NTNU

Norwegian University of
Science and Technology

Pricing a Bermudan Swaption using the LIBOR Market Model

A LSM approach

Ole-Petter Bård Anstensrud

Master of Science in Physics and Mathematics
Submission date: July 2008
Supervisor: Jacob Laading, MATH

Norwegian University of Science and Technology
Department of Mathematical Sciences

Problem Description

The aim of this study is to present the theoretical framework behind the LIBOR market model and use the model to price an exotic interest rate derivative. The model will be implemented and calibrated using real data. The exotic interest rate derivative is taken to be a Bermudan swaption and the The Least Squares Monte Carlo (LSM) algorithm will play a central role in the pricing procedure.

Assignment given: 18. February 2008
Supervisor: Jacob Laading, MATH

Preface

This report represents my work on my Master's thesis during the spring semester of 2008. It represents one full semester of work and completes my five year Master of Technology program at the Norwegian University of Science and Technology. The study is performed at the Department of Mathematical Sciences.

My project work fall 2007 discussed the use of Monte Carlo methods in the pricing procedure of some common types of stock options. This study concerns the pricing of various types of interest rate derivatives within the framwork of the LIBOR Market Model. This is in general more difficult than the preceding and often requires the use of higher dimensional simulations. The Least Squares Monte Carlo algorithm was used for pricing american options in my project work, and it was mentioned in the concluding chapter that this algorithm was especially well suited for multidimensional simulations. In this study we pursue that statement and use a multidimensional version of the algorithm to price a Bermudan swaption. The study depends heavily on computer simulation and a lot of C++ programming. This has been the source to some frustrating debugging, but in the end it always feels quite satisfactory to finally see your code up and working. I've tried to provide a fundamental understanding of the whole subject and not just a superficial treatment. This approach made the use of some stochastic calculus and the description of the relevant simulation techniques almost unavoidable, but I hope it's possible to gather the details together into a whole.

I would like to thank my supervisor Associate Professor II Jacob K. Laading for providing useful information and giving constructive feedback throughout the entire semester. I would also like to thank my student colleagues for a lot of fruitful discussions and my closest family for all the motivation and encourage they have given me.

Trondheim, July 2008

Ole-Petter B. Anstensrud

Abstract

This study will focus on the pricing of interest rate derivatives within the framework of the LIBOR Market Model. First we introduce the mathematical and financial foundations behind the basic theory. Then we give a rather rigouros introduction to the LIBOR Market Model and show how to calibrate the model to a real data set. We use the model to price a basic swaption contract before we choose to concentrate on a more exotic Bermudan swaption. We use the Least Squares Monte Carlo (LSM) algorithm to handle the early exercise features of the Bermuda swaption. All major results are vizualised and the C++ implementation code is enclosed in appendix B.

Contents

Preface **i**

Abstract **iii**

1 Introduction **1**

2 Mathematical and Financial Foundations **3**
2.1 Definitions 3
2.2 Choice of Numeraire 10
2.3 Historical Development of Interest Rate Models 15

3 Simulation and Monte Carlo Methods **19**
3.1 An Introduction to Monte Carlo Methods 19
3.2 Simulation of Random Variables 20
3.3 Discretization of Stochastic Differential Equations 22

4 The LIBOR Market Model **25**
4.1 Theory 25
4.2 Calibration to Market Data 27
4.3 Pricing a Swaption 29

5 The LSM Algorithm **35**
5.1 Theory 35
5.2 Pricing a Bermudan Swaption 36

6 Conclusion **41**

References **43**

A Results from Stochastic Calculus **45**

B Computer Code **51**
B.1 Main Code 51
B.2 Multinorm Header 61
B.3 Multinorm Class 61

1 Introduction

A swap is an agreement between two parties to exchange, or swap, future cashflows. This study will discuss interest rate swaps. These are swaps where two parties exchange cashflows represented by the interest rate on a notional principal. Typically one side agrees to pay the other a fixed interest rate. The cashflow in the opposite direction is a floating rate. One of the most common floating rates is the London Interbank Offer Rate, known as the LIBOR rate. Swap agreements are used both by speculators who possess a particular market view, and hedgers who want to manage their daily risk. The swap market is huge and very liquid. According to the Bank for International Settlements [1] the total notional amount of outstanding interest rate swaps was about $309.6 trillions by the end of 2007. A swap option, called a swaption, is an option on a swap agreement. This option provide one party with the right at a future time to enter into a swap where a predetermined fixed rate is exchanged for a floating rate. A Bermudan swaption is a swaption that can be exercised on some or all the agreed payment dates of the underlying swap. The precise definitions of these contracts will be stated in chapter 2.1. But how much is such a contract worth? The aim of this study is essentially to provide an answer to this question.

Interest rate derivatives are in many ways much harder to deal with than derivatives on stocks, like common stock options. Interest rates are for one thing much harder to model than stocks, but in addition we'll often have to work with a multiple set of correlated interest rates. This leads to a lot of calibration issues and higher dimensional simulations. The LIBOR Market Model is the modern way of pricing exotic interest rate derivatives, and will play the leading part in this thesis. To price the bermudan swaption we'll use the LSM (Least Squares Monte Carlo) algorithm modified to handle forward rates under a particular forward measure. The LSM algorithm was first proposed by Francis A. Longstaff and Eduardo S. Schwartz in 2001, see [11] for the original paper, and is particular suited to handle higher dimensional simulations with early exercise features.

Chapter 2 explains the basic concepts and notation, and give a brief introduction to the subject. This chapter deals with the necessary mathematical tools as well as the financial aspects of the contracts. It also offer a guided tour through the historical development of interest rate models. Chapter 3 gives a brief introduction to computer based simulation and Monte Carlo methods. Chapter 4 deduce the LIBOR Market Model from scratch and suggest a possible calibration scheme. We also show how to use the model to price a swaption. Chapter 5 is dedicated to the LSM algorithm and the pricing of a bermudan swaption. Chapter 6 contains come conclusions and final remarks. Appendix A states some useful results from stochastic calculus and reviews some of the most fundamental concepts of the theory.

All the major algorithms are implemented in C++, as listed in appendix B, and the LIBOR Market Model are calibrated to real data from DnB Nor.

[1] http://www.bis.org/

2 Mathematical and Financial Foundations

2.1 Definitions

Most people understand the basic mechanism of a bank account. They expect the amount of deposited money to grow as time goes by, and they expect their spare money to be safe from theft and financial risk. In every-day life this is what you really need to know, but in mathematical terms we need a more precise definition.

Definition 2.1 (Bank account). *We define $B(t)$ to be the value of a bank account at time $t \geq 0$. We assume $B(0) = 1$ and that the bank account evolves according to*

$$dB(t) = r_t B(t) dt$$

where r_t is a positive, and possibly stochastic, function of time, known as the instantaneous spot rate. r_t is assumed to be risk-free. Solving this differential equation gives

$$B(t) = \exp\left(\int_0^t r_s ds\right)$$

We assume there is no way to loose money on an investment at the risk-free rate. There is no such thing as a totally risk-free investment in real life. But even though there is a positive probability a bank can go bankrupt, we assume the bank account to be free of risk. There is no such thing as an instantaneous interest rate either, but in practice we can take the one-month spot rate quoted in the market as the instantaneous rate. The overnight rate is usually too illiquid for this purpose.

In many situations we want to know what an amount of money at time $T > t$ is worth at time t. If we allow the evolution of the interest rate to be a possible stochastic process we have the following definition of a discounting process

Definition 2.2 (Stochastic discount factor). *The possibly stochastic discount factor $D(t, T)$ between two time instants t and T is the amount at time t that is equivalent to one unit of currency payable at time T, and is given by*

$$D(t, T) = \frac{B(t)}{B(T)} = \exp\left(-\int_t^T r_s ds\right)$$

If the interest rate is a deterministic process in the form of a constant, say $r_t = r$, we obtain the well known formula $D(t, T) = e^{-r(T-t)}$. In the pricing of for example stock options we often make this assumption because we assume the variability of the interest rate to contribute considerably less to the price than the underlying stock. In very interest rate sensitive products, like interest rate derivatives, we just can't make this assumption and we need to model the interest rate as a stochastic process.

Bonds are instruments of crucial importance in the analysis of interest rate derivatives. A bond is a debt security, in which the authorized issuer owes the holders a debt. The bond's principal is received at some given date in the future known as the maturity date. Most bonds pay a periodically interest known as coupons to the holder. A zero-coupon bond is a bond where all interest is paid at maturity or more precice

Definition 2.3 (Zero-coupon bond). *A T-maturity zero-coupon bond (pure discount bond) is a contract that guarantees its holder the payment of one unit of currency at time T, with no intermediate payments. The contract value at time $t < T$ is denoted by $P(t,T)$. Clearly $P(T,T)=1$ for all T.*

It's possible to multiply $P(t,T)$ with an arbitrary principal, but we find it convenient to work with a scaled version of the zero-coupon bond. The zero-coupon bond can be seen as the expectation of the random variable $D(t,T)$ under the risk-neutral measure as seen in chapter 2.2.

LIBOR is short for London Interbank Offer Rate and is the spot rate at which a bank is prepared to make a deposit with other banks. Large banks and other financial institutions quote 1-month, 3-month, 6-month and 12-month LIBOR in all major currencies. In according to quoting LIBOR rates, large banks also quote LIBID rates. This is the London Interbank Bid Rate and is the spot rate at which they will accept deposits from other banks. Due to active trading there is usually a small spread between the quoted LIBID and LIBOR rates. In this study we only work with LIBOR rates and ignore this difference. LIBOR rates are simply-compounded rates and we denote them by $L(t,T)$ in accordance with the following definition:

Definition 2.4 (Simply-compounded spot interest rate). *The simply-compounded spot interest rate prevailing at time t for the maturity T is denoted by $L(t,T)$ and is the constant rate at which an investment has to be made to produce an amount of one unit of currency at maturity, starting from $P(t,T)$ units of currency at time t, when accruing occurs proportionally to the investment time. In formulas:*

$$P(t,T)(1 + L(t,T)\tau(t,T)) = 1$$
$$\Longrightarrow L(t,T) = \frac{1 - P(t,T)}{\tau(t,T)P(t,T)}$$

where $\tau(t,T)$ is the time difference $T - t$, usually quoted in years.

Bond prices can be expressed in terms of $L(t,T)$ as

$$P(t,T) = \frac{1}{1 + L(t,T)\tau(t,T)}$$

Although we will not make much use of other compounding types in this study we mention the continously-compounded interest rate, $R(t,T)$, given by $e^{R(t,T)\tau(t,T)}P(t,T) = 1$ and the annually compounded interest rate, $Y(t,T)$, given by $P(t,T)(1+Y(t,T))^{\tau(t,T)} = 1$.

The effective rate of interest achieved from a bond is usually referred to as the bond yield. A set of zero-coupons of different maturities can build up what we call a yield curve. The yield curve is also known as the term structure of interest rates. A yield curve is a plot of yield against time. An investor would in general expect a higher yield on a long-term investment than a short-term investment. The yield curve is therefore usually an increasing function of time to maturity. A flat, or maybe decreasing, yield

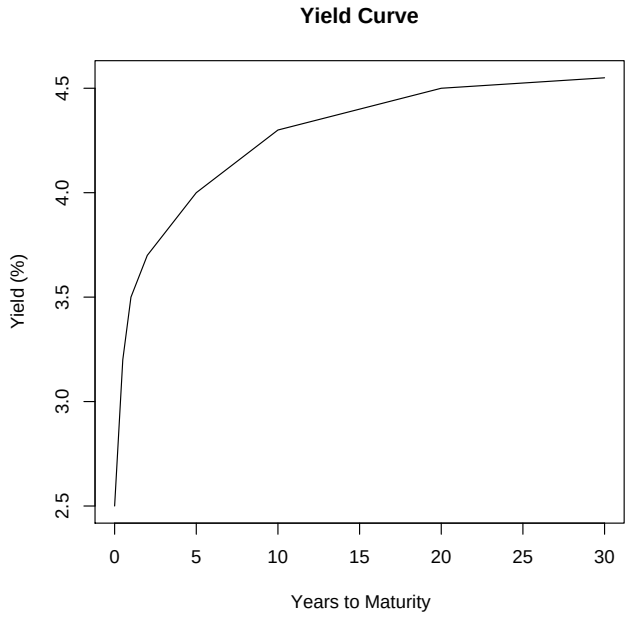

Figure 1: A normal yield curve

curve imply that the market expect a lower interest rate in the future. A typically shaped yield curve can be seen in figure 1.

Forward rates are interest rates that can be locked in today for a future time period, and are set consistently with the current term structure of interest rates. Assume we are at time t and buy one zero-coupon bond of maturity S and sell one zero-coupon bond of maturity T with the same principal, where $T < S$. At time T we pay back a principal with interests and at time S you receive a principal with interests. Measured by the yield with normalized zero-coupon bonds we pay $(1 + L(t,T)\tau(t,T))$ at time T and receive $(1 + L(t,S)\tau(t,S))$ at time S. If we call the forward rate between T and S at time t for $F(t;T,S)$ we have the equation

$$(1 + L(t,T)\tau(t,T))(1 + F(T,S)\tau(T,S)) = (1 + L(t,S)\tau(t,S))$$

By definition 2.4 we obtain

$$\begin{aligned}(1 + \frac{1 - P(t,T)}{P(t,T)})(1 + F(T,S)\tau(T,S)) &= (1 + \frac{1 - P(t,S)}{P(t,S)}) \\ \implies F(t;T,S) &= \frac{1}{\tau(T,S)}\left(\frac{P(t,T)}{P(t,S)} - 1\right)\end{aligned} \tag{1}$$

If we negligate the bid-ask spread, $F(t;T,S)$ is the (simply-compounded) interest rate at which you at time t can borrow or lend money between time T and S. A forward-rate

agreement is a contract between two parties involving the three time instants t, T and S. At maturity, S, one party pays the other a fixed payment based on a fixed rate K, and the other party pays a floating payment based on the spot rate $L(T,S)$ resetting in T and with maturity S. At time S the receiver of the fixed payment receives

$$\begin{aligned} & N\tau(T,S)(K-L(T,S)) \\ &= N\left[\tau(T,S)K - \frac{1}{P(T,S)} + 1\right] \end{aligned}$$

where N is the notional principal. $\frac{1}{P(T,S)}$ is worth $P(T,S)\frac{1}{P(T,S)} = 1$ at time T. This is equivalent to an amount of $P(t,T)$ at time t. $\tau(T,S)K+1$ is worth $P(t,S)(\tau(T,S)K+1)$ at time t. Hence at time t the contract is worth

$$\begin{aligned} & N[P(t,S)\tau(T,S)K - P(t,T) + P(t,S)] \\ =& NP(t,S)\tau(T,S)(K - F(t;T,S)) \end{aligned}$$

where we have used the definition of the forward rate in the last equation. Therefore, to value a forward rate agreement, we replace the LIBOR rate $L(T,S)$ with the corresponding forward rate $F(t;T,S)$ and discount with the corresponding zero-coupon bond $P(t,S)$. Note that the initial value of the forward rate agreement equates to zero if $K = F(t;T,S)$. We also show in chapter 2.2 that the forward rate can be seen as the expectation of the corresponding spot rate under the corresponding forward measure. When the maturity of the forward rate collapses towards its expiry, we have the notion of instantaneous forward rate. The instantaneous forward rate, $f(t,T)$, is expressed as

$$\begin{aligned} f(t,T) = \lim_{S\to T^+} F(t;T,S) &= -\lim_{S\to T^+} \frac{1}{P(t,S)} \frac{P(t,S)-P(t,T)}{S-T} \\ &= -\frac{1}{P(t,T)} \frac{\partial P(t,T)}{\partial T} \\ &= -\frac{\partial \ln P(t,T)}{\partial T} \end{aligned}$$

If an investor believes that rates in the future will be quite different from the forward rates observed in the market today, he can speculate in according to something called yield curve play. If an investor for example believes the 1-year interest rates won't change much the next 5 years, he can lend money for 1 year and invest them in a 5 year bond. The 1-year borrowings can be rolled over for further 1-year periods at the end of the first, second, third and fourth years. This is can be a lucrative, but dangerous business. Robert Citron, once the Treasurer at Orange County, used yield curve plays similar to this one very successfully in 1992 and 1993. The profit from Mr. Citron's trade became an important contributor to Orange County's budget and he was re-elected. In 1994 he expanded his yield curve play. If short-term interest rates had remained the same or declined he would have continued to do well, but interest rates rose sharply. On December 1, the same year, Orange County announced that its investment portfolio had

lost $1.5 billion. At that point Orange County was left with no recourse other than to file for bankruptcy. While in bankruptcy, every county program budget was cut, about 3,000 public employees were discharged and all services were reduced. Mr. Citron was found guilty of violating state investment laws and was sentenced to one year of community service. The lesson must be to never underestimate the risk potential of an investement.

A generalization of the forward-rate agreeement is the interest rate swap. An interest rate swap is a contract that exchanges payments between two differently indexed legs, starting from a future time instant. At every instant T_i in a prespecified set of dates $T_{\alpha+1,\ldots,T_\beta}$ the fixed leg pays out the amount $N\tau_i K$ corresponding to a fixed interest rate K, a nominal value N and a year fraction τ_i between T_{i-1} and T_i. The floating leg pays out the amount $N\tau_i L(T_{i-1}, T_i)$ at dates $T_{\alpha+1,\ldots,T_\beta}$ and resets at dates $T_{\alpha,\ldots,T_{\beta-1}}$. When the fixed leg is paid and the floating leg is received we call it a payer interest swap. Otherwise we call it a receiver interest rate swap. The swap agreement can be seen as a portifolio of forward rate agreements and hence has the valuation

$$N \sum_{i=\alpha+1}^{\beta} \tau_i P(t, T_i)(K - F(t; T_{i-1}, T_i))$$

for a receiver swap (for a payer swap we just change the sign).

Now we find the fixed rate, $K = S_{\alpha,\beta}(t)$, which equates the value of the whole agreement to zero:

$$\begin{aligned}
&N \sum_{i=\alpha+1}^{\beta} \tau_i P(t, T_i)(K - F(t; T_{i-1}, T_i)) \\
&= N \sum_{i=\alpha+1}^{\beta} [\tau_i P(t, T_i)K - P(t, T_{i-1}) + P(t, T_i)] = 0 \\
\Longrightarrow &K \sum_{i=\alpha+1}^{\beta} \tau_i P(t, T_i) = P(t, T_\alpha) - P(t, T_\beta) \\
\Longrightarrow &K = S_{\alpha,\beta}(t) = \frac{P(t, T_\alpha) - P(t, T_\beta)}{\sum_{i=\alpha+1}^{\beta} \tau_i P(t, T_i)}
\end{aligned}$$

Definition 2.5 (Swap rate). *The forward swap rate $S_{\alpha,\beta}(t)$ is the rate in the fixed leg of an interest rate swap that makes the contract fair at the present time, i.e. the fixed rate which equates the value of the agreement to zero.*

$$S_{\alpha,\beta}(t) = \frac{P(t, T_\alpha) - P(t, T_\beta)}{\sum_{i=\alpha+1}^{\beta} \tau_i P(t, T_i)} \tag{2}$$

As we would expect the swap rate coincides with the forward rate if there is only on time period in consideration.

Some algebraic manipulation of equation 1 yields $\frac{P(t,S)}{P(t,T)} = \frac{1}{1+F(t;T,S)\tau(T,S)}$. If we divide both the numerator and the denominator in equation 2 by $P(t, T_\alpha)$ and use the

described manipulation we obtain

$$\begin{aligned} S_{\alpha,\beta}(t) &= \frac{1-\frac{P(t,T_\beta)}{P(t,T_\alpha)}}{\sum_{i=\alpha+1}^{\beta}\tau_i\frac{P(t,T_i)}{P(t,T_\alpha)}} \\ &= \frac{1-\frac{P(t,T_\beta)}{P(t,T_\alpha)}\cdot 1}{\sum_{i=\alpha+1}^{\beta}\tau_i\frac{P(t,T_i)}{P(t,T_\alpha)}\cdot 1} \\ &= \frac{1-\frac{P(t,T_\beta)}{P(t,T_\alpha)}\prod_{j=\alpha+1}^{\beta-1}\frac{P(t,T_j)}{P(t,T_j)}}{\sum_{i=\alpha+1}^{\beta}\tau_i\frac{P(t,T_i)}{P(t,T_\alpha)}\prod_{j=\alpha+1}^{i-1}\frac{P(t,T_j)}{P(t,T_j)}} \\ &= \frac{1-\prod_{j=\alpha+1}^{\beta}\frac{P(t,T_j)}{P(t,T_{j-1})}}{\sum_{i=\alpha+1}^{\beta}\tau_i\prod_{j=\alpha+1}^{i}\frac{P(t,T_j)}{P(t,T_{j-1})}} \\ &= \frac{1-\prod_{j=\alpha+1}^{\beta}\frac{1}{1+\tau_j F_j(t)}}{\sum_{i=\alpha+1}^{\beta}\tau_i\prod_{j=\alpha+1}^{i}\frac{1}{1+\tau_j F_j(t)}} \end{aligned} \tag{3}$$

This is a practical formula because it's often convenient to calculate the swap rate in terms of forward rates. We will use this formula in the simulations in chapter 4.3 and 5.2.

A cap is a contract that can be viewed as a payer swap where each exchange payment is executed only if it has positive value. The cap discounted payoff is therefore given by

$$\sum_{i=\alpha+1}^{\beta} D(t,T_i)N\tau_i(L(T_{i-1},T_i)-K)^+$$

where the $^+$-operator has the meaning $\max(L(T_{i-1},T_i)-K,0)$. A floor can be viewed as a "positive" receiver swap with payoff $(K-L(T_{i-1},T_i))^+$. A cap contract can be used to guarantee that the borrower of an arbitrary loan tied to a floating rate won't pay higher interest rate than K. To see this suppose L exceed the fixed rate K for some time interval. Then the borrower pays the floating rate L but receives $(L-K)$. In general the borrower always pays $L-(L-K)^+ = \min(L,K)$.

A cap can be decomposed additively as a sum of caplets, $D(t,T_i)N\tau_i(L(T_{i-1},T_i)-K)^+$. A floor contract can be decomposed additively as the sum of floorlets (which are quite similar). It's common market practice to price a cap with the following sum of Black's formulas

$$N\sum_{i=\alpha+1}^{\beta} P(0,T_i)\tau_i \mathrm{Bl}(K,F(0,T_{i-1},T_i),\nu_i,1) \tag{4}$$

where, if we denote by Φ the standard Gaussian cumulative distribution function,

$$\begin{aligned}
\mathrm{Bl}(K,F,\nu,\omega) &= F\omega\Phi(\omega d_1(K,F,\nu)) - K\omega\Phi(\omega d_2(K,F,\nu)) \\
d_1(K,F,\nu) &= \frac{\ln(F/K)+\nu^2/2}{\nu} \\
d_2(K,F,\nu) &= \frac{\ln(F/K)-\nu^2/2}{\nu} \\
\nu_i &= \sigma_{\alpha,\beta}\sqrt{T_{i-1}}
\end{aligned}$$

The volatility parameter $\sigma_{\alpha,\beta}$ must be retrieved from market quotes. The Black formula is similar to the Black-Scholes formula for valuing stock options except that the spot price of the underlying is replaced by the forward price F. We will justify this formula in chapter 4.1. A floor has a similar formula, see for instance [16]. There exist a similiar formula for the floor case. Because of the similarity with a swap contract a cap is said to be at the money if $K = S_{\alpha,\beta}$, where $S_{\alpha,\beta}$ is the representative swap rate. The cap is said to be in the money if $K < S_{\alpha,\beta}$ and out of the money if $K > S_{\alpha,\beta}$. To use the same notation on a floor we simply reverse the inequalities.

A swaption is an option on a swap contract. There are two versions, payer swaptions and receiver swaptions. A European payer swaption is an option giving the right (and no obligation) to enter a payer swap at a given future time, the swaption maturity. Usually the swaption maturity coincides with the first reset date of the underlying swap. The underlying swap length, say $(T_\beta - T_\alpha)$ is called the tenor of the swaption. The payoff of a payer swaption at time $t < T_\alpha$ is equal to

$$\begin{aligned}
&ND(t,T_\alpha)\left(\sum_{i=\alpha+1}^{\beta} P(T_\alpha,T_i)\tau_i(F(T_\alpha;T_{i-1},T_i)-K)\right)^+ \\
=&ND(t,T_\alpha)\left(\sum_{i=\alpha+1}^{\beta} P(T_\alpha,T_i)\tau_i F(T_\alpha;T_{i-1},T_i) - K\sum_{i=\alpha+1}^{\beta} P(T_\alpha,T_i)\tau_i\right)^+ \\
=&ND(t,T_\alpha)\left(\sum_{i=\alpha+1}^{\beta} P(T_\alpha,T_i)\tau_i \frac{1}{\tau_i}\left(\frac{P(T_\alpha,T_{i-1})-P(T_\alpha,T_i)}{P(T_\alpha,T_i)}\right) - K\sum_{i=\alpha+1}^{\beta} P(T_\alpha,T_i)\tau_i\right)^+ \\
=&ND(t,T_\alpha)\left(\sum_{i=\alpha+1}^{\beta} (P(T_\alpha,T_{i-1})-P(T_\alpha,T_i)) - K\sum_{i=\alpha+1}^{\beta} P(T_\alpha,T_i)\tau_i\right)^+ \\
=&ND(t,T_\alpha)\left(\frac{\sum_{i=\alpha+1}^{\beta}\tau_i P(T_\alpha,T_i)}{\sum_{i=\alpha+1}^{\beta}\tau_i P(T_\alpha,T_i)}(P(T_\alpha,T_\alpha)-P(T_\alpha,T_\beta)) - K\sum_{i=\alpha+1}^{\beta} P(T_\alpha,T_i)\tau_i\right)^+ \\
=&ND(t,T_\alpha)\left(\sum_{i=\alpha+1}^{\beta}\tau_i P(T_\alpha,T_i)S_{\alpha,\beta}(T_\alpha) - K\sum_{i=\alpha+1}^{\beta} P(T_\alpha,T_i)\tau_i\right)^+ \\
=&ND(t,T_\alpha)(S_{\alpha,\beta}(T_\alpha)-K)^+\sum_{i=\alpha+1}^{\beta}\tau_i P(T_\alpha,T_i)
\end{aligned}$$

This last expression clearly illustrates the idea of a swapion in, out and at the money. We will use this expression in chapter 4.3.

Contrary to the cap case, this payoff cannot be decomposed in more elementary products. From an algebraic point of view, this is essentially due to the fact that the summation is inside the $^+$-operator. As a consequence, in order to value and manage swaption contracts, we will need to consider the joint action of the rates involved in the contract payoff. We will observe the consequences of this fact in chapter 4.3. In fact we have the inequality

$$\left(\sum_{i=\alpha+1}^{\beta} P(T_\alpha, T_i)\tau_i(F(T_\alpha; T_{i-1}, T_i) - K)\right)^+ \leq \sum_{i=\alpha+1}^{\beta} P(T_\alpha, T_i)\tau_i(F(T_\alpha; T_{i-1}, T_i) - K)^+ \tag{5}$$

with no equality in general. This means that a payer swaption in practice has a value that is always smaller than the value of a corresponding cap contract. Similar to the cap case it's common market practice to value swaptions with a Black-like formula given by

$$N\text{Bl}(K, S_{\alpha,\beta}(0), \sigma_{\alpha,\beta}\sqrt{T_\alpha}, 1) \sum_{i=\alpha+1}^{\beta} \tau_i P(0, T_i) \tag{6}$$

where $\sigma_{\alpha,\beta}$ is a volatility parameter quoted in the market that in general is different from the corresponding volatility parameter in the cap/floor case. A similar formula is used for a receiver swaption. [1] contains a rigorous treatment and derivation of both of these formulas.

2.2 Choice of Numeraire

A numeraire is the unit of account in which other assets are denominated. One usually takes the numeraire to be the currency of the country. In some applications one must change the numeraire in which one works because of financial of modeling considerations. Here we will restrict ourself to the bank account numeraire and the zero coupon bond numeraire. First we need som basic definitions. We assume we work in an economy, or trade in a financial market, that satisfies the following definition:

Definition 2.6 (Financial market). *By a market we mean a collection of $K+1$ non dividend paying securities under a filtration, $\mathcal{F}(t)$, that is traded continously from time 0 until time T. Their prices are modeled by a $K+1$ dimensional adapted semimartingale $S = \{S_t : 0 \leq t \leq T\}$ whose components $S^0, S^1, \ldots, S^K$ are positive. The asset indexed by 0 is a bank account. Its price then evolves according to $dS_t^0 = r_t S_t^0 dt$ with $S_0^0 = 1$ and r_t the instantaneous short-term rate at time t.*

Next we define precisely what we mean be a trading strategy.

Definition 2.7 (Trading strategy). *A trading strategy is a $K+1$ dimensional stochastic process $\phi = \{\phi_t : 0 \leq t \leq T\}$, whose components $\phi^0, \phi^1, \ldots \phi^K$ are locally bounded*

and predictable. The value process associated with a strategy ϕ is defined by

$$V_t(\phi) = \phi_t S_t = \sum_{k=0}^{K} \phi_t^k S_t^k, \quad 0 \leq t \leq T$$

and the gain process associated with a strategy ϕ by

$$G_t(\phi) = \int_0^t \phi_u dS_u = \sum_{k=0}^{K} \int_0^t \phi_u^k dS_u^k, \quad 0 \leq t \leq T$$

The k-th component ϕ_t^k of the strategy ϕ_t at time t, for each t, is interpreted as the number of units of security k held by an investor at time t. The predictability condition on each ϕ^k means that the value ϕ_t^k is known immediately before time t. This is a quite natural definition. If we think in terms of stock trader, he cannot decide the quantity of stocks held after the market has spoken. This also makes the gains process an Itô process. $V_t(\phi)$ and $G_t(\phi)$ are respectively interpreted as the market value of the portfolio ϕ_t and the cumulative gains realized by the investor until time t by adopting the strategy ϕ. A trading strategy is called self-financing if its value changes only due to changes in the asset prices. In other words, no additional cash inflows or outflows occur after the intial time. This motivates the following definition

Definition 2.8 (Self-financing trading strategy). *A trading strategy ϕ is self-financing if $V(\phi) \geq 0$ and*

$$V_t(\phi) = V_0(\phi) + G_t(\phi), \quad 0 \geq t < T$$

The next definition states exactly what we mean by an risk-neutral measure.

Definition 2.9 (Risk-neutral measure). *A risk-neutral measure $\tilde{\mathbb{P}}$ is a probability measure on the space $(\Omega, \mathcal{F})$ such that*

(i) $\tilde{\mathbb{P}}$ and the objective measure $\mathbb{P}$ is equivalent, that is $\tilde{\mathbb{P}}(A) = 0$ if and only if $\mathbb{P}(A) = 0$ for every $A \in \mathcal{F}$.

(ii) The Radon-Nikodym derivative $\frac{d\tilde{\mathbb{P}}}{d\mathbb{P}}$ belongs to $L^2(\Omega, \mathcal{F}, \mathbb{P})$ (the space of square Lebesgue-integrable functions, see [14] for a good reference).

(iii) The discounted asset price process $D(0,\cdot)S$ is an $(\mathcal{F}(t), \tilde{\mathbb{P}})$-martingale, that is $\tilde{\mathbb{E}}(D(0,t)S_t^k|\mathcal{F}_u) = D(0,u)S_u^k$ for all $k = 0, 1, \ldots, K$ and for all $0 \leq u \leq t \leq T$, with $\tilde{\mathbb{E}}$ denoting the expectation under $\tilde{\mathbb{P}}$.

An arbitrage opportunity is an opportunity to create something out of nothing. This should, in theory, not be possible in a modern and sufficient efficient financial market. It should not be possible to start with nothing and earn money with no risk involved. A mathematical precise definition can be stated as

Definition 2.10 (Arbitrage). *A self-financing trading strategy ϕ is called an arbitrage if $V_0(\phi) = 0$, $\mathbb{E}[V_T(\phi) \geq 0] = 1$ and $\mathbb{E}[V_T(\phi) > 0] > 0$.*

In words, there exists an arbitrage opportunity if it's possible to start with nothing, impossible to loose money, but a positive probability for a gain. A contingent claim is a contract that mandate payments that are contingent on uncertain events. The distinctions between contingent claims and derivatives are pretty weak, and they are often interchanged, but in general there are contracts that are contingent claims and not derivatives. Examples of such contracts are casualty and life insurance contracts. In this study we will only treat derivatives, but choose to work with the slightly more general contingent claims. We now define what we mean by a contingent claim.

Definition 2.11 (Contingent claims). *A contingent claim is a square Lebesgue-integrable and positive random variable on $(\Omega, \mathcal{F}, \mathbb{P})$. A contingent claim H is attainable if there exists some self-financing trading strategy ϕ such that $V_T(\phi) = H$. Such a ϕ is said to generate H and $\pi_t = V_t(\phi)$ is the price at time t associated with H.*

With attainable we essentially mean a hedge. There must be some way to replicate the payoff generated from the contingent claim. We now define what we mean by a complete market.

Definition 2.12. *A financial market is complete if and only if every contingent claim is attainable.*

In this study we assume every market to be complete. The next three theorems are of fundamental importance for all asset pricing. The proofs are rather difficult, but can be found in [5] and [6].

Theorem 2.1. *Assume there exists an risk-neutral measure $\tilde{\mathbb{P}}$ and let H be an attainable contingent claim. Then, for each time t, $0 \leq t \leq T$, there exists an unique no-arbitrage price H_t, associated with H, i.e.,*

$$H_t = \tilde{\mathbb{E}}(D(t,T)H_T|\mathcal{F}_t)$$

where H_T is the payoff function at time T.

We have worked within a finite dimensional market, but this theorem can be extended to infinite dimensional markets too. See [1] for references. We can use this theorem to find the expectation of the zero-coupon bond, $P(t,T)$, under the risk-neutral measure. Because $P(T,T) = 1$ we have $P(t,T) = \tilde{\mathbb{E}}_t[D(t,T)]$. This proves the statement in chapter 2.1 that the zero-coupon bond can be seen as the expectation of the random variable $D(t,T)$ under the risk-neutral measure.

Theorem 2.2 (First fundamental theorem of asset pricing). *If a market model has a risk-neutral probability measure, then it does not admit arbitrage.*

Theorem 2.3 (Second fundamental theorem of asset pricing). *Consider a market model that has a risk-neutral measure. The model is complete if and only if the risk-neutral measure (associated with the given numeraire) is unique.*

Now we give a precise statement of a numeraire.

Definition 2.13 (Numeraire). *A numeraire is any positive non-dividend-paying asset.*

In general, a numeraire Z is identifiable with a self-financing strategy ϕ in that $Z_t = V_t(\phi)$ for each t. Intuitively, a numeraire is a reference asset that is chosen as to normalize all other asset prices with respect to it. Choosing a numeraire Z then implies that the relative prices S^k/Z, $k = 0, 1, \ldots, K$ are considered instead of the securities prices themselves. The value of the numeraire will be often used to denote the numeraire itself. If we take the bank account as a numeraire, [5] proved that a trading strategy is self-financing if and only if the trading strategy expressed in terms of the bank account is a numeraire. This is stated in the next theorem.

Theorem 2.4. *Let ϕ be a trading strategy. Then, ϕ is self-financing if and only if $D(0,t)V_t(\phi) = V_0(\phi) + \int_0^t \phi_u d(D(0,u)S_u)$.*

This proof is extended to an arbitrary numeraire in [3]. Therefore, an attainable claim is also attainable under any numeraire. The next theorem is also proven by [3]. As we will see, its consequences are of crucial importance for the rest of this study.

Theorem 2.5. *Assume there exists a numeraire N and a probability measure $\mathbb{E}^N$, equivalent to the initial $\mathbb{E}$, such that the price of any traded asset X (without intermediate payments) relative to N is a martingale under $\mathbb{E}^N$, i.e.,*

$$\frac{X_t}{N_t} = \mathbb{E}^N \left[\frac{X_T}{N_T} | \mathcal{F}_t \right], \quad 0 \leq t \leq T$$

Let U be an arbitrary numeraire. Then there exists a probability measure $\mathbb{E}^U$, equivalent to the initial $\mathbb{E}$, such that the price for any attainable claim Y normalized by U is a martingale under $\mathbb{E}^U$, i.e.,

$$\frac{Y_t}{U_t} = \mathbb{E}^U \left[\frac{Y_T}{U_T} | \mathcal{F}_t \right], \quad 0 \leq t \leq T$$

If we assume a complete market we have a risk-neutral measure and hence theorem 2.1 holds. By taking N as the bank account numeraire (and noting that any tradet asset is also a trivial contingent claim) the assumptions of theorem 2.5 holds. We are then in position to deduce two crucial facts from theorem 2.5.

FACT ONE. *The price of any asset divided by a numeraire is a martingale under the measure associated with that numeraire.*

FACT TWO. *The unique price H_t (at time t and under the filtration $\mathcal{F}_t$) associated with a risk-neutral measure $\mathbb{E}_t^B$, that is*

$$H_t = \mathbb{E}_t^B \left[B_t \frac{H_T}{B_T} \right]$$

where H_T is the payoff at time T, is invariant by change of numeraire. If S is any other numeraire, we have

$$H_t = \mathbb{E}_t^S \left[S_t \frac{H_T}{S_T} \right]$$

The change-of-numeraire technique described in fact two is typically emplyed as follows. A payoff $H(X_T)$ is given, which depends on an underlying variable X (for example an interest rate or an stock) at time T. Pricing such a payoff amounts to compute the risk-neutral expectation $H_0 = \tilde{\mathbb{E}}_0[D(0,T)H(X_T)] = \tilde{\mathbb{E}}_0[\frac{B(0)}{B(T)}H(X_T)] = B(0)\tilde{\mathbb{E}}_0[\frac{H(X_T)}{B(T)}]$, where $B(t)$ is the bank account numeraire. Due to fact two this is equal to

$$H_0 = \tilde{\mathbb{E}}_0[D(0,T)H(X_T)] = S(0)\mathbb{E}_0^S\left[\frac{H(X_T)}{S(T)}\right]$$

for an arbitrary numeraire S.

As a numeraire we will now choose the zero-coupon bond whose maturity T coincides with that of the derivative to price. This imply $S_T = P(T,T) = 1$. This is referred to as the T-forward measure and will be denoted by $\mathbb{P}^T$. The related expectation will be denoted by $\mathbb{E}^T$. Denoting H_t the price of the derivative at time t yields $H_t = P(t,T)\mathbb{E}^T[H_T|\mathcal{F}_t]$ for $0 \leq t \leq T$. We will now show that any simply-coumponded forward rate spanning a time interval ending in T is a martingale under this measure and that the forward rate can be seen as the expectation of the future simply-compounded spot rate under this measure.

Theorem 2.6. *Any simply-compounded forward rate spanning a time interval ending in T is a martingale under the T-forward measure, i.e..*

$$\mathbb{E}^T[F(t;S,T)|\mathcal{F}_u] = F(u;S,T)$$

for each $0 \leq u \leq t \leq S < T$. In particular, the forward rate spanning the interval $[S,T]$ is the $\mathbb{P}^T$-expectation of the future simply-compunded spot rate at time S for the maturiy T, i.e,

$$\mathbb{E}^T[L(S,T)|\mathcal{F}_t] = F(t;S,T)$$

for each $0 \leq t \leq S < T$.

Proof. From the definition of a simply-compunded forward rate we have

$$\begin{aligned} F(t;S,T) &= \frac{1}{\tau(S,T)}\left[\frac{P(t,S)}{P(t,T)} - 1\right] \\ \implies F(t;S,T)P(t,T) &= \frac{1}{\tau(S,T)}[P(t,S) - P(t,T)] \end{aligned}$$

This is a multiple of the difference of two bonds, and hence a traded asset. Theorem 2.5 then states that $\frac{F(t;S,T)P(t,T)}{P(t,T)} = F(t;S,T)$ is a martingale under the T-forward measure. The relation $F(S;S,T) = L(S,T)$ and the now proven martingale property of the forward rate yields

$$\mathbb{E}^T[L(S,T)|\mathcal{F}_t] = \mathbb{E}^T[F(S;S,T)|\mathcal{F}_t] = F(t;S,T)$$

This proof the final assertion. □

2.3 Historical Development of Interest Rate Models

In 1973 Fischer Black and Myron S. Scholes published their celebrated option pricing formula that now bears their name. They worked in close cooperation with Robert C. Merton. He also published an article which included the formula and various extensions the same year. The Black-Scholes formula is therefore also known as the Black-Scholes-Merton formula. Merton and Scholes received the 1997 Nobel Prize in Economics for their work. Fischer Black sadly passed away in 1995, but was mentioned as an important contributor by the Nobel prize committee. The Black-Scholes model provided for the first time, under some restrictions, a theoretical method for fairly pricing an attainable contingent claim, more common known as an (attainable) financial derivative. By attainable we stress the existence of a self-financing dynamic trading strategy that reproduces the final payoff of the derivative, as defined in 2.1. [7] defines a financial derivative as a financial instrument whose value depends on the value of other, more basic, underlying variables. These underlying variables must be traded assets in order to satisfy the reasoning behind the Black-Scholes model. Examples of common derivatives are options, swaps and future contracts. The Black-Scholes equation solved for simple vanilla calls and puts are usually called the Black-Scholes option pricing formula. A 'fair' price in this context must coincide with the cost of succesfully hedging the derivative. A higher price would admit arbitrage because it would allow the seller to sell a derivative, hedge away all risk and earn 'free money'. In finance there is no such thing as 'free lunch'. At least not in theory. A lower price would in the long run (with probability one) ruin the seller.

A naive use of the Black-Scholes formula was the first attempt to price interest-rate derivatives. The most common approach was to calculate the Black-Scholes option price using the spot bond price as the underlying. This approach soon came under heavy criticism because of the so-called pull-to-par phenomena. In its original form the Black-Scholes formula requires a constant percentage volatility of the underlying. But since the bond price has to converge to par at maturity, its volatility can certainly not be a constant. This was not considered a big problem if the expiry of the bond option was much shorter than the maturity of the underlying bond, but could cause a serious headache if the expiry and maturity was comparable. An easy fix for the problem was to consider a non-traded quantity, like the bond yield, as the underlying. This was certainly not a theoretically justifiable use of the Black-Scholes formula, because of the non-traded asset as the underlying, but was a popular approach because the volatility of the yield was more independent of the underlying instrument than the volatility of the spot price. This approach was not surpricingly hated by the academic society and researchers in the field, and it proved to be a blind alley. A more fruitful market practice at that time was to price caplets and swaptions using the Black formula, see 4 and 5. These formulas were first proven theoretical justifiable almost a decade after they became standard market practice. None the less they showed up to be correct, and this rather sketchy market practice actually paved the way for the LIBOR Market Model

In 1977 the Czech mathematician Oldrich Vasicek introduced the first yield-curve model known as the Vasicek model. In search for a coherent and self-consistent descrip-

tion Vasicek made the sweeping assumption that the dynamics of the whole yield curve could be driven by the instantaneous short rate alone. The model specified that the instantaneous interest rate followed the following stochastic differential equation:

$$dR(t) = (\alpha - \beta R(t))dt + \sigma dW(t) \tag{7}$$

where α and β are constants to be calibrated and σ is a constant volatility factor. A good thing about this model is its mean reverting ability. That is, the interest rate won't move too far away from the mean. Unfortunately the model is also capable of producing negative interest rates. If we're not in Japan in 1998 or Switzerland in the 1960's this is not very realistic. Anyway, given the existence of a single source of uncertainty in the form of the instantaneous short rate, the stochastic movements of any bond can be perfectly hedged by simultaneously taking a suitable position in another bond of different maturity (see for instance [16]). An appropriate hedge ratio could then be found by the use of ItÃ´'s lemma. The solution of the problem could then be reduced to fit into the Black-Scholes model. The Vasicek model was not a big practical success, but its influence upon the academic society was enormous, and a lot of similar one-factor models were developed in the years to come. The model could be used both in a presciptive and a descriptive way. If applied in a fundamental way, by estimating the parameters econometrically, the model would indicate what the yield curve should look like today. The other possibility is to fit the model directly to the current yield curve in the best possible way. In either cases a believer in the model would indicate a bond traded in the market as a trading opportunity if the price of the bond differ from the bond price implied by the model. But for the option trader any such mismatch between the model and the market would be equal to force him to use the wrong price for the underlying in a Black-Scholes framework. This was definetely not a satisfactory solution. There's a little technical issue here worth mentioning. To price a derivative we often need to work under a risk-neutral measure. The volatility, or the diffusion coeffisient, will remain the same both in the real and the risk-neutral world, but the other coeffisients will in general not. Therefore we will often need to calibrate the model to the market price of a given set of instruments (which could be the zero coupon bonds implied from the yield curve) and not the yield curve directly, but the fundamental problem remains the same. The one-factor models are unable to reproduce the current yield curve implied by the market. Another well-known model from this period is the CIR model namned after its inventors; Cox, Ingersoll and Ross.

The second generation yield-curve models featured time-dependent parameters which allowed an exact fit to the observed yield curve given by the market. Great news for the option traders who finally where in position to use the correct underlying values for their option pricing formulas. Important second generation yield-curve models are the Hull and White model, the extended Vasicek model and the extended CIR model. These models were of course useless for the bond trader because such "overfitting" to the yield curve makes the model loose all its explanatory power. And even worse, the day to day refitting of the yield curve typically showed serious relative inconsistencies. Even for the option traders one big problem remained unsolved. The interest rate models could now reproduce any yield curve, but failed to match the market data on vanilla

options like swaptions and caplets. If the models were implemented as recommended by their inventors, with a constant volatility parameter, the models could only be calibrated to match one single vanilla option quoted in the market at a time. Similarly to the way bond traders used the last generation of yield-curve models to pick "cheap" bonds, vanilla option traders could now use the new models to pick "cheap" vanilla options in the market. This was a big issue for traders of more exotic interest rate options like knock-out caps, indexed-principal swaps, callable inverse floaters, index accruals and Bermudan swaptions. Exotic traders wanted their hedges (caplets and swaptions) correctly priced by the model as much as the vanilla option traders wanted their hedges (bonds and swaps) correctly priced by the model. The problem just grew bigger in the early 90's as new exotic products really began to hit the market with great pace.

The development of multifactor models like the G2++ model made some improvement but the real revolution came with the Heath-Jarrow-Morton (HJM) model in 1992. This was a quite general framework for the modelling of interest-rate dynamics. By choosing the instantaneous forward rates as fundamental quantities to model, they derived an arbitrage-free framwork for the stochastic evolution of the entire yield curve. There were no more one single volatility factor, but an entire instantaneous volatility structure for the whole forward-rates dynamics. Moreover there can be showed that every short-rate model free of arbitrage is a subset of the HJM family. The only problem with this promising model was the dependence of the instantaneous forward rates. The instantaneous forward rate is a theoretical construction not observable in the market. This fact made the model hard to implement and calibrate. The model actually impose a mathematically quite interesting problem of random dynamics in infinite dimensions.

The real breakthrough came in 1994 in the form of the LIBOR Market Model (LMM), created by Kristian Miltersen, Klaus Sandmann and Dieter Sondermann. The LIBOR Market Model was later modified to its current form in 1997 by Farshid Jamshidian based on the work of Alan Brace, Dariusz Gatarek and Marek Musiela. This is why the LIBOR Market Model also is known as the BGM or BGM/J model. The LMM can be looked upon as a discrete version of the HJM model and models the discrete forward rates directly observable in the market instead of the theoretical construction of instantaneous forward rates. This makes the model easy to calibrate to market data. Actually it's possible to carry out a joint calibration to both discount bonds, swaptions and caplets at the same time. The model is also, as the first interest model in history, compatible with Black's formula for caplets. The forward rates are assumed lognormal under its measure, and because of this the model is also called the lognormal forward-LIBOR model (LFM). There exists a different version of the model which assume the forward swap rates to be lognormal under its measure. This model is called the lognormal forward-swap model (LSM, not to be confused with the LSM algorithm) and is compatible with Black's formula for swaptions. Unfortunately it can be showed that LFM and LSM are incompatible models, see for instance [1]. In the upcoming chapters we will solely talk about the LFM and we prefer to use the name LMM.

3 Simulation and Monte Carlo Methods

3.1 An Introduction to Monte Carlo Methods

Monte Carlo methods are a class of computational algorithms that rely heavily on repeated random sampling to compute their results. In this study we will make use of Monte Carlo integration. We want to find an expectation of a discounted contingent payoff which depend on one or more assets under a certain martingale measure. To make the idea clear suppose we want to price the expectation of a certain contingent claim C under some specified dynamics. In other words we want to calculate $\mathbb{E}[C]$. We do so by generating n realizations of C, indexed by i, and average. The final result is $\mathbb{E}[C] \approx \bar{C}_n = \frac{1}{n}\sum_{i=0}^{n-1} C_i$, where n is taken as a large number. By common sense a large n will increase the accuracy of the answer, but also slow down the algorithm. To find a large, but not too large, n is of crucial importance for all practical implementations of the algorithm. To further analyze the situation we can calculate the variance of the Monte Carlo estimate

$$\mathrm{Var}[\bar{C}_n] = \mathrm{Var}\left[\frac{1}{n}\sum_{i=0}^{n-1} C_i\right] = \frac{1}{n^2}\sum_{i=0}^{n-1}\mathrm{Var}[C_i] = \frac{1}{n^2}\sum_{i=0}^{n-1}\sigma^2 = \frac{1}{n^2}\cdot n\sigma^2 = \frac{\sigma^2}{n}$$

where σ is the standard deviation of C_i. σ is in general an unknown parameter but an consistent estimate is given by

$$\sigma \approx s_C = \sqrt{\frac{1}{n-1}\sum_{i=0}^{n-1}(C_i - \bar{C}_n)}$$

Remark the $(n-1)$ quantity in the denominator which is not a misprint. For large n we can invoke the central limit theorem to obtain an asymptotic $100(1-\delta)\%$ confidence interval for $\mathbb{E}[C]$ given by

$$\bar{C}_n \pm z_{\delta/2}\frac{s_C}{\sqrt{n}}$$

where $z_{\delta/2}$ is the $1-\frac{\delta}{2}$ quantile in the standard normal distribution. The rate of convergence is thus $\mathcal{O}\left(\frac{1}{\sqrt{n}}\right)$. Hence an increase in the accuracy by a factor of 10 implies an increase in the number of repetitions by 100. For low dimensional problems this may be a serious drawback, but the methods are very competitative for large or maybe infinite dimensional problems because the rate of convergence is constant and doesn't increase with the dimension of the problem like regular methods usually do. The use of Monte Carlo methods in financial engineering were for many years considered as some kind of last resort if everything else failed, but with greatly increasing computer power, more efficient algorithms and an increasing demand for higher dimensional simulations, time has changed. The use of Monte Carlo methods is indeed essential in the modern world of quantitative finance.

3.2 Simulation of Random Variables

At the core of all simulations is the ability to create sequences of apparently random numbers. A computer is a very deterministic machine unable of creating real random number by itself. Instead it makes use of so-called pseudorandom numbers. If we really need genuine random numbers we must use some kind of physical device attached to the computer. For most practical applications this is not necessary.

A pseudorandom number generator must be sufficiently good at mimicking genuine randomness. Informally, given a true random sequence and a sequence of pseudorandom numbers we must be unable to tell the difference between the sequences. Humorously there's often stated that; "If it looks like a duck, talk like a duck and walk like duck. It's a duck." But even though we can't tell the difference it's important to be aware of the fact that all pseudorandom numbers are produced by completely deterministic algorithms.

Normally we would want to mimick a sequence of uniformly independent random numbers between 0 and 1. The uniform distribution is easy to handle and can be transformed into virtually any other thinkable distribution. General linear congruential generators are one common way of creating such pseudorandom numbers.

$$\begin{aligned} x_{i+1} &= (ax_i + c) \bmod m \\ u_{i+1} &= x_{i+1}/m \end{aligned} \tag{8}$$

where a, c and m are integer constants that determine the values generated. The generation is initialized by an initial seed x_0 between 1 and $m-1$ normally specified by the user. The operation $y \bmod m$ returns the remainder of y (an integer) after division by m. For example $3 \bmod 7 = 10 \bmod 7 = 3$ because $3 = 7 \cdot 0 + 3$ and $10 = 7 \cdot 1 + 3$. The fact that $(k \cdot m + d) \bmod m = d \bmod m = d$, k an integer, makes the generated numbers range from 0 to $m-1$. Dividing by m will then create numbers in the range $[0, 1)$. The algorithm will eventually start repeating itself. A linear congruential generator that produces all $m-1$ distinct values before repeating is said to have full period. This is a desired feature. In practice we would like the generator to generate at least tens of millions of distinct values before repeating itself. Simply choosing m to be very large does not ensure this property because of the possibility that a poor choice of the parameters a, c and m may result in short cycles amoung the generated values. Other important properties of a pseudorandom generator are the speed at which it generates numbers and what can be called the degree of randomness of the generator. In many applications speed is important, but with modern computer power it's not recommended to compromise good distributional properties for a gain in speed. Randomness is a hard property to define, but in a nutshell the generated values should be uniformly distributed and independent. Given a sequence of generated numbers we should not be able to guess any future numbers. It's also very important that the the generated numbers are approximately uniformly distributed along the whole interval and don't leave any gaps. To ensure good distributional properties it's often necessary to run a couple of statistical tests on the pseudorandom outputs. For the interested reader [10] gives a comprehensive coverage of the topic.

To sample from the standard normal distribution we can use the Box-Muller algorithm. The algorithm takes two drawings from the uniform distribution and generates two standard normal distributed variables. The algorithm is based on two properties of the bivariate normal distribution. If $Z \sim N(0, I_2)$ then $R = Z_1^2 + Z_2^2$ is exponentially distributed with mean 2 and given R the point (Z_1, Z_2) is uniformly distributed on a circle of radius $\sqrt{R}$ centered at the origin. To sample from the exponential distribution we may set $R = -\log(U_1)$, where U_1 is a uniformly (pseudo)random variable in the interval $[0, 1]$. For a proof of this see for example [12]. To generate a random point on a circle we may generate another uniform variable U_2 and multiply it by 2π. The normal variables are then given by $(Z_1, Z_2) = (\sqrt{R}\cos(2\pi U_2), \sqrt{R}\sin(2\pi U_2))$ (by a transformation from polar to Cartesian coordinates).

Algorithm 1 Box-Muller

generate U_1, $U_2 \sim$ Unif$[0, 1]$
set $R = -2\log(U_1)$
set $V = 2\pi U_2$
$Z_1 = \sqrt{R}\cos(V)$, $Z_2 = \sqrt{R}\sin(V)$
return Z_1, Z_2

To sample from a univariate normal distribution with mean μ and standard deviation σ we may use the transform $N(\mu, \sigma) = \sigma Z + \mu$. This is very basic probability theory and a proof can be found in for example [2].

We have now the necessary tools available to simulate from a multivariate normal distribution. By p independent drawings from the univariate standard normal distribution we can simulate from $\boldsymbol{Z} \sim N_p(\boldsymbol{0}, \boldsymbol{I})$, where $\boldsymbol{0}$ is a null vector and $\boldsymbol{I}$ the identity matrix. We want to simulate from a general multivariate normal distribution, that is $\boldsymbol{X} \sim N_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, where Σ is a covariance matrix. If $\boldsymbol{A}\boldsymbol{A}^T = \boldsymbol{\Sigma}$, where $\boldsymbol{A}^T$ is the transpose of $\boldsymbol{A}$, then $\boldsymbol{\mu} + \boldsymbol{A}\boldsymbol{Z} \sim N_p(\boldsymbol{\mu}, \boldsymbol{A}\boldsymbol{A}^T) = N_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. A proof of this fact can be found in [8]. Hence we can simulate from $N_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ by finding a matrix A such that $\boldsymbol{A}\boldsymbol{A}^T = \boldsymbol{\Sigma}$, simulate $\boldsymbol{Z} \sim N_p(\boldsymbol{0}, \boldsymbol{I})$ and calculate $\boldsymbol{X} = \boldsymbol{\mu} + \boldsymbol{A}\boldsymbol{Z}$. If $\boldsymbol{\Sigma}$ is a positive definite matrix, we can use the Cholesky decomposition to find such an $\boldsymbol{A}$ which is unique up to changes in sign. A positive definite matrix is a matrix which every eigenvalue is strictly greater than zero. This is equivalent to $\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} > 0$, for all nonzero $\boldsymbol{x}$. An extensive discussion of positive definite matrices and facts about the Cholesky decomposition can be found in [15]. Although it will make the implementation of the algorithm somewhat more involved, it's also possible to perform a Cholesky decomposition on a positive semidefinite matrix. A positive semidefinite matrix is a matrix where every eigenvalue is greater than or equal to zero, or equivalently $\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} \geq 0$, for all nonzero $\boldsymbol{x}$. Details on the algorithm can be found in [4]. Remark the fact that, by definition, a positive definite matrix is also a positive semidefinite matrix. Positive definite is a stronger statement than positive semidefinite. If a matrix is positive semi definite, but not positive definite, one or more of its eigenvalues must equal zero. Because the determinant of a matrix can be expressed as the product of its eigenvalues, the matrix must be singular. By a short argument we

can deduce the fact that every covariance matrix is at least positive semidefinite. The variance of the linear combination $\boldsymbol{aX}$, where $\boldsymbol{a}$ is a p-dimensional vector (not z zero-vector) and $\boldsymbol{X} \sim N_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, can according to [8] be expressed as $\boldsymbol{a}^T\boldsymbol{\Sigma}\boldsymbol{a}$. The variance of a random variable can under no circumstances become negative, so we must have $\boldsymbol{a}^T\boldsymbol{\Sigma}\boldsymbol{a} \geq 0$. Because $\boldsymbol{a}$ was an arbitrary vector and $\boldsymbol{\Sigma}$ an arbitrary covariance matrix, every covariance matrix is at least positive indefinite, and hence has a Cholesky decomposition. A "random" variable with zero variance is in fact not a random variable, but a constant. Obtaining this from a linear combination of random variables (with nonzero variance) requires one or more subsets of the random variables to be perfectly correlated and in this case we have a positive semidefinite covariance matrix. This is not very likely to happen with a real data set, but singular covariance matrices can arise from factor models in which a vector of length d is determined by $k < d$ sources of uncertainty (factors).

Algorithm 2 Simulating from a p-dimensional multivariate normal distribution

compute the Cholesky decomposition, $\boldsymbol{A}$, of the covariance matrix
generate $\boldsymbol{Z} \sim N_p(\boldsymbol{0}, \boldsymbol{I})$
return $\boldsymbol{\mu} + \boldsymbol{A}\boldsymbol{\Sigma}$

The algorithm is implemented in appendix B.3 with $\boldsymbol{\mu} = \boldsymbol{0}$.

3.3 Discretization of Stochastic Differential Equations

A stochastical differential equation, or SDE, is an equation of the form

$$dX(t) = a(t, X(t))dt + b(t, X(t))dW(t)$$

where $a(t, X(t))$ and $b(t, X(t))$ are given functions respectively called the drift and the diffusion coefficient of the equation and must be adapted stocastic processes. The problem is to find an Itô process $X(T)$, defined for $T \geq 0$ such that

$$X(T) = X(0) + \int_0^T a(u, X(u))du + \int_0^T b(u, X(u))dW(u)$$

The integrals $\int_0^T a(u, X(u))du$ and $\int_0^T b(u, X(u))dW(u)$ must exist. The latter as an Itô integral. Most SDE cannot be solved explicitly, but if there exist an unique solution (which may or may not be found explicitly) it's possible to simulate its trajectories through a discretization scheme. We will discuss the Euler and the Milstein Scheme.

The Euler scheme is the most intuitive. First we integrate the SDE between t and $t + \Delta t$ to obtain

$$X(t + \Delta t) = X(t) + \int_t^{t+\Delta t} a(u, X(u))du + \int_t^{t+\Delta t} b(u, X(u))dW(u) \tag{9}$$

We then approximate this integral equation by

$$\widetilde{X}(t + \Delta t) = \widetilde{X}(t) + a(t, \widetilde{X}(t))\Delta t + b(t, \widetilde{X}(t))(W(t + \Delta t) - W(t))$$

If we apply this formula iteratively we obtain a discretized approximation of the solution. This is called the Euler scheme.

To analyze the situation further we examine the evolution of $b(t, X(t))$. From the Itô-Doeblin formula, theorem A.5, we get

$$\begin{aligned} db(t,X(t)) &= b_t(t,X(t))dt + b_x(t,X(t))dX(t) + \frac{1}{2}b_{xx}(X(t))dX(t)dX(t) \\ &= b_t(t,X(t))dt + b_x(t,X(t))[a(t,X(t))dt + b(t,X(t))dW(t)] \\ &\quad + \frac{1}{2}b_{xx}(X(t))b^2(X(t))dt \\ &= [b_t(t,X(t)) + b_x(t,X(t))a(t,X(t)) + \frac{1}{2}b_{xx}(X(t))b^2(X(t))]dt \\ &\quad + b_x(t,X(t))b(t,X(t))dW(t) \end{aligned}$$

We then approximate $b(X(u))$, $t \leq u \leq t+h$ by the Euler scheme obtaining

$$\begin{aligned} b(X(u)) = b(t,X(t)) &+ [b_t(t,X(t)) + b_x(t,X(t))a(t,X(t)) + \frac{1}{2}b_{xx}(X(t))b^2(X(t))](u-t) \\ &+ b_x(t,X(t))b(t,X(t))(W(u)-W(t)) \end{aligned}$$

Because $W(u) - W(t)$ is $\mathcal{O}(\sqrt{u-t})$, which is of lower order than the $\mathcal{O}(h)$ obtained by the drift term, we cut off the drift term obtaining

$$b(X(u)) \approx b(t,X(t)) + b_x(t,X(t))b(t,X(t))(W(u)-W(t))$$

We are now in possession of a refined discretization of the drift term in equation 9. The approximation now reads

$$\begin{aligned} \int_t^{t+\Delta t} b(X(u))dW(u) &\approx \int_t^{t+\Delta t} (b(t,X(t)) + b_x(t,X(t))b(t,X(t))[W(u)-W(t)])\,dW(u) \\ &= b(t,X(t))[W(t+h)-W(t)] \\ &\quad + b_x(t,X(t))b(t,X(t))\left(\int_t^{t+\Delta t}[W(u)-W(t)]dW(u)\right) \end{aligned}$$

This can be simplified to

$$\begin{aligned} \int_t^{t+\Delta t} b(X(u))dW(u) &\approx b(t,X(t))[W(t+h)-W(t)] \\ &\quad + \frac{1}{2}b_x(t,X(t))b(t,X(t))([W(t+h)-W(t)]^2 - \Delta t) \end{aligned}$$

This leads to a refined approximation of 9 given by

$$\begin{aligned} \widetilde{X}(t+\Delta t) &= \widetilde{X}(t) + a(t,\widetilde{X}(t))\Delta t + b(t,\widetilde{X}(t))(W(t+\Delta t)-W(t)) \\ &\quad + \frac{1}{2}b_x(t,\widetilde{X}(t))b(t,\widetilde{X}(t))([W(t+h)-W(t)]^2 - \Delta t) \end{aligned}$$

This is called the Milstein scheme. We have increased the accuracy of the diffusion coeffisient from $\mathcal{O}(\sqrt{h})$ to $\mathcal{O}(h)$. This leads to better strong convergence properties than the Euler scheme. We also notice the fact that the Euler scheme coincides with the Milstein scheme if the diffusion coeffisient is deterministic with respect to the time, say $b(t, X(t)) = b(t)$, because this will make the last term be equal to zero. In chapter 4.3 we will exploit this fact by carry out a smart transformation of the given SDE. A rather rigorous treatment of stochastical differential equations can be found in [9] and a detailed discussion on various discretization schemes can be found in [4].

4 The LIBOR Market Model

4.1 Theory

We let $t = 0$ be the current time and consider a set $T = \{T_0, \ldots, T_M\}$ from which expiry-maturity pairs of dates (T_{i-1}, T_i) for a familiy of spanning forward rates are taken. We denote $\{\tau_0, \ldots, \tau_M\}$ the corresponding years fractions where τ_i is associated with the expiry-maturity pair (T_{i-1}, T_i) for $i \geq 1$. τ_0 is the year fraction from settlement to T_0. Time will be expressed in years. We use the notation $F_k(t) = F(t; T_{k-1}, T_k)$, $k = 1, \ldots, M$. This forward rate is "alive" up to time T_{k-1} where it coincides with the simply-compounded spot rate $L(T_{k-1}, T_k)$.

Theorem 2.6 states that $F_k(t)$ is a martingale under the the T_k-forward measure $\mathbb{P}^k$. The martingale representation theorem (theorem A.7) then states that $F_k(t)$ can be represented as an Itô integral. We assume the following driftless dynamics for F_k under $\mathbb{P}^k$

$$dF_k(t) = \boldsymbol{\sigma}_k(t) F_k(t) d\boldsymbol{Z}^k(t), \quad t \leq T_{k-1}$$

where $\boldsymbol{Z}^k(t)$ is a M-dimensional column-vector Brownian motion (under $\mathbb{P}^k$) with instantaneous correlation matrix $\boldsymbol{\rho} = (\rho)_{i,j=1,\ldots,M}$ such that

$$d\boldsymbol{Z}^k(t) d\boldsymbol{Z}^k(t)^T = \boldsymbol{\rho} dt$$

The instantaneous correlation matrix must be calibrated to market data. $\boldsymbol{\sigma}_k(t)$ is the M-vector volatility coeffisient for the forward rate $F_k(t)$. We will from now on assume that

$$\boldsymbol{\sigma}_j(t) = [0\, 0 \ldots \sigma_j(t) \ldots 0\, 0]$$

with the only non-zero entry $\sigma_j(t)$ occurring at the j-th position in the vector $\boldsymbol{\sigma}_k(t)$. This yields the following dynamics for $F_k(t)$ under $\mathbb{P}^k$

$$dF_k(t) = \sigma_k(t) F_k(t) dZ_k(t), \quad t \leq T_{k-1}$$

where $Z_k = Z_k^k$ is the k-th component of Brownian motion vector $\boldsymbol{Z}^k$ under the $\mathbb{P}^k$ forward measure. $\sigma_k(t)$ is the instantaneous volatility at time t for the forward rate F_k and must be calibrated to the market.

This representation of the forward rates is known as the LIBOR Market Model. The dynamic of $F_k(t)$ under a measure $\mathbb{P}^i$ different from $\mathbb{P}^k$ is stated in theorem 4.1. The proof is somewhat involved but can be found in [1]. We assume $t \leq \min(T_i, T_{k-1})$ because both the chosen numeraire, $P(t, T_i)$, and the modeled forward rate have to be alive at time t.

Theorem 4.1. *The dynamics of F_k under the forward measure $\mathbb{P}^i$ in the three cases*

$i < k$, $i = k$ and $i > k$ are respectively

$$i < k, \quad t \leq T_i : \quad dF_k(t) = \sigma_k(t)F_k(t) \sum_{j=i+1}^{k} \frac{\rho_{k,j}\tau_j\sigma_j(t)F_j(t)}{1+\tau_j F_j(t)}dt + \sigma_k(t)F_k(t)dZ_k(t)$$

$$i = k, \quad t \leq T_{k-1} : \quad dF_k(t) = \sigma_k(t)F_k(t)dZ_k(t)$$

$$i > k, \quad t \leq T_{k-1} : \quad dF_k(t) = -\sigma_k(t)F_k(t) \sum_{j=k+1}^{i} \frac{\rho_{k,j}\tau_j\sigma_j(t)F_j(t)}{1+\tau_j F_j(t)}dt + \sigma_k(t)F_k(t)dZ_k(t)$$

As seen in chapter 2 the discounted payoff of a cap at time zero with $N = 1$ is equal to

$$\sum_{i=\alpha+1}^{\beta} D(0,T_i)\tau_i(L(T_{i-1},T_i) - K)^+$$

By theorem 2.1 the unique no arbitrage-price associated with the cap is equal to the risk-neutral expectation at time zero, that is

$$\tilde{\mathbb{E}}\left[\sum_{i=\alpha+1}^{\beta} D(0,T_i)\tau_i(L(T_{i-1},T_i) - K)^+\right] = \sum_{i=\alpha+1}^{\beta} \tilde{\mathbb{E}}\left[D(0,T_i)\tau_i F(T_{i-1};T_{i-1},T_i) - K\right]^+$$

Where we have used the fact that $L(T_{i-1},T_i) = F(T_{i-1};T_{i-1},T_i)$. Fact two in chapter 2.2 states that this expectation is invariant under the choice of numeraire. Hence we can switch from the bank account numeraire to the zero coupon bond numeraire to obtain the more convenient pricing formula

$$\sum_{i=\alpha+1}^{\beta} P(0,T_i)\tau_i\mathbb{E}^i\left[F(T_{i-1};T_{i-1},T_i) - K\right]^+$$

The price of a cap is therefore equal to the price of an additively sum of caplets of the form $P(0,T_i)\tau_i\mathbb{E}^i[F_i(T_{i-1}) - K]^+$. This imply that correlations between the forward rates are not involved in this calculation. To calculate the expectation $\mathbb{E}^i[F_i(T_{i-1}) - K]^+$ at time 0 we can simulate $F_i(t)$ by a Monte Carlo method and calculate the average payoff. The drift of $F_i(t)$, under $\mathbb{P}^i$, is by theorem 4.1 given by $dF_i(t) = \sigma_i(t)F_i(t)dZ_i(t)$. Hence the above expectation can be calculated as a Black-Scholes price for a stock call option whose underlying is $F_i(0)$, instead of a stock, with maturity T_{i_1}, zero constant "risk-free

rate", strike K and volatility $\sigma_i(t)$. This justify the Black formula for caps as stated in equation 4 and makes the LIBOR Market Model compatible with Black's formula for caplets. If $\sigma_i(t)$ is a time dependent parameter we may set $\sigma_{\alpha,\beta} = \sqrt{\frac{1}{T_{i-1}} \int_0^{T_{i-1}} \sigma_i(t)^2 dt}$ in equation 4.

4.2 Calibration to Market Data

We choose to calibrate the model to the 1st of January 2008. The data set from DnB Nor contains five forward rates with representatively 1, 2, 3, 4 and 5 years to expiry and each with one year to maturity. The forward rates are quoted back to 1998. As of January the 1st 2008, the representative forward rates are

$$\begin{pmatrix} F(0;T_0,T_1) \\ F(0;T_1,T_2) \\ F(0;T_2,T_3) \\ F(0;T_3,T_4) \\ F(0;T_4,T_5) \end{pmatrix} = \begin{pmatrix} F_1(0) \\ F_2(0) \\ F_3(0) \\ F_4(0) \\ F_5(0) \end{pmatrix} = \begin{pmatrix} 5.56\% \\ 5.20\% \\ 5.15\% \\ 5.56\% \\ 5.55\% \end{pmatrix} \tag{10}$$

where T_i is the date January the 1st, year $2008 + i + 1$. The forward rates have a rather uncommon u-shape, but this must be seen in context with all the turbulence in the financial market at that time. The forward rate structure tells us that the market believes the rates are going to fall, and then raise again.

Caps with the same expiry and maturity are also quoted back to 1998. The cap prices, with $N = 1$, at January the 1st 2008 were quoted to be

$$\begin{pmatrix} \text{Cap}(T_0,T_1) \\ \text{Cap}(T_1,T_2) \\ \text{Cap}(T_2,T_3) \\ \text{Cap}(T_3,T_4) \\ \text{Cap}(T_4,T_5) \end{pmatrix} = \begin{pmatrix} 0.0126 \\ 0.0145 \\ 0.0147 \\ 0.0147 \\ 0.0145 \end{pmatrix}$$

in the currency NOK (Norwegian kroner).

These caps can be seen as caplets with respect to their corresponding forward rates. The caplet prices are quoted "at the money". Hence we can take the quoted forward rates as the underlying in Black's formula (equation 4). The first zero-coupon was calculated by formula 2.1, where we assumed $L(0,1) \approx 5.50\%$. The rest of the zero-coupons were calculated by iteratively use of equation 1. All these calculations were performed in Microsoft Excel. By inverting the standard Black-Scholes option pricing formula for a call, we can retrieve the desired volatility factors from the cap prices as quoted in the market. This can be done by standard root finding techniques because the option price is a monotonically increasing function of the volatility, see for instance [16]. By use of a standard option [2] calculator we retrieved the following volatility factors:

[2] http://www.oslobors.no/ob/opsjonskalkulator

$$\begin{pmatrix} \sigma_{T_0,T_1} \\ \sigma_{T_1,T_2} \\ \sigma_{T_2,T_3} \\ \sigma_{T_3,T_4} \\ \sigma_{T_4,T_5} \end{pmatrix} = \begin{pmatrix} 63.74\% \\ 84.39\% \\ 91.16\% \\ 88.96\% \\ 93.14\% \end{pmatrix} \tag{11}$$

These volatility factors are pretty large, but again they must be seen in context with all the financial turbulence.

With our usual notation, $F_k(t) = F(t; T_{k-1}, T_k)$, $k = 1. \ldots, M$, we assume the following instantaneous volatility structur for the forward rates:

Instant. Vol.	Time: $t \in (0, T_0]$	$(T_0, T_1]$	$(T_1, T_2]$	$\ldots$	$(T_{M-2}, T_{M-1}]$
Fwd Rate: $F_1(t)$	s_1	Dead	Dead	$\ldots$	Dead
$F_2(t)$	s_2	s_2	Dead	$\ldots$	Dead
$\vdots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
$F_M(t)$	s_M	s_M	s_M	$\ldots$	s_M

By setting $s_i = \sigma_{T_{i-1},T_i}$ for $i = 1, \ldots, 5$ we have successfully calibrated the LIBOR Market Model to the cap prices as quoted by the market.

It now remains to calibrate the instantaneous correlation structure to the market data. We estimate the correlation between the forward rates throughout the year 2007 by the following Gaussian approximation

$$\left[\ln\left(\frac{F_1^*(t+\Delta t)}{F_1^*(t)} \right), \ldots, \ln\left(\frac{F_5^*(t+\Delta t)}{F_5^*(t)} \right) \right] \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

where $\boldsymbol{\mu}$ is a 5-dimensional drift vector and $\boldsymbol{\Sigma}$ is the covariance (not correlation) matrix. $F_i^*(t)$ is the forward rate for the period T_{i-1}^*-T_i^* depending on t by a fixed constant. Hence starting at the year $t+1$ for $F_1^*(t)$ and generally at time $t+i$ for $F_i^*(t)$. Δt is one day. The correlation matrix $\boldsymbol{\rho}$ is then estimated by the matrix $\hat{\boldsymbol{\rho}}$ with elements $\hat{\rho}_{i,j} = \frac{\hat{V}_{i,j}}{\sqrt{\hat{V}_{i,i}}\sqrt{\hat{V}_{i,i}}}$ where

$$\begin{aligned} \hat{\mu}_i &= \frac{1}{n} \sum_{k=0}^{n-1} \ln\left(\frac{F_i^*(t_{k+1})}{F_i^*(t_k)} \right) \\ \hat{V}_{i,j} &= \frac{1}{n} \sum_{k=0}^{n-1} \left[\left(\ln\left(\frac{F_i^*(t_{k+1})}{F_i^*(t_k)} \right) - \hat{\mu}_i \right) \left(\ln\left(\frac{F_j^*(t_{k+1})}{F_j^*(t_k)} \right) - \hat{\mu}_j \right) \right] \end{aligned}$$

and n is the number of quoted forwards rates. Because of holidays and weekends, the number of quoted forward rates in one year is approximately 250. During 2007 there were exactly 250 quoted forward rates, so in this calibration $n = 250$. The result of this

historical estimation si given below.

$$\begin{pmatrix} 1.0000000 & 0.6036069 & 0.4837154 & 0.3906583 & 0.2847411 \\ 0.6036069 & 1.0000000 & 0.5462708 & 0.4847784 & 0.3399323 \\ 0.4837154 & 0.5462708 & 1.0000000 & 0.4631405 & 0.2109093 \\ 0.3906583 & 0.4847784 & 0.4631405 & 1.0000000 & 0.2191104 \\ 0.2847411 & 0.3399323 & 0.2109093 & 0.2191104 & 1.0000000 \end{pmatrix} \tag{12}$$

It seems like the quoted forward rates have a tendency to correlate more with former than future rates, but otherwise the correlation structure looks nice with forward rates close to each other having the greatest correlation.

To save computation time it's possible to simulate under a reduced correlation structure. We can assume the correlation matrix $\boldsymbol{\rho}$ to be a positive definite matrix because a correlation matrix is essentially a scaled covariance matrix, and a covariance matrix is almost always positive definite, se chapter 3.2. Then we can find a matrix $\boldsymbol{A}$, possibly by performing a Cholesky decomposition, such that $\boldsymbol{\rho} = \boldsymbol{A}\boldsymbol{A}^T$. We can try to mimick this decomposition by means of a suitable m-rank $M \times m$ matrix $\boldsymbol{B}$ such that $\boldsymbol{B}\boldsymbol{B}^T$ is an m-rank correlation matrix with typically $m << M$. We can then replace the M-dimensional random shocks $d\boldsymbol{Z}$ with the m-dimensional standard Brownian motion $d\boldsymbol{W}$ by the transformation $d\boldsymbol{Z}(t) \approx \boldsymbol{B}d\boldsymbol{W}(t)$. This imply that we move from the correlation structure $d\boldsymbol{Z}d\boldsymbol{Z}^T = \boldsymbol{\rho}dt$ to $\boldsymbol{B}d\boldsymbol{W}(\boldsymbol{B}d\boldsymbol{W})^T = \boldsymbol{B}d\boldsymbol{W}d\boldsymbol{W}^T\boldsymbol{B}^T = \boldsymbol{B}\boldsymbol{B}^T dt$ where we have used the fact that $d\boldsymbol{W}d\boldsymbol{W}^T = \boldsymbol{I}$. We set $\boldsymbol{\rho}^B = \boldsymbol{B}\boldsymbol{B}^T$. There are many techniques and algoithms available for choosing a suitable parametric form for $\boldsymbol{B}$. For a comprehensive discussion see for instance [1]. In this study we choosed to run full scale simulations because of the relatively modest original dimension of the problem, but rank-reducing techniques should be considered in higher dimensional simulations.

4.3 Pricing a Swaption

We consider the price of a payer swaption with strike K given by

$$\tilde{\mathbb{E}}\left[D(0,T_\alpha)(S_{\alpha,\beta}(T_\alpha)-K)^+ \sum_{i=\alpha+1}^{\beta} \tau_i P(T_\alpha,T_i)\right]$$

A change of numeraire to the T_α forward measure gives

$$P(0,T_\alpha)\mathbb{E}^\alpha\left[(S_{\alpha,\beta}(T_\alpha)-K)^+ \sum_{i=\alpha+1}^{\beta} \tau_i P(T_\alpha,T_i)\right]$$

This expression depends on the joint distribution of the forward rates $F_{\alpha+1}(T_\alpha), F_{\alpha+2}(T_\alpha), \dots F_\beta(T_\alpha)$, so correlation do matter in the pricing of swaptions. From theorem 4.1 we see that the dynamics of the forward rates under the $\mathbb{P}^\alpha$ measure is given by

$$dF_k(t) = \sigma_k(t)F_k(t)\sum_{j=i+1}^{k} \frac{\rho_{k,j}\tau_j\sigma_j(t)F_j(t)}{1+\tau_j F_j(t)} + \sigma_k(t)F_k(t)dZ_k(t)$$

We need to generate n realizations of each forward rate from time $t = 0$ to $t = T_\alpha$, calculate the payoff $(S_{\alpha,\beta}(T_\alpha) - K)^+ \sum_{i=\alpha+1}^{\beta} \tau_i P(T_\alpha, T_i)$ for each realization and average. We can use equation 3 to calculate $S_{\alpha,\beta}(T_\alpha)$ and equation 1 to calculate $P(T_\alpha, T_i)$.

Before we discretizise the forward rate dynamics we do a simple logarithm transformation. By Itô-Doeblin's formula, equation A.5, we obtain

$$\begin{aligned} d\ln F_k(t) &= \frac{1}{F_k(t)} dF_k(t) - \frac{1}{F_k^2(t)} dF_k(t) dF_k(t) \\ &= \frac{1}{F_k(t)} \left(\sigma_k(t) F_k(t) \sum_{j=i+1}^{k} \frac{\rho_{k,j} \tau_j \sigma_j(t) F_j(t)}{1 + \tau_j F_j(t)} + \sigma_k(t) F_k(t) dZ_k(t) \right) \\ &\quad - \frac{1}{2} \frac{1}{F_k^2(t)} \sigma_k^2(t) F_k^2(t) dt \\ &= \sigma_k(t) \sum_{j=i+1}^{k} \frac{\rho_{k,j} \tau_j \sigma_j(t) F_j(t)}{1 + \tau_j F_j(t)} - \frac{\sigma_k^2(t)}{2} dt + \sigma_k(t) dZ_k(t) \end{aligned}$$

This is a stochastic differential equation with a deterministic diffusion coeffisient. In chapter 3.3 we proved that this imply that the Euler scheme coincides with the more sophisticated Milstein scheme. The Euler discretization applied to the transformed SDE yields

$$\begin{aligned} \ln \widetilde{F}_k(t + \Delta t) = \ln \widetilde{F}_k(t) &+ \sigma_k(t) \sum_{j=\alpha+1}^{k} \frac{\rho_{k,j} \tau_j(t) \widetilde{F}_j(t)}{1 + \tau_j \widetilde{F}_j(t)} \Delta t \\ &- \frac{\sigma_k^2(t)}{2} \Delta t + \sigma_k(t) (Z_k(t + \Delta t) - Z_k(t)) \end{aligned} \tag{13}$$

By the definition of the correlation structure we have $\boldsymbol{Z}(t + \Delta t) - \boldsymbol{Z}(t) \sim \sqrt{\Delta t}\, N(\boldsymbol{0}, \boldsymbol{\rho})$. Because of the independence property of Brownian motion, these joint shocks can be taken as independent draws from the multivariate normal distribution $N(\boldsymbol{0}, \boldsymbol{\rho})$. Algorithm 2 decribes how to perform these draws.

We now consider a payer swaption with a tenor of 5 years starting one year from January the 1st 2008. The initial forward rates are given in equation 10. We take $\Delta t = \frac{1}{250}$ years (which imply 250 time steps) and use equation 13 to simulate the forward rates through one year. The volatility coeffisients and the correlation matrix are given by respectively equation 11 and 12. We summarize the whole process in algorithm 3.

To price a receiver swaption we simply change the payoff function from $(S_{\alpha,\beta}(T_\alpha) - K)^+$ to $(K - S_{\alpha,\beta}(T_\alpha))^+$. The corresponding swap rate as of January the 1st 2008 was calculated to be 5.40%. Figure 2 and 3 shows representatively a payer swaption and a receiver swaption with the just described tenor structure (expiry in one year, tenor 5 year and January the 1st 2008 as "today"). The swap rate is labeled with a red rectangle.

Figure 4 shows a payer and a receiver swaption plotted against each other. The swap rate is again labeled with a red triangle. We see that the price of the payer swaption equals that of the receiver swaption then $K = S_{T_0,T_5}$. This is not a coincidence. The

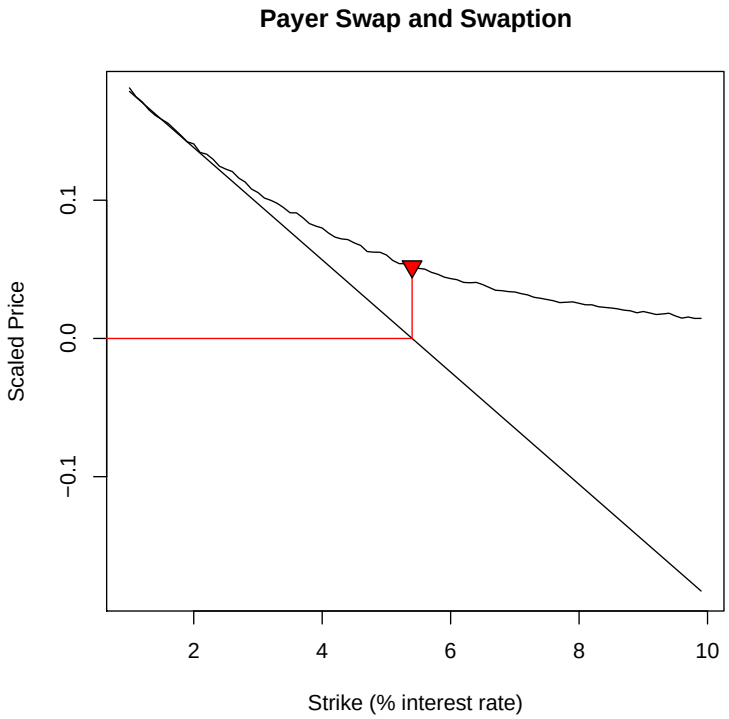

Figure 2: A plot of a payer swaption and a payer swap as a function of strike K. The swap rate is labeled by a red triangle.

Algorithm 3 Pricing a swaption

initialize the forward rates $F_{\alpha+1}(0), F_{\alpha+2}(0), \dots F_{\beta}(0)$ at time zero
initialize the volatility coeffisients and the correlation structure
for $i = 0 \dots n-1$ **do**
  for $t = 0 \dots$ number of time steps **do**
    $\boldsymbol{Z}(t+\Delta t) - \boldsymbol{Z}(t) \sim \sqrt{\Delta t}\, N(\boldsymbol{0}, \boldsymbol{\rho})$
    for $i = 0 \dots$ number of forward rates **do**
      $\ln \widetilde{F}_k(t+\Delta t) = \ln \widetilde{F}_k(t) + \sigma_k(t) \sum_{j=\alpha+1}^{k} \frac{\rho_{k,j}\tau_j(t)\widetilde{F}_j(t)}{1+\tau_j \widetilde{F}_j(t)} \Delta t$
      $-\frac{\sigma_k^2(t)}{2}\Delta t + \sigma_k(t)(Z_k(t+\Delta t) - Z_k(t))$
    end for
  end for
  evaluate $(S_{\alpha,\beta}(T_\alpha) - K)^+ \sum_{i=\alpha+1}^{\beta} \tau_i P(T_\alpha, T_i)$
end for
return $P(0,T_\alpha)\frac{1}{n}\sum_{i=1}^{n} \left[(S_{\alpha,\beta}(T_\alpha) - K)^+ \sum_{i=\alpha+1}^{\beta} \tau_i P(T_\alpha, T_i) \right]$

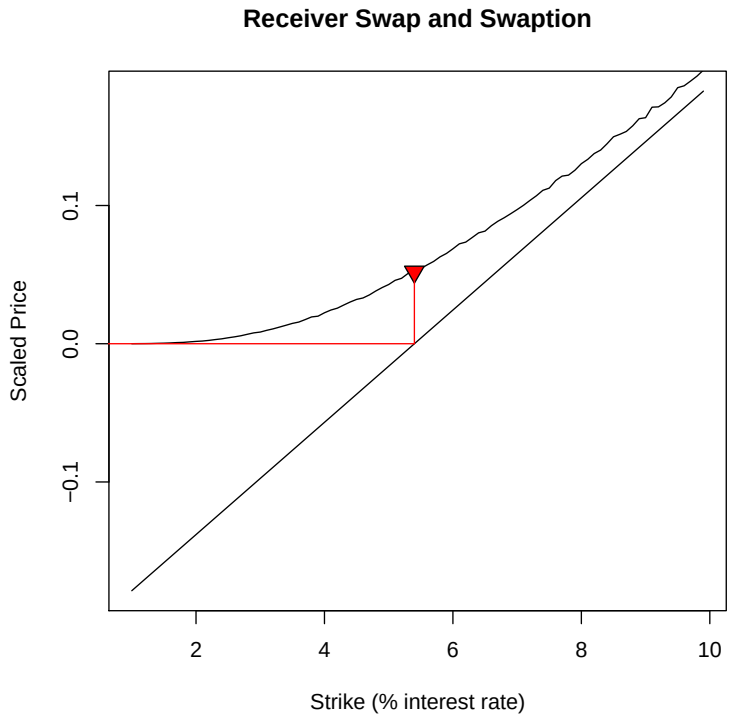

Figure 3: A plot of a receiver swaption and a receiver swap as a function of strike K. The swap rate is labeled by a red triangle.

difference between a payer swaption and the corresponding receiver swaption is easily calculated to be the price of a swap contract. Because a swap contract at the money is not worth anything, the price of a payer swaption and a receiver swaption must equal each other then they are at the money. Here they are both worth approximately 0.051 NOK (with $N = 1$ NOK). The number of repetitions in the simulations are, in both cases, taken to be $n = 10000$.

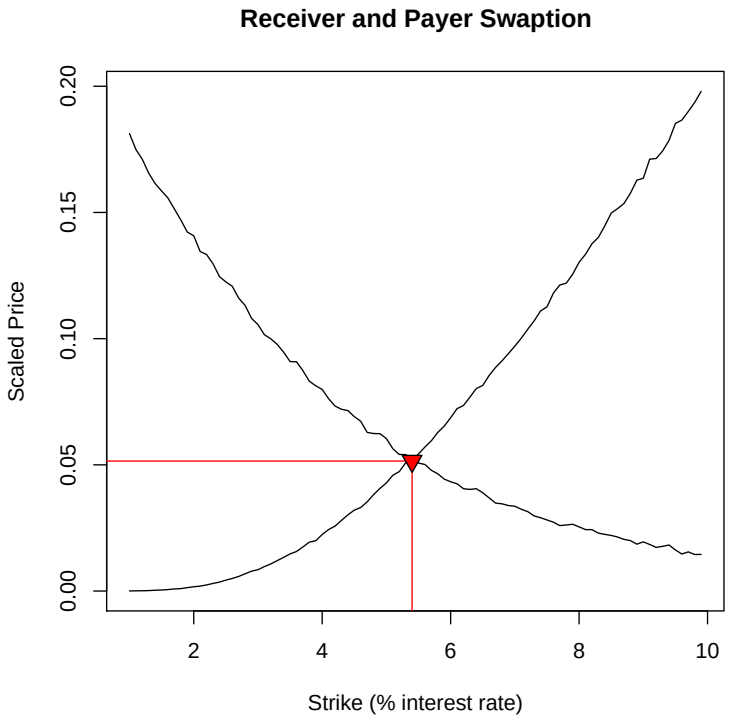

Figure 4: A plot of a payer swaption and a receiver swaption as a function of strike K. The swap rate is labeled by a red triangle.

5 The LSM Algorithm

5.1 Theory

Bermudan swaptions are options which gives the holder the right to enter a swap agreement at a sequence of reset dates, or more precise

Definition 5.1 (Bermudan payer swaption). *A Bermudan payer swaption is a swaption characterized by three dates $T_\alpha < T_h < T_\beta$, giving its holder the right to enter at any time T_l in the time interval $T_\alpha \leq T_l \leq T_h$ into a payer swaption with first reset in T_α, last payment in T_β and fixed rate K.*

Traditional numerical methods like the finite difference techniques or binomial trees, are generally unsuited to handle higher-dimensional problems like the pricing of a Bermudan swaption because their computation time grows too large as the dimension of the problem increases. Monte Carlo methods are very well suited for higher dimensional problems and path dependency, but have serious problems with early exercise features. In 2001 Francis A. Longstaff and Eduardo S. Scwartz at UCLA proposed a promising new algorithm, known as the Least Squares Monte Carlo (LSM) algorithm, for pricing early exercise products by Monte Carlo simulation. The key idea behind the algorithm is to approximate the conditional expected payoff from continuation with least squares approximation down to a set of basic functions.

As usual we assume an underlying probability space with a finite time horizon, say $t \in [0, T]$, equipped with a filtration $\mathcal{F}(t)$. We also assume the existence of a risk-neutral measure and consider square Lebesgue-integrable contingent claims. The objective of the LSM algorithm is to provide a pathwise approximation to the optimal stopping rule that maximizes the value of a derivative with early exercise features. We assume the intelligent investor will exercise as soon as the immediate exercise value is greater than or equal to the value of continuation. We assume further that the derivative must be exercised at the K discrete times $0 < t_1 \leq t_2 \leq \ldots \leq T_K$. An american option can be approximated by choosing a sufficient large K. We introduce the notation $C(s; t, T)$ to denote the path of cash flows generated by the derivative, conditional on that the derivative are not exercised at or prior to time t, and that the optimal stopping strategy are followed for all $s \in (t, T]$. The original paper, [11], expresses the value of continuation at time t_k by taking the value of the remaining discounted cash flows under the risk-neutral measure

$$\tilde{\mathbb{E}}\left[\sum_{j=k+1}^{K} D(t_k, t_j) C(t_j; t_k, T) | \mathcal{F}_{t_k}\right] \tag{14}$$

The LSM approach uses least squares to approximate the conditional expectation at $t_{K-1}, t_{K-2}, \ldots, t_1$. We work backwards since the path of cash flows generated by the derivative is defined recursively. $C(s; t_k, T)$ may differ from $C(s; t_{k+1}, T)$ since it may be optimal to stop at time t_{k+1}. We assume that the expectation in equation 14 may be represented as a linear combination of a countable set of $\mathcal{F}_{t_k}$-measurable basis functions. This assumption can be formally justified if the expectation is an element in L^2, the

space of square-integrable functions, usually with respect to the Lebsgue measure (see [14] for a good introduction to this topic). Because L^2 is a Hilbert space, and every Hilbert space has a countable orthonormal basis, we can represent the expectation as a linear combination of some orthogonal basis functions. Typical choices would be the Laguerre, Hermite or Legendre polynomials. Numerical tests indicate that even simple polynomials of the state variables give accurate results. But in higher order polynomial approximations a more sophisticated form of basis functions would be advisory because of their orthogonal properties. We only consider paths in the money. The reason for this is somewhat involved, but paths out of the money is not considered to be good predictors for future cash flows. In addition they would contribute to a bigger least square problem and slow down the algorithm.

To implement the LSM algorithm we first simulate n paths. Then we work backwards and approximate the value of continuation by regressing the discounted values of $C(s;T_k,T)$ onto the chosen set of the $M < \infty$ basic functions. In every possible stopping time we make a rule by regression to determine if a given derivative (on a given path) should be exercised or not. In the end every path is either decided exercised (possibly at maturity and only once) or not decided to be exercised. We add up all the discounted cash flows and divide by the n number of paths to calculate the expected price of the derivative.

It can be shown, see [11], that the LSM price of the derivative is always less or equal to the objective price when we let $n \to \infty$. This provides an objective criterion for convergence because we can increase the number M of basic functions until the value implied by LSM algorithm no longer increases.

5.2 Pricing a Bermudan Swaption

We will price a Bermudan swaption similar to the swaption priced in chapter 4.3, but we can enter the "swaption" at time T_0, T_1 ,T_2, T_3 and T_4. We are forced to simulate under the $\mathbb{P}^\beta = \mathbb{P}^{T_5}$ measure because we must simulate through 5 time periods (year) and the numeraire has to be alive. By a change of numeraire equation 14 now yields

$$\mathbb{E}^\beta \left[\sum_{j=k+1}^{K} \frac{P(T_k,T_\beta)}{P(T_j,T_\beta)} C(t_j;t_k,T) | \mathcal{F}_{t_k} \right]$$

The $\mathbb{P}^\beta$ measure and the Bermudan exercise feature imply that the date of the payoff and the maturity of the numeraire no longer coincides. This make the expression somewhat harder to evalute. The dynamics of the forward rates under this measure is according to equation 4.1 equal to

$$dF_k(t) = -\sigma_k(t)F_k(t) \sum_{j=k+1}^{i} \frac{\rho_{k,j}\tau_j\sigma_j(t)F_j(t)}{1+\tau_j F_j(t)} dt + \sigma_k(t)F_k(t)dZ_k(t)$$

By a Euler scheme approximation we obtain

$$\widetilde{F}_k(t+\Delta t) = \widetilde{F}_k(t) - \sigma_k(t)\widetilde{F}_k(t) \sum_{j=k+1}^{i} \frac{\rho_{k,j}\tau_j\sigma_j(t)\widetilde{F}_j(t)}{1+\tau_j\widetilde{F}_j(t)}(\Delta t) + \sigma_k(t)\widetilde{F}_k(t)dZ_k(t) \quad (15)$$

For each time period we will loose one forward rate. Table 5.2 visualizes the flow of the algorithm.

0-T_0	T_0-T_1	T_1-T_2	T_2-T_3	T_3-T_4	T_4-T_5
$F_1(t)$	Dead	Dead	Dead	Dead	Dead
$F_2(t)$	$F_2(t)$	Dead	Dead	Dead	Dead
$F_3(t)$	$F_3(t)$	$F_3(t)$	Dead	Dead	Dead
$F_4(t)$	$F_4(t)$	$F_4(t)$	$F_4(t)$	Dead	Dead
$F_5(t)$	$F_5(t)$	$F_5(t)$	$F_5(t)$	$F_5(t)$	Dead

First we simulate 5 rates, then 4 rates, and in the end there is only one rate remaining. All rates are simulated according to the discretization in equation 15. One such realization is called a path. We will create approximately $n = 100000$ such paths. When these paths are simulated, we begin to work backwards. The only rate still remaining at T_4 is $F_5(t)$. The payoff is the possible positive value of a swap contract for the last time interval. At time T_3 we discount down all the payoffs from the n paths by the ratio $\frac{P(T_3,T_5)}{P(T_4,T_5)}$. We also calculate the paths which are in the money at time T_3. For all the paths that are in the money, we calculate the swap rate. We then make a rule for optimal stopping by regressing all the representative discounted payoffs down to a second order polynomial of the swap rate, and to all the forward rates still alive. This choice of basic functions is recommended by [11] when they price a pretty similar Bermudan swaption with a 20-factor string model. The regression is performed by standard matrix algebra with the GNU Scientific Library (GSL)[3]. For all the paths that are in the money we use the calculated stopping role to decide whether we should exercise or continue. If we decide to exercise, we simply discount this value down to T_2 and repeat the procedure. If we decide to continue we will discount down the current available cashflow (which could be zero). At time T_1 we simply discount the possible n cashflows down to time zero, and average. This is the price of the Bermudan swaption. It's important to notice that we never take the approximated value of continuation as a cashflow. We only use this approximated value as our stopping role, to decide whether we should continue or exercise. If we used the approximated cashflow of continuation directly we would make an overestimation of the price of the derivative. Actually, we should ideally have used our stopping rules on a new dataset, but [11] argues that this not necessary on behalf of a lot of numerical testing.

The LSM algorithm applied to the market data as stated in chapter 4.2 is visualized in figure 5. The Bermuda exercise feature adds significant value to the contract. As seen in chapter 4.3 a payer swaption at the money is worth 0.051 NOK. In comparison

[3] `http://www.gnu.org/software/gsl/`

the corresponding Bermuda payer swaption is worth 0.092 NOK. Figure 6 shows how the price of the Bermuda swaption converges to the objective price as the number of regression coefficients increases. The blue line is obtained by using only a second degreee polynomial of swap-rates as the basic functions. The black line use both forward rates and a second degreee polynomial of swap-rates. The red line is identical to the black line but use a third degree polynomial of swap rates. This clearly don't improve the accuracy of the algorithm and is not recommended. Even higher degree polynomials can make the regression unstable and is not recommended if the polynomials don't possess some orthogonal characteristics. Hence, a sensible choice of basic functions in this situation is a second order polynomial of the swap rate and all the living forward rates. To achieve a satisfactory accuracy we used $n = 100000$ in most our simulations. The computation time was about 7 minutes on a modern computer. If this algorithm was to be implemented for commercial use it would be advisable to implement some variance reduction methods to decrease computation time without lack of accuracy. The whole implementation of the algorithm in C++ is listed in appendix B.

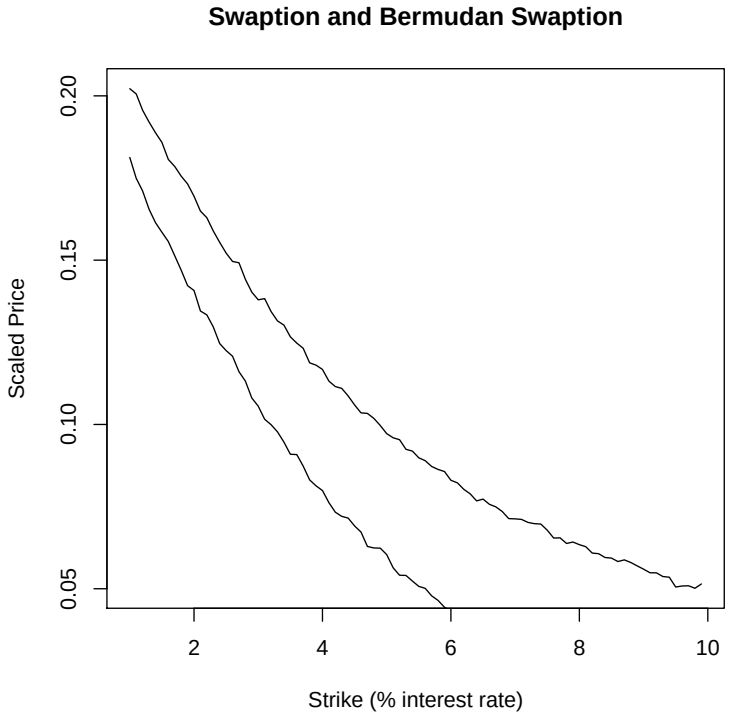

Figure 5: A swaption and a Bermuda swaption.

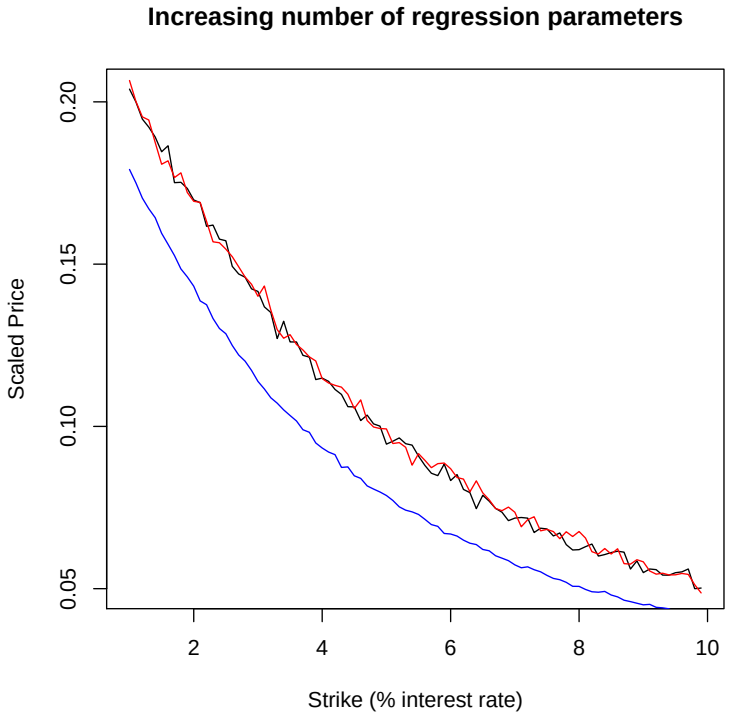

Figure 6: A Bermuda swaption with an increasing number of regression coefficients.

6 Conclusion

The LIBOR Market Model is a modern and excellent framework for pricing interest rate derivatives. The model can be calibrated to fit a huge amount of market data, is perfectly suited for Monte Carlo simulation, and is compatible with Black's formula for caplets.

Unfortunately the heavy dependence on Monte Carlo simulation has a drawback when it comes to interest rate derivatives with early exercise features, such as Bermudan swaptions. Standard Monte Carlo methods are in general not suited for pricing these kind of derivatives, but the LSM algorithm yields a very promising solution to the problem. The LSM algorithm really show off its potential when it comes to pricing contracts with higher dimensional underlying variables and early exercise features.

In the particular case of a Bermudan swaption, it's recommended to use both the still «living» forward rates and a second order degree polynomial of the swap rates as regression coeffisients in the regression procedure. Numerical tests with higher degree polynomials showed no significant effects, with the possibly exception of a slightly higher computation time. We also showed that the Bermudan exercise feature added considerable value to the contract.

If this model, and especially the LSM algorithm, was to be implemented for commercial purposes it would be highly recommendable to use variance reduction techniques to decrease computation time. For even higher dimensional simulations, a reduced-rank parametrization, as briefly discussed in chapter 4.2, should be considered.

References

[1] D. Brigo and F. Mercurio. *Interest Rate Models - Theory and Practice*. Springer, second edition, 2006.

[2] G. Casella and Berger R.L. *Statistical Inference*. Duxberry Press, second edition, 2001.

[3] H. Geman, N. El Karoui, and J.C. Rochet. Changes of numeraire, changes of probability measures and pricing of options. *Journal of Applied Probability*, 32:443–458, 1995.

[4] P. Glassermann. *Monte Carlo Methods in Financial Engineering*. Springer, first edition, 2000.

[5] J.M. Harrison and S.R. Pliska. Martingales and stochastic integrals in the theory of continous trading. *Stochastic Processes and their Applications*, 11(3):215–260, 1981.

[6] J.M. Harrison and S.R. Pliska. Martingales and stochastic integrals in the theory of continous trading: Complete markets. *Stochastic Processes and their Applications*, 15:313–316, 1983.

[7] J.F. Hull. *Options, Futures And Other Derivatives*. Pearson Prentice Hall, sixth edition, 2006.

[8] R.A. Johnson and D.W Wichern. *Applied Multivariate Statistical Analysis*. Pearson Prentice Hall, sixth edition, 2007.

[9] F.C. Klebaner. *Introduction to Stochastical Calculus with Applications*. Imperial College Press, second edition, 2005.

[10] D.E. Knuth. *The Art of Computer Programming, Volume II: Seminumerical Algorithms*. Addison-Wesley, second edition, 1998.

[11] F.A. Longstaff and Schwartz E.S. Valuing american options by simulation: A simple least-squares approach. *Review of Financial Studies*, 14(1):113–147, 2001.

[12] S.M. Ross. *Probability Models*. Academic Press, eight edition, 2003.

[13] S.E. Shreve. *Stochastic Calculus for Finance II - Continous-Time Models*. Springer, first edition, 2004.

[14] E.M. Stein and R. Shakarchi. *Real Analysis: Measure Theory, Integration, and Hilbert Spaces (Princeton Lectures in Analysis)*. Princeton University Press, first edition, 2005.

[15] G. Strang. *Linear Algebra and its Applications*. Harcourt Brace Jovanovich Inc, third edition, 1976.

[16] P. Wilmott. *Paul Wilmott on Quantitative Finance*. Wiley, second edition, 2006.

A Results from Stochastic Calculus

This appendix will review some of the most fundamental probability theory and state som useful results from stochastic calculus. A good introduction to this rather difficult subject can be found in [13] or [9].

Definition A.1 (σ-algebra). *Let $\mathcal{F}$ be a collection of subsets of a nonempty set Ω. We say that $\mathcal{F}$ is a σ-algebra if:*

(i) $\emptyset \in \mathcal{F}$

(ii) $A \in \mathcal{F} \Rightarrow A^c \in \mathcal{F}$

(iii) whenever the sequence of sets $A_1, A_2, \ldots \in \mathcal{F}$ *their union* $\cup_{n=1}^{\infty} A_n \in \mathcal{F}$

Definition A.2 (Borel σ-algebra). *The minimal σ-algebra over $\mathbb{R}$ containing all the open sets is called the Borel σ-algebra and is written $\mathfrak{B}$. We also say that $\mathfrak{B}$ is a σ-algebra generated by the open sets in $\mathbb{R}$.*

By definition $\mathfrak{B}$ contains all the closed and half-closed sets (every closed set is the complement of an open set). $\mathbb{R}$ is of course in $\mathfrak{B}$ since $\mathbb{R} = \emptyset^c$. The Borel sets are all measurable, see [14], and behaves nicely in all analysis. Every subset of the real line we encounter in this study is taken to be a Borel set.

Definition A.3 (Probability space). *Let Ω be a nonempty set, and let $\mathcal{F}$ be a σ-algebra of subsets of Ω. A probability measure $\mathbb{P}$ is a function that, to very set $A \in \mathcal{F}$, assigns a number in $[0, 1]$, called the probability of A and written $\mathbb{P}(A)$. We require:*

(i) $\mathbb{P}(\Omega) = 1$

(ii) whenever a sequence of disjoint sets $A_1, A_2, \ldots \in \mathcal{F}$*, then*

$$\mathbb{P}\left(\bigcup_{n=1}^{\infty} A_n\right) = \sum_{n=1}^{\infty} \mathbb{P}(A_n)$$

The triple $(\omega, \mathcal{F}, \mathbb{P})$ is called a probability space.

There is a rather subtle technical point in connection with events of probability measure zero. An event may be assigned probability measure zero, even though the event may occur. To see the difference between impossible and improbable events, assume a coin is tossed an unlimited number of times, and we note the result of every toss (head or tail). What is the probability that every toss results in a tail? The probability is zero, and so are actually any other predetermined sequence of heads and tails (an event with 50% probability would for example be the set of outcomes where the first toss is a tail). We say such an event is almost sure not to happen, but it's not an impossible event. There's no law of nature which states this can't happen, it's just very, very improbable. An impossible event would be any other event than a sequence of heads and tails, because

this is impossible by definition. Technically speaking, an event of probability zero is a set of outcomes with Lebesgue measure zero. Whenever an event is said to be almost sure, we mean it has probability one, even though it may not include every possible outcome. But the set of outcomes not included has probability zero. This is the motivation behind the next definition.

Definition A.4 (Almost surely). *Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space. If a set $A \in \mathcal{F}$ satisfies $\mathbb{P}(A) = 1$, we say that the event A occurs almost surely.*

Next we give a precise statement of a random variable.

Definition A.5 (Random variable). *Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space. A random variable is a real valued function X defined on Ω with the property that for every $B \in \mathfrak{B}$ the subset of Ω given by*

$$\{X \in B\} = \{\omega \in \Omega; X(\omega) \in B\}$$

is in the σ-algebra $\mathcal{F}$.

Imprecisely this definition states that given an outcome of the experiment, it should be possible to determine the value of every random variable defined on the representative probability space. The contrary that given the value of a random variable you should be able to determine the outcome of the experiment, is of course not true.

Definition A.6 (Filtration). *Let Ω be a nonempty set. Let T be a fixed positive number, and assume that for each $t \in [0, T]$ there is a σ-algebra $\mathcal{F}(t)$. Assume further that if $s \leq t$, then every set in $\mathcal{F}(s)$ is also in $\mathcal{F}(t)$. Then we say that $\mathcal{F}(t)$ has a finer resolution than $\mathcal{F}(s)$ and we call the collection of σ-algebras $\mathcal{F}(t)$, $0 \leq t \leq T$, a filtration.*

Definition A.7. *Let X be a random variable defined on a nonempty sample space Ω. The σ-algebra generated by X, denoted $\sigma(X)$, is the collection of all subsets of Ω of the form $\{X \in B\}$ where $B \in \mathfrak{B}$.*

Informally we can say $\sigma(X)$ is the smallest available σ-algebra such that X is a random variable on Ω.

Definition A.8. *Let X be a random variable defined on a nonempty sample space Ω. Let $\mathcal{G}$ be a σ-algebra on Ω. We say that X is $\mathcal{G}$ measurable if every set in $\sigma(X)$ is in $\mathcal{G}$.*

Definition A.9 (Adapted stochastic process). *Let Ω be a nonempty sample space equipped with a filtration $\mathcal{F}(t)$, $0 \leq t \leq T$. Let $X(t)$ be a collection of random variables indexed by $t \in [0, T]$. We say this collection of random variables is an adapted stochastic process if, for each t, $X(t)$ is $\mathcal{F}(t)$ measurable.*

Definition A.10 (Brownian motion). *Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space. For each $\omega \in \Omega$ assume there exists a continous function $W(t)$, $t \geq 0$, which satisfy $W(0) = 0$ and depends on ω. We say that $W(t)$ is a Brownian motion if the intervals*

$$W(t_1) - W(t_0), W(t_2) - W(t_1), \ldots, W(t_m) - W(t_{m-1})$$

are independent and normally distributed for every $0 = t_0 < t_1 < \ldots < t_m$ *with*

$$\mathbb{E}[W(t_{i+1}) - W(t_i)] = 0$$
$$\text{Var}[W(t_{i+1}) - W(t_i)] = t_{i+1} - t_i$$

Ω can be interpreted as the set of every uncountable infinite paths the Brownian motion can follow from $t = 0$ to $t = t_m$. A Brownian motion is also known as a Wiener process. Figure 7 visualize an arbitrary realized Brownian motion.

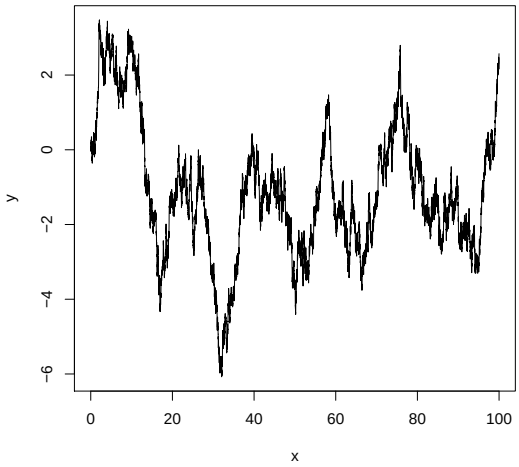

Figure 7: Brownian motion

We will now define the Itô integral and state some of its properties. Let $W(t)$ be a Brownian motion, $\mathcal{F}$ a filtration and $b(t)$ an adapted stochastic process. First we consider $b(t)$ to be a constant in every time interval $[t_j, t_{j+1})$. This is called a simple process. Then we define

$$I(t) = \int_0^t b(u)dW(u) = \sum_{j=0}^{k-1} b(t_j)[W(t_{j+1}) - W(t_j)] + b(t_k)[W(t) - W(t_k)]$$

This integral can be interpreted as the limit of a stochastic sum. $W(t)$ og $b(t)$ are dependent on the same sample space Ω and $b(t)$ can only depend on the information available at time t. The latter is of crucial importance for the existence of a self-financing trading strategy as seen in chapter 2.2. For more general integrands we use the following definition:

Definition A.11 (Itô integral). *Let $W(t)$ be a Brownian motion and $b(t)$ an adapted stochastic process satisfying $\mathbb{E}\int_0^T b^2(t)dt < \infty$. Let $b_n(t)$ be a sequence of simple processes converging to $b(t)$ in the sense $\lim_{n\to\infty}\mathbb{E}\int_0^T |b_n(t) - b(t)|^2 dt = 0$. We then define*

$$\int_0^t b(u)dW(u) = \lim_{n\to\infty}\int_0^t b_n(u)dW(u), 0 \leq t \leq T$$

This is called an Itô integral.

To summarize, we approximate $b(t)$ as a sum of infinitly many simple processes, $b_n(t)$ converging to $b(t)$.

Definition A.12 (Itô process). *Let $W(t)$, $t \geq 0$ be a Brownian motion and let $\mathcal{F}$, $t \geq 0$ be an associated filtration. An Itô process is a stochastic process of the form*

$$X(t) = X(0) + \int_0^t a(u)du + \int_0^t b(u)dW(u)$$

where $X(0)$ is a constant and $a(u)$ and $b(u)$ are adapted stochastic processes. We also assume the usual integrability assumptions $\mathbb{E}\int_0^T b^2(t)dt < \infty$ and $\int_0^t |a(u)|du < \infty$. In the future we will not always explicitly states these technical conditions, but we always assume them to hold.

Definition A.13. *Let $X(t)$, $t \geq 0$, be an Itô process and $\Gamma(t)$, $t \geq 0$, an adapted process. We define the integral with respect to an Itô process as*

$$\int_0^t \Gamma(u)dX(u) = \int_0^t \Gamma(u)a(u)du + \int_0^t \Gamma(u)b(u)dW(u)$$

Definition A.14 (Martingales). *Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space, and T a positive fixed number. Let $\mathcal{F}(t)$, $0 \leq t \leq T$, be a filtration of σ-algebras contained in $\mathcal{F}$. An adapted stochastic process $M(t)$ is a martingale if $\mathbb{E}[M(t)|\mathcal{F}(s)] = M(s)$ holds for all $0 \leq s \leq t \leq T$. A martingale has no tendency to rise or fall. If $\mathbb{E}[M(t)|\mathcal{F}(s)] \geq M(s)$ holds for all $0 \leq s \leq t \leq T$ we say the process is a submartingale. A submartingale has no tendency to fall, but may have a tendency to rise. If $\mathbb{E}[M(t)|\mathcal{F}(s)] \leq M(s)$ holds for all $0 \leq s \leq t \leq T$ we say the process is a supermartingale. A supermartingale has no tendency to rise, but may have a tendency to fall. A process that is either a submartingale or a supermartingale is usually termed semimartingale.*

Theorem A.1. *An Itô integral as defined in definition A.11 is a martingale.*

Definition A.15 (Quadratic variation). *Let $f(t)$ be a function defined for $0 \leq t \leq T$. The quadratic variation og f up to time T is defined as*

$$[f, f](T) = \lim_{\|\Pi\|\to 0}\sum_{j=0}^{n-1}[f(t_{j+1} - f(t_j)]^2$$

where $\Pi = \{t_0, t_1, \ldots, t_n\}$ and $\|\Pi\| = \max_{j=0,\ldots,n-1}(t_{j+1} - t_j)$.

Continously differentiable functions in ordinary calculus have zero quadratic variation. A Brownian motion is not continously differentiable, in fact it's almost surely not differentiable at any point, and has quadratic variation. The non-zero quadratic variation for Brownian motion is considered to be the main difference between stochastic calculus and ordinary calculus.

Theorem A.2. *Let f a continously differentiable function. Then we have $[f, f](t) = 0$. Informally we write $dtdt = 0$.*

Theorem A.3. *Let W be a Brownian motion. The quadratic variation is almost surely $[W, W](T) = T$ for every $T \geq 0$. Informally we write $dW(t)dW(t) = dt$.*

Theorem A.4. *Let X be an Itô process. We have then $[X, X](t) = \int_0^t b^2(u)du$. Informally we write $dX(t)dX(t) = b^2(t)dt$*

Theorem A.5 (Itô-Doeblin). *Let $X(t)$, $t \geq 0$, be an Itô process and $f(t, x)$ a function where the partial derivatives $f_t(t, x)$, $f_x(t, x)$ and $f_{xx}(t, x)$ are defined and continous. Then we have for every $T \geq 0$*

$$f(T, X(T)) = f(0, X(0)) + \int_0^T f_t(t, X(t))\, dt + \int_0^T f_x(t, X(t))\, dX(t) + \frac{1}{2}\int_0^T f_{xx}(t, X(t))\, d[X, X](t)$$

Informally we write

$$\begin{aligned} df(t, X(t)) &= f_t(t, X(t))dt + f_x(t, X(t))dX(t) + \frac{1}{2}f_{xx}(t, X(t))dX(t)dX(t) \\ &= f_t(t, X(t))dt + f_x(t, X(t))dX(t) + \frac{1}{2}f_{xx}(t, X(t))b^2(t)dt \end{aligned} \tag{16}$$

Theorem A.5 is called the Itô-Doeblins formula for an Itô process and can be considered as the stochastic counterpart of the chain rule from ordinary calculus. It's important to notice how the quadratic variation contributes to an extra term in the formula.

We now state the Girsanov theorem in multiple dimensions. This theorem show how to change probability measure. This is the foundation for all risk-neutral pricing formulas.

Theorem A.6 (Girsanov theorem). *Let T be a fixed positive time, and let $\Theta(t) = (\Theta_1(t), \ldots, \Theta_d(t))$ be a d-dimensional adapted process. Define*

$$Z(t) = \exp\left\{-\int_0^t \Theta(u)dW(u) - \frac{1}{2}\int_0^t ||\Theta^2(u)||du\right\}$$

$$\widetilde{W}(t) = W(t) + \int_o^t \Theta(u)du$$

and assume that

$$\mathbb{E}\int_0^T ||\Theta^2||(u)Z^2(u)du < \infty$$

Set $Z = Z(T)$. Then $\mathbb{E}Z = 1$, and under the probability measure $\widetilde{\mathbb{P}}$ given by

$$\widetilde{\mathbb{P}}(A) = \int_A Z(\omega) d\mathbb{P}(w), \quad \forall A \in \mathcal{F}$$

the process $\widetilde{W}(t)$ is a d-dimensional Brownian motion. The Itô integral is calculated as

$$\int_0^t \Theta(u) \cdot dW(u) = \int_0^t \sum_{j=1}^d \Theta_j(u) dW_j(u) = \sum_{j=1}^d \int_0^t \Theta_j(u) dW_j(u)$$

$\widetilde{W}(t) = (\widetilde{W}_1(t), \ldots, \widetilde{W}_d(t))$ share the same multidimensional interpretation. $||\Theta(u)||$ denotes the usual Euclidean norm given by

$$||\Theta(u)|| = \left(\sum_{j=1}^d \Theta_j^2(u) \right)^{\frac{1}{2}}$$

Finally we state the Martingale representation theorem in multiple dimensions.

Theorem A.7 (The martingale representation theorem). *Let T be a fixed positive time, and assume that $\mathcal{F}$, $0 \leq t \leq T$, is the filtration generated by the d-dimensional Brownian motion $W(t)$, $0 \leq t \leq T$. Let $M(t)$, $0 \leq t \leq T$, be a martingale with respect to this filtration under $\mathbb{P}$. Then there is an adapted, d-dimensional process $\Gamma(u) = (\Gamma_1(u), \ldots, \Gamma_d(u))$, $0 \leq t \leq T$, such that*

$$M(t) = M(0) + \int_0^t \Gamma(u) \cdot dW(u), \quad 0 \leq t \leq T.$$

If, in addition, we assume the notation and assumptions of the Girsanov theorem in multiple dimensions and if $\widetilde{M}(t)$, $0 \leq t \leq T$, is a $\widetilde{\mathbb{P}}$-martingale, then there is an adapted, d-dimensional process $\widetilde{\Gamma}(u) = (\widetilde{\Gamma}_1(u), \ldots, \widetilde{\Gamma}_d(u))$ such that

$$\widetilde{M}(t) = \widetilde{M}(0) + \int_0^t \widetilde{\Gamma}(u) \cdot d\widetilde{W}(u), \quad 0 \leq t \leq T.$$

The martingale representation theorem states that every martingale, with respect to the filtration induced by the representative Brownian motion, can be represented by an Itô integral plus an initial condition.

B Computer Code

This appendix contains the source code for all the implemented algorithms. Matrix calculations and linear algebra operations are carried out using the GNU Scientific Library (GSL). GSL is a well tested and freely available library for numerical calculations and can be downloaded from its homepage `http://www.gnu.org/software/gsl/`. The pseudo-random numbers are generated in according to equation 8 with $a = 25214903917$, $c = 11$ and $m = 2^{48}$. The normal random variables are calculated with the Box-Muller algorithm as described in algorithm 1. To sample from a multivariate distribution a Cholesky decomposition of the correlation matrix is performed as explained in chapter 3.2. GSL's standard Cholesky decomposition routine is used for this purpose. The sampling routine is implemented as an object, but the rest of the code is of purely procedural nature.

B.1 Main Code

```
#include <cstdlib>
#include <iostream>
#include <fstream>
#include <cmath>
#include <gsl/gsl_linalg.h>
#include <gsl/gsl_blas.h>
#include <gsl/gsl_rng.h>
#include "multinorm.h"

using namespace std;
void init(gsl_vector* forwardVec, gsl_matrix* corrMatrix,
  gsl_vector* sigmaVec, int dim);
double swaption(const double K, const gsl_vector* forwardVec,
  const gsl_matrix* corrMatrix, const gsl_vector* sigmaVec, int dim);
double bermuda(const double K, const gsl_vector* inForwardVec,
  const gsl_matrix* corrMatrix, const gsl_vector* sigmaVec, int dim);

int main(int argc, char *argv[])
{
  // Initializing with the desired dimension
  const int DIM = 5;
  gsl_vector *forwardVec = gsl_vector_alloc(DIM);
  gsl_matrix *corrMatrix = gsl_matrix_alloc(DIM,DIM);
  gsl_vector *sigmaVec = gsl_vector_alloc(DIM);

  // Importing the correlation matrix
  // and initiatlizing the forward rates and volatilities
  init(forwardVec, corrMatrix, sigmaVec, DIM);
```

```
  // Example of code to run the pricing procedures
  // swaption(0.054, forwardVec, corrMatrix, sigmaVec, DIM);
  // bermuda(0.054, forwardVec, corrMatrix, sigmaVec, DIM);

  // Deallocates memory
  gsl_vector_free(sigmaVec);
  gsl_matrix_free(corrMatrix);
  gsl_vector_free(forwardVec);

  system("PAUSE");
  return EXIT_SUCCESS;
}

void init(gsl_vector* forwardVec, gsl_matrix* corrMatrix,
  gsl_vector *sigmaVec, int dim)
{
  // Open input stream and reads the correlation matrix
  ifstream inCorr;
  inCorr.open("corr.txt");
  double corr;
  int counter = 0;
  while (!inCorr.eof()) {
    inCorr >> corr;
    gsl_matrix_set(corrMatrix, int(counter/dim), counter%dim, corr);
    counter++;
  }
  inCorr.close();

  // Manually defines the initial forward rates
  gsl_vector_set(forwardVec, 0, 0.0556);
  gsl_vector_set(forwardVec, 1, 0.0520);
  gsl_vector_set(forwardVec, 2, 0.0515);
  gsl_vector_set(forwardVec, 3, 0.0556);
  gsl_vector_set(forwardVec, 4, 0.0555);

  // Manually defines the initial volatilities
  gsl_vector_set(sigmaVec, 0, 0.64);
  gsl_vector_set(sigmaVec, 1, 0.84);
  gsl_vector_set(sigmaVec, 2, 0.91);
  gsl_vector_set(sigmaVec, 3, 0.89);
  gsl_vector_set(sigmaVec, 4, 0.93);
}
```

```
double swaption(const double K, const gsl_vector* inForwardVec,
  const gsl_matrix* corrMatrix, const gsl_vector* sigmaVec, int dim)
{
  /*
  Initializing
  Generates a new object to simulate from the
  multivariate normal distribution
  */
  Multinorm mn = Multinorm(corrMatrix, dim);
  gsl_vector *sim = gsl_vector_alloc(dim);
  gsl_vector *forwardVec = gsl_vector_alloc(dim);
  // No of simulations
  const int NRUNS = 1000;
  // No of steps per simulation
  const int STEPS = 250;
  // Time stepconst double DELTA = 1.0/STEPS;
  // P0: Zero coupon bond for the desired time period
  const double P0 = 0.9473;
  double sum = 0;

  // Looping through NRUNS MC simulations
  for (int run=0; run<NRUNS; run++) {
    // Resets the forward vector
    for (int i=0; i<dim; i++) {
      gsl_vector_set(forwardVec, i,
      gsl_vector_get(inForwardVec, i));
    }

    // Looping through the timesteps
    for (int i=0; i<STEPS; i++) {
      // Draws a new random vector
      mn.getRand(sim);

      // Simulating each rate under the Q^alpha forward measure
      for (int j=0; j<dim; j++) {
        double forward = gsl_vector_get(forwardVec, j);
        double sigma = gsl_vector_get(sigmaVec, j);
        double corr = 0;
        double temp = log(forward);
        temp -= 0.5*sigma*sigma*DELTA;
        temp += sigma*gsl_vector_get(sim, j)*sqrt(DELTA);
```

```
        double temp2 = 0;
        for (int k=0; k<=j; k++)  {
          corr = gsl_matrix_get(corrMatrix, j, k);
          sigma = gsl_vector_get(sigmaVec, k);
          forward = gsl_vector_get(forwardVec, k);
          temp2 += (corr*sigma*forward)/(1+forward);
        }
        temp2 *= DELTA*gsl_vector_get(sigmaVec, j);
        temp += temp2;
        gsl_vector_set(forwardVec, j, exp(temp));
      }
    }

    // Calculating the swap rate
    double temp, swap = 0;
    double temp2 = 1;
    for (int i=0; i<dim; i++) {
      temp2 = 1;
      for (int j=0; j<=i; j++)  {
        temp2 *= 1/(1+gsl_vector_get(forwardVec, j));
      }
      temp += temp2;
    }
    swap = (1-temp2)/temp;
    /*
    Calculating the corresponding zero coupons
    The first forward rate is "dead" and can be
    considered as the LIBOR interest rate.
    The first zero coupons must therefore be treated individually
    */
    double zero = 1/(1+gsl_vector_get(forwardVec, 0));
    double zeroSum = zero;

    for (int i=1; i<dim; i++) {
      zero = zero/(1+gsl_vector_get(forwardVec, i));
      zeroSum += zero;
    }
    double swaption = 0;
    if (swap-K > 0) swaption = (swap-K)*zeroSum;
    else swaption = 0;

    sum += swaption;
  }
```

```
  // Dellocating vectors from memory
  gsl_vector_free(sim);
  sl_vector_free(forwardVec);
  return P0*(sum/NRUNS);
}

double bermuda(const double K, const gsl_vector* inForwardVec, const
  gsl_matrix* corrMatrix, const gsl_vector* sigmaVec, const int daim) {

  // Initalize the dimension variable
  int dim = daim;
  // No of simulations
  const int NRUNS = 100000;
  // No of steps per simulation
  const int STEPS = 250;
  // Time step
  const double DELTA = 1.0/STEPS;
  /*
  No of regression variables.
  This parameter need som modification of the code to work.
  The necessary code to increase the number of regression
  variables by one are commented out in the code
  */
  const int NVAR = 3;
  // Zero coupon with maturity T_beta
  const double Tbeta = 0.729;

  // Matrix that contains all the paths
  gsl_matrix *scenarioes = gsl_matrix_alloc(daim*NRUNS, daim+1);

  // Fills in the initial forward rates
  for (int i=0; i<daim*NRUNS; i++)  {
    gsl_matrix_set(scenarioes, i, 0,
      gsl_vector_get(inForwardVec, i%daim));
  }

  // Loop that run through all the time periods
  for (dim; dim>0; dim--) {

    // Initalizing temporary correlation matrix and random vector
    // with the correct dimension
    gsl_matrix *tempCorr = gsl_matrix_alloc(dim, dim);
```

```
gsl_vector *tempRand = gsl_vector_alloc(dim);
gsl_vector *tempFor = gsl_vector_alloc(dim*NRUNS);
gsl_vector *tempSigma = gsl_vector_alloc(dim);

// Copying elements to the new reduced correlation matrix
for (int i=0; i<dim; i++) {
  for (int j=0; j<dim; j++) {
    gsl_matrix_set(tempCorr, i, j,
      gsl_matrix_get(corrMatrix, i+(daim-dim), j+(daim-dim)));
  }
}

// Copying the desired volatilites
for (int i=0; i<dim; i++) {
  gsl_vector_set(tempSigma, i,
    gsl_vector_get(sigmaVec, daim-dim+i));
}

// Initiate a new multivariate random object
// with the new desired correlation matrix
Multinorm mn = Multinorm(tempCorr, dim);

for (int run=0; run<NRUNS; run++) {
  // Copying initial forward rates to the
  // temporary forward vector
  for (int i=0; i<dim; i++) {
    double temp = gsl_matrix_get(scenarioes,
      (run*daim)+(daim-dim)+i, daim-dim);
    gsl_vector_set(tempFor, i+(run*dim), temp);
  }

  // Looping through all the discretisized time steps
  // in one time period
  for (int step=0; step<STEPS; step++)  {
    mn.getRand(tempRand);

    // Looping through all the "living" forward rates
    for (int f=0; f<dim; f++) {
      double temp = 0;
      for (int i=f+1; i<dim; i++) {
        double corr = gsl_matrix_get(tempCorr, f, i);
        double sigma = gsl_vector_get(tempSigma, i);
        double forward = gsl_vector_get(tempFor, i+run*dim);
```

```
            temp += (corr*sigma*forward) / (1+forward);
          }
          double sigma = gsl_vector_get(tempSigma, f);
          double forward = gsl_vector_get(tempFor, f+run*dim);
          double shock = gsl_vector_get(tempRand, f);
          temp *= -sigma*forward*DELTA;
          temp += sigma*forward*sqrt(DELTA)*shock;
          gsl_vector_set(tempFor, f+run*dim, forward+temp);
        }
      }
      // Updates the scenarioes matrix
      for (int i=0; i<daim; i++)  {
        if (i < daim-dim)
          gsl_matrix_set(scenarioes, (daim*run)+i, 1+(daim-dim), 0);
        else  {
          double temp = gsl_vector_get(tempFor, i-(daim-dim)+run*dim);
          gsl_matrix_set(scenarioes, (daim*run)+i, 1+(daim-dim), temp);
        }
      }
    }

    // Free memory
    gsl_vector_free(tempSigma);
    gsl_vector_free(tempFor);
    gsl_vector_free(tempRand);
    gsl_matrix_free(tempCorr);
  }

  // Calculating the final payoff
  gsl_vector *payoff = gsl_vector_alloc(NRUNS);
  for (int i=0; i<NRUNS; i++) {
    double swapr = gsl_matrix_get(scenarioes, daim*(i+1)-1, daim);
    if (swapr-K>0)  {
      double zero = 1/(1+swapr);
      gsl_vector_set(payoff,i , (swapr-K)*zero);
    }
    else gsl_vector_set(payoff, i, 0);
  }

     // Swaps contain the most recent calculated swap rates
     gsl_vector *swaps = gsl_vector_alloc(NRUNS);

  // Looping through all the remaining time periods
```

```
for (int i=0; i<daim-1; i++)  {
  int inTheMoney = 0;
  // Discounting the last payoffs and calculating new swap rates
  for (int j=0; j<NRUNS; j++) {
    double P0 = 1/(1+gsl_matrix_get(scenarioes,
      j*daim + daim-2-i, daim-1-i));
    for (int k=0; k<i+1; k++) {
      P0 = P0/(gsl_matrix_get(scenarioes,
        j*daim + daim-2-i+k, daim-1-i)+1);
    }
    double P1 = 1/(1+gsl_matrix_get(scenarioes,
      j*daim + daim-1-i, daim-i));
    for (int k=0; k<i; k++) {
      P1 = P1/(gsl_matrix_get(scenarioes,
        j*daim + daim-1-i+k, daim-i)+1);
    }
    double cash = gsl_vector_get(payoff, i)*(P0/P1);
    gsl_vector_set(payoff, i, cash);

    // Calculating the swaprate
    double swap = 0;
    double temp = 1;
    for (int k=0; k<i+2; k++) {
      temp = 1;
      for (int l=0; l<=k; l++)  {
        temp *= 1/(1+gsl_matrix_get(scenarioes,
          j*daim + daim-2-i+l, daim-1-i));
      }
      swap += temp;
    }
    swap = (1-temp)/swap;

    // Counting up the paths which are in the money
    if (swap-K>0) inTheMoney++;
    gsl_vector_set(swaps, j, swap);
  }
  // These are dummy matrices/vectors for the least square calculations
  gsl_vector *Atb = gsl_vector_alloc(NVAR+i+1);
  gsl_matrix *AtA = gsl_matrix_alloc(NVAR+i+1,NVAR+i+1);
  gsl_vector *x = gsl_vector_alloc(NVAR+i+1);

  // Doing a regression on the the paths in money
  gsl_matrix *sr = gsl_matrix_alloc(inTheMoney, NVAR+i+1);
```

```
gsl_vector *index = gsl_vector_alloc(inTheMoney);
gsl_vector *b = gsl_vector_alloc(inTheMoney);
int tempATM = 0;
for (int j=0; j<NRUNS; j++) {
  double swap = gsl_vector_get(swaps, j);
  if (swap-K>0) {
    gsl_vector_set(index, tempATM, j);
    gsl_matrix_set(sr, tempATM, 0, 1);
    gsl_matrix_set(sr, tempATM, 1, swap);
    gsl_matrix_set(sr, tempATM, 2, swap*swap);
    //gsl_matrix_set(sr, tempATM, 3, swap*swap*swap);
    gsl_vector_set(b, tempATM, gsl_vector_get(payoff, j));
    for (int k=0; k<i+1; k++) {
      gsl_matrix_set(sr, tempATM, k+NVAR,
        gsl_matrix_get(scenarioes, j*daim+daim-k-1, daim-i));
    }
    tempATM++;
  }
}

// Finds the best polynomial fit with respect to the swap rate
gsl_blas_dgemm(CblasTrans, CblasNoTrans, 1.0, sr, sr, 0.0, AtA);
gsl_blas_dgemv(CblasTrans, 1.0, sr, b, 0.0, Atb);
int surf;
gsl_permutation *P = gsl_permutation_alloc(NVAR+i+1);
gsl_linalg_LU_decomp(AtA, P, &surf);
gsl_linalg_LU_solve(AtA, P, Atb, x);

double x0 = gsl_vector_get(x, 0);
double x1 = gsl_vector_get(x, 1);
double x2 = gsl_vector_get(x, 2);
//double x3 = gsl_vector_get(x, 3);

// Exercise the option if the value is greater than
// the expected value of continuation
int continuation = 0;
for (int j=0; j<inTheMoney; j++)  {
  int status = int(gsl_vector_get(index, j));
  double swap = gsl_vector_get(swaps, status);
  double temp = swap-K;
  double zero = 1/(1+
    gsl_matrix_get(scenarioes, status*daim+daim-2-i, daim-1-i));
  double zerosum = zero;
```

```
      for (int k=1; k<i+2; k++) {
        zero = zero/(
          gsl_matrix_get(scenarioes,
            status*daim+daim-2-i+k, daim-1-i)+1);
        zerosum += zero;
      }
      temp *= zerosum;
      double estpay = x0 + swap*x1 + swap*swap*x2;
      //estpay += swap*swap*swap*x3;
      for (int k=0; k<i+1; k++) {
        estpay += gsl_vector_get(x, k+NVAR)*
          gsl_matrix_get(scenarioes, status*daim+daim-k-1, daim-i);
      }
      if (estpay < temp) gsl_vector_set(payoff, status, temp);
      else continuation++;
    }

    // Frees memory
    gsl_vector_free(x);
    gsl_vector_free(Atb);
    gsl_matrix_free(AtA);
    gsl_vector_free(b);
    gsl_vector_free(index);
    gsl_matrix_free(sr);
  }

  // Calculating the discounted sum of the payoffs
  double sum = 0;
     for (int i=0; i<NRUNS; i++)  {
      double temp = 1/(1+gsl_matrix_get(scenarioes, i*daim, 1));
      for (int j=0; j<daim-1; j++)  {
        temp = temp/(gsl_matrix_get(scenarioes, i*daim + j+1, 1)+1);
      }
      sum += (Tbeta/temp)*gsl_vector_get(payoff, i);
     }

     // Frees memory
     gsl_vector_free(swaps);
     gsl_vector_free(payoff);
     gsl_matrix_free(scenarioes);

     // Returning the averaged result
     return (sum/NRUNS);
```

```
}
```

B.2 Multinorm Header

```
#ifndef MN_H_
#define MN_H_

class Multinorm
{
private:
  int dim;
  gsl_rng * rng;
  gsl_rng * initRandGen();
  gsl_matrix * chol;
  gsl_vector * normVec;
public:
  Multinorm(const gsl_matrix * cov, int dim);
  ~Multinorm();
  void getRand(gsl_vector * sim);
};

#endif
```

B.3 Multinorm Class

```
#include <iostream>
#include <gsl/gsl_rng.h>
#include <gsl/gsl_linalg.h>
#include <gsl/gsl_randist.h>
#include <gsl/gsl_blas.h>
#include "multinorm.h"
using namespace std;

// Constructor.
// Takes a covarianse matrix and its dimension as inputs
Multinorm::Multinorm(const gsl_matrix* cov, int daim)
{
  // Initializing
  dim = daim;
  rng = initRandGen();
  chol = gsl_matrix_alloc(dim,dim);
  normVec = gsl_vector_alloc(dim);

  // Copying the covariance matrix
  for (int i=0; i<dim; i++) {
```

```
    for (int j=0; j<dim; j++) {
      gsl_matrix_set(chol, i, j, gsl_matrix_get(cov,i,j));
    }
  }

  // Cholesky decomposes the covariance matrix
  gsl_linalg_cholesky_decomp(chol);

  /*
  Sets the upper triangular of the  decomposed matrix equal to zero
  Necessary because of technical reasons due to the GSL
  implementation of the output from the Cholesky decomposition
  */
  for (int i=0; i<dim; i++) {
    for (int j=0; j<dim; j++)  {
      if (j>i) gsl_matrix_set(chol, i, j, 0);
    }
  }
}

// Destructor
// Frees memory
Multinorm::~Multinorm()
{
  gsl_rng_free(rng);
  gsl_matrix_free(chol);
  gsl_vector_free(normVec);
}

// Simulating a new random vector
void Multinorm::getRand(gsl_vector* sim)
{
  // Fills the vector with draws from the standard
  // normal distribution
  for (int i=0; i<dim; i++) {
    gsl_vector_set(normVec, i, gsl_ran_gaussian(rng, 1));
  }
  /*
  Multiplies the S.N. vector with the desired Cholesky
  decompostion of the covariance matrix to attain
  the correct correlations
  */
  gsl_blas_dgemv(CblasNoTrans, 1, chol, normVec, 0, sim);
```

```
}

gsl_rng * Multinorm::initRandGen()
{
  /*
  Initializing the random generator
  Using the number of seconds from
  1st of january 1970 as seed
  */
  time_t seconds;
  seconds = time (NULL);
  long seed = seconds;
  gsl_rng * rng;
  rng = gsl_rng_alloc (gsl_rng_rand48);
  gsl_rng_set (rng, seed);
  return rng;
}
```