2 =
exp(-r*T).*payoffM(1,:); valueM = VM(M); conflM(1,i) =
mean(VM2(1,1:i)) - 1.96*std(VM2(1,1:i))./sqrt(i); confrM(1,i) =
mean(VM2(1,1:i)) + 1.96*std(VM2(1,1:i))./sqrt(i);

VHA = exp(-r*T)*(cumsum(payoffHA)./(1:M)); VHA(M); VHA2=
exp(-r*T).*payoffHA; valueBMA = VHA(M); conflBMA(1,i) =
mean(VHA2(1,1:i)) - 1.96*std(VHA2(1,1:i))./sqrt(i); confrBMA(1,i) =
mean(VHA2(1,1:i)) + 1.96*std(VHA2(1,1:i))./sqrt(i);

VMA = exp(-r*T)*(cumsum(payoffMA)./(1:M)); VMA(M); VMA2 =
exp(-r*T).*payoffMA; valueMA = VMA(M); conflMA(1,i) =
mean(VMA2(1,1:i)) - 1.96*std(VMA2(1,1:i))./sqrt(i); confrMA(1,i) =
mean(VMA2(1,1:i)) + 1.96*std(VMA2(1,1:i))./sqrt(i); end



%Plot1: the reltiv error in option value (in comparison to Black-Scholes)
%Vexakt = divput (S0,0,K,r,sigma,T,delta); %The Black-Scholes value for a  European put
Vexakt = divcall (S0,0,K,r,sigma,T,delta); %The Black-Scholes value for a European call
```

```
figure(1) hold on
%axis([0,M+2000,9,12])%100
axis([0,M+2000,2,4])%80
plot(VI,'b') plot(Vexakt.*ones(1,M),'k') plot(confl,'c')
plot(confr,'m') title('Monte Carlo simulation') xlabel('number of
simulations') ylabel('option value') hold off

figure(2) hold on
%axis([0,M+2000,9,11])%100
axis([0,M+2000,2.3,4])%80
plot(VIA,'b') plot(Vexakt.*ones(1,M),'k') plot(conflA,'c')
plot(confrA,'m') title('Monte Carlo simulation with AV')
xlabel('number of simulations') ylabel('option value') hold off

figure(3) hold on
%axis([0,M+2000,9,11])%100
axis([0,M+2000,2.3,4])%80
plot(VH,'b') plot(Vexakt.*ones(1,M),'k') plot(conflBM,'c')
plot(confrBM,'m') title('Quasi Monte Carlo simulation (Boxmuller)')
xlabel('number of simulations') ylabel('option value') hold off

figure(4) hold on
%axis([0,M+2000,9,11])%100
axis([0,M+2000,1.8,3.5])%80
plot(VM,'b') plot(Vexakt.*ones(1,M),'k') plot(conflM,'c')
plot(confrM,'m') title('Quasi Monte Carlo simulation (Moro)')
xlabel('number of simulations') ylabel('option value') hold off

figure(5) hold on
%axis([0,M+2000,9,11])%100
axis([0,M+2000,2.3,4])%80
plot(VHA,'b') plot(Vexakt.*ones(1,M),'k') plot(conflBMA,'c')
plot(confrBMA,'m') title('Quasi Monte Carlo simulation (Boxmuller)
with AV') xlabel('number of simulations') ylabel('option value')
hold off

figure(6) hold on
%axis([0,M+2000,9,11])%100
axis([0,M+2000,2.3,4])%80
plot(VMA,'b') plot(Vexakt.*ones(1,M),'k') plot(conflMA,'c')
plot(confrMA,'m') title('Quasi Monte Carlo simulation (Moro) with
AV') xlabel('number of simulations') ylabel('option value') hold off

figure(7) hold on
plot(abs(VI-Vexakt*ones(1,M))/Vexakt,'c')%normal
plot(abs(VIA-Vexakt*ones(1,M))/Vexakt,'m')%AV
plot(abs(VH-Vexakt*ones(1,M))/Vexakt,'k')%LD
%plot(abs(VM-Vexakt*ones(1,M))/Vexakt,'r')%LD moro
plot(abs(VHA-Vexakt*ones(1,M))/Vexakt,'y')%LD and AV
%plot(abs(VMA-Vexakt*ones(1,M))/Vexakt,'m')%LD and AV moro
%axis([0,M+100,0,0.05])%100
axis([0,M+2000,0,0.2])%80

legend('MC', 'MC with AV', 'QMC (boxmuller applied to the halton
sequence)', 'QMC (boxmuller applied to the halton sequence) and AV')
xlabel('number of simulations') ylabel('relative error')
title('European call with dividends')
%title('European put')
hold off
```

```
figure(8) hold on
plot(abs(VH-Vexakt*ones(1,M))/Vexakt,'c')%LD
plot(abs(VM-Vexakt*ones(1,M))/Vexakt,'m')%LD moro
title('comparison Boxmuller,Moro') legend('Boxmuller','Moro')
%axis([0,M+1000,0,0.02])%100
axis([0,M+2000,0,0.02])%80
hold off

figure(9) hold on plot(abs(VHA-Vexakt*ones(1,M))/Vexakt,'c')
plot(abs(VMA-Vexakt*ones(1,M))/Vexakt,'m') title('comparison AV
Boxmuller, AV Moro') legend('Boxmuller AV','Moro AV')
axis([0,M+1000,0,0.005]) hold off

%Histogram fr haltonboxmueller
figure(16) hist(HBM, [-3.8:0.1:3.8]) title('Boxmueller auf Halton')

figure(17) hist(HM, [-3.8:0.1:3.8]) title('moro auf halton')
```

C.2.5 Stochastic volatility

The plots for the example of stochastic volatility resulting in a three dimensional stochastic process were computed with this program. Again, we used the Euler-Maruyama discretization algorithm introduced in Appendix (B) for approximating the solution of the stochastic process.

```
%stochastic volatility

%Plots stochastic and mean volatility (figure1)

%Plots asset price movement with and without stochastc volatility (figure2)

%Option value with and without stochastic volatility   (figure3)

randn('state',5)
K = 100; %exercise/strike price
r = 0.1; %risk free interest rate
delta = 0.07, %dividend rate
ssigma0 = 0.3; %stochastic volatility
sigma = 0.3; % fixed volatility for Black-Scholes Vergleich
msigma0 = 0.3; % mean volatility
delta = 0.1 %dividend rate
alpha = 0.3; beta = 10; T = 1;
S0 =120; %asset price at time t=0
n = 250; % number of steps
h = T/n; %delta t
M = 10000; %number of simulations

%Initialising the two Wiener Processes:
dW1 = sqrt(h) * randn(n,M); dW2 = sqrt(h) * randn(n,M);

%Simultaneous calculation of the asset price movement with stochastic
%volatility:

%Initialising:
S = zeros(n+1,M); %asset price movement
```

```
S(1,:) = S0;
ssigma = zeros(n+1,M); %stochastic volatility
ssigma(1,:) = ssigma0;
msigma = zeros(n+1,M); %averaged volatility
msigma(1,:) = msigma0;


%Euler-Maruyama for solving the tree-dimensional stochastic process:

for i= 1:n
    S(i+1,:) = S(i,:).*(1 + (r-delta).*h + ssigma(i,:).* dW1(i,:));

    ssigma(i+1,:) = h .* msigma(i,:) + ssigma(i,:).*(1 - h + alpha.* dW2(i,:));

    msigma(i+1,:) = h .* ssigma(i,:).*beta + msigma(i,:).*(1 - beta * h );
end

%---------------------------------------------------------------------
%one-dimensional process: constant volatility:

%Simultaneous calculation of the asset price movement with constant volatility:
%Initialising:
SM = zeros(n+1,M); %asset price movement
SM(1,:) = S0;

%Euler-Maruyama for solving the one-dimensional stochastic process:
for i = 1:n
    SM(i+1,:) = SM(i,:).*(1 + (r-delta) *h + sigma *dW1(i,:));

end

%---------------------------------------------------------------------

%Simultaneous calculation of the Payoff funktion (put):
%payoff = max(0, K-S(n+1,:)); %stochastic volatility

%payoffM = max(0, K-SM(n+1,:)); %constant volatility


%Simultaneous calculation of the Payoff funktion (call):
payoff = max(0, S(n+1,:)-K); %stochastic volatility

payoffM = max(0, SM(n+1,:)-K); %constant volatility

%Calculation of the estiamtor for the option value
V = exp(-r*T)*(cumsum(payoff)./(1:M)); %stochastic volatility

VM = exp(-r*T)*(cumsum(payoffM)./(1:M)); %constant volatility



%Plot 1: realization of the volatility tandem stochastic volatility and mean
%volatility:
figure(1) hold on plot(ssigma(:,1)) plot(msigma(:,1),'m') hold off
legend('stochastic volatility', 'mean volatility') xlabel('time t')
ylabel('volatility') title('volatility tandem')

%Plot 2: Asset price movement with and without stochastic volatility
figure(2) hold on plot(S(:,1)) plot(SM(:,1),'m')
legend('stochastic volatility', 'constant volatility')%,
'stochastic volatility with AV','constant volatility with AV')
xlabel('number of timesteps') ylabel('asset price') title('asset
price movement’)


%Plot 3: Option value with and without stochastic volatility

figure(3) hold on plot(V) plot(VM,’m’) hold off
legend(’stochastic volatility’, ’constant volatility’)%,
’stochastic volatility with AV’, ’constant volatility with AV’)
xlabel(’number of timesteps’) ylabel(’option value’) title(’option
value development’)
```

C.2.6 Historical volatility

This is the MATLAB code for computing the example of historical volatility.

```
%vektor with asset price values:
A=[9.64,9.10,9.32,9.70,9.79,10.34,11.29,11.43,11.08,11.75,11.34,...
11.87,11.57,12.17,12.44,12.75,12.54,12.38,12.11,11.68,11.42,11.64,...
11.71,11.90,11.65,10.77,11.67,12.17,12.39,12.04,12.36,12.25,12.17,...
12.11,11.91,11.78,11.85,11.32,12.95,12.43,13.36,13.60,13.88,13.70,...
13.85,13.97,14.10,13.88,13.84,14.47,14.80,12.65,12.99];


for i=1:51
    y(i)=log(A(i+1))-log(A(i));
end
ysum=cumsum(y);
yq=1/51*(ysum(51));
for i=1:51
    ydach=(y(i)-yq)^2;
end

vhist = sqrt(262)*sqrt((1/51) *ydach)
```

C.3 MATLAB code for Broadie and Glasserman algorithm

C.3.1 Broadie and Glasserman algorithm

```
%Broadie and Glasserman algorithm for AO

d = 3; %number of possible exercise dates t=0,t=T/2,t=T
b = 50; %number of branches per node in a tree 50
S = 80; %underlying asset value at start %100
n=200;
```

```
K = 100; %exercise\strike price 110
r = 0.07; %risk-free interest rate 0.05
sigma = 0.3; %volatility 0.2
delta = 0;% 0.1; %dividends
T = 1;

randn('state',2) for i=1:n
  Lesthelp(i) = brandglasLest(d,b,S,K,r,sigma,delta,T);
  Lest(i) = mean(Lesthelp);

%confidence intervall:
confl(i) = max(max((S-K),0), Lest(i) -
1.96*std(Lesthelp(1:i))/sqrt(i)); end

randn('state',2) for j=1:n
 Hesthelp(j) = brandglasHest(d,b,S,K,r,sigma,delta,T);
 Hest(j) =  mean(Hesthelp);

 %confidence interval:
 confr(j) = Hest(j) + 1.96* std(Hesthelp(1:j))/sqrt(j);
end

%point estimate:
V(1,:) = 0.5* max(max((S - K),0), Lest(1,:)) + 0.5* Hest(1,:);

Vexakt= divcall(S,0,K,r, sigma, T,delta)

figure(1) hold on plot(Vexakt.*ones(1,n),'k') plot(confl,'c')
plot(confr,'m') plot(V,'b') title('b=50') xlabel('number of
simulations') ylabel('option value') hold off
```

C.3.2 Low estimator tree

```
%Broadie and Glasserman Algorithmus
function result = brandglasLest(d,b,S,K,r,sigma,delta,T);

lest = 0; h = T/(d-1);


w = ones(1,d); %vector that helps decide where in the matrix the asset price is derived from
v = ones(b,d); %matrix that holds node value and asset price
x = 1;%helps compute asset price in the right place in matrix (case 1; case 3.1)
cv = 0; %Helps sum up all node values in that loop (for 3.1 and 4)
nue = 0; %helps sum up all values for the loop for nu in (3 and 4)

%initialize parameters
v(1,1) = S; for j = 2:d

    %the first branch of the asset price tree including the first final
    %node:
    v(1,j) = v(1,j-1) *exp(((r-delta)-1/2 *sigma^2) *h + sigma* sqrt(h) *randn(1,1));

end

%process tree
j = d; while (j > 0)
   %w,j
   if j == d & w(j) < b %terminal node (j=d)
        %case 1
        v(w(j),j) = max(v(w(j),j) - K, 0);

        v(w(j)+1,j) = v(x,j-1) *exp(((r-delta)-1/2 *sigma^2) *h + sigma* sqrt(h) *randn(1,1));
        w(j) = w(j) + 1;

    elseif j == d & w(j) == b %terminal node (j=d)
        %case2

        v(w(j),j) = max(v(w(j),j) - K, 0);
        w(j) = 0;
        j = j-1;

    elseif j < d & w(j) < b %intermediate node (j<d)
        %case3

         for sh = 1: b
            for i = 1 : sh -1 %unser sh ist das i=j
                cv = cv + exp(-r*h)*v(i,j+1);
            end
            for i = sh + 1  : b
                cv = cv + exp(-r*h)*v(i,j+1);
            end
            if max(v(w(j),j) - K, 0) >= 1/(b-1) * cv
               nu = max(v(w(j),j) - K, 0);
            else
               nu = exp(-r*h)*v(sh,j+1);
           end
           cv = 0;
           nue = nue + nu;

       end
       v(w(j),j) = (1/b) * nue;
       nue = 0;
        %'node value';
        if j > 1
            %case3.1

            v(w(j)+1,j) = v(w(j-1),j-1)*exp(((r-delta)-1/2 *sigma^2)*h+sigma*sqrt(h)*randn(1,1));
            w(j) = w(j) + 1;
            for i = j + 1 : d
                v(1,i) = v(w(j),j)*exp((r-delta-1/2 *sigma^2)*h+sigma*sqrt(h)*randn(1,1));

                w(i) = 1;
            end
            j = d;

            if x < b
                x = x+1;
            else x = 1;
            end


        else j =0;
            %case3.2

        end
    elseif j < d & w(j) == b %intermediate node (j<d)
        %case4

        for sh = 1: b
            for i = 1 : sh -1 %unser sh ist das i=j
                cv = cv + exp(-r*h)*v(i,j+1);
            end
            for i = sh + 1  : b
                cv = cv + exp(-r*h)*v(i,j+1);
            end
            if max(v(w(j),j) - K, 0) >= 1/(b-1) * cv
               nu = max(v(w(j),j) -K, 0);
            else
               nu = exp(-r*h)*v(sh,j+1);
           end
           cv = 0;
           nue = nue + nu;

       end
       v(w(j),j) = (1/b) * nue;
       nue = 0;
        w(j) = 0;
        j = j - 1;

    end


end lest = lest + v(1,1);

result = lest;
```

C.3.3 High estimator tree

```
%Broadie and Glasserman Algorithmus for the High estimator
%AO
function result = brandglasHest(d,b,S,K,r,sigma,delta,T)

h = T/(d-1);


w = ones(1,d); %vector that helps decide where in the matrix the asset price is derived from
v = ones(b,d); %matrix that holds node value and asset price
x = 1;%helps compute asset price in the right place in matrix (case 1; case 3.1)
cv = 0; %Helps sum up all node values in that loop (for 3.1 and 4)


%initialize parameters
v(1,1) = S; for j = 2:d

    %the first branch of the asset price tree including the first final
    %node:
    v(1,j) = v(1,j-1) *exp(((r-delta)-1/2 *sigma^2) *h + sigma* sqrt(h) *randn(1,1));

end

%process tree
j = d; while (j > 0)
   %w,j
   if j == d & w(j) < b %terminal node (j=d)
        %case 1
        v(w(j),j) = max(v(w(j),j) - K, 0);

        v(w(j)+1,j) = v(x,j-1) *exp(((r-delta)-0.5 *sigma^2) *h + sigma* sqrt(h) *randn(1,1));

        w(j) = w(j) + 1;
    elseif j == d & w(j) == b %terminal node (j=d)
        %case2


        v(w(j),j) = max(v(w(j),j) - K, 0);
        w(j) = 0;
        j = j-1;

    elseif j < d & w(j) < b %intermediate node (j<d)
        %case3


        for i = 1:b
            cv = cv + exp(-r*h)*v(i,j+1);
        end
        v(w(j),j) = max(max(v(w(j),j)-K,0),1/b * cv);
        cv = 0;
        %'node value';
        if j > 1
            %case3.1

            v(w(j)+1,j) = v(w(j-1),j-1)*exp(((r-delta)-1/2 *sigma^2)*h+sigma*sqrt(h)*randn(1,1));

            w(j) = w(j) + 1;
            for i = j + 1 : d
                v(1,i) = v(w(j),j)*exp((r-delta-1/2 *sigma^2)*h+sigma*sqrt(h)*randn(1,1));


                w(i) = 1;
            end
            j = d;

            if x < b
                x = x+1;
            else x = 1;
            end


        else j =0;
            %case3.2

        end
    elseif j < d & w(j) == b %intermediate node (j<d)
        %case4
        for i = 1:b
            cv = cv + exp(-r*h)*v(i,j+1);
        end
        v(w(j),j) = max(max(v(w(j),j)-K,0),1/b * cv);
        cv = 0;

        w(j) = 0;
        j = j - 1;

    end
    %j
    %l=l+1;

end result = v(1,1);
```

Bibliography

[1] L. Arnold, *Stochastische Differentialgleichungen - Theorie und Anwendungen*; Oldenburg Verlag 1973.

[2] P. Boyle, M. Broadie, P. Glasserman: Monte Carlo methods for security pricing *Journal of Economic Dynamics and Control* vol. 21, 1997 pp 1267-1321.

[3] P. Brandimarte, *Numerical Methods in Finance - A MATLAB-Based Introduction*; Wiley series in probability and statistics. Financial engeneering section, John Wiley & Sons Inc., New York 2002.

[4] M. Broadie, P. Glasserman: Pricing American-style securities using simulation, *Journal of Economic Dynamics and Control* vol. 21, 1997 pp 1323-1352.

[5] L. Devroye, *Non-Uniform Random Variate Generation*; Springer New York 1986

[6] A. Etheridge, *A course in financial calculus*; Cambridge University Press 2004.

[7] M.C. Fu, S.B. Laprise, D.B. Madan, Y.Su, R. Wu: Pricing American Options: A Comparison of Monte Carlo Simulation Approaches, *Journal of Computational Finance* vol. 4, 2001 pp 39-88.

[8] S. Galanti, A. Jung : Low-Discrepancy Sequences: Monte Carlo Simulation Of Option Prices, *Journal of Derivatives* vol. 5, Fall 1997 pp 63-83.

[9] D. Grant, G. Vora, D. Weeks: Simulation of the Early-Exercise Option Problem, *Journal of Financial Engineering 3* vol. 5, 1997 pp 211-227.

[10] L. Grüne, *Skript zur Vorlesung Modellierung mit Differentialgleichungen*; 2003.

[11] L. Grüne, *Skript zur Vorlesung Numerische Mathematik II: Differentialgleichungen*; 2003.

[12] M. Günther, A. Jüngel, *Finanzderivate mit MATLAB*; Vieweg Verlag, 2003 (1st printing).

[13] D.J. Higham, *An introduction to Financial Option Valuation*; Cambridge University Press 2004.

[14] I. Karazas, S. Shreve, *Brownian Motion and Stochastic Calculus*; Springer Verlag 1988.

[15] P.E. Kloeden, E. Platen, *Numerical Solution of stochastic Differential Equations*; Springer Verlag 1999 (corrected 3rd printing).

[16] E. Korn, R. Korn, *Optionsbewertung und Portfolio-Optimierung - Methoden Moderner Finanzmathematik* Vieweg 1999 (1st printing).

[17] Y.K. Kwok, *Mathematical Models of Financial Derivatives*; Springer Verlag Singapore.

[18] C.B. Moler, *Numerical Computing with MATLAB*; SIAM 2004.

[19] O. Mußhoff, N. Hirschauer, K. Palmer: Bounded Recursive Stochastic Simulation - a simple and efficient method for pricing complex American type options, *Working Paper, Humboldt-Universität zu Berlin* 65/2002.

[20] H. Niederreiter, *Random Number Generation and Quasi-Monte Carlo Methods*; SIAM, Philadelphia 1992.

[21] D. Revuz, M. Yor, *Continuous Martingales and Brownian Motion*; Springer Verlag 1999.

[22] N. Schmitz, *Vorlesungen über Wahrscheinlichkeitstheorie*; Teubner Verlag Stuttgart 1996.

[23] R. Seydel, *Tools for Computational Finance*; Springer Verlag 2002.

[24] J. Tietze, *Einführung in die Finanzmathematik* Vieweg Verlag 6. Auflage

[25] P. Wilmott, J. Dewynne, S. Howison, *Option Pricing - Mathematical models and computation*; Oxford Financial Press 1997.

[26] P. Wilmott, S. Howison, J. Dewynne, *The mathematics of financial Derivatives*; Cambridge University Press 1997.

ERKLÄRUNG

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel angefertigt habe.

Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen.

Bayreuth, den September 28, 2005 ....

Andrea Kölz