An Examination and Implementation of the Libor Market Model

James Jardine
B.Sc. (Hons) Computer Science & Applied Mathematics

supervised by
Prof. Ronald Becker

Dissertation

presented to the Faculty of Science
of the University of Cape Town

in partial fulfilment of the requirements for the degree of
M.Sc. in the Mathematics of Finance

October 10, 2006

Abstract

The relatively young field of quantitative finance has grown over the past thirty years with the cherry-picking of a wide variety of techniques from the disciplines of finance, mathematics and computer science. The Libor Market Model, a model for pricing and risk-managing interest rate derivatives, is a prime example of this cherry-picking, requiring an understanding of the interest rate markets to understand the problem to be modelled, requiring some deep mathematics from probability theory and stochastic calculus to build the model, and requiring a level of computer expertise to efficiently implement the computationally demanding requirements of the model. This dissertation intends to draw from a wide literature to bring into one body of work a treatment of the Libor Market Model from start to finish.

Contents

1 Introduction

The relatively young field of quantitative finance has grown over the past thirty years with the cherry-picking a wide variety of techniques from the disciplines of finance, mathematics and computer science. The Libor Market Model, a model for pricing and risk-managing interest rate derivatives, is a prime example of this cherry-picking, requiring an understanding of the interest rate markets to understand the problem to be modelled, requiring some deep mathematics from probability theory and stochastic calculus to build the model, and requiring a level of computer expertise to efficiently implement the computationally demanding requirements of model.

This dissertation intends to draw from a wide literature to bring into one body of work a treatment of the Libor Market Model from start to finish. Careful attention has been paid to ensuring consistency in style and notation in the material sourced from different works in the literature. It is intended that the reader will, by the end of this dissertation, have understood the basics of the Libor Market Model and be able to go about implementing a reasonable version of the model for themselves without having to piece together information from other sources. That said, where one approach has been chosen over another, references are made to those alternatives in the literature.

We start in Section 2 with a brief history of the field of interest rate derivative modelling. Section 3 then presents the financial background and mathematical foundation required for the mathematical development of the Libor Market Model in Section 4. Section 5 goes into the details of the model, turning the mathematical model into something that can model the market. Section 6 covers some of the technical details required in the implementation of the model, mainly topics from computer science but also including the code for modelling some interest rate derivatives. Section 7 discusses some results from using the model. We close with Section 8, which summarises the work done and points to potential areas of further research.

2 A brief history of interest rate models

In 1979 the Fed changed their monetary policy from one where interest rates were historically static quantities to one where interest rates play a vital role in the steering of economic variables. Since then there has been a considerable increase in the volatility of interest rates inside the USA and around the world [50]. Market players have become more and more dependent on interest rate derivatives as vital sources of insurance against adverse moves in interest rates. This huge increase in demand for interest rate derivatives has inspired a lot of research into creating new and useful interest rate products and into how these derivatives should be priced.

This research has spawned a plethora of models that are used as "extrapolation tools" to determine the prices of exotic interest rate derivatives, each with its own set of assumptions and solutions. No model solves all problems: traders regularly use several similar but inconsistent models to model securities with the same underlying [17]. Indeed, [65] states that all models he has ever seen can go wrong under some circumstances: its just a question of how wrong, and whether or not it is a problem for the pricing task at hand.

The modelling of interest rate derivatives has up to now fallen into two broad categories: those modelling instantaneous spot and forward rates, rates which are themselves not visible in the market but can be used to synthesize market rates; and those modelling rates that are visible in the market, like Libor and swap rates.

2.1 Instantaneous spot and forward rate models

The first category of models contains the spot and forward rate models. They model an instantaneous interest rate (be it the instantaneous spot rate or forward rate) to generate the term structure of interest rates. An instantaneous rate is the amount of interest one earns on a risk-free investment in an infinitesimal amount of time. Because of the simplicity of the underlying, these models generally have elegant closed form solutions for the instantaneous spot rate and forward rates - rates that are mathematical constructs that do not really exist in the market. However, it becomes quite com-

plicated to price real-world instruments like caps and swaptions using these models. Another problem is that their elegantly simple structure limits their pricing potential and those few models with a rich analytical structure do not describe interest rate derivatives very well [50].

These models started emerging in the late 1970s and are still used today [55]. Although they started as a hodgepodge of different approaches using PDEs, equilibrium models and expectations, most have been resolved into the single HJM framework [24] that prescribes arbitrage free dynamics using equivalent martingale measures with the risk-free bank account as numeraire. [28] shows how all forward rate models can be written as a spot rate model.

Although a detailed treatment of the various spot and forward rate models can be found in [55] and [62], we present a brief chronology of the models that we feel were significant in leading to the development of the market models described in the next section, and which form the subject of this dissertation:

- 1977 - the Vasicek model [63] introduces the technique of valuing interest rate derivatives using partial differential equations.
- 1985 - the Cox-Ingersoll-Ross model [16] models an equilibrium model of the economy and then derives bond prices from variables in this economy.
- 1986 - the Ho-Lee model [25] models the entire yield curve rather than just the instantaneous rate.
- 1990 - the Hull-White model [27] adds reversion to a time-dependent drift. This allows for much simpler calibration of the model to the initial term structure, allowing the model to correctly price products in the market.
- 1990 - the Black-Derman-Toy model [6] introduces a discrete time model of the term structure.
- 1992 - the Longstaff-Schwartz model [39] extends the Cox-Ingersoll-Ross model to model the dynamics of the short rate and its instantaneous volatility.

- 1992 - the HJM framework [24] provides a single environment inside which the arbitrage-free dynamics of most of the previous models could be compared.
- 1997 - the market models emerge, as described in the next section.

2.2 Market models

The second category of models contains the market models, so called because they model interest rates that are visible in the market, not the non-observable mathematical rates that were the foundation of the models in the previous section. Several market models, modelling either the Libor forward rates or the market swap rates, emerged almost simultaneously in 1997: the BGM model [4], Jamshidian's swap rate model [30], the Milterstein-Sandmann-Sondermann model [45], and the model of Musiela and Rutkowski [46]. These models follow the spirit of HJM [24], where the drift conditions of the modelled rates are forced by arbitrage considerations once the numeraire and volatility structure is specified. However they model the dynamics of discretely compounded forward and swap rates that are directly visible in the market rather than instantaneous continuously compounded forward rates of the HJM framework. Although more complicated to derive, they can reproduce the market prices of common interest rate derivatives with very little effort and so have become the methods of choice for pricing complex interest rate derivatives [59]. Because of the complexity of these models, no useful analytical solutions to derivative prices are available in the literature, but rather Monte Carlo techniques are relied on for their solution.

The model implemented in this dissertation is a combination of those introduced by [45] for the single payoff derivative model presented in Section 4.2 and [4] for the general Libor Market Model presented in Section 4.3. The model presented in [4] is straightforward to derive and can be implemented very efficiently.

3 Financial and mathematical foundations

This section presents the mathematical and financial foundations that we will require when we build the Libor model in the next section. It starts with an introduction to the financial concepts that are relevant to pricing interest rate derivatives. It then presents in Section 3.2 the main theorems and results from probability theory and stochastic calculus that will be used in the pricing of simple options in Section 3.3, pricing simple options on interest rate products in Section 3.4 and then in constructing the mathematics of the Libor Market Model in Section 4. Section 3.3 shows how to price general options, which is then used in Section 3.4 to price caplets and swaptions using the Black76 formula.

3.1 Simple finance

This section introduces the various financial instruments that are relevant to the design and implementation of the Libor Market Model. The most important concepts introduced are the forward rates and caplets: the forward rates form the crux of the Libor Market Model as they are the quantity that we model, while the caplets help describe the behaviour of the forward rates.

3.1.1 Tenors and time spans

The tenors T_j are simply names for specific points in time that correspond to some interesting event in the market such as the payment or receipt of cash-flows, the expiry of an option or the reset of an interest rate. In the interest rate markets these may be spaced roughly every three, six or twelve months, depending on the market and the instruments in which we are interested. Note that although they are generally spaced fairly equally apart, they need not be and will generally be influenced by the day-count-convention for each market.

This dissertation regards tenor T_0 as today and it is sometimes written as t. The tenors are generally written in units of years, so for example, the

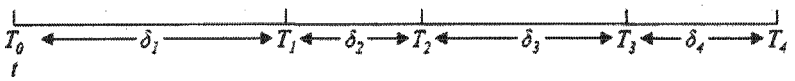

Figure 1: Tenors T_i and time spans δ_i.

first two tenors, in six months and twelve months, will be written

$$\begin{aligned} T_1 &= 0.5 \\ T_2 &= 1.0 \end{aligned}$$

The time span δ_j is simply the space of time between tenors T_{j-1} and T_j and is defined as

$$\delta_j = T_j - T_{j-1}$$

The δ_j are generally not equal since the tenors are generally not equally spaced. Figure 1 illustrates tenors and time spans.

3.1.2 Discount or zero-coupon bonds

Discount or zero-coupon bonds are instruments that are bought at less than face value and at their expiry they pay their face value. This implies that some interest is earned over the period between paying for the bond and receiving a larger payment on expiry of the bond. Generally speaking, these instruments are not traded in their own right, but rather are stripped from the more common coupon-bearing bonds and interest rate swaps using methods like those described in [22].

The discount bond $P(t, T_j)$ is the time-t value of the bond with face value 1 expiring at time T_j, i.e. $P(T_j, T_j) = 1$. Note that if positive interest is accrued in the market, $P(t, T_j) < 1$ for $t < T_j$. Also $P(t, T_j) > 0$, or we have arbitrage where zero money at time t is worth 1 at time T_j. Using arbitrage arguments, $P(t, T_j)$ is the amount by which we multiply any cashflows arising at time T_j to calculate their equivalent value today at time t. See [26] for more details on bond mathematics.

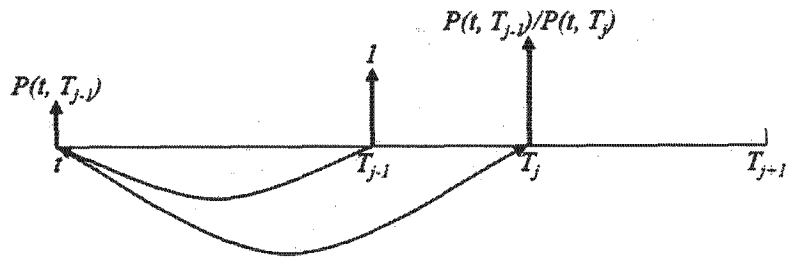

Figure 2: The forward continuous discount where a cashflow of 1 at time T_{j-1} is worth $P_j(t) = \frac{P(t,T_{j-1})}{P(t,T_j)}$ at time T_j.

Theorem 1 *In an arbitrage-free bond market we have that* $P(t, T_{j-1}) > P(t, T_j)$.

Proof. This proof is adapted from [46]. Suppose, on the contrary, that $P(t, T_{j-1}) \leq P(t, T_j)$. Then purchase one $P(t, T_{j-1})$ bond and sell one $P(t, T_j)$ bond for a positive initial cashflow. At time T_{j-1} we receive 1 unit cash, which we store (or invest for further profit) until time T_j, when we pay 1 unit cash and pocket the interest gained over the period $[T_{j-1}, T_j]$. This is an arbitrage.

Note that if we have zero interest rates then no arbitrage implies only the weaker $P(t, T_{j-1}) \geq P(t, T_j)$. ■

3.1.3 Forward continuous discount

The forward continuous discount $P_j(t) = P(t, T_{j-1}, T_j)$ is defined as

$$P_j(t) = P(t, T_{j-1}, T_j) = \frac{P(t, T_{j-1})}{P(t, T_j)}$$

It is a shorthand notation for calculating the time T_j value of cashflows at time T_{j-1}. Figure 2 shows this relationship: a cashflow of 1 at time T_{j-1} is worth $P(t, T_{j-1})$ at time t and $\frac{P(t,T_{j-1})}{P(t,T_j)}$ at time T_j. Note that $P_j(t) > 1$ is a direct consequence of Theorem 1.

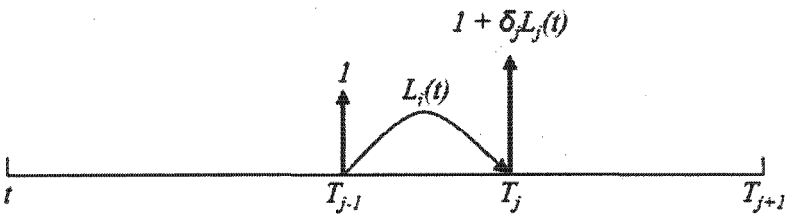

Figure 3: The forward rate $L_j(t)$ is the simple rate of return agreed at time t that an investment of 1 earns from time T_{j-1} till T_j.

3.1.4 Forward rate and Libor forward rate

Having defined our discount bonds and our forward continuous discount, we are in a position to define the rate that is the foundation of the various Libor Market Models: the forward rate. The forward rate $L_j(t)$ (or equivalently $L(t, T_{j-1}, T_j)$) is the simple rate of return agreed at time t that an investment of 1 earns from time T_{j-1} till T_j, as shown in Figure 3.

Notice that the cashflows in Figures 2 and 3 are identical and so we can write the forward rate $L_j(t) = L(t, T_{j-1}, T_j)$ in terms of discount bonds as

$$\begin{aligned} 1 + \delta_j L_j(t) &= \frac{P(t, T_{j-1})}{P(t, T_j)} \\ \implies L_j(t) &= \frac{1}{\delta_j}\left[P_j(t) - 1\right] \end{aligned} \tag{1}$$

Note that $L_j(t) > 0$ is a direct consequence of Theorem 1. So the forward rates can be calculated from the various discount bonds, and we would expect different institutions to agree on them. [59] describes how these rates are actually determined by market forces, derived not only from discount bonds, but from a variety of instruments such as FRAs, swaps and secured deposits.

So far we have talked only about forward rates. The BBA Libor (London InterBank Offer Rate) is a reference forward rate calculated daily from the interest rates that banks lend unsecured funds to other banks on the London interbank wholesale money market, although there are different Libor rates

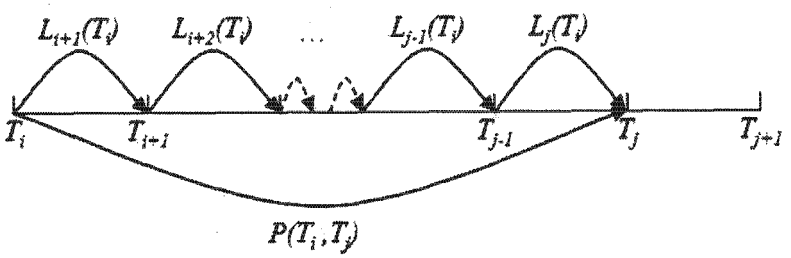

Figure 4: Synthesizing $P(T_i, T_j)$ discount bonds from the forward rates spanning from times T_i to T_j.

in many currencies and markets. Practitioners of the Libor Market Model use the Libor rates of the market to which they have access.

The next theorem gives us the ability to synthesize $P(T_i, T_j)$ discount bonds from the forward rates spanning from times T_i to T_j. This relationship is shows graphically in Figure 4.

Theorem 2 *Given a set of forward Libor rates, $L_k(T_i)$, $k = i+1 \ldots j$, we can calculate the value of $P(T_i, T_j)$ by*

$$P(T_i, T_j) = \frac{1}{\prod_{k=i+1}^{j} (1 + \delta_k L_k(T_i))} \tag{2}$$

Proof.

$$\begin{aligned} P_j(T_i) &= \frac{P(T_i, T_{j-1})}{P(T_i, T_j)} \\ \Longrightarrow P(T_i, T_j) &= \frac{P(T_i, T_{j-1})}{P_j(T_i)} = \frac{P(T_i, T_{j-2})}{P_j(T_i) P_{j-1}(T_i)} = \cdots \\ &= \frac{P(T_i, T_i)}{P_j(T_i) P_{j-1}(T_i) \cdots P_{i+1}(T_i)} \\ &= \frac{1}{\prod_{k=i+1}^{j} P_k(T_i)} \end{aligned}$$

Now from (1) we have $P_j(t) = 1 + \delta_j L_j(t)$, so

$$P(T_i, T_j) = \frac{1}{\prod_{k=i+1}^{j} (1 + \delta_k L_k(T_i))}$$

■

Note that this formula only holds for T_i falling on a Libor reset date. For an arbitrary $t < T_i$ we require the existence of an additional bond in the market $P(t, T_i)$ to calculate $P(t, T_j)$ by

$$P(t, T_j) = P(t, T_i)P(T_i, T_j)$$

Now that we have a definition for $P(t, T_j)$ and $L_j(t)$ let us briefly show that the quantity $L_j(t)P(t, T_j)$ is an asset that is traded in the market. We will use this fact in Section 4.2 to create ratios between two traded assets, $L_j(t)P(t, T_j)$ and $P(t, T_j)$.

Lemma 3 *$L_j(t)P(t, T_j)$ is a tradable asset.*

Proof. From (1) we have

$$\begin{aligned} L_j(t) &= \frac{1}{\delta_j}\left[\frac{P(t, T_{j-1})}{P(t, T_j)} - 1\right] \\ \Longrightarrow L_j(t)P(t, T_j) &= \frac{P(t, T_{j-1}) - P(t, T_j)}{\delta_j} \end{aligned}$$

Note that $L_j(t)P(t, T_j)$ can be written as a portfolio of tradable assets, and is therefore itself tradable. ■

3.1.5 Swap

Swaps are simply financial commitments to exchange one set of cashflows for another over a given period of time. They are used primarily in the risk management of future cashflows, examples of which are covered in [26]. In our case we are interested in vanilla interest rate swaps where one counterparty pays a fixed rate of interest at each period while the other counterparty

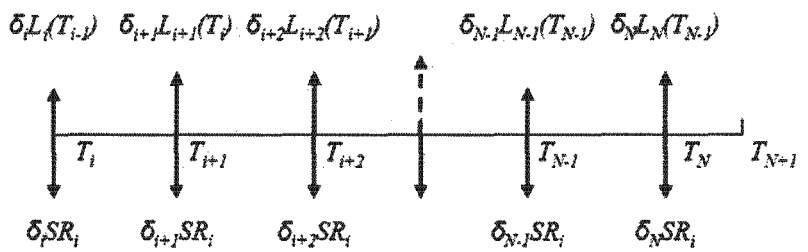

Figure 5: The cashflows of a payer swap with fixed rate SR_i from period T_i to T_N.

pays the floating Libor rate. A payer swap is one that is valued from the viewpoint of the counterparty paying the fixed rate. Figure 5 shows the cashflows of a payer swap with fixed rate SR_i from period T_i to T_N. Upward arrows represent positive cashflows, while downward ones represent negative cashflows. Notice that while the negative cashflows (downward pointing arrows) are at the constant rate SR_i the positive cashflows (upward pointing arrows) are variable, depending on the Libor rate $L_i(T_{i-1})$ set at time T_{i-1}.

We should like to calculate the swap rate SR_i that balances the positive and negative cashflows. The fixed leg of the swap from time T_i to T_N pays $SR_i\delta_j$ at time T_j for $j = i \ldots N$. The present value of all these cashflows is

$$Value_{\text{fixed}} = SR_i \sum_{j=i}^{N} \delta_j P(T_j)$$

By converting each of the floating rates of the swap into a fixed rate using a FRA (a zero cost structure that locks in future forward rates - see [26] for details), the floating leg of the swap pays $L_j\delta_j$ at time T_j for $j = i \ldots N$. The present value of these cashflows is

$$Value_{\text{floating}} = \sum_{j=i}^{N} L_j \delta_j P(T_j)$$

We set these cashflows equal, yielding

$$\begin{aligned} Value_{\text{fixed}} &= Value_{\text{floating}} \\ SR_i \sum_{j=i}^{N} \delta_j P(T_j) &= \sum_{j=i}^{N} L_j \delta_j P(T_j) \\ \implies SR_i &= \frac{\sum_{j=i}^{N} L_j \delta_j P(T_j)}{\sum_{j=i}^{N} \delta_j P(T_j)} \\ \implies SR_i &= \sum_{j=i}^{N} w_{i,j} L_j \end{aligned} \tag{3}$$

where

$$w_{i,j} = \frac{\delta_j P(T_j)}{\sum_{k=i}^{N} \delta_k P(T_k)}$$

Note that

$$\sum_{j=i}^{N} w_{i,j} = 1$$

so our swap rate is just a weighted average of our forward rates and so must always lie between the lowest and highest forward rates.

We have seen that the swap rate SR_i is chosen to make the swap initially worthless. As time passes and the underlying interest rates move with market pressure, the swap may gain or lose value. Consider if we entered into a swap at time t at fair swap rate SR_i, which initially had zero value. Lets imagine that at some later time t' (still before the first swap payment) we wished to enter into the identical swap and found that its fair swap rate is SR_i'. What is the value of our original swap? We follow the argument presented in [32]. Write the original swap rate as $SR_i = SR_i' + (SR_i - SR_i')$.

Then our original swap will be paying on the fixed leg the amount

$$\begin{aligned} SR_i \sum_{j=i}^{N} \delta_j P(T_j) &= [SR_i' + (SR_i - SR_i')] \sum_{j=i}^{N} \delta_j P(T_j) \\ &= SR_i' \sum_{j=i}^{N} \delta_j P(T_j) + (SR_i - SR_i') \sum_{j=i}^{N} \delta_j P(T_j) \end{aligned}$$

where the first term is the same as what a swap entered into today would pay, and the second term is some extra payment. Note that the floating legs will pay the same amounts. Thus if the fixed leg is paying some extra amount on top of an initially worthless swap, then the value of the swap entered into at time t must be equal to the negative of that amount, i.e.

$$\begin{aligned} V_i(t) &= -(SR_i - SR_i') \sum_{j=i}^{N} \delta_j P(T_j) \\ &= (SR_i' - SR_i) \sum_{j=i}^{N} \delta_j P(T_j) \end{aligned} \tag{4}$$

We will use Equation (4) when calibrating to swaptions in Section 5.5.

3.1.6 Coterminal swaps

The concept of coterminal swaps is important when we come to calibrate our Libor Market Model to swaptions in Section 5.5. Coterminal swaps are a collection of swaps who have different initial payment dates but identical final payment dates. Figure 6 shows a set of coterminal swaps SR_j for $j = i \ldots N$, where the first cashflow for swap SR_i is at time T_i and final cashflow is at time T_N. As you might imagine, and as will be seen in Section 5.5, given two consecutive swap rates $SR_i(t)$ and $SR_{i+1}(t)$ it should be possible to determine some of the properties of the one forward rate $L_{i-1}(t)$ in which they differ. Indeed, the entire latter part of the yield curve can be bootstrapped from a set of coterminal swaps [22].

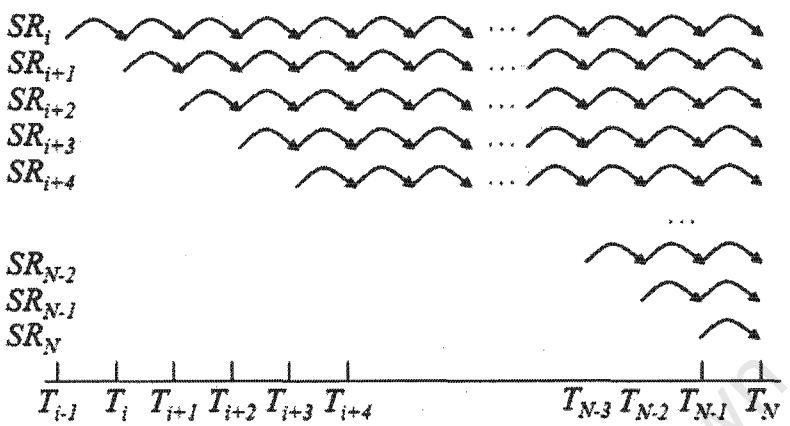

Figure 6: Coterminal swaps are a collection of swaps who have different initial payment dates but identical final payment dates.

3.1.7 Caplets and caps

A caplet is an option on a forward rate. A caplet $caplet_i$ pays $\delta_i \left(L_i(T_{i-1}) - K\right)^+$ at time T_i. It is generally used to hedge future payments of interest on borrowings: buying a caplet $caplet_i$ with strike K in the same quantity as the amount borrowed ensures that the maximum interest payable over the period from T_{i-1} to T_i is capped at the caplet strike K since the caplet would pay out the same amount as would be incurred by any interest rate higher than the caplet strike.

Single caplets are not generally traded but rather come in bundles of consecutive caplets called caps, designed, for example, to last the same amount of time as an interest bearing loan. Caplet prices (or implied volatilities) can be approximately stripped from market cap prices using the various methods described in [26] or obtained directly from the trading desk. They are generally priced using the Black76 model with the assumption that the underlying forward rates are lognormal, as is described in detail in Section 3.4.1.

3.1.8 Swaption

A swaption is an option on a swap. The holder of a European call swaption on a payer swap has the right at option expiry to enter into an underlying swap at a predetermined (fixed-leg) swap rate. Clearly the option will be exercised only if the option strike is lower than the prevailing par swap rate at option expiry. Swaptions are generally priced using the Black76 model with the assumption that the underlying swap rates are lognormal, as is described in detail in Section 3.4.2.

A set of coterminal swaptions have corresponding coterminal swaps as their underlyings. These instruments are used to calibrate the Libor Market Model to market swaptions, as will be seen in Section 5.5.

3.2 Stochastic calculus

This section presents the main theorems and results from probability theory and stochastic calculus that will be used in the remainder of the dissertation. Most theorems are stated without proof. Generally, the results from stochastic calculus can be found in [48] and [44] while the probability theory is concisely covered in [3].

Let us first introduce some definitions that are used by several of the theorems in this section. Let $W(t) = (W_1(t), ..., W_n(t))$ be an n-dimensional standard Brownian motion and $\mathcal{F}_t^{(n)}$ be the σ-algebra generated by $W(t)$. Let $X(t)$ be the n-dimensional Itô process of the form

$$dX(t) = \mu(t)dt + \sigma(t)dW_t$$

where $X(t)$, $\mu(t)$, and dW_t are n-dimensional vectors and $\sigma(t)$ is an $n \times n$ matrix.

$\mu(t)$ is the drift of the stochastic process and is

- $\mathcal{F}_t^{(n)}$-adapted
- $\mathbb{P}\left[\int_0^t |\mu(s)ds| < \infty \text{ for all } t > 0\right] = 1$

$\sigma(t)$ is the volatility of the stochastic process and determines the exposure of $X(t)$ to the underlying Brownian motion, and is (see [48] for technicalities)

- $\mathcal{F}_t^{(n)}$-adapted
- càdlàg
- $\mathbb{E}\left[\int \|\sigma\|^2(s)ds\right] < \infty$

The first three theorems cover the dynamics of a stochastic process. Itô's Theorem provides us with the chain-rule equivalent of calculus and allows us to build up the complex functions of underlying random processes that we will use to model forward rates. Theorems 5 and 6 show how the stochastic components of a stochastic process behave under the expectation operator. We use these when calculating the expected values of our models.

Theorem 4 (Itô's theorem) *Let $g(t,x) = (g_1(t,x), \ldots, g_p(t,x))$ be a $\mathcal{C}^2$ map from $[0,\infty)\times\mathbb{R}^n$ into $\mathbb{R}^p$. Then the process $Y(t,\omega) = g(t,X(t))$ is again an Itô process given by*

$$dY_k = \frac{\partial g_k}{\partial t}(t,X)dt + \sum_i \frac{\partial g_k}{\partial x_i}(t,X)dX_i + \frac{1}{2}\sum_{i,j}\frac{\partial^2 g_k}{\partial x_i \partial x_j}(t,X)dX_i dX_j, \quad k = 1\ldots p$$

The proof of this theorem can be found in [48].

Theorem 5

$$\mathbb{E}_{\mathbb{P}}\left[\int_t^T \sigma(s)dW(s) \mid \mathcal{F}_t\right] = 0$$

The proof and more rigorous technical specification of this theorem can be found in [48].

Theorem 6 (The Itô isometry)

$$\mathbb{E}_{\mathbb{P}}\left[\left(\int_t^T \sigma(s)dW(s)\right)^2 \mid \mathcal{F}_t\right] = \mathbb{E}_{\mathbb{P}}\left[\int_t^T \sigma^2(s)ds \mid \mathcal{F}_t\right]$$

The proof and more rigorous technical specification of this theorem can be found in [48].

The martingale representation theorem tells us that if we have a martingale process with mild restrictions (and in our case we are going to use the fact that various ratios of the forward rates fit this description) then that process can be written as an integral equation of Brownian motion. Once we are in the world of Brownian motion we have at our disposal the tools of stochastic calculus.

Theorem 7 (The martingale representation theorem) *Suppose M_t is an $\mathcal{F}_t^{(n)}$ martingale (w.r.t $\mathbb{P}$) and that $M_t \in \mathcal{L}^2(\mathbb{P})$ for all $t \geq 0$. Then there exists a unique previsible stochastic process $\sigma(s,\omega)$ such that*

$$M_t(\omega) = \mathbb{E}[M_0] + \int_0^t \sigma(s,\omega) dW(s) \text{ a.s., for all } t \geq 0.$$

The proof and more rigorous technical specification of this theorem can be found in [48].

A process that is a martingale under one probability measure is generally not a martingale under a different equivalent probability measure. Girsanov's theorem tells us how to change the drift of the process so that it is again a martingale under the new equivalent probability measure. We will use this to turn all our forward rate processes into martingales under a single "terminal" probability measure.

Theorem 8 (Girsanov's theorem) *Let $W^{\mathbb{P}}$ be a standard d-dimensional Brownian motion on $(\Omega, \mathcal{F}, \mathbb{P}, \{\mathcal{F}\}_t)$ and let φ, the Girsanov kernel, be any d-dimensional adapted row vector process. Define the process L on $[0,T]$ by*

$$\begin{aligned} dL_t &= \varphi_t L_t dW_t^{\mathbb{P}} \qquad (5) \\ L_0 &= 1 \\ \Longrightarrow L_t &= \exp\left(\int_0^t \varphi_s dW_s^{\mathbb{P}} - \frac{1}{2}\int_0^t \|\varphi_s\|^2 ds\right) \end{aligned}$$

Assume that

$$\mathbb{E}^{\mathbb{P}}[L_t] = 1$$

and define the probability measure $\mathbb{Q}$ *on* $\mathcal{F}_T$ *equivalent to* $\mathbb{P}$ *by*

$$\begin{aligned} L_T &= \frac{d\mathbb{Q}}{d\mathbb{P}} \\ \implies d\mathbb{Q} &= L_T d\mathbb{P} \\ \implies \mathbb{Q}(A) &= \int_A L_T(\omega) d\mathbb{P}(\omega), \quad A \in \mathcal{F}_T \end{aligned}$$

Then,

$$dW_t^{\mathbb{P}} = \varphi_t dt + dW_t^{\mathbb{Q}}$$

where $W^{\mathbb{Q}}$ *is a* $\mathbb{Q}$*-Brownian motion. That is,*

$$W_t^{\mathbb{Q}} = W_t^{\mathbb{P}} - \int_0^t \varphi_s ds$$

is a $\mathbb{Q}$*-Brownian motion. The proof and more rigorous technical specification of this theorem can be found in [3].*

By putting an integrability restriction on the Girsanov kernel, the Novikov condition ensures that things don't blow up when performing the Girsanov transform. We will see in Section 4.3 that the kernel we use in the Libor Market Model is a combination of our forward rates and the discount bonds, a combination that satisfies the Novikov condition.

Lemma 9 *If the Girsanov kernel* φ *is such that*

$$\mathbb{E}^{\mathbb{P}}\left[\exp\left(\frac{1}{2}\int_0^T \|\varphi_t\|^2 dt\right)\right] < \infty$$

then L_t *defined by (5) is a martingale and* $\mathbb{E}^{\mathbb{P}}[L_t] = 1$. *The proof of this theorem can be found in [3].*

The change of numeraire is a powerful technique for valuing derivatives using the martingale approach to arbitrage pricing theory. We first define

what a numeraire is, show that an equivalent probability measure does indeed exist for this numeraire, and then describe how to use them to price arbitrary derivative products.

Definition 10 (Numeraire) *A numeraire is simply the unit of measure in terms of which we find the worth of financial instruments, be it in terms of currency, gold, index levels, etc. In arbitrage pricing theory, risk-neutral valuation uses the risk-free bank account as the numeraire, but pricing problems can sometimes be significantly simplified by using a different instrument as the numeraire. In an arbitrage-free market any traded asset can be used as a numeraire [23]. The Libor Market Model uses various discount bonds as numeraire.*

Theorem 11 (The First Fundamental theorem) *A financial model is arbitrage free iff there exists a (local) martingale measure $\mathbb{P}_X$ (equivalent to the risk-neutral measure $\mathbb{P}$) for the numeraire X. The proof of this theorem can be found in [3].*

Theorem 12 (General pricing formula) *Let X and V be the price processes of two assets in our financial model. Let X be the numeraire with a corresponding martingale measure $\mathbb{P}_X$. Then the price of $V(t)$ at time t with arbitrary payoff $V(T)$ at time T can be calculated using*

$$V(t) = X(t)\mathbb{E}_{\mathbb{P}_X}\left[\frac{V(T)}{X(T)} \mid \mathcal{F}_t\right]$$

The proof of this theorem can be found in [3].

The abstract Bayes theorem allows us to find the expectation of a random variable under a probability measure we know little about by finding rather the expectation of the variable under a known equivalent probability measure but weighted by some known likelihood process. We need it to prove the theorem that follows.

Theorem 13 (Abstract Bayes theorem) *Assume that X is a random variable on $(\Omega, \mathcal{F}, \mathbb{P})$ and let $\mathbb{Q}$ be another probability measure on $(\Omega, \mathcal{F})$ with*

Radon-Nikodym derivative $L = \frac{d\mathbb{Q}}{d\mathbb{P}}$ *on* $\mathcal{F}$. *Assume that* $X \in \mathcal{L}^1(\Omega, \mathcal{F}, \mathbb{Q})$ *and that* $\mathcal{G}$ *is a* σ*-algebra with* $\mathcal{G} \subseteq \mathcal{F}$. *Then*

$$\mathbb{E}_{\mathbb{Q}}[X \mid \mathcal{G}] = \frac{\mathbb{E}_{\mathbb{P}}[L \cdot X \mid \mathcal{G}]}{\mathbb{E}_{\mathbb{P}}[L \mid \mathcal{G}]}, \quad \mathbb{Q}\text{-}a.s.$$

The proof of this theorem can be found in [3].

The following theorem helps us choose the likelihood ratio that will generate a new numeraire's martingale measure from an existing one. We will use this result repeatedly to move between the equivalent martingale measures where each discount bond is the numeraire.

Theorem 14 *We have a probability measure* $\mathbb{P}_0$ *for the numeraire* S_0 *and wish to generate a new martingale measure* $\mathbb{P}_1$ *for the numeraire* S_1. *If we define* $\mathbb{P}_1$ *using Girsanov's theorem (Theorem 8) with the likelihood process*

$$L_0^1(t) = \frac{S_0(0)}{S_1(0)} \cdot \frac{S_1(t)}{S_0(t)}, \quad 0 \leq t \leq T$$

then $\mathbb{P}_1$ *is a martingale measure for numeraire* S_1.

Proof. Consider any arbitrage free price process Π. We wish to show that the normalised process $\Pi(t)/S_1(t)$ is a $\mathbb{P}_1$-martingale. Note that $\Pi(t)/S_0(t)$ and $L_0^1(t)$ are $\mathbb{P}_0$-martingales. For $s \leq t$, using the Abstract Bayes theorem

(Theorem 13), we have

$$\begin{aligned}
\mathbb{E}^{\mathbb{P}_1}\left[\frac{\Pi(t)}{S_1(t)} \mid \mathcal{F}_s\right] &= \frac{\mathbb{E}^{\mathbb{P}_0}\left[L_0^1(t)\frac{\Pi(t)}{S_1(t)} \mid \mathcal{F}_s\right]}{\mathbb{E}^{\mathbb{P}_0}\left[L_0^1(t) \mid \mathcal{F}_s\right]} \\
&= \frac{\mathbb{E}^{\mathbb{P}_0}\left[L_0^1(t)\frac{\Pi(t)}{S_1(t)} \mid \mathcal{F}_s\right]}{L_0^1(s)} \\
&= \frac{\mathbb{E}^{\mathbb{P}_0}\left[\frac{S_0(0)}{S_1(0)}\frac{S_1(t)}{S_0(t)}\frac{\Pi(t)}{S_1(t)} \mid \mathcal{F}_s\right]}{L_0^1(s)} \\
&= \frac{\mathbb{E}^{\mathbb{P}_0}\left[\frac{S_0(0)}{S_1(0)}\frac{\Pi(t)}{S_0(t)} \mid \mathcal{F}_s\right]}{L_0^1(s)} \\
&= \frac{S_0(0)}{S_1(0)}\frac{\mathbb{E}^{\mathbb{P}_0}\left[\frac{\Pi(t)}{S_0(t)} \mid \mathcal{F}_s\right]}{L_0^1(s)} \\
&= \frac{S_0(0)}{S_1(0)}\frac{\frac{\Pi(s)}{S_0(s)}}{L_0^1(s)} \\
&= \frac{S_0(0)}{S_1(0)}\frac{\Pi(s)}{S_0(s)}\frac{S_1(0)}{S_0(0)}\frac{S_0(s)}{S_1(s)} \\
&= \frac{\Pi(s)}{S_1(s)}
\end{aligned}$$

Thus $\Pi(t)/S_1(t)$ is a $\mathbb{P}_1$-martingale and $\mathbb{P}_1$ is a martingale measure for numeraire S_1. ■

3.3 General option theory

This section presents two important results from stochastic calculus for the pricing of vanilla European options. The first describes the statistical qualities of lognormal processes and the second uses these qualities to obtain a closed formula for a vanilla European option. For an in-depth treatment of martingale pricing theory, refer to [23].

Theorem 15 *If we have a process with the following dynamics*

$$dL(t) = L(t)\sigma(t)dW(t)$$

where $W_t(t)$ is a standard Brownian motion under some probability measure $\mathbb{P}$ and $\sigma(t)$ is deterministic, then $L(T)$ is lognormal under $\mathbb{P}$ with

$$\begin{aligned} Var_{\mathbb{P}}\left[\ln L(T) \mid \mathcal{F}_t\right] &= \int_t^T \|\sigma(s)\|^2 \, ds \\ \mathbb{E}_{\mathbb{P}}\left[\ln L(T) \mid \mathcal{F}_t\right] &= \ln L(t) - \frac{1}{2}\int_t^T \|\sigma(s)\|^2 \, ds \end{aligned}$$

Proof. We have $dL(t) = L(t)\sigma(t)dW(t)$. Put $Y(t) = \ln L(t)$. Let us find the $\mathbb{P}$-dynamics for $\ln L(T)$ using Itô:

$$\begin{aligned} dY &= \frac{1}{L}dL - \frac{1}{2}\frac{1}{L^2}d\langle L\rangle \\ &= -\frac{1}{2}\|\sigma(t)\|^2 dt + \sigma(t)dW(t) \\ \implies d(\ln L) &= -\frac{1}{2}\|\sigma(t)\|^2 dt + \sigma(t)dW(t) \\ \implies \ln L(T) &= \ln L(t) - \frac{1}{2}\int_t^T \|\sigma(s)\|^2 ds + \int_t^T \sigma(s)dW(s) \end{aligned}$$

Consider the mean of $\ln L(T)$ under $\mathbb{P}$:

$$\begin{aligned} \mathbb{E}_{\mathbb{P}}\left[\ln L(T) \mid \mathcal{F}_t\right] &= \mathbb{E}_{\mathbb{P}}\left[\ln L(t) - \frac{1}{2}\int_t^T \|\sigma(s)\|^2 ds + \int_t^T \sigma(s)dW(s) \mid \mathcal{F}_t\right] \\ &= \mathbb{E}_{\mathbb{P}}\left[\ln L(t) \mid \mathcal{F}_t\right] + \mathbb{E}_{\mathbb{P}}\left[-\frac{1}{2}\int_t^T \|\sigma(s)\|^2 ds \mid \mathcal{F}_t\right] + \mathbb{E}_{\mathbb{P}}\left[\int_t^T \sigma(s)dW(s) \mid \mathcal{F}_t\right] \\ &= \ln L(t) - \frac{1}{2}\int_t^T \|\sigma(s)\|^2 ds \; + \; 0 \\ &\quad [(\ln L(t) \text{ is } \mathcal{F}_t\text{-msbl}) + (\text{deterministic fns msbl}) + (\text{by Theorem 5})] \\ &= \ln L(t) - \frac{1}{2}\int_t^T \|\sigma(s)\|^2 ds \end{aligned}$$

Consider the variance of $\ln L(T)$ under $\mathbb{P}$:

$$\begin{aligned}
Var_{\mathbb{P}}\left[\ln L(T) \mid \mathcal{F}_t\right] &= \mathbb{E}_{\mathbb{P}}\left[\left(\ln L(T) - \mathbb{E}_{\mathbb{P}}\left[\ln L(T) \mid \mathcal{F}_t\right]\right)^2 \mid \mathcal{F}_t\right] \\
&= \mathbb{E}_{\mathbb{P}}\left[\begin{pmatrix} \ln L(t) - \frac{1}{2}\int_t^T \|\sigma(s)\|^2 ds + \int_t^T \sigma(s) dW(s) \\ -\ln L(t) + \frac{1}{2}\int_t^T \|\sigma(s)\|^2 ds \end{pmatrix}^2 \mid \mathcal{F}_t\right] \\
&= \mathbb{E}_{\mathbb{P}}\left[\left(\int_t^T \sigma(s) dW(s)\right)^2 \mid \mathcal{F}_t\right] \\
&= \mathbb{E}_{\mathbb{P}}\left[\int_t^T \sigma^2(s) ds \mid \mathcal{F}_t\right] \quad \text{(by Theorem 6)} \\
&= \int_t^T \sigma^2(s) ds \quad \text{(deterministic functions are always measurable)}
\end{aligned}$$

■

Theorem 16 *Suppose* $\ln X \sim N(\mu, \sigma^2)$. *Then for any* $x > 0$,

$$\mathbb{E}\left[(X - x)^+\right] = \mathbb{E}[X]\,\Phi(d_1) - x\Phi(d_2)$$

where $d_1 = (\mu + \sigma^2 - \ln x)/\sigma$ *and* $d_2 = d_1 - \sigma = (\mu - \ln x)/\sigma$. *Note that* $(X - x)^+ = \max(X - x, 0)$, *and is used throughout the text.*

Proof. Let $\mathcal{I}_{\{X>x\}}$ be the indicator function on the set $\{X > x\}$, i.e.

$$\mathcal{I}_{\{X>x\}} = \begin{cases} 1 & \text{when } X > x \\ 0 & \text{when } X \leq x \end{cases}$$

Then,

$$\begin{aligned}\mathbb{E}\left[(X-x)^+\right] &= \mathbb{E}\left[(X-x)\mathcal{I}_{\{X>x\}}\right] \\ &= \mathbb{E}\left[X\mathcal{I}_{\{X>x\}}\right] - \mathbb{E}\left[x\mathcal{I}_{\{X>x\}}\right] \\ &= (\mathtt{A}) - (\mathtt{B})\end{aligned}$$

Let us first consider (A). We modify a proof from [26]. Let $g(X)$ be the probability distribution function of X. Now $\ln X \sim N(\mu, \sigma^2)$, so define the standard normal variable $Q = (\ln X - \mu)/\sigma$, whose probability distribution function is simply $h(Q) = \frac{1}{\sqrt{2\pi}} e^{-\frac{Q^2}{2}}$. Note also that a property of the lognormal distribution is that $\mu = \ln[\mathbb{E}X] - \frac{1}{2}\sigma^2$. Then we can solve

$$\begin{aligned}(\mathtt{A}) &= \mathbb{E}\left[X\mathcal{I}_{\{X>x\}}\right] \\ &= \int_x^\infty X g(X) dX \\ &= \int_{(\ln x-\mu)/\sigma}^\infty e^{Q\sigma+\mu} h(Q) dQ \\ &= \int_{(\ln x-\mu)/\sigma}^\infty e^{Q\sigma+\mu} \frac{1}{\sqrt{2\pi}} e^{-\frac{Q^2}{2}} dQ \\ &= e^{\mu+\frac{1}{2}\sigma^2} \int_{(\ln x-\mu)/\sigma}^\infty \frac{1}{\sqrt{2\pi}} e^{\frac{1}{2}[-(Q-\sigma)^2]} dQ \\ &= e^{\mu+\frac{1}{2}\sigma^2} \int_{(\ln x-\mu)/\sigma}^\infty h(Q-\sigma) dQ \\ &= e^{\mu+\frac{1}{2}\sigma^2} \left[1 - \Phi\left((\ln x - \mu)/\sigma - \sigma\right)\right] \\ &= e^{\mu+\frac{1}{2}\sigma^2} \Phi\left(\sigma - (\ln x - \mu)/\sigma\right) \\ &= e^{\ln[\mathbb{E}X]-\frac{1}{2}\sigma^2+\frac{1}{2}\sigma^2} \Phi\left(\sigma - (\ln x - \mu)/\sigma\right) \\ &= \mathbb{E}X\Phi\left((\mu + \sigma^2 - \ln x)/\sigma\right) \\ &= \mathbb{E}[X]\Phi(d_1)\end{aligned}$$

Now let us consider (B). We could solve this in the same way we solved for (A), but we will use a more elegant measure theoretic approach:

$$\begin{aligned}
(\mathtt{B}) &= \mathbb{E}\left[x\mathcal{I}_{\{X>x\}}\right] \\
&= x\mathbb{E}\left[\mathcal{I}_{\{X>x\}}\right] \quad (x \text{ is constant and thus measurable}) \\
&= x\mathbb{Q}\left(X > x\right) \\
&= x\mathbb{Q}\left(\ln X > \ln x\right) \\
&= x\mathbb{Q}\left(\frac{\ln X - \mu}{\sigma} > \frac{\ln x - \mu}{\sigma}\right) \\
&= x\mathbb{Q}\left(\frac{\mu - \ln X}{\sigma} < \frac{\mu - \ln x}{\sigma}\right) \\
&= x\Phi\left(\frac{\mu - \ln x}{\sigma}\right) \\
&= x\Phi(d_2)
\end{aligned}$$

■

3.4 Using Black76 to price interest rate options

The Black76 model [5] is the industry-standard method for pricing European options on a number of underlying instruments [26]. Its sole assumption is that the underlying is lognormally distributed under the risk-neutral measure at expiry. No restrictions are placed on the behaviour of the underlying before expiry. In addition, it assumes that the risk-neutral interest rate is a non-stochastic process. We shall see during the following derivations of the Black76 formula that this assumption does not make sense when pricing two common interest rate derivatives: caplets and swaptions. In fact, some motivation for the development of the Libor Market Model was to address the problems of using the Black76 model for interest rate derivatives [4].

3.4.1 Derivation of Black76 for caplets

Let us first derive the Black76 formula for caplets. It is a worthwhile exercise to derive the Black76 formula for caplets because we will need to use market

implied volatilities of the caplets when we calibrate our Libor Market Model in Section 5.4. This market volatility is the implied volatility that is used in the Black76 formula to correctly price the caplet at market value. So let us now derive the Black76 formula for caplets.

Consider a caplet $V(t)$ on the Libor forward rate $L_j(t)$ with strike K and expiry T_{j-1} with payoff $\delta_j \left(L_j(T_{j-1}) - K\right)^+$ at time T_j. We assume that the forward rate $L_j(T_{j-1})$ is lognormal in the risk-neutral world, so that its logarithm $\ln L_j(T_{j-1})$ has variance $\sigma_{j-1}^2 T_{j-1}$ and hence mean $\mathbb{E}\left[L_j(T_{j-1})\right] - \frac{1}{2}\sigma_{j-1}^2 T_{j-1}$. Note that the variance $\sigma_{j-1}^2 T_{j-1}$ will generally be dependent on the time span $[t,\ T_{j-1}]$. Suppose $\mathbb{Q}$ is the risk-neutral measure for the bank account (earning the risk-free rate r_t) as numeraire. Then by Theorem 12 we have

$$\begin{aligned} V(t) &= 1 \times \mathbb{E}_{\mathbb{Q}}\left[\frac{V(T_j)}{e^{\int_t^{T_j} r_t dt}} \mid \mathcal{F}_t\right] \\ &= \mathbb{E}_{\mathbb{Q}}\left[e^{-\int_t^{T_j} r_t dt} \delta_j \left(L_j(T_{j-1}) - K\right)^+ \mid \mathcal{F}_t\right] \end{aligned}$$

The first assumption in using Black's formula is that the option payoff, a function of the underlying, is independent of the prevailing interest rates, and so we can split the expectation. This assumption has proved to work with a variety of European options on different underlyings such as equity and commodities [26], but when the underlying is an interest rate itself (in our case a Libor forward rate) this assumption is clearly absurd. The independence assumption implies

$$\begin{aligned} V(t) &\simeq \mathbb{E}_{\mathbb{Q}}\left[e^{-\int_t^{T_j} r_t dt} \mid \mathcal{F}_t\right] \mathbb{E}_{\mathbb{Q}}\left[\delta_j \left(L_j(T_{j-1}) - K\right)^+ \mid \mathcal{F}_t\right] \\ &= P(t,T_j)\delta_j \mathbb{E}_{\mathbb{Q}}\left[\left(L_j(T_{j-1}) - K\right)^+ \mid \mathcal{F}_t\right] \end{aligned}$$

Now using the fact that $L_j(T_{j-1})$ is lognormal in the risk-neutral world and the results of Theorem 16,

$$V(t) = P(t,T_j)\delta_j \left[\mathbb{E}\left[L_j(T_{j-1})\right]\Phi(d_1) - K\Phi(d_2)\right]$$

where

$$\begin{aligned} d_1 &= (\ln(\mathbb{E}[L_j(T_{j-1}) \mid \mathcal{F}_t]) - \frac{1}{2}\sigma_{j-1}^2 T_{j-1} + \sigma_{j-1}^2 T_{j-1} - \ln K)/\sigma_{j-1}\sqrt{T_{j-1}} \\ &= (\ln(\mathbb{E}[L_j(T_{j-1}) \mid \mathcal{F}_t]/K) + \frac{1}{2}\sigma_{j-1}^2 T_{j-1})/\sigma_{j-1}\sqrt{T_{j-1}} \\ d_2 &= d_1 - \sigma_{j-1}\sqrt{T_{j-1}} \end{aligned}$$

The second assumption that is made when using Black's formula is that the expected future value of the underlying is equal to its current forward rate, i.e. that $\mathbb{E}[L_j(T_{j-1}) \mid \mathcal{F}_t] = L_j(t)$. However the expected value of an instrument under the risk-neutral measure is its futures price, not its forward price. So this assumption is reasonable only if interest rates are non-stochastic since then the futures price of an instrument is the same as its forward price [26]. In the case of modelling interest rate derivatives we are modelling stochastic interest rates so this assumption is invalid. Substituting our second assumption we arrive at

$$V(t) = P(t, T_j)\delta_j [L_j(t)\Phi(d_1) - K\Phi(d_2)] \tag{6}$$

where

$$d_1 = (\ln[L_j(t)/K] + \frac{1}{2}\sigma_{j-1}^2 T_{j-1})/\sigma_{j-1}\sqrt{T_{j-1}}, \quad d_2 = d_1 - \sigma_{j-1}\sqrt{T_{j-1}}$$

3.4.2 Derivation of Black76 for swaptions

When it comes time to calibrate our Libor Market Model in Section 5.5 we will need to use the market prices for swaption volatilities. This market volatility is the implied volatility that is used in the Black76 formula to correctly price the swaption at market value. So let us now derive the Black76 formula for swaptions.

To derive the analytical Black76 price for European swaptions we follow a very different line of reasoning to that which we followed in the previous section when we determined the Black76 caplet pricing formula. In the case of pricing the caplet we used the risk-free bank account as our numeraire,

which required several ugly assumptions about the relationship between the risk-free interest rate and the forward rates. To price the swaption we will make use of a different numeraire to obviate the need for any of these assumptions. This will lead the way to the next section where we again price the caplet, but in the framework of the Libor Market Model, and in the process we will find that we do not need to make any assumptions about the interaction between the different interest rates.

The only assumption we do have to make is that the swap rates are lognormal. It can be shown that swap rates and forward rates can not simultaneously be lognormal, but this will be covered in Section 5.5.

From (3) we have that the swap rate for a swap from time T_i to T_N is

$$SR_i = \frac{\sum_{j=i}^{N} L_j \delta_j P(T_j)}{\sum_{j=i}^{N} \delta_j P(T_j)}$$

But from (1) we can write L_j in terms of our zero coupon bonds $P(T_j)$ as

$$\begin{aligned} L_j(t) &= \frac{1}{\delta_j}\left[\frac{P(T_{j-1})}{P(T_j)} - 1\right] \\ \Longrightarrow L_j\delta_j &= \frac{P(T_{j-1})}{P(T_j)} - 1 \end{aligned}$$

hence

$$\begin{aligned} SR_i &= \frac{\sum_{j=i}^{N}\left[\frac{P(T_{j-1})}{P(T_j)}-1\right]P(T_j)}{\sum_{j=i}^{N}\delta_j P(T_j)} \\ &= \frac{\sum_{j=i}^{N}[P(T_{j-1})-P(T_j)]}{\sum_{j=i}^{N}\delta_j P(T_j)} \\ &= \frac{P(T_{i-1})-P(T_N)}{\sum_{j=i}^{N}\delta_j P(T_j)} \end{aligned}$$

So our swap rate SR_i can be written as the ratio of two tradeable assets $P(T_{i-1}) - P(T_N)$ and $\sum_{j=i}^{N}\delta_j P(T_j)$. Let $X_i(t) = \sum_{j=i}^{N}\delta_j P(T_j)$ be our numeraire and invoke Theorem 11 to note that there exists a measure $\mathbb{P}_X$ under which $SR_i(t) = (P(T_{i-1}) - P(T_N))/X_i(t)$ is a martingale (as are all tradable assets divided by $X_i(t)$). Notice that $SR_i > 0$ since $P(T_{i-1}) > P(T_N)$ and $P(T_j) > 0$ (Theorem 1). Let us pick (as a basic assumption of our model) a deterministic form for $\sigma_i(t)$, and use Theorem 7 to write

$$dSR_i(t) = SR_i(t)\sigma_i(t)dW^{X_i}(t)$$

where $W^{X_i}(t)$ is an $\mathbb{P}_{X_i}$-martingale. Since $\sigma_i(t)$ is deterministic, we use Theorem 15 to find that $SR_i(T_{i-1})$ is lognormal under $\mathbb{P}_{X_i}$ with

$$\begin{aligned} Var_{\mathbb{P}_{X_i}}\left[\ln SR_i(T_{i-1}) \mid \mathcal{F}_t\right] &= \int_t^{T_{i-1}} \|\sigma_i(s)\|^2 ds \\ \mathbb{E}_{\mathbb{P}_{X_i}}\left[\ln SR_i(T_{i-1}) \mid \mathcal{F}_t\right] &= \ln SR_i(t) - \frac{1}{2}\int_t^{T_{i-1}} \|\sigma_i(s)\|^2 ds \end{aligned}$$

Now using (4) a swaption whose strike is K is worth at time T_{i-1} the amount $V_i(T_{i-1}) = X_i(T_{i-1})(SR_i(T_{i-1}) - K)^+$. Using Theorem 12 we get

$$\begin{aligned} \frac{V_i(t)}{X_i(t)} &= \mathbb{E}_{\mathbb{P}_{X_i}}\left[\frac{V_i(T_{i-1})}{X_i(T_{i-1})} \mid \mathcal{F}_t\right] \\ \implies V_i(t) &= X_i(t)\mathbb{E}_{\mathbb{P}_{X_i}}\left[(SR_i(T_{i-1}) - K\ )^+ \mid \mathcal{F}_t\right] \end{aligned}$$

And using Theorem 16 we have,

$$V_i(t) = X_i(t)\left[SR_i(t)\Phi(d_1) - K\Phi(d_2)\right] \tag{7}$$

where

$$d_1 = \frac{\ln(SR_i(t)/K) + \frac{1}{2}\sigma_v^2}{\sigma_v}, \quad d_2 = d_1 - \sigma_v, \quad \sigma_v^2 = \int_t^{T_{i-1}} \|\sigma_i(s)\|^2\, ds$$

4 The Libor Market Model

This section is the most important of the dissertation: it derives the Libor Market Model. We start off in Section 4.1 with an outline of the procedure we will follow when deriving the Libor Market Model. Section 4.2 derives the equivalent of the Black76 formula for caplets using the $P(t, T_j)$ discount bond as numeraire instead of the risk-free bank account as was done in Section 3.4.1. We find that we arrive at the same expression as (6) without having to make the unrealistic assumptions that we did before. Section 4.3 then uses change of measure technology to be able to price more complex derivatives with payoffs at multiple time periods. This leads us to a stochastic differential equation that describes the dynamics of the forward rates when using any arbitrary discount bond as numeraire. This stochastic differential equation is too complex to solve analytically, so Section 4.4 describes how to discretise the stochastic differential equation into a form that is compatible with the Monte Carlo integration technique. Section 4.5 shows how we transform the multiple cashflows from different time periods to payoffs at the same time as the expiry of the numeraire so that we can use martingale theory to calculate the time-t prices of a derivative product. Finally Section 4.6 shows how we calculate and use the Greeks to hedge our derivative products.

4.1 The plan

- Section 4.2 prices a simple caplet $caplet_j$. This is rather straightforward because it has one payoff $\delta_j \left(L_j(T_{j-1}) - K\right)^+$ at time T_j.
  - Choose the discount bond $P(t, T_j)$ to be our numeraire because it has the value of 1 at time T_j coinciding with our caplet payoff. This will greatly simplify our expectation integral.
  - Divide the tradable asset $L_j(t)P(t, T_j)$ by our numeraire to show that $L_j(t) = L_j(t)P(t, T_j)/P(t, T_j)$ is a martingale under the measure $\mathbb{P}_{T_j}$ (corresponding to the numeraire $P(t, T_j)$).

- We choose an equation whose solution is a positive lognormal martingale for $L_j(t)$ under $\mathbb{P}_{T_j}$.
- With the dynamics of $L_j(t)$ in hand we can calculate the distribution of $L_j(T_{j-1})$ and hence find the value of the caplet.

- Section 4.3 shows how to price the more complex derivatives that use more than just one forward rate and generate payoffs at more than one time. This requires that we find the dynamics of all our forward rates under a single measure.
  - We find a suitable Girsanov transform to move our forward rate dynamics from one measure $\mathbb{P}_{T_j}$ to its neighbouring measure $\mathbb{P}_{T_{j-1}}$.
  - We use this transform inductively to move from any measure $\mathbb{P}_{T_j}$ to any other measure $\mathbb{P}_{T_p}$, and specifically choose the terminal measure $\mathbb{P}_{T_N}$ as that in which we will work.

4.2 Pricing a single caplet

The derivation in this section follows closely the treatment presented in [13], although we present the market model in discrete time. We showed in Lemma 3 that $L_j(t)P(t,T_j)$ is a tradable asset. Let $V(t) = L_j(t)P(t,T_j)$ and let the numeraire be $X(t) = P(t,T_j)$. Then we have that our forward rate can be defined by $L_j(t) = V(t)/X(t)$. Since both $V(t)$ and $X(t)$ are tradable assets we invoke Theorem 11 to note that there exists a measure $\mathbb{P}_{T_j}$ under which $L_j(t) = V(t)/X(t)$ is a martingale (as are all tradable assets divided by $P(t,T_j)$). We call $\mathbb{P}_{T_j}$ the forward measure because we are using the discount bond $P(t,T_j)$ as numeraire. Since $P(t,T_{j-1}) > P(t,T_j)$ (Theorem 1), we know that $L_j(t) > 0$ and because $L_j(t)$ is a positive $\mathbb{P}_{T_j}$-martingale we can use Theorem 7 to write

$$dL_j(t) = L_j(t)\sigma_j(t)dW^{T_j}(t) \tag{8}$$

where $W^{T_j}(t)$ is a correlated M-dimensional $\mathbb{P}_{T_j}$-martingale and $\sigma_j(t)$ is some deterministic vector of functions. Note that $W^{T_j}(t)$ models M sources of

noise with correlation matrix ρ (ρ_{ij} is the correlation between noise source i and j). When modelling only a single caplet we could replace this with a single source of noise and adjust $\sigma_j(t)$ accordingly. However, in the next section we will be pricing more complex derivatives based on multiple forward rates with multiple payoffs at various times, and for this we require several sources of noise so that the correlation between the forward rates can be modelled realistically [59]: we derive a so-called multi-factor version of the Libor Market Model. In Section 5.6 we will describe a technique that reduces this number of factors.

Since $\sigma_j(t)$ is deterministic, we use Theorem 15 to find that $L_j(T_{j-1})$ is lognormal under $\mathbb{P}_{T_j}$ with

$$\begin{aligned} Var_{\mathbb{P}_{T_j}}\left[\ln L_j(T_{j-1}) \mid \mathcal{F}_t\right] &= \int_t^{T_{j-1}} \|\sigma_j(s)\|^2 \, ds \\ \mathbb{E}_{\mathbb{P}_{T_j}}\left[\ln L_j(T_{j-1}) \mid \mathcal{F}_t\right] &= \ln L_j(t) - \frac{1}{2}\int_t^{T_{j-1}} \|\sigma_j(s)\|^2 \, ds \end{aligned}$$

Now a caplet pays at time T_j the amount $V(T_j) = \delta_j(L_j(T_{j-1}) - K)^+$. At time $T_{j-1} < t < T_j$ we have $V(t) = P(t,T_j)\delta_j\left(L_j(T_{j-1}) - K\right)^+$. Using the numeraire $X(t) = P(t,T_j)$ we use Theorem 12 to get

$$\begin{aligned} V(t) &= P(t,T_j)\delta_j \mathbb{E}_{\mathbb{P}_{T_j}}\left[\frac{V(T_j)}{P(T_j,T_j)} \mid \mathcal{F}_t\right] \\ &= P(t,T_j)\delta_j \mathbb{E}_{\mathbb{P}_{T_j}}\left[V(T_j) \mid \mathcal{F}_t\right] \\ &= P(t,T_j)\delta_j \mathbb{E}_{\mathbb{P}_{T_j}}\left[\left(L_j(T_{j-1}) - K\right)^+ \mid \mathcal{F}_t\right] \end{aligned}$$

And since $L_j(T_{j-1})$ is lognormal under $\mathbb{P}_{T_j}$ we use Theorem 16 to obtain

$$V(t) = P(t,T_j)\delta_j\left[L_j(t)\Phi(d_1) - K\Phi(d_2)\right] \tag{9}$$

where

$$d_1 = \frac{\ln(L_j(t)/K) + \frac{1}{2}\sigma_v^2}{\sigma_v}, \quad d_2 = d_1 - \sigma_v, \quad \sigma_v^2 = \int_t^{T_{j-1}} \|\sigma_j(s)\|^2 \, ds$$

Notice that the equations for the Libor caplet price (9) and the Black76 caplet price (6) are identical. However their derivations are quite different: in the case of the Black76 derivation, we make two messy assumptions when taking expectations under the risk-neutral measure $\mathbb{Q}$; in the case of the Libor derivation, we make no such assumptions under the forward measure $\mathbb{P}_{T_j}$. Also, when pricing several caplets simultaneously under the risk-neutral measure we assume that all the forward rates are simultaneously lognormal, leading to explosive rates; under the forward measure, each caplet expiring at T_{j-1} is lognormal under its own measure $\mathbb{P}_{T_j}$, avoiding the problem [59].

The fact that the Black76 equation can be used to correctly price caplets under their own measure is fundamental to the calibration of the Libor Market Model to current market data. We will see this used in section 5.4.

4.3 Pricing more complex instruments

The previous section has built a model that describes the dynamics of each $L_j(t)$ process under its own measure $\mathbb{P}_{T_j}$. The model can price only relatively simple derivatives (e.g. caplets) that depend on the process $L_j(t)$ with a single payoff at time T_{j-1}. We wish to develop the model further to allow us to price more complex instruments that depend on several $L_j(t)$ processes with payoffs at arbitrary times T_p. The derivation in this section follows closely the treatment presented in [3].

Assuming $L_j(t)$ dynamics of the form (8), where we have no drift term, we use successive Girsanov transforms to arrive at the dynamics with drift of the form:

$$dL_j(t) = L_j(t) \left[\mu_j(t)dt + \sigma_j(t)dW^{T_p}(t)\right] \tag{10}$$

Since we can apply Girsanov's theorem in either direction, we will have developed a suitable choice of $\mu_j(t)$ that will allow us to move between dynamics

of the form (8) and (10).

From (1) we have that

$$\begin{aligned} L_j(t) &= \frac{1}{\delta_j}\left[\frac{P(t,T_{j-1})}{P(t,T_j)} - 1\right] \\ &= \frac{1}{\delta_j}\left[P_j(t) - 1\right] \qquad (11) \\ \implies P_j(t) &= 1 + \delta_j L_j(t) \qquad (12) \end{aligned}$$

Instead of jumping directly from the measure $\mathbb{P}_{T_j}$ to $\mathbb{P}_{T_p}$ lets see what dynamics we find for a single step in change of measure from $\mathbb{P}_{T_j}$ to $\mathbb{P}_{T_{j-1}}$. Define the likelihood process:

$$\begin{aligned} \eta_j^{j-1}(t) &= \frac{P(0,T_j)}{P(0,T_{j-1})}\frac{P(t,T_{j-1})}{P(t,T_j)} \\ &= \frac{P(0,T_j)}{P(0,T_{j-1})}P_j(t) \\ &= \frac{P(0,T_j)}{P(0,T_{j-1})}\left(1+\delta_j L_j(t)\right) \\ \implies d\eta_j^{j-1}(t) &= \frac{P(0,T_j)}{P(0,T_{j-1})}\delta_j dL_j(t) \\ &= \frac{P(0,T_j)}{P(0,T_{j-1})}\delta_j L_j(t)\sigma_j(t)dW^{T_j}(t) \\ &= \frac{\eta_j^{j-1}(t)}{(1+\delta_j L_j(t))}\delta_j L_j(t)\sigma_j(t)dW^{T_j}(t) \\ &= \eta_j^{j-1}(t)\frac{\delta_j L_j(t)}{(1+\delta_j L_j(t))}\sigma_j(t)dW^{T_j}(t) \end{aligned}$$

Girsanov's theorem (Theorem 8) tells us that a kernel of

$$\frac{\delta_j L_j(t)}{(1+\delta_j L_j(t))}\sigma_j(t)$$

which we will ensure satisfies the Novikov condition (Lemma 9) in our choice of $\sigma_j(t)$, will give the relation between the Brownian motions under two

successive measures as

$$dW^{T_j}(t) = \frac{\delta_j L_j(t)}{(1+\delta_j L_j(t))}\sigma_j(t)dt + dW^{T_{j-1}}(t)$$

Rearranging to get $dW^{T_{j-1}}(t)$ in terms of $dW^{T_j}(t)$,

$$dW^{T_{j-1}}(t) = dW^{T_j}(t) - \frac{\delta_j L_j(t)}{(1+\delta_j L_j(t))}\sigma_j(t)dt$$

Similarly for $dW^{T_{j-2}}(t)$,

$$\begin{aligned} dW^{T_{j-2}}(t) &= dW^{T_{j-1}}(t) - \frac{\delta_{j-1} L_{j-1}(t)}{(1+\delta_{j-1} L_{j-1}(t))}\sigma_{j-1}(t)dt \\ &= dW^{T_j}(t) - \frac{\delta_j L_j(t)}{(1+\delta_j L_j(t))}\sigma_j(t)dt - \frac{\delta_{j-1} L_{j-1}(t)}{(1+\delta_{j-1} L_{j-1}(t))}\sigma_{j-1}(t)dt \end{aligned}$$

We can now apply this single time step transformation inductively to move from any measure $\mathbb{P}_{T_j}$ to $\mathbb{P}_{T_p}$ using the relation

$$dW^{T_j}(t) = \begin{cases} dW^{T_p}(t) - \sum_{k=j+1}^{p} \frac{\delta_k L_k(t)}{(1+\delta_k L_k(t))}\sigma_k(t)dt & \text{for } j < p \\ dW^{T_p}(t) & \text{for } j = p \\ dW^{T_p}(t) + \sum_{k=p}^{j-1} \frac{\delta_k L_k(t)}{(1+\delta_k L_k(t))}\sigma_k(t)dt & \text{for } j > p \end{cases}$$

Plugging this into (8) we obtain dynamics of the form (10):

$$\begin{aligned} dL_j(t) &= L_j(t)\sigma_j(t)dW^{T_j}(t) \\ &= \begin{cases} L_j(t)\sigma_j(t)dW^{T_p}(t) - L_j(t)\sum_{k=j+1}^{p} \frac{\delta_k L_k(t)}{(1+\delta_k L_k(t))}\sigma_j(t)\sigma_k(t)\rho_{jk}dt & \text{for } j < p \\ L_j(t)\sigma_j(t)dW^{T_p}(t) & \text{for } j = p \\ L_j(t)\sigma_j(t)dW^{T_p}(t) + L_j(t)\sum_{k=p}^{j-1} \frac{\delta_k L_k(t)}{(1+\delta_k L_k(t))}\sigma_j(t)\sigma_k(t)\rho_{jk}dt & \text{for } j > p \end{cases} \end{aligned} \tag{13}$$

where we have used the convention $\sum_{p+1}^{p}(...) = 0$.

In general (13) will not have a solution except in the special case when the $\sigma_i(t)$ are bounded over any time interval $[0, t]$. Fortunately this special case is relevant to most practical finance and the proof of the existence of a solution is given in [28]. Although we are only interested in the derivation of the dynamics of the forward rates, [28] proves in detail that the model is arbitrage free by showing that all discount bonds $P(0, T_i)$ divided by the terminal discount bond $P(0, T_N)$ (as defined by our forward rate dynamics) are indeed martingales under the terminal measure and then extends the proof to all numeraire-rebased assets in the economy. It is also worth mentioning that a large motivation for the more complex Libor Market Model presented in [30] is that we makes absolutely sure that his mathematical model is descriptive of the market and is arbitrage free.

4.4 Discretisation

In the previous section we derived Equation (13), which describes the arbitrage-free simultaneous dynamics for all our Libor forward rates $L_j(t)$ under any chosen measure $\mathbb{P}_{T_p}$. Ideally, we would like to find a closed form solution to the stochastic differential equation, but in the case of (13) the dynamics are too complex to solve analytically. Instead we have to resort to approximate numerical schemes where we discretise the dynamics so that they can be simulated in software. We follow the treatment in [46], where we focus on discretisation under the terminal measure $\mathbb{P}_{T_N}$, corresponding to a numeraire of $P(0, T_N)$. The arbitrage-free dynamics of the forward rates under the terminal measure are

$$dL_j(t) = L_j(t)\sigma_j(t)dW^{T_N}(t) - L_j(t)\sum_{k=j+1}^{N}\frac{\delta_k L_k(t)}{(1+\delta_k L_k(t))}\sigma_j(t)\sigma_k(t)\rho_{jk}dt \quad (14)$$

The first thing we notice in our dynamics in (14) is that both the drift (the dt) and the volatility (the dW^{T_N}) terms are state dependent since they contain $L_j(t)$. This state dependence make it difficult for us to discretise

the dynamics accurately because we have to approximate the continuous drift and volatility terms over discrete time intervals. We should like to remove as much state dependence from our discretisation as possible.

Using the very useful structure of the Ito derivative of $\ln L_j(t)$, we can eliminate the state dependence of the volatility term by discretising $\ln L_j(t)$ rather than just $L_j(t)$. Putting $X_j(t) = \ln L_j(t)$, we have by Ito,

$$\begin{aligned} dX_j(t) &= \frac{1}{L_j(t)} dL_j(t) - \frac{1}{2}\frac{1}{L_j(t)^2} d\langle L_j(t) \rangle \\ &= \left[-\sum_{k=j+1}^{N} \frac{\delta_k L_k(t) \sigma_j(t) \sigma_k(t) \rho_{jk}}{(1+\delta_k L_k(t))} - \frac{1}{2} \|\sigma_j(t)\|^2 \right] dt + \sigma_j(t) dW^{T_N}(t) \end{aligned}$$

Notice that the state dependence (the $L_j(t)$ factor) has fallen out of the volatility term but we are still left with a state dependent drift term.

To begin our discretisation we integrate both sides over our discretisation timestep and apply a truncated Ito-Taylor expansion [37] to get

$$X_j(t_{i+1}) = X_j(t_i) + \int_{t_i}^{t_{i+1}} \left[-\sum_{k=j+1}^{N} \frac{\delta_k L_k(u) \sigma_j(u) \sigma_k(u) \rho_{jk}}{(1+\delta_k L_k(u))} - \frac{1}{2} \|\sigma_j(u)\|^2 \right] du + \int_{t_i}^{t_{i+1}} \sigma_j(u) dW^{T_N}(u)$$

We now approximate the stochastic terms $L_k(u)$ over the time period $u \in [t_i, t_{i+1}]$ with constants L_k^* to get

$$\begin{aligned} X_j(t_{i+1}) &\simeq X_j(t_i) + \int_{t_i}^{t_{i+1}} \left[-\sum_{k=j+1}^{N} \frac{\delta_k L_k^* \sigma_j(u) \sigma_k(u) \rho_{jk}}{(1+\delta_k L_k^*)} - \frac{1}{2} \|\sigma_j(u)\|^2 \right] du + \int_{t_i}^{t_{i+1}} \sigma_j(u) dW^{T_N}(u) \\ &= X_j(t_i) - \sum_{k=j+1}^{N} \frac{\delta_k L_k^*}{(1+\delta_k L_k^*)} \rho_{jk} \int_{t_i}^{t_{i+1}} \sigma_j(u) \sigma_k(u) du \\ &\quad -\frac{1}{2} \int_{t_i}^{t_{i+1}} \|\sigma_j(u)\|^2 du + \int_{t_i}^{t_{i+1}} \sigma_j(u) dW^{T_N}(u) \end{aligned}$$

How do we choose these L_k^*? The simplest choice would be to use

$L_k^* = L_k(t_i)$, since we know this information at time t_i. [32] states that most banks use this approach and that, with a step size of about three months, the error in the drift approximation is negligible. However [59] found that this approximation is inaccurate for discretisations of three months or more (between two Libor reset dates) and recommends that the Predictor-Corrector (PC) method be used. This method works as follows:

1. Initially, choose $L_k^* = L_k(t_i)$ to evolve $X_j^*(t_{i+1})$ and hence $L_j^*(t_{i+1}) = \exp(X_j^*(t_{i+1}))$.

2. Repeat the computation using $L_k^* = \frac{1}{2}\left[L_k(t_i) + L_j^*(t_{i+1})\right]$ to evolve $X_j(t_{i+1})$ and hence $L_j(t_{i+1}) = \exp(X_j(t_{i+1}))$.

As can be seen from the PC algorithm, we are effectively using the "average" Libor rates between times t_i and t_{i+1} to determine the Libor rate $L_j(t_{i+1})$. It is not the true average because we have to estimate $L_j(t_{i+1})$ using $L_j^*(t_{i+1})$ but [59] notes that this improved approximation of $L_k(u)$ with L_k^* yields very accurate results when compared to improving accuracy by using much finer discretisations.

So the discretised dynamics we implement are of the form,

$$\begin{aligned} X_j(t_{i+1}) &= X_j(t_i) - \frac{1}{2}\int_{t_i}^{t_{i+1}} \|\sigma_j(u)\|^2 du + \int_{t_i}^{t_{i+1}} \sigma_j(u) dW^{T_N}(u) \\ &\quad - \sum_{k=j+1}^{N} \frac{\delta_k L_k^*}{(1+\delta_k L_k^*)} \rho_{jk} \int_{t_i}^{t_{i+1}} \sigma_j(u)\sigma_k(u) du \\ L_j(t_{i+1}) &= e^{X_j(t_{i+1})} \end{aligned} \tag{15}$$

When performing our discretisation we chose to apply a truncated Ito-Taylor expansion to our stochastic differential equation. This generally obtains an order of strong convergence $\gamma = 0.5$ [37], which describes how the error in discretisation diminishes with the decreasing size of the discretisation mesh. It is possible to apply discretisation expansions that have better

orders of convergence: [37] recommends using the Milstein scheme as it is hardly more complex than the simple Euler scheme and has an order of strong convergence $\gamma = 1$. However [29] shows how we are already applying the Milstein scheme to $L_j(t)$ by our simulating its logarithm, $X_j(t)$: little advantage is realised applying the Milstein approximation to $X_j(t)$.

Of more concern than the errors introduced by our discretisation is the fact that our discretisation removes the martingale property from the dynamics of $X_j(t)$: the model is no longer arbitrage free [20]. Two adjustments to the discretisation process are described in [20], where an adjustment is made to the drift of $X_j(t)$, and [43], where a completely different transform of $L_j(t)$ is discretised. Both describe how their improvements are marginal under most conditions. We implement neither approach in this dissertation.

4.5 Pricing financial instruments

Equation (15) of the preceding section allows us to calculate $L_j(t)$ at any time $t \in T_i$. The market data we have available are the initial term structure $L_j(T_0)$ and the various modelled volatilities and correlations that allow us to calculate the various integrals. These then allow us to build up an upper triangular matrix of forward rates as shown in the left-hand matrix of Figure 7. Using (2) we can calculate the discount bonds $P(T_i, T_j)$ in terms of these forward rates as shown in the right-hand matrix of Figure 7.

Using these forward rates and discount bonds we should be able to model the cashflows of pretty much any interesting interest rate derivative. Section 6.4 explains what interest rate derivatives can be priced using the Libor Market Model. We saw from Theorem 16 that if $P(t, T_N)$ is our numeraire and $\mathbb{P}_{T_N}$ its corresponding equivalent measure, then the value of our derivative $V(t)$ is calculated by

$$V(t) = P(t, T_N)(t)\mathbb{E}_{\mathbb{P}_{T_N}}\left[\frac{V(T)}{P(T_N, T_N)} \mid \mathcal{F}_t\right]$$

So to price a derivative for a particular set of evolved forward rates and discount bonds at time t, we have to:

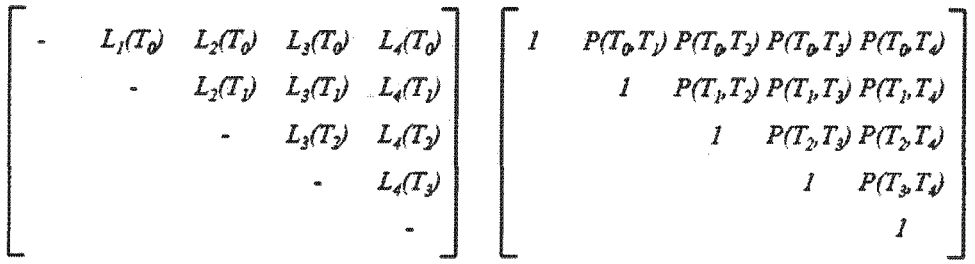

Figure 7: Examples of the matrices of forward rates and discount bonds that are generated with each evolution of the forward rates. The rows correspond to measurements at particular points in time while the columns represent the future time periods at which the rates are relevant.

1. Calculate the cashflows of the derivative at each time T_i.
2. Transport these cashflows forward to time T_N by dividing by our evolved discount bonds $P(T_i, T_N)$.
3. Transport the combined T_N payoffs back to today (time t) by multiplying by today's discount bond $P(t, T_N)$.

4.6 Hedging financial instruments: the Greeks

Section 4.5 describes how to use the Libor Market Model to calculate the price of a financial instrument. This is the "fair" price of the instrument because it is the same amount that theoretically can be realised in the market by a synthetic instrument with identical payoffs constructed using (dynamic) hedging [42]. So it is important that once a trader has sold a financial instrument (at a slight premium) she is able to lock in her profit by hedging the instrument: she requires some recipe for her hedging activity. It is almost certain that the ability to calculate a hedging strategy is more important than the ability to calculate a price: often prices are visible in the market, but it is the hedging strategy that allows the trader to mitigate risk and realise profit. This section describes this hedging process.

Continuous-time arbitrage theory states that we need to be able to hedge

continuously to realise exactly the theoretical price of a financial instrument. Hedging is achieved by shorting the delta of the instrument, making the combined portfolio "delta-neutral". This means that the portfolio value should remain almost constant over a small change in the value of the underlying variables. Delta is calculated as the first derivative of the instrument price with respect to its underlying variables. Delta is one of the "Greeks", a collection of sensitivities of the option to various underlying variables that are used for hedging purposes and risk management. In the case of the Libor Market Model we have many underlying variables, the forward rates L_i, so we require a delta for each of them:

$$\Delta_i = \frac{\partial V}{\partial L_i}$$

Obviously it is impossible to hedge continuously, and once liquidity restrictions and transaction costs are taken into account it might be impractical to hedge more often than every few hours or days. This discrete delta-hedging strategy is inaccurate and it is necessary to attempt to mitigate the costs that are incurred by the less frequent hedging. To this end traders also look at the gamma of the instrument and ensure that their portfolios are both delta- and gamma-neutral. Gamma (another Greek) is calculated as the second derivative of the instrument price (or the first derivative of delta) with respect to its underlying variables. Notice that for each delta we have one gamma for each underlying variable, so we have very many gammas (equal to the square of the number of underlyings):

$$\Gamma_{ij} = \frac{\partial^2 V}{\partial L_i \partial L_j}$$

Among other reasons (as is discussed in Section 4.6.1), this explosive number of gammas makes them difficult to work with, and generally only the diagonal gammas are calculated. In applications where the off-diagonal gammas are required, they are referred to as cross-gammas. [18] briefly discusses how response surface techniques can be applied to approximate these cross-gammas more efficiently.

In addition to neutralising her portfolio against changes in the underlying forward rates, the trader is generally also interested in the effect that changes in the market volatilities (vega is the Greek that measures sensitivity to volatility) and correlations will have on the value of her portfolio. By calculating sensitivities to these values the trader can attempt to hedge against them. In the Libor Market Model we have to decide whether we wish to calculate the sensitivities to the model volatilities (i.e. those of the forward rates) or to the market implied volatilities (i.e. those of the caplets and swaptions to which we calibrate). The first option, although computationally straightforward, yields sensitivities to rather ephemeral measurements of volatility: they are not readily visible in the market and so it is hard to get a feel for them. The second option is extremely computationally expensive because it requires recalibration of the Libor Market Model each time a volatility is bumped. [51] and [53] present very lucid discussions of these problems and the trade-offs involved.

Once we have the Greeks in terms of changes in the underlying forward rates, we can quite easily calculate the Greeks in terms of discount bonds, swaps and other market instruments used to build the yield curve by a simple application of the chain rule [19].

In general, calculating the Greeks inside the Monte Carlo framework faces several challenges: it can be slow and inaccurate compared to other pricing techniques in the literature [18]. Additionally, Monte Carlo techniques are normally required once a pricing problem becomes too complex for other methods. When calculating the Greeks this complexity brings to the table complications, like discontinuous payoffs, where the Greeks are not smooth or even extant! Although some progress has been made at improving the performance of calculating the Greeks inside the Monte Carlo framework [19], the complexity of the Libor Market Model renders most of these techniques impractical.

There are two fundamental approaches to calculating the Greeks inside the Monte Carlo framework:

- Finite difference approximations. This is certainly the simpler of the

two approaches, that naively uses the existing Monte Carlo pricing methodology and calculates the value of a derivative at several points in the underlying. A finite difference calculation then provides an approximation of the Greek. This method is discussed in Section 4.6.1.

- Simulation of Greeks. Here numerical differentiation is replaced with exact calculation. The pathwise method differentiates each simulated outcome with respect to the parameter of interest by placing the dependence on the parameter as a drift change in the process. The likelihood ratio method differentiates the derivative price by putting the change in the underlying into the probability density function. These methods are covered briefly in Section 4.6.2.

4.6.1 The finite difference method

The finite difference method is not specific to Monte Carlo simulation: it can be used with any option valuation method as long as we can recalculate the option price after varying the underlying parameter of interest. A simple finite difference approximation gives an estimate for the derivative. Our treatment of the finite difference method combines material from [18] and [29].

As it is the most fundamental Greek in the hedging of securities, let us focus on calculating delta:

$$\Delta_i = \frac{\partial V}{\partial L_i}$$

We can approximate this continuous derivative with the simplest finite difference calculation, Newton's forward-difference formula using one additional valuation at $L_i + \Delta L_i$:

$$\Delta_i \simeq \frac{V(L_i + \Delta L_i) - V(L_i)}{\Delta L_i} \tag{16}$$

A better approximation is Stirling's central-difference formula, although it requires the computation of the option value at two additional valuations

in the underlying at $L_i + \Delta L_i$ and $L_i - \Delta L_i$:

$$\Delta_i \simeq \frac{V(L_i + \Delta L_i) - V(L_i - \Delta L_i)}{2\Delta L_i} \tag{17}$$

Figure 8 provides a visual comparison of the two approximations. Notice how the central-difference formula is superior to the forward-difference formula when approximating the tangent of a curve. Although it may seem inefficient to do twice as much work to calculate two additional points for approximating delta using Stirling's formula (17) its bias and variance properties are vastly superior to that of Newton's formula (16) [18], and in addition we can reutilise the same points when calculating the major gammas using the approximation

$$\begin{aligned} \Gamma_{ii} &= \frac{\partial^2 V}{\partial L_i \partial L_i} \\ &\simeq \frac{V(L_i + \Delta L_i) - 2V(L_i) + V(L_i - \Delta L_i)}{(\Delta L_i)^2} \end{aligned} \tag{18}$$

To determine how well our approximation of delta matches the actual delta of the option we consider the bias and the variance of our approximation. The bias tells us to what error our approximation will converge, while its variance tell us how quickly we will converge there. Let us first examine the bias of our delta approximation:

$$\begin{aligned} bias_i &= \mathbb{E}[\Delta_i] - \frac{\partial V}{\partial L_i} \\ &\simeq \mathbb{E}\left[\frac{\bar{V}(L_i + \Delta L_i) - \bar{V}(L_i - \Delta L_i)}{2\Delta L_i}\right] - V'(L_i) \\ &= \frac{1}{2\Delta L_i}\mathbb{E}\left[\bar{V}(L_i + \Delta L_i) - \bar{V}(L_i - \Delta L_i)\right] - V'(L_i) \end{aligned}$$

where $\bar{V}$ is the average of the Monte Carlo simulations, and then Taylor

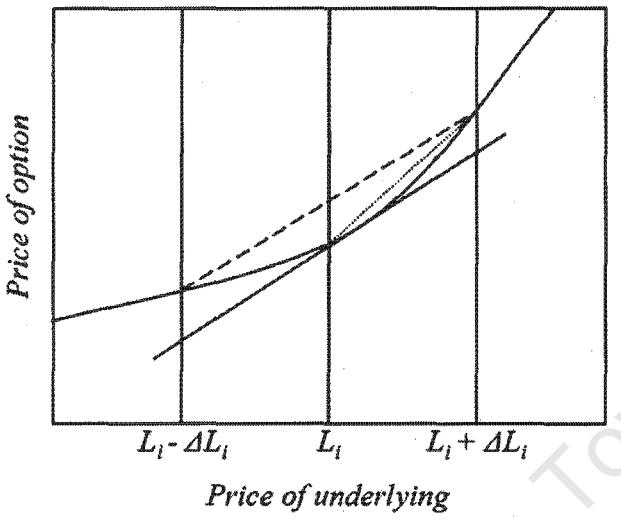

Figure 8: Comparison of Newton's (dotted) and Stirling's (dashed) approximation to the actual (solid) gradient of the curve. Notice how much better the approximation is when using Stirling's central difference formula.

expanding our values around L_i we get

$$
\begin{aligned}
bias_i &= \frac{1}{2\Delta L_i}\mathbb{E}\left[\begin{array}{c}\left(\bar{V}(L_i)+\Delta L_i'\bar{V}(L_i)+\frac{1}{2}\Delta L_i^{2\bar{V}''}(L_i)+O(\Delta L_i^2)\right)\\ -\left(\bar{V}(L_i)-\Delta L_i'\bar{V}(L_i)+\frac{1}{2}\Delta L_i^2\bar{V}''(L_i)-O(\Delta L_i^2)\right)\end{array}\right]-V'(L_i)\\
&= \frac{1}{2\Delta L_i}\mathbb{E}\left[\begin{array}{c}\bar{V}(L_i)+\Delta L_i\bar{V}'(L_i)+\frac{1}{2}\Delta L_i^2\bar{V}''(L_i)+O(\Delta L_i^2)\\ -\bar{V}(L_i)+\Delta L_i\bar{V}'(L_i)-\frac{1}{2}\Delta L_i^2\bar{V}''(L_i)+O(\Delta L_i^2)\end{array}\right]-V'(L_i)\\
&= \frac{1}{2\Delta L_i}\mathbb{E}\left[2\Delta L_i\bar{V}'(L_i)+O(\Delta L_i^2)\right]-V'(L_i)\\
&= \frac{1}{2\Delta L_i}\mathbb{E}\left[O(\Delta L_i^2)\right]\\
&= O(\Delta L_i)
\end{aligned}
$$

So we see that we can arbitrarily reduce the bias of our estimate by reducing ΔL_i, although we must remain aware of machine precision issues

[7]. Let us now consider the variance of our delta approximation:

$$\begin{aligned} variance_i &= Var\left[\Delta_i\right] \\ &\simeq Var\left[\frac{\bar{V}(L_i+\Delta L_i)-\bar{V}(L_i-\Delta L_i)}{2\Delta L_i}\right] \\ &= \frac{1}{4\Delta L_i^2} Var\left[\bar{V}(L_i+\Delta L_i)-\bar{V}(L_i-\Delta L_i)\right] \\ &= \frac{1}{4\Delta L_i^2 n} Var\left[V_n(L_i+\Delta L_i)-V_n(L_i-\Delta L_i)\right] \end{aligned}$$

The last step assumes that the V_n, which comprise the average $\bar{V}$ are i.i.d as is argued in [18]. The $\frac{1}{4\Delta L_i^2}$ term suggests that variance might become explosive as we reduce ΔL_i unless the variance term contains a ΔL_i or ΔL_i^2 term in the numerator to compensate for the ΔL_i^2 in the leading denominator. [18] gives three scenarios for the order of $Var\left[V(L_i+\Delta L_i)-V(L_i-\Delta L_i)\right]$:

- $O(1)$: If $V_n(L_i+\Delta L_i)$ and $V_n(L_i-\Delta L_i)$ are computed independently.
- $O(\Delta L_i)$: If $V_n(L_i+\Delta L_i)$ and $V_n(L_i-\Delta L_i)$ are computed using common random numbers.
- $O(\Delta L_i^2)$: If $V_n(L_i+\Delta L_i)$ and $V_n(L_i-\Delta L_i)$ are computed using common random numbers and that V_n is sufficiently smooth in L_i.

Clearly the third scenario would be optimal, but we can only guarantee the second scenario, where we use the same random numbers when calculating the value of our option at the different points of the finite difference approximation. In this case our variance is:

$$\begin{aligned} variance_i &= \frac{1}{4\Delta L_i^2 n} O(\Delta L_i) \\ &= O\left(\frac{1}{\Delta L_i}\right) \end{aligned}$$

So although we can improve our bias arbitrarily by decreasing ΔL_i, the variance of our measurement simultaneously grows as we decrease ΔL_i. So we have to trade-off between reducing bias and increasing variance. [49]

claims that market practitioners use the arbitrary choice of a 1% shift (i.e. $\Delta L_i = 0.01 L_i$). [18] gives a fairly comprehensive account of basing this trade-off on an optimal mean square error procedure. [29] describes a more compact and lucid approach, balancing bias, variance and machine precision. His choice of ΔL_i is

$$\Delta L_i = \sqrt[4]{\varepsilon} L_i$$

where ε is the smallest number that the machine can differentiate from zero (we use the C# constant `Single.Epsilon`).

It is all fair and well to naively use the finite difference method to calculate the Greeks: calculating delta and gamma are rather straightforward. However, for most option payoffs the first derivative (delta) of the payoff with respect to the underlying price is not continuous (specifically Lipschitz continuous) and hence the second derivative (gamma) may not even exist at some points (or at least behave with a Dirac delta spike). Path dependent options like the barrier options have even more ill-behaved Greeks. The finite difference method is not well equipped to detect these "spiked" deltas and gammas and can simply generate garbage results. A large value of the variance of the Greeks estimates might be useful in detecting this problem. [49] and [51] discuss how the pathwise methods (discussed in the next section) are better alternatives in these circumstances. [29] describes how to use importance sampling to focus the Monte Carlo simulations on the sample space where the Greeks are significant (e.g. in the region of the gamma spike) rather than wasting time in its usual broad evaluation of the entire sample space where the Greeks are uninteresting. This technique generally takes advantage of the structure of the process being simulated and of the structure of the derivative being priced and it is not clear how easily this might be applied to the Libor Market Model.

One might expect that having to calculate three times as many prices to generate the Greeks might cause a threefold slowdown in computational time, but this is certainly not the case. A large proportion of time in the Monte Carlo simulation is spent generating the random numbers that are used to build each simulation instance. It is rather fortunate that our estimates of

the Greeks are better when we reuse these random numbers. Experimentally we found that the computation of delta and gamma for each forward rate takes only around an additional 30% of the time it takes to price the option itself.

4.6.2 Other approaches: the pathwise method and the likelihood ratio method

Not implemented for this dissertation, but worth describing briefly are two other methods exist for estimating the Greeks of various derivative products. They attempt to produce unbiased estimates by replacing numerical differentiation with exact calculations. The pathwise method differentiates each simulated outcome with respect to the parameter of interest by placing the dependence on the parameter as a drift change in the process; the likelihood ratio method differentiates by putting the dependence in the probability density function. The two methods are covered in detail in [12]. The pathwise method seems to be the more stable of the two approaches [51] and is appreciably faster to calculate than the finite difference method [49]. Its only drawback is that it requires continuous derivative payoffs. The likelihood ratio method works well with options with discontinuous payoffs but can generate unstable estimations for the Greeks. Neither method is as general as the finite difference method, which can be applied to any derivative product. Both the pathwise method and the likelihood ratio method require some mathematics to be performed on the payoff of each derivative being priced.

5 Model details

The Libor Market Model requires several inputs: today's forward rates, forward volatilities and correlations between forward rates. These three sources of information can not simply be looked up on a market data terminal. They are instead derived using somewhat complex procedures from instruments traded in the market. Forward rates are bootstrapped from deposit rates, bonds and swaps; forward rate volatilities are estimated from caplet volatilities, which are in turn bootstrapped from traded caps; forward rate correlations can be inferred from swaption volatilities. All these choices of market data, bootstrap procedures and inference algorithms can yield vastly different input parameters for the Libor Market Model, and understandably can lead to hugely varying estimates for modelled derivative prices. Indeed [60] presents a study on how many different prices one can arrive at for the same option using different calibrations that are consistent with market data. [59] and [60] describe how successful application of the Libor Market Model lies somewhere between science and an art. This unfortunate state of affairs is summed up by [65], who states that all models he has ever seen are demonstrably wrong, but in experienced hands they are powerful pricing tools.

This section describes how the information in the market is intelligently translated into a form that is acceptable and reasonable as inputs to the Libor Market Model.

5.1 The initial forward rates

Forward rates lie at the heart of the Libor Market Model and are generally implied from the market yield curve. This yield curve is built up using a variety of techniques from the deposit rates, bond prices and swap rates that are available in the market. There are a plethora of techniques in the literature on how to bootstrap this yield curve, each with its advantages and disadvantages: [22] presents many of them in detail. On top of the choice of bootstrap methodologies to use, the implementor must choose between the "grade" of interest rate product that is used in the bootstrap. Government bonds yield lower returns than riskier corporate bonds; swaps between A-

grade counterparties carry different rates to those between less credit-worthy customers; overnight Libor rates are available only to the larger banks. Also, [59] explains how each investment bank might have its own views on the future realisations of the yield curve and so when pricing, a trader might not want to use the equilibrium rates in the market caused by supply and demand, but rather might wish to incorporate her own subjective view of the future forward rates.

Once the forward rates prevailing today have been established we can move on to describing the dynamics of the forward rates, first as solitary entities (by looking at their volatilities, see Section 5.2), and then as a group of entities that influence each others' behaviour (by looking at their correlation, see Section 5.3).

5.2 Volatility specification

Probably the most important parameters of the Libor Market Model are the volatilities of the forward rates. They play a role not only in the stochastic part of the evolution of our forward rates, but more importantly in the deterministic drift adjustments that each rate enjoys under the terminal measure. Inside the model, the specification of the instantaneous volatility $\sigma_i(t)$ for each forward rate $L_i(t)$ is pretty flexible, with the only pressing requisites being that it be deterministic and integrable (the Novikov condition of Lemma 9).

This infinite-dimensional (over continuous time) instantaneous volatility parameter allows us great flexibility in calibrating the forward rate volatilities to those found in the market. When deciding how we want to specify our forward volatilities we have to be careful that we do not give our model so much flexibility that it is easily allowed to become over-calibrated. An over-calibrated model generally performs poorly at modeling the future because it has learned the temporal (and even noisy) characteristics of the present market rather than the general time-invariant characteristics of the present market that might model the future market effectively.

The primary means of calibrating the volatilities of the forward rates

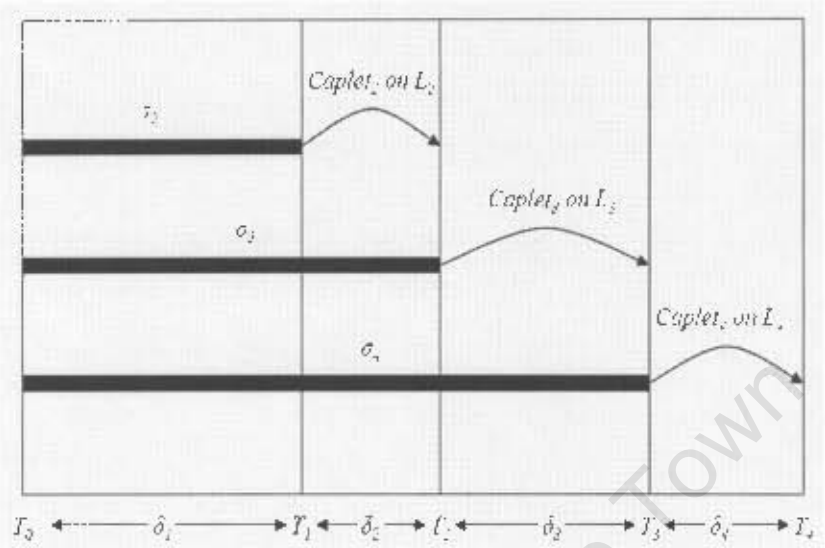

Figure 9: Hypothetical caplet market data

in the Libor Market Model is using the caplet implied volatilities that are quoted in the market. We will see how this is done in Section 5.4. For now let us focus on what information is actually provided by the market and determine a parsimonious specification for the volatility of our forward rates that might capture the essence of the volatilities of the forward rates. Figure 9 show some hypothetical caplet data that is available in the market.

We are given three caplets $caplet_i$ with expiries T_{i-1}, $i = 2 \ldots 4$, each with their own underlying forward rate L_i with implied Black76 volatility σ_i. There is no initial caplet $caplet_1$ because by definition it is a caplet on the forward rate L_1 which is the (forward) rate from time T_0 to time T_1, which is known at T_0. Now the Black76 volatility σ_i is just the "average squared volatility over time" of the underlying forward rate L_i, i.e.

$$\sigma_i^2 = \frac{1}{T_{i-1} - T_0} \int_{T_0}^{T_{i-1}} \sigma_i^2(u) du$$

The simplest specification might set $\sigma_i^2(u)$ to the constant σ_i^2, but this leads to an instantaneous volatility specification that is not time-homogeneous

[59]. This means that at some snapshot of the future, our model will describe forward rate volatilities differently than today. We have no reason to expect a forward rate today with one week to expiry to have significantly different behaviour to a forward rate in a year's time with one week to expiry. A time-homogeneous model will ensure that behaviour modelled today is the same as behaviour modelled in the future. Another problem with the constant specification is that it does not take into account the humped shape of the term structure of instantaneous volatility that is visible in the market: for long-dated caplets, a large portion of the total volatility of caplets seems to be derived in the period from six-months to two-years of its inception [57].

One method of ensuring that our specification is time-homogeneous is to parametrise it solely on the remaining time to expiry of the caplet, i.e.

$$\sigma_i(t) = f\left(T_{i-1} - t\right)$$

where T_{i-1} is the expiry of the caplet, t is the point where we are measuring the instantaneous volatility of the underlying forward rate and f is some function.

After analysing numerous sets of market data, [57] suggests the specification of

$$\sigma_i(t) = \left[a + b\left(T_{i-1} - t\right)\right] e^{-c\left(T_{i-1}-t\right)} + d \tag{19}$$

which he argues is suitable because

- it is time-homogeneous;
- it has a flexible form able to reproduce humped or monotonically decreasing instantaneous volatilities that pervade the market;
- its parameters have relatively transparent economic interpretation; and
- it has an efficient analytical representation of the square integral $\int \sigma_i(u)\sigma_j(u)du$, which is required during calibration and evaluation of the covariances during simulation

An example of the term structure of volatilities produced by (19) is shown in Figure 10. [59] provides lucid descriptions of the characteristic shape of

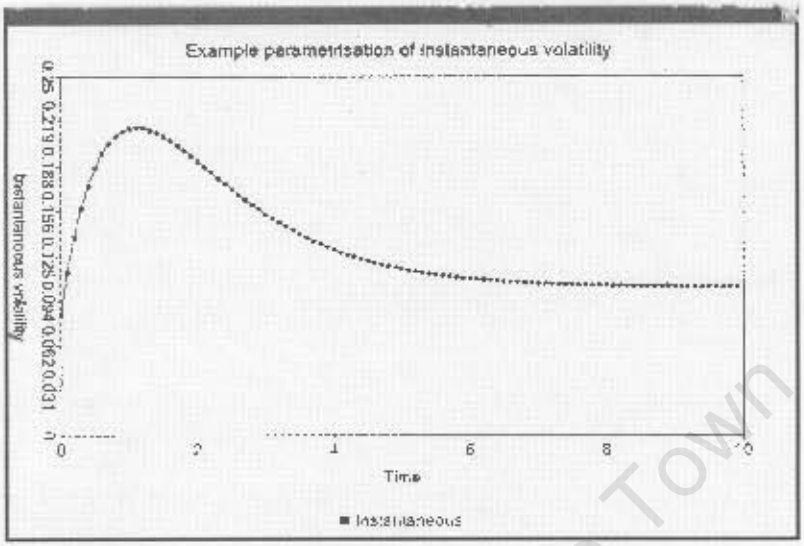

Figure 10: An example of the term structure of volatilities. Notice the characteristic "hump" around the one- to two-year period.

the market instantaneous volatilities and of the economic interpretation of the specification parameters.

The determination of the four parameters a, b, c, d are covered in Section 5.4. For completeness (and for practicality's sake), I include not the mathematical representation of the square-integral of (19), but rather the C# code for it in Section 10.1. The code was translated from the c++ code available at [43].

Although (19) provides a time-homogenous description of the volatility of the market, it is not likely to exactly fit all the market data simultaneously. When pricing certain derivatives, especially products similar to caplets (e.g. barrier caplets), it is important that the model correctly calculates the current market price of the caplets used for calibration. [59] suggests adding a touch of inhomogeneity to the specification by writing it in the form

$$\sigma_i(t) = K_i \left([a + b\,(T_{i-1} - t)]\, e^{-c(T_{i-1}-t)} + d \right)$$

where K_i is a scaling constant that allows complete calibration to the market

data. Generally speaking, if (19) provides a reasonable fit to the market data these K_i should be very close to unity, and thus the level of time-inhomogeity introduced is minimal. The addition of this constant does not substantially alter the complexity of the square-integral: it merely introduces the factor $K_i K_j$ in front of the integral.

5.3 Correlation specification

Having specified the instantaneous volatilities of the forward rates in our model we are half way to completely describing their dynamics. What remains is the specification of the correlation, and thus covariance, between the various forward rates. The covariance between the forward rates appears in the deterministic drift adjustments that each rate enjoys under the terminal measure, and has a particularly large impact of correlation sensitive instruments such as long-dated Bermudan swaptions [59].

We are faced with two problems when specifying our forward rate correlation matrix:

- Correlations are not directly visible in the market. They have either to be approximately derived from swaption prices (covered in Section 5.5), estimated by an experienced market participant or calculated from historical data. [1] describes how calibration to historical data leads to calibration that is not at all time-homogenous and is rather unstable over time.

- Being able to specify a large number of correlation coefficients may yield an over-calibrated model that may not model the future very well.

Correlations between forward rates can become quickly unmanageable: given n forward rates, there are $\frac{n(n-1)}{2}$ distinct correlation coefficients between them. For example, a long dated Bermudan swaption spanning 20 years with six-monthly cashflows spans 40 forward rates and 780 correlation coefficients! Rather than calibrate each individual correlation coefficient, we therefore seek a parsimonious specification for the forward rate correlations that adequately describes what scant information is available in the market.

Such specifications are the topic of active research [11], [8], [14], [15], [43], [59]. The model we choose is one described in [59]. It has the form:

$$\rho_{ij} = \alpha + (1 - \alpha)\, e^{-\beta|T_i - T_j|} \tag{20}$$

where α is a long term minimum correlation between two forward rates and β describes the sensitivity of the correlation between two forward rates to the difference between their expiries. [32] recommends the parameters $\alpha = \beta = 0.1$ as being reflective of the UK market. Under this model two forward rates with nearby expiries (e.g. 1-year and 2-year) will have a high correlation, while two forward rates with greatly different expiries (e.g. 1-year and 10-year) will have a low correlation. This seems in keeping with the market [59]. Some authors argue that the correlation between the 1- and 2-year forward rates should be lower than that of the 19- and 20-year forward rates [8] because market participants have more strongly conflicting views about the short end of the yield curve than the long end, but [31] and [59] defend it. An example of the correlation matrix produced by (20) is shown in Figure 11, where the floor axes are the expiries of each of the underlying forward rates and the vertical axis is the correlation between them.

We have chosen this correlation specification because of its concise formulation and because its financial implications can be readily explained. The downside of our choice is that this specification leads to a multi-factor Libor Market Model that has as many factors as there are forward rates. Section 5.6 describes why this might be unreasonable and shows how we can reduce this number of factors. Alternative specifications are presented in [8], [14] and [59] where correlation matrices are produced that implicitly have a small number of factors, but they lose the intuitive relation of their specifications to financial reality.

5.4 Calibration to caplets

We begin the calibration of our Libor Market Model to the market by ensuring that it correctly prices a set of caplets available in the market. We showed in Section 3.4.1 how the market arrives at the Black76 implied volatil-

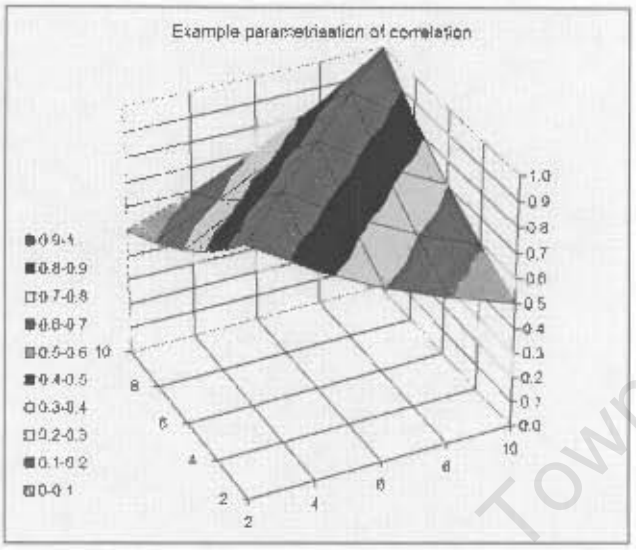

Figure 11: An example of the parametrisation of the correlation between forward rates. The floor axes represent the year of expiry of the forward rates and the vertical axis their correlation.

ity for a caplet. Ideally we would like to find in the market a set of caplets that span the time periods over which we wish to price more complex derivatives using our model. One generally will not go to the market to find the market prices for the various caplets as these prices are likely to be biased (the spread and skews that the traders add to customer facing prices) or stale (if they are illiquid and seldom traded). Generally speaking, these implied market volatilities will be provided in-house by the interest rate traders [10], probably stripped from their cap prices combined with personal adjustments applied to cap volatilities. [9] describes how to perform this stripping.

Having collected the implied (Black76) volatilities $\bar{\sigma}_i$ for N spanning caplets expiring at T_{i-1} and paying out at T_i, our task is simply to find the combination of parameters a, b, c, d of Equation (19) that minimises the objective function

$$error = \sum_{i=1}^{N} \left(\bar{\sigma}_i^2 T_{i-1} - \int_0^{T_{i-1}} \sigma_i^2 (u; a, b, c, d)\, du \right)^2$$

The location of the optimal parameters requires an iterative global optimisation method, several of which are described in [54]. We use the Nelder-Mead global search algorithm that is described in Section 6.3.

Once we have discovered the optimal set of parameters a, b, c, d we have calibrated the continuous specification for the instantaneous volatility such that

$$\bar{\sigma}_i^2 T_{i-1} \simeq \int_0^{T_{i-1}} \sigma_i^2(u; a, b, c, d)\, du$$

Note that it is unlikely that we will have exact parametrisations for the market volatilities so we define K_i such that

$$\bar{\sigma}_i^2 T_{i-1} = K_i^2 \int_0^{T_{i-1}} \sigma_i^2(u; a, b, c, d)\, du$$

i.e.

$$K_i = \sqrt{\frac{\bar{\sigma}_i^2 T_{i-1}}{\int_0^{T_{i-1}} \sigma_i^2(u; a, b, c, d)\, du}} \tag{21}$$

Section 7.1 shows the results of our calibrating to caplets using this technique.

5.5 Calibration to caplets and swaptions

The previous section showed how to calibrate the Libor Market Model to the implied volatilities of caplets in the market. That calibration ensures that our model correctly prices the caplets. This should then ensure that the prices of more complex derivatives are realistic - especially those that have very similar characteristics to caplets. Unfortunately there are many important interest rate derivative products that rely on not only the instantaneous volatility of the underlying forward rates but also (and generally more importantly) on the correlations between them. The instrument of this form that has received the most attention is the Bermudan swaption [2], [36] and

[59].

One might expect that the various swap market models (in contrast to the Libor Market Model) might be better suited to modelling derivatives that exhibit features that are more similar to swaptions than to forward rate derivatives, but [59] argues that the Libor Market Model is a better modelling framework for reasons of simplicity, understandability and efficiency. In addition, being able to calibrate the Libor Market Model simultaneously to caplets and swaptions allows us to automatically price derivatives that employ features from both markets.

As was seen in Section 5.3, our specification of the correlation between the forward rates was simply a time-homogenous specification that was monotonically decreasing on the absolute difference between forward rate expiries. Other than the broad market calibration of the parameters α and β, no attempt was made to capture any more information that might be available in the market. This section motivates an approach that ensures that at least the market price of swaptions is correctly determined by our Libor Market Model because a large number of derivative products behave quite similarly to these swaptions. We showed in Section 3.4.2 how the market arrives at the Black76 implied volatility for a swaption.

Now any attempt to jointly calibrate to the market prices of caplets and swaptions assumes that their prices are mutually consistent. Both [40] and [57] describe where this assumption is sometimes violated. Generally speaking, swaptions are priced higher than they should be because the market requirement for them is very one-sided (everyone is buying), so we rely heavily on the information provided by the market caplet prices and adjust only slightly to the information provided by the market swaption prices.

We combine the methodologies presented in [33], [35], [58] and [32].

From (3) we have the swap rate

$$\begin{aligned} SR_i &= \sum_{j=i}^{N} w_{i,j} L_j \\ \implies dSR_i &= \sum_{j=i}^{N} (dw_{i,j} L_j + w_{i,j} dL_j + dw_{i,j} dL_j) \end{aligned}$$

[59] gives the compelling argument that $w_{i,j}$ remains pretty much constant for first-order (roughly parallel shifts) and second-order (more complex shape changes) changes to the yield curve and that these changes are most prevalent in practice. If we make the assumption that $w_{i,j}$ is constant then the $dw_{i,j}$ terms fall away, leaving

$$dSR_i = \sum_{j=i}^{N} w_{i,j} dL_j$$

So we know we can write our swap rates in terms of our forward rates. Now a fundamental assumption of the Libor Market Model is that the forward rates have lognormal dynamics. [30] and [56] show that if we assume lognormal dynamics for the forward rates then the dynamics of the swap rates can not be lognormal. Indeed not only are the swap rate dynamics not lognormal, but their drift and volatility terms are stochastic [56]. To proceed with our calibration to swaptions we make the simplifying assumption that the swap rates have lognormal dynamics. [34] and more recently [35] describe how accurate this assumption is in practice.

If we assume the forward rates and swap rates have lognormal dynamics, then we can attempt to write the swap rate volatilities in terms of the forward rate volatilities. Focussing only on the volatility terms we can write the dynamics for $\ln SR_i$ under the terminal measure $\mathbb{P}_{T_N}$:

$$\begin{aligned} dSR_i &= \sigma_i^{SR} SR_i dW^{T_N} + (\cdots) SR_i dt \\ \frac{dSR_i}{SR_i} &= \sigma_i^{SR} dW^{T_N} + (\cdots) dt \\ d\ln SR_i &= \sigma_i^{SR} dW^{T_N} + (\cdots) dt \end{aligned}$$

We can write the equivalent expression for the dynamics of $\ln SR_j$ in

terms of forward rates as,

$$\begin{aligned} d\ln SR_i &= \sum_{j=i}^{N} \frac{\partial \ln SR_i}{\partial \ln L_j} d\ln L_j - (\cdots)\, dt \\ &\simeq \sum_{j=i}^{N} \frac{\partial \ln SR_i}{\partial \ln L_j} d\ln L_j \\ &= \sum_{j=i}^{N} \frac{L_j}{SR_i} \frac{\partial SR_i}{\partial L_j} d\ln L_j \\ &= \sum_{j=i}^{N} Z_i d\ln L_j \end{aligned}$$

where

$$\begin{aligned} Z_{ij} &= \frac{L_j}{SR_i} \frac{\partial SR_i}{\partial L_j} \\ &\simeq \frac{L_j}{SR_i} \times w_{i,j} \end{aligned} \tag{22}$$

and

$$w_{i,j} = \frac{\delta_j P(T_j)}{\sum_{k=i}^{N} \delta_k P(T_k)} \tag{23}$$

We wish to determine the relationship between the covariance matrices of swap rates and the Libor forward rates, so

$$\begin{aligned} \mathbb{E}[d\ln SR_i d\ln SR_j] &= \mathbb{E}\left[\sum_{m=i}^{N} Z_{im} d\ln L_m \sum_{n=j}^{N} Z_{jn} d\ln L_n\right] \\ &= \mathbb{E}\left[\sum_{m=i}^{N}\sum_{n=j}^{N} Z_{im} d\ln L_m d\ln L_n Z'_{nj}\right] \\ \Longrightarrow C^{SR} &= ZC^L Z' \end{aligned}$$

where C_{SR} and C^L are the covariance matrices for the swap rates and the Libor forward rates, respectively.

If we consider (22) we see that Z_{ij} is upper triangular: $Z_{ij} = 0$ for $j < i$

(since the swap rate does not depend on the forward rates before its initial cashflow) and $Z_{ij} \neq 0$ for $i = j$ (since the swap rate is certainly dependent on the forward rate of its reset date). Thus Z is invertible and we can move freely between the two covariance matrices using

$$C^{SR} = ZC^{L}Z' \tag{24}$$

$$C^{L} = Z^{-1}C^{SR}\left(Z'\right)^{-1} \tag{25}$$

Note that the specification of Z by (22) is a continuous time specification. More interestingly, it is stochastic because it contains L_j. This means that the covariance matrix for our swap rates is also stochastic. To avoid the theoretical complexity of stochastic covariances and the computational complexity of calculating Z at each timestep we make the further approximation of using the $T = 0$ value of Z throughout the evolution of our forward rates and swap rates. [35] gives evidence that this approximation is reasonable and describes its (occasional) shortcomings. Thus we use

$$C^{SR}(0, T_j) = Z(0)C^{L}(0, T_j)Z'(0)$$

We are now in a position to calibrate to the market swaption prices. If $\hat{\sigma}_j$ is the market implied volatility of swaption j, then it should be equal to

$$\sigma_j = \sqrt{\frac{1}{T_j}C^{SR}_{jj}(0, T_j)}$$

It is unlikely that the market volatility $\hat{\sigma}_j$ matches our model volatility σ_j, so we set

$$\lambda_j = \frac{\hat{\sigma}_j}{\sigma_j}$$

and

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \lambda_n \end{bmatrix}$$

Our swaption calibrated Libor covariance matrix $\hat{C}^L$ is then

$$\hat{C}^L(0,T_j) = Z(0)^{-1}\Lambda Z(0)C^L Z(0)'\Lambda\left(Z(0)'\right)^{-1}$$

Although this method works well for most yield curves, both [32] and [33] suggest improvements in the approximation of the $w_{i,j}$ for very steep yield curves. Note that this method of calibration to swaption volatilities makes no mention of the covariance between the swap rates. The correlations between the swap rates and the forward rates are determined only by the correlations between the forward rates, with a little influence creeping in from the calibrated swaption volatilities. [32] notes that this is not completely unnatural: coterminal swap price covariances are not something that are readily visible in the market.

The method of calibrating to swaption volatilities used in this dissertation is by no means the only approach covered in the literature: [14], [15] and [59] present other approaches that they feel have more compelling justification.

5.6 Factor reduction

As was explained in Sections 4.2 and 5.3, we have implemented a multi-factor Libor Market Model. That means that our forward rates react in different ways to several sources of noise through their sensitivity (expressed via their volatilities $\sigma_j(t)$) to each dimension of the Brownian motion $dW^{T_j}(t)$. [32] shows how several factors are indeed necessary to adequately capture the relationship between the various forward rates and correctly price derivatives such as Bermudan swaptions that are highly dependent on the correlation structure between the forward rates. That said, the model we have implemented so far has as many factors as there are forward rates and [32] notes that there is no noticeable improvement to pricing using more than three or four factors. We really are performing unnecessary work and this can slow down computation almost linearly in the number of factors because we have to generate a random number for each of the factors and update the forward rates according to their sensitivities to each factor.

This section describes a method of reducing the number of factors represented by our correlation matrix by picking out the "most important features" of the correlation specification. There are many approaches in the literature to deciding exactly what we mean by the "most important features" [21], [32] and [59]. We implement the method in [32] that builds a correlation matrix by retaining only the eigenvectors corresponding to the largest eigenvalues of the original correlation matrix.

Assume that we have a correlation matrix C of the form described in Section 5.3. We find the N eigenvalues λ_i and corresponding eigenvectors e_i of matrix C: our correlation matrices are real and symmetric and so can be quickly reduced to Householder form and the eigensystem efficiently solved using the QL algorithm [54]. The matrix R with N columns $\sqrt{\lambda_i}e_i$ is a pseudo-square root of C. Let $R^{(M)}$ be the matrix whose first M columns are the $\sqrt{\lambda_i}e_i$ corresponding to the largest (in absolute value) eigenvalues λ_i and remaining $N - M$ columns set to zero. Then $R^{(M)}$ is the M-factor pseudo-square root of C, and $\tilde{C}^{(M)} = R^{(M)}\left(R^{(M)}\right)^T$ is a matrix with only M factors. Note that $\tilde{C}^{(M)}$ will not be a correlation matrix because the diagonal elements are not likely to be unity. It will, however, be symmetric and so we can turn it into the correlation matrix $C^{(M)}$ by setting

$$C_{ij}^{(M)} = \frac{\tilde{C}_{ij}^{(M)}}{\sqrt{\tilde{C}_{ii}^{(M)}\tilde{C}_{jj}^{(M)}}}$$

$C^{(M)}$ is then the M-factor correlation matrix we use in our Monte Carlo simulations exactly as we would have used C before. We will see the effect that factor reduction has on option prices in Section 7.4.

6 Technical details

6.1 Monte Carlo integration

Ideally, we would like to be able to solve the forward rate dynamics (13) analytically so that we might have a closed form solution $L_j(t)$ at any time $t \in [0, T_N]$. Unfortunately (13) is far too complex to solve analytically, and the problem becomes worse when we wish to calculate the expected value of derivatives with optionality, as we then need also to know the distribution of the $L_j(t)$.

To this end we have to turn to numerical methods to calculate our derivative prices. These numerical methods include the various tree techniques, finite difference PDE methods and Monte Carlo technique [66]. The tree techniques and finite difference PDE methods work well for problems with low dimensionality, but once the number of dimensions becomes too large their computation time becomes unreasonable - the so-called curse of dimensionality [66]. The method of choice for solving problems with complicated dynamics and high-dimensionality is the Monte Carlo technique [18].

Let us examine the mathematics of the Monte Carlo technique. We should like to examine how our Monte Carlo approximations converge to the actual solution. The mathematics is covered in more detail in [29].

We use Monte Carlo simulations in finance to calculate the expectation of various derivative payoffs $f(x)$, under generally complex distributions of underlying processes $\psi(x)$, over some domain $\mathcal{D}$.

$$\begin{aligned} V &= \mathrm{E}_{\psi(x)}\left[f(x)\right] \\ &= \int_{x \in \mathcal{D}} f(x)\psi(x)dx \end{aligned}$$

The Monte Carlo approximation to the above expectation is simply

$$\hat{V}_N = \frac{1}{N}\sum_{i=1}^{N} f(x_i) \text{ where } x_i \sim \psi(x)$$

If the sequence of random variates $f(x_i)$ has mean μ and variance σ^2,

then the Central Limit Theorem [29] tells us that

$$\hat{V}_N \longrightarrow \mathcal{N}\left(\mu, \frac{\sigma}{\sqrt{N}}\right)$$

Although we do not know the variance of $f(x_i)$ we can combine the Central Limit Theorem and the Continuous Mapping Theorem [29] to use the variance of our Monte Carlo simulation as an estimate for σ^2:

$$\hat{\sigma}_N = \sqrt{\left(\frac{1}{N}\sum_{i=1}^{N} f(x_i)^2\right) - \left(\frac{1}{N}\sum_{i=1}^{N} f(x_i)\right)^2}$$

We define the standard error ϵ_N as

$$\epsilon_N = \frac{\hat{\sigma}_N}{\sqrt{N}}$$

Note that the standard error does not provide a guaranteed range inside which the true answer lies; it is a statistical measure. This means that there is a chance that the simulation answer lies outside the range of the standard error. We use the standard error to help determine when to stop the simulation and some give some indication of the error of the result. But it is up to the user of the model to decide on the convergence of the simulation, generally by inspecting the "average-so-far" as the simulation progresses. Figure 12 shows an example of the convergence of the Monte Carlo simulation. Notice how the price starts off erratically but after several thousand iterations converges towards a solution. Also note the inverse square-root behaviour of the standard error: it becomes increasingly difficult to further reduce the standard error as the number of iterations increases.

There exists a vast literature on the various techniques that can be applied to increase the convergence speed of the Monte Carlo method. [7] covers these in some detail. We will make explicit use only of antithetic variables in Section 6.2.3 and implicit use of moment matching with Sobol sequences in Section 6.2.2. Control variates are a powerful mechanism for reducing simulation variance but are generally specific to the product being priced,

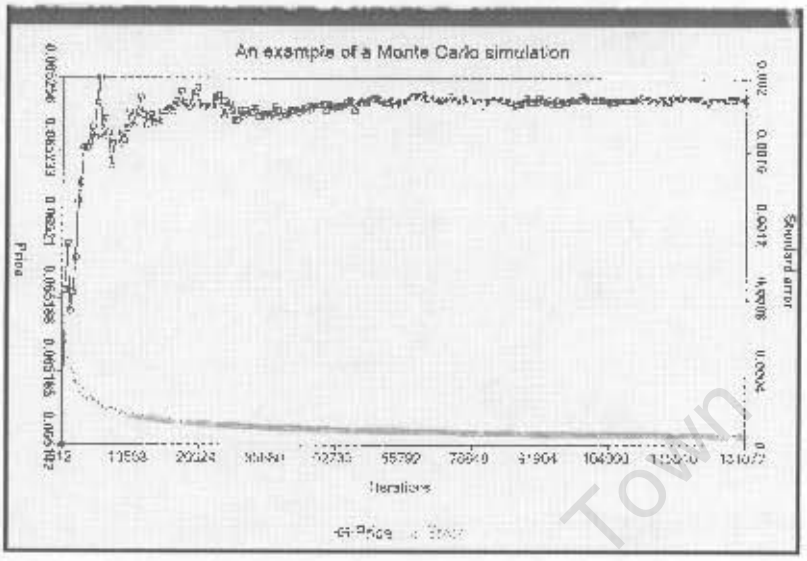

Figure 12: An example of a Monte Carlo simulation. Notice that the price converges towards a solution as the standard error drops with the inverse square-root of the number of iterations.

and so are not covered in this dissertation.

6.2 Random numbers

This section describes the various mechanisms we have at our disposal for generating the "random" numbers for the Brownian factors in our Monte Carlo simulations. Excluding custom-built hardware for generating truly random numbers, there is no such thing as a computer-generated random number: the computer has to follow an algorithm to generate these sequences, and any algorithm is deterministic and can be repeated. We cover the generation of two broad categories of random numbers: pseudo- and quasi-random. The former attempts to mimic truly random numbers by using interesting properties of mathematical constructs and is presented in Section 6.2.1; the latter attempts to cover a domain as comprehensively as possible while maintaining the same distribution as a random number and is presented in Section 6.2.2. Section 6.2.3 then explains how a pseudo-random source can be made

to better cover a domain through the use of antithetic variables. Finally, Section 6.2.4 discusses how uniformly distributed numbers are converted to numbers from the Gaussian distribution: we need Gaussian draws to simulate Brownian motion.

6.2.1 Pseudo-random numbers

Pseudo-random numbers are produced by deterministic algorithms to form sequences that attempt to imitate the distribution of truly random numbers. As was said before, there is no such thing as a computer-generated truly random number. Instead the inventors of the random number generators rely on the ingenious manipulation of mathematical properties of numbers to extract numbers that appear random. Although most computer languages come with libraries for generating uniformly distributed pseudo-random numbers, both [7] and [54] give compelling reasons for why we should implement a robust pseudo-random number source for Monte Carlo simulation. The most compelling is probably that some platform provide random number generators that are hardly acceptable at all: they may produce numbers that are not uniformly distributed; sequences that repeat after only several thousand draws; or sequences that exhibit undesirable patterns in high dimensions. Another useful reason is that if the pseudo-random number generator is specified, we can be sure that we will generate exactly the same sequence of numbers across two Monte Carlo runs if we wish to. This can be helpful for debugging and calculating prices and the Greeks in a distributed computing environment, where multiple computers might need to use the same set of random numbers.

We implement the Mersenne Twister algorithm as presented in [41]. [29] describes it as being the best available, both in terms of speed and in terms of satisfying the properties we would like from a Monte Carlo random number source. The code for the algorithm is presented in Section 10.4.

6.2.2 Quasi-random numbers

Although the Mersenne Twister random numbers discussed in the previous section provide a reliable source of uniformly distributed pseudo-random numbers, it has been found that for Monte Carlo integration techniques, better results are achieved using uniformly distributed quasi-random numbers that better cover the region of integration [7]. Two of the most established sources of these quasi-random numbers are the Niederreiter [47] and Sobol [61] sequences. [29] argues that the Sobol sequences are better than Niederreiter sequences for high-dimensional problems as long as they have been correctly initialised. [49] also gives recommendations of how the Brownian bridge should be used with high-dimensional Sobol sequences to avoid any problems with the higher dimension sequences becoming coplanar.

The next section describes the implementation of antithetic variables. By their nature, Sobol sequences have a degree of implicit antithetics (and moment matching for that matter) as long as you take care to run your Monte Carlo simulation with a number of iterations that is a power of 2.

Note that if computation speed is important in pricing many different derivatives in a portfolio, both the pseudo-random and quasi-random numbers can be precalculated and used again and again.

6.2.3 Antithetics

The variance of a Monte Carlo simulation can be reduced by producing negative correlations between the numerous paths generated during the simulation. The simplest approach to producing these negatively correlated paths is the method of antithetic variables. It relies on the fact that if random variable U is uniformly distributed then so is $\hat{U} = 1 - U$. This translates to Gaussian draws using the technique in section 6.2.4 that have the same magnitude but opposite signs, i.e. $Z \equiv F^{-1}(U) = -F^{-1}(\hat{U}) \equiv -\hat{Z}$, where F^{-1} is the inverse transform of the Gaussian distribution. This in turn translates to antithetic sequences of random draws $(Z_1, \ldots Z_N)$ and $(\hat{Z}_1, \ldots \hat{Z}_N)$ that produce antithetic simulations of the Libor rate using equation 15.

These antithetic simulations of the Libor rate, L and $\hat{L}$ should be nega-

tively correlated: e.g. a large upswing in L will be associated with a large downswing in $\hat{L}$. This negative correlation should be reflected in the pricing of the options V and $\hat{V}$ for that particular simulation instance. It should be noted that (i) the Libor discretisations have a very complex drift term that depend on factors other than the Libor rate and (ii) the option prices might depend non-linearly on the instance of the simulated Libor rates. This might mean that the resulting option prices are less negatively correlated than we would expect them to be.

[18] elucidates when the use of antithetics is useful. Suppose that it takes twice as long to produce the pair of option prices $(V, \hat{V})$ as it does to price V alone. Then we can produce half as many antithetic simulation pairs as normal simulations in the same amount of time. So using antithetics reduces variance if

$$\begin{aligned}
Var\left[\hat{V}_{ave}\right] &< Var\left[V_{ave}\right] \\
\implies Var\left[\frac{1}{n}\sum_{i=1}^{n}\left(\frac{V_i+\hat{V}_i}{2}\right)\right] &< Var\left[\frac{1}{2n}\sum_{i=1}^{2n}V_i\right] \\
\implies \frac{1}{4n^2}Var\left[\sum_{i=1}^{n}\left(V_i+\hat{V}_i\right)\right] &< \frac{1}{4n^2}Var\left[\sum_{i=1}^{2n}V_i\right] \\
\implies Var\left[\sum_{i=1}^{n}\left(V_i+\hat{V}_i\right)\right] &< Var\left[\sum_{i=1}^{2n}V_i\right] \\
\implies Var\left[V+\hat{V}\right] &< 2Var\left[V\right] \quad \text{(since there are } 2n \text{ i.i.d variables)} \\
\implies Var[V]+Var[\hat{V}]+2Covar[V,\hat{V}] &< 2Var\left[V\right] \\
\implies 2Var[V]+2Covar[V,\hat{V}] &< 2Var\left[V\right] \quad \text{(since } V,\hat{V} \text{ have the same distribution)} \\
\implies Covar[V,\hat{V}] &< 0
\end{aligned}$$

Thus as long as the covariance of our antithetic simulations is negative, we will get a better reduction in variance using antithetic variables than by doubling the number of simulations. This negative covariance might not always be extant, in which case the use of antithetic variables may actually *increase* the variance of the simulation.

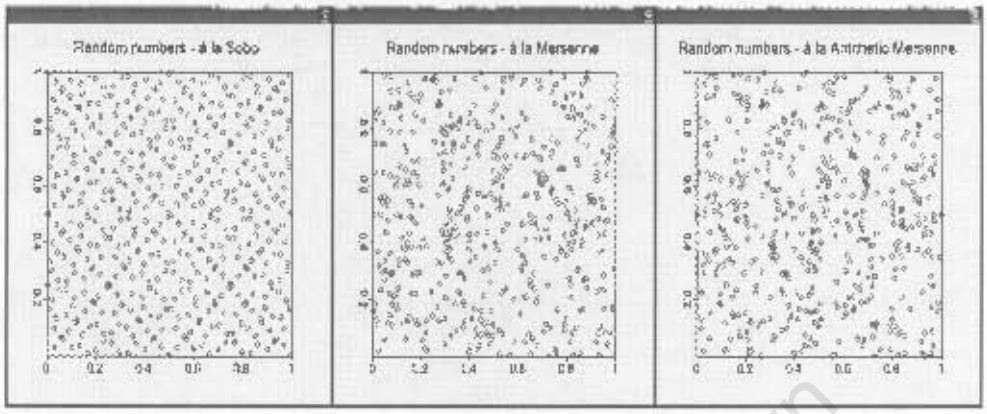

Figure 13: Examples of the various random number sources. From left to right: Sobol sequence; Mersenne Twister; and Mersenne Twister with antithetic variables.

As mentioned in the previous section, Sobol sequences automatically includes the concept of antithetic variables.

6.2.4 Gaussian draws

[64] discusses the care that should be taken when generating random numbers from the Normal distribution using uniform draws, especially when it comes to the translation of points near the extremes of the distributions. We have selected Moro's algorithm for transforming the "uniformly distributed" numbers between 0 and 1 (generated using the Mersenne Twister or via Sobol sequences) into Normally distributed numbers as it is reportedly accurate to 15 decimal places, even out in the tails of the distribution! I should like to thank Dr. Graeme West for his c++ version of the Moro algorithm, whose C# translation is included in Section 10.2.

6.3 Global optimisation - Nelder-Mead algorithm

During the process of the calibration of the Libor Market Model to caplet market prices, as described in Section 5.4, we need to determine the parame-

ters a, b, c, d that minimise the sum of squares error

$$error = \sum_{i=1}^{N} \left(\bar{\sigma}_i^2 T_{i-1} - \int_0^{T_{i-1}} \sigma_i^2 (u; a, b, c, d)\, du \right)^2$$

This error surface is a highly non-linear surface in five-dimensions (four parameter variables plus one objective value) with many local minima. The calibration process has two broad objectives:

- The first and most important is to locate the a, b, c, d that gives the smallest *error* value;
- The second, and of less importance, is to locate the a, b, c, d that not only give a small *error* value, but which are also quite stable around their target: generally we would like our calibration to be robust enough that a small change in one of the $\bar{\sigma}_i^2$ does not significantly alter the optimal parameters.

[54] dedicates an entire chapter to the various optimisation techniques available in the literature and the Nelder-Mead algorithm was chosen for our calibration process for the following reasons:

- It is a stochastic global search algorithm designed to locate a global minimum on highly non-linear surfaces with high probability. No global optimisation algorithm can guarantee an optimal solution [54]. Owing to the random nature of the algorithm, chance of locating the global minimum the procedure can be increased by repeating the search from different initial configurations.
- It is easily scalable to problems of high dimension, so can be tested visually and intellectually in two or three dimensions and then scaled up to the 4 dimensions we require.
- It does not require the gradient (first derivative) of the objective function as do some of the stochastic steepest-descent algorithms.

- Due to the random nature of the algorithm, it is most likely to find the floor of the larger, flatter basins in complex search spaces, which translates into optimal and stable results.
- Although it is not the fastest of the global search algorithms, it is relatively simple to implement, and it is more than adequate for the task of calibration, as will be seen in Section 7.1.

The working of the algorithm is rather straightforward. Please refer to Figure 14 during the discussion.

1. An optimisation in N dimensions requires $N + 1$ sample points, so we have $N + 1$ sets of points (a, b, c, d).
2. Initialise these points randomly inside the search space, ensuring that no three points are colinear.
3. Repeat the following steps until the $N + 1$ sample points are close enough together, i.e. inside the precision wanted from the optimisation.
   (a) Evaluate the objective function at each of the sample points and sort them.
   (b) Transform A: Move the worst of the sample points (always point 3 in Figure 14) to its reflection in the plane through the remaining points. The motivation is that we want to move as far away (in a stable fashion) from our worst point as possible. If this yields an improvement in the objective function, keep the new point and try Transform B. Otherwise try Transform C.
   (c) Transform B: Double the reflection jump that was done in Transform A. The idea is that if Transform A found an improvement, then it is likely that there will be an even bigger improvement if we reach out a little further out. If we found a better point still, keep it. We are done with this step, so jump back to the top of the loop.

(d) Transform C: Move the worst of the sample points halfway to the plane through the remaining points. We are doing this step only if Transform A failed, which means that a minimum lies somewhere between the worst point and the other points. If the objective function is indeed reduced by this point, keep it and jump back to the top of the loop. If there is no improvement, do Transform D.

(e) Transform D: Move all the points but the best point towards the best point. This transform happens only if the other three transforms have failed to locate a better point. In this case, we want to close in on the best point. Jump back to the top of the loop.

The Nelder-Mead algorithm can we easily modified with constraints on the underlying parameters. After each transform has been performed, make sure that the transformed point is inside the search domain. If it is not, bump it into the search domain. Always ensure that no three points become colinear because otherwise it reduces the number of dimensions inside which the algorithm is able to search.

The C# code for the Nelder-Mead algorithm is given in Section 10.3.

6.4 Interest rate derivative products

The Libor Market Model can be used to price any instrument whose payoffs can be decomposed into a finite set of forward rates [32]. Ideally, these payoffs should occur at the same time as the expiry of liquid interest rate products (bonds, swaps, caplets and swaptions) so that we glean from them accurate descriptions of the forward rates around the payoff (in terms of the initial forward rates, their volatilities and their correlations). If a payoff falls on some arbitrary date we can still interpolate these parameters approximately from known neighbours [22], [59].

The model as we have implemented it requires that all interest rate derivative products implement the following simple interface:

```
void priceAgainstCurve(Matrix Li, Matrix Pi, Vector delta, Vector cashflows)
```

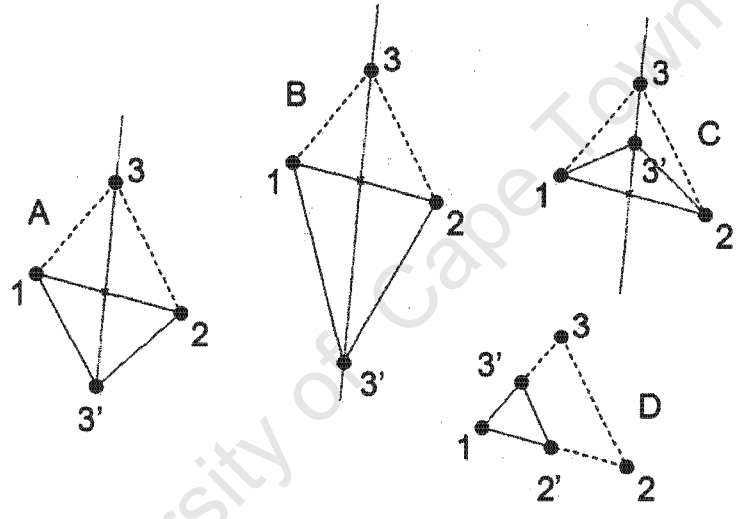

Figure 14: The four transforms of the 2D Nelder-Mead algorithm. Under Transforms A, B & C the worst point (labelled 3) is moved to point 3' in an attempt to find an improvement in the error function. Under Transform D all points but the best are moved towards the best point.

The input variables `Li` and `Pi` contain matrices of the forward rates and discount bonds of the form shown in Figure 7. The input variable `delta` contains a vector of the length of the periods between each of the rates as shown in Figure 1. The output variable `cashflows` is where the product implementation code must insert the payoffs corresponding to each tenor: e.g. `cashflows[2]` should be assigned the cashflow occurring at time T_2.

The developer of a new interest rate derivative does not have to worry about how these cashflows are taken forward to the numeraire tenor and then discounted back to T_0: that is all covered by the pricing engine itself. All the developer must do is calculate the cashflow at each tenor and assign it to the `cashflows` vector.

We now examine a few interest rate derivatives on the market and implement the code to price their cashflows.

6.4.1 Caplet

Probably the most straightforward interest rate derivative to price using the Libor Market Model is the caplet. To price $caplet_i$ expiring at T_{i-1} with strike K on the forward rate $L_i(T_{i-1})$ and payoff $\delta_i \left(L_i(T_{i-1}) - K\right)^+$ at T_i we use the code:

```
void priceAgainstCurve(Matrix Li, Matrix Pi, Vector delta, Vector cashflows)
{
  cashflows[i] = delta[i] * Math.Max(0.0, Li[i-1,i] - strike);
}
```

6.4.2 Cap

A cap is just a sequence of caplets. To price the cap containing caplets $caplet_i$ to $caplet_j$ with strike K we use the code:

```
void priceAgainstCurve(Matrix Li, Matrix Pi, Vector delta, Vector cashflows)
{
  for (int t = i; t <= j; ++t)
  {
```

```
    cashflows[t] = delta[t] * Math.Max(0.0, Li[t-1,t] - strike);
  }
}
```

6.4.3 Limit cap and chooser limit cap

A limit cap is the same as a cap in that it is comprised of a sequence of caplets, but only the first M caplets that are in the money pay out: the rest expire worthless. To price the limit cap containing caplets $caplet_i$ to $caplet_j$ with strike K we use the code:

```
void priceAgainstCurve(Matrix Li, Matrix Pi, Vector delta, Vector cashflows)
{
  int limit_count = 0;

  for (int t = i; t <= j; ++t)
  {
    if (Li[t-1,t] - strike > 0)
    {
      ++limit_count;
    }

    if (limit_count <= limit)
    {
      cashflows[t] = delta[t] * Math.Max(0.0, Li[t-1,t] - strike);
    }
  }
}
```

The chooser limit cap allows the holder of the option to choose at each caplet expiry whether or not she wishes to use the expiring caplet towards her limit. This American feature adds complications that are outside the scope of this dissertation. To see some approaches to pricing American options using Monte Carlo techniques and the Libor Market Model see [32] and [55].

6.4.4 Barrier caplet

A barrier caplet is a path-dependent caplet that has either: a knockout feature that causes the option to immediately terminate if the underlying reaches a specific barrier level; or a knock-in feature that causes the option to pay out only if the underlying has reached a specified barrier level during the lifetime of the option. It must be carefully stated in the option contract how often the underlying is monitored for a breach of the barrier. Due to the contingent nature of the option they tend to have lower prices than their corresponding vanilla options.

To price a discrete knock-out barrier caplet with inspections of the underlying for knock-out only on tenor dates we use the code:

```
void priceAgainstCurve(Matrix Li, Matrix Pi, Vector delta, Vector cashflows)
{
    bool barrier_breached = false;

    // Check for a breach in barrier
    for (int j = 0; j < i; ++j)
    {
        if (Li[j,i+1] < down_barrier)
        {
            barrier_breached = true;
            break;
        }
    }

    // Define the payoffs only if the barrier was not breached
    if (!barrier_breached)
    {
        cashflows[i] = delta[i] * Math.Max(0.0, Li[i-1,i] - strike);
    }
}
```

Up to now we have priced only derivatives that work directly with the forward rates and look and behave very much like the vanilla caplet. We now move onto a set of products that work with swap rates.

6.4.5 Swaption

The formula for determining a swap rate in terms of the forward rates is (3). The time T_i payoff a swaption on a swap with cashflows from time T_i to time T_N is just the positive part of the value of the swap at time T_i minus the strike price.

```
void priceAgainstCurve(Matrix Li, Matrix Pi, Vector delta, Vector cashflows)
{
    int N = delta.cols;

    // Calculate the swap rate
    double w_denominator = 0.0;
    for (int k = first_included_time; k < N; ++k)
    {
        w_denominator += delta[k]*Pi[first_included_time,k];
    }

    // Loop through the time periods that comprise the swap
    double swap_rate = 0.0;
    for (int j = first_included_time; j < N; ++j)
    {
        double w_numerator = delta[j] * Pi[first_included_time,j];
        swap_rate += w_numerator / w_denominator * Li[first_included_time,j];
    }

    cashflows[first_included_time] = Math.Max(0.0, swap_rate - strike);
}
```

6.4.6 Trigger swap

The trigger-swap provides interest rate protection while ensuring full benefit if interest rates move in a desired direction. A trigger-swap is an agreement whereby the borrower pays the lower of the Libor rate (plus some margin) or a specified fixed level. By entering into a trigger swap, the borrower may benefit from a decline in interest rates while still being protected against an interest rate hike. The T_0 value of the trigger swap on a nominal of 1 with cashflows from time T_i to T_N can be calculated with the code:

```
void priceAgainstCurve(Matrix Li, Matrix Pi, Vector delta, Vector cashflows)
{
  int N = delta.cols;
  for (int i = first_included_time; i < N; ++i)
  {
    cashflows[i] = delta[i] * Math.Min(ceiling, Li[i-1, i]);
  }
}
```

6.4.7 Bermudan swaption

Bermudan swaptions are probably the most popular of the exotic interest rate products in the market as they are fundamental to the pricing of home loans where the borrower can pay off their loan early. This ability to settle a loan at any time gives the holder of the loan an American (or Bermudan) option on the underlying interest rate. This American feature adds complications that are outside the scope of this dissertation. The pricing of Bermudan swaptions is currently an area of active research: [2], [36], [38] and [52] treat specifically the pricing of Bermudan swaptions using the Libor Market Model.

T_i	σ_i	T_i	σ_i
1	0.180253	11	0.137982
2	0.191478	12	0.134708
3	0.186154	13	0.131428
4	0.177294	14	0.128148
5	0.167887	15	0.127100
6	0.158123	16	0.126822
7	0.152688	17	0.126539
8	0.148709	18	0.126257
9	0.144703	19	0.125970
10	0.141259		

Table 1: 20 year caplet volatility view of the Banca IMI traders at 16 May 2000

7 Results

This section presents the results of several experiments. They were implemented in C# and run on a 3.0GHz Pentium 4 with 1Gb RAM. The first four sections examine the issues around the calibration of volatility and correlation matrices to caplets and swaptions and also the effects of factor reduction on the correlation matrix. Then Section 7.5 shows the effect of using different random number generators on the prices calculated by the Monte Carlo method. Section 7.6 shows the various issues surrounding the calculation of the Greeks. Section 7.7 shows that the Libor Market Model generates the same price for different interest rate derivatives no matter our choice of numeraire. Finally, Section 7.8 shows the prices of the various options covered in Section 6.4.

7.1 Calibration to caplets

We first calibrate our model to some caplet market data using the method described in Section 5.4. As was explained, caplet market data is generally obtained from the trading desk. Table 1 shows the market data supplied by [10], which contains the 20 year caplet volatility view of the Banca IMI traders at 16 May 2000.

Figures 15 and 16 show the results of calibration to caplet market data.

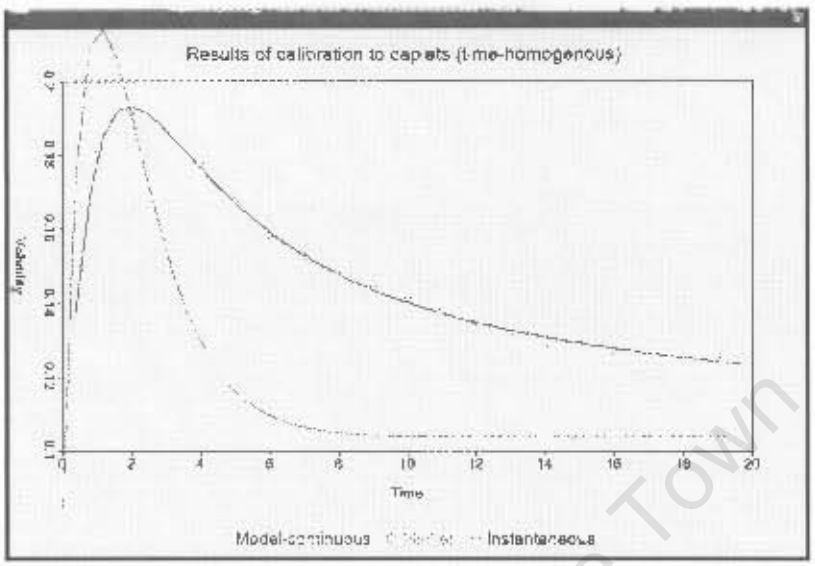

Figure 15: Calibration of model volatility to market data using a specification of the time-homogenous form (19).

Calibration using the Nelder-Mead takes a couple of seconds. The first figure shows how well the time-homogenous accumulated volatility parametrisation (solid line) of the form in Equation (19) fits the market data (circles). The dashed line shows the corresponding instantaneous volatility. To be precise, the solid line shows $\sqrt{\frac{1}{T}\int_0^T \sigma^2(u)du}$ while the dashed line shows $\sigma(T)$.

The resulting parameters for Equation (19) are:

$a =$ -0.0195607358161403 $\quad c =$ 0.959891548098344

$b =$ 0.307959414926732 $\quad d =$ 0.103684750097087

Naturally the parametrisation does not fit the market data exactly, especially in the term structure from years 13 to 16. Figure 16 shows the resulting calibration once we add some time-inhomogeneity as proscribed by Equation (21). Notice that the scaling factors K_i are very close to unity, as we should like.

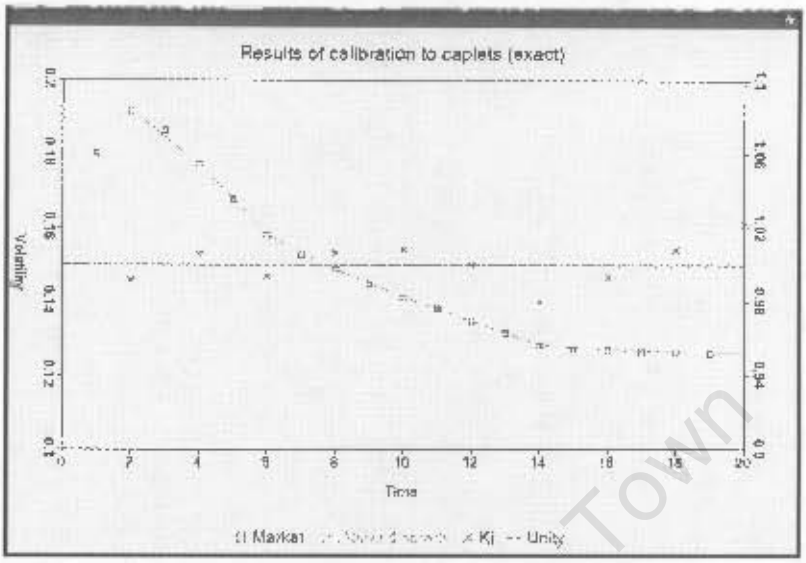

Figure 16: Calibration of model volatility to market data after adding the time-inhomogenous factors (21).

7.2 Sensitivity of caplet calibration to initial conditions

We now review how sensitive our calibration methodology is to minor changes in the underlying caplet market data. Ideally we would like small changes in the market data to have a small effect on our calibration parameters. If this is the case, then the behaviour of the prices and Greeks generated by our model should remain fairly smooth throughout the hedging lifetime of the derivatives we are modelling.

We test three scenarios in which we would like the model to remain stable:

1. There is an increase in implied volatility across the board. This is consistent with a significant economic change that affects the trading of interest rate derivatives across the entire term structure. We simulate this by increasing all market volatilities by 1%.

2. There is an increase in implied volatility over a particular period of time. This might happen if a significant event is known to be happen-

	a	b	c	d
Original	-0.0195607	0.307959	0.959892	0.103685
Bumped up	-0.0175795	0.302518	0.953801	0.114031
Bumped up (few)	-0.0121908	0.325393	0.951690	0.100538
Bumped random	-0.0195607	0.307959	0.959892	0.103685

Table 2: Parameters of the continuous time specification after various changes to the caplet market data.

ing at a certain time, but the consequences of the event are not known. We simulate this by increasing the market volatilities of the periods from 2 to 5 years by 1%.

3. There are random changes in implied volatility. This might be consistent with the normal market over time: market forces bump implied volatilities by "random" amounts (the so-called volatility-of-volatility). We simulate this by changing the market volatilities up or down by up to 10% of their current value.

Figure 17 shows the resulting parameters which are listed in Table 2:

Notice that the parameters do not vary wildly across each scenario. Most important is the fact that the random bumps of the market data yield a set of parameters that is almost identical to the original set.

Figure 18 shows the term structures of volatility generated by the various scenarios. Notice the line second-from-top, which corresponds to the 3 to 5 year bump in market vols: the model has done its best to capture these bumps while matching the remaining values as closely as possible. The top line shows the increase across the board of the market volatilities. The line corresponding to the random bumps up and down is indistinguishable from the original. This is very encouraging as it shows that the parametrisation is robust in the presence of corrections to the market data.

Occasionally (about one out of every twenty runs), the Nelder-Mead algorithm does not manage to find an optimal solution to the calibration. An example is shown in Figure 19. It is important therefore to manually check the results of the calibration routine to make sure that this has not happened.

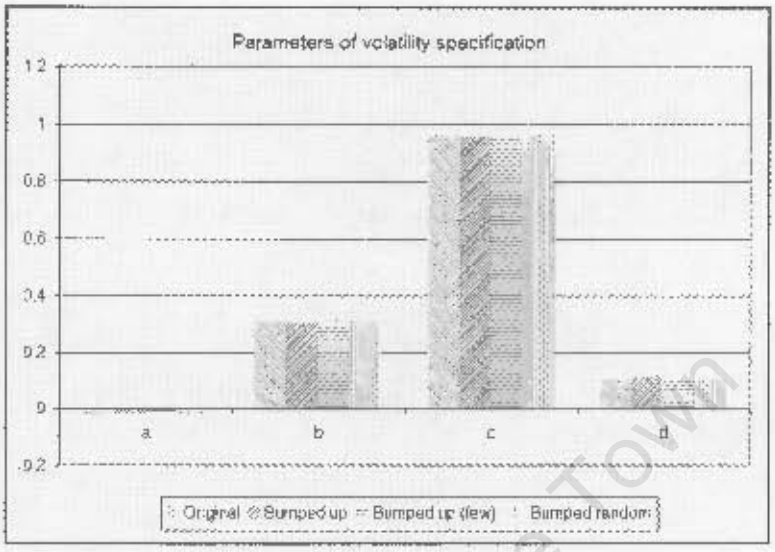

Figure 17: Parameters of the continuous time specification after various changes to the caplet market data.

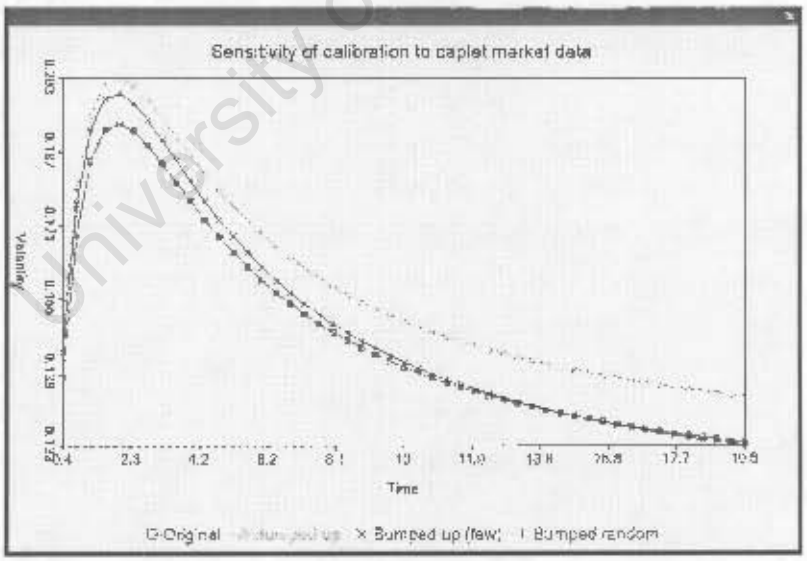

Figure 18: Term structures of the calibrated market volatilities after various changes to the caplet market data.

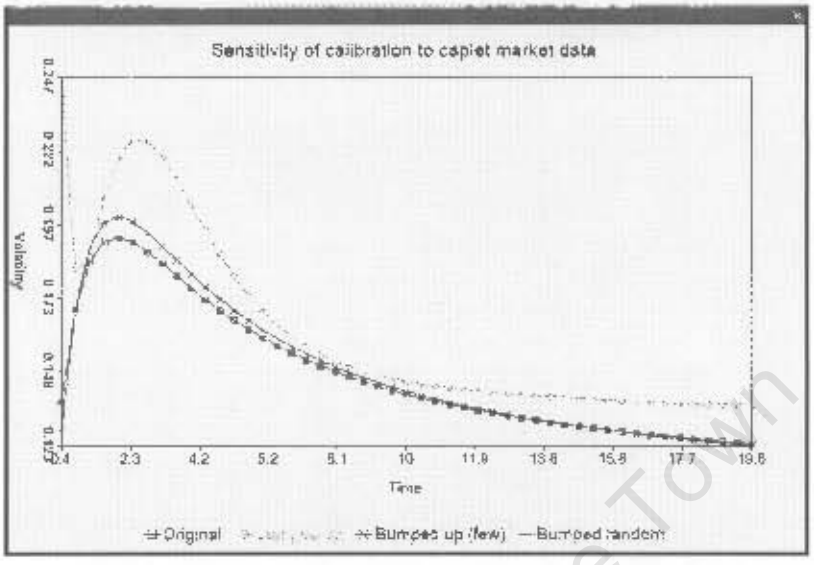

Figure 19: An example of where the Nelder-Mead algorithm does not find an optimum solution during calibration.

If we require a completely automated system, we could run the calibration routine several times and select the best solution.

7.3 Calibration to caplets and swaptions

Let us now see the effect of calibrating the Libor Market Model to caplets and swaptions in the market using the technique described in Section 5.5. The design of the calibration requires that the availability of coherent caplet and coterminal swaption market data, something I was not able to locate: unfortunately [10] was missing values from years 7-9! To this end I have "simulated" swaption volatility market data by calculating using Equation (24) the swaption volatility that is implied by the caplet prices and then bumped them down, giving swaption values that had similar shape to those in [59]. This might yield market conditions that are not entirely realistic, but it provides at least a dataset of market data against which we can test the calibration method.

The first pair of series in Figure 20 show the caplet-implied and my

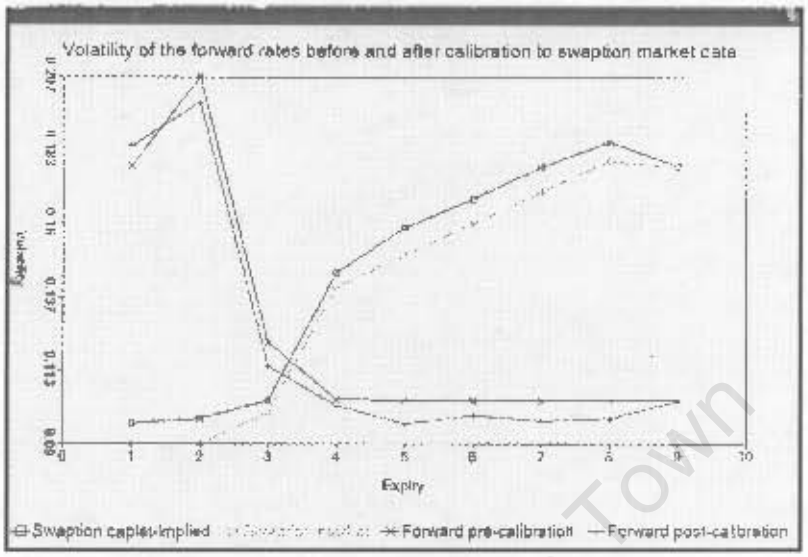

Figure 20: Volatilities of the swaptions that are implied by the caplet prices and the "market" volatilities; and volatilities of the forward rates before and after calibration to swaption volatilities.

"bumped" swaption volatilities. The graph shows the volatilities over only the first time period δ_1 (from 0 to 1 years); there is a similar set of volatilities over each time period δ_i. These swaption volatilities may seem strange at first: you might expect the same characteristic hump exhibited by the forward rates, but the weight of each forward rate volatility in the weighted sum of the swaption volatility is determined by the initial forward rate term structure and Equation (23). In this case the very non-linear weighting mechanism has found that the later swaption volatilities are very dependent on the early (and large) forward volatilities.

The second pair of series in Figure 20 shows the forward rate volatilities before and after calibration to the swaptions. Note that as the market implied lower swaption volatilities than did our model, the calibration forces the forward volatilities lower. Figures 21 and 22 show the difference the calibration to swaptions has made on the correlation matrix for the first time period δ_1. It has left the correlation structure predominantly untouched except in increasing the correlation between the earlier (L_2 and L_3) and later

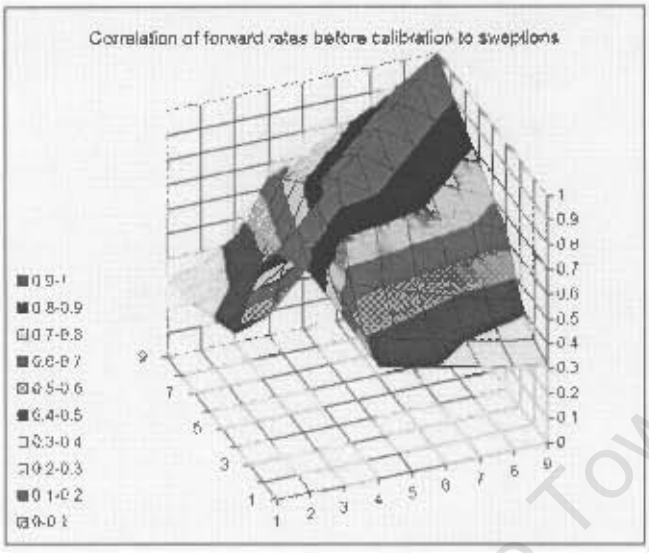

Figure 21: Correlation of forward rates before calibration to swaptions in the market. The floor axes represent the year of expiry of the forward rates and the vertical axis their correlation.

(L_4 through L_9) forward rates. This is shown more clearly in Figure 23, where the correlation matrix after calibration to swaptions has been subtracted from the initial correlation matrix.

7.4 Reduced factor matrices

This experiment determines the effect of factor reduction on the prices of a caplet and a swaption and on their computation time. In both simulations the discount bond chosen as numeraire is that maturing in ten years. The caplet runs from year 8 to year 9. The swaption expires on year 2 and lasts till year 10. The initial forward curve and volatilities are not that important for the results of this experiment and are chosen arbitrarily: the yield curve is gently decreasing from 9% out to 10 years while the volatility curve is the one calibrated in Section 7.1.

Figures 21 and 22 show, respectively, the correlation matrix before and after calibration to swaptions. Figure 23 shows their difference. Figure 24

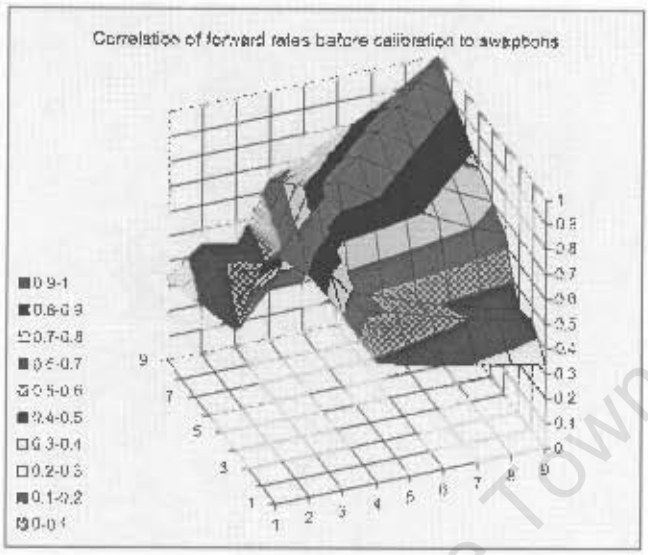

Figure 22: Correlation of forward rates after calibration to swaptions in the market. The floor axes represent the year of expiry of the forward rates and the vertical axis their correlation.

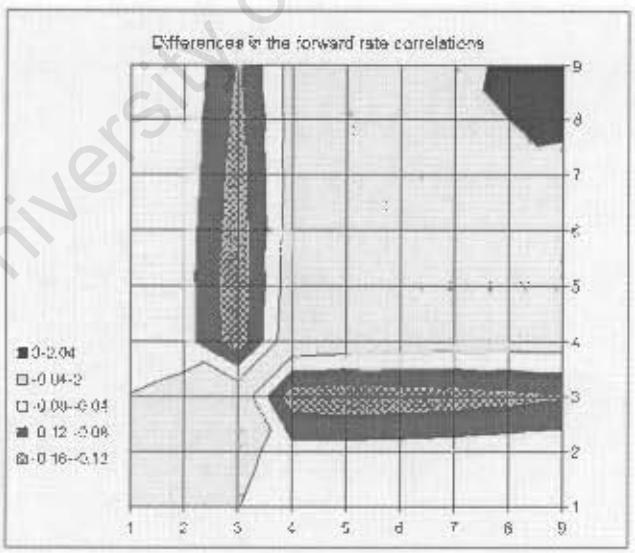

Figure 23: Differences in the forward rate correlations before and after calibration to swaptions in the market.

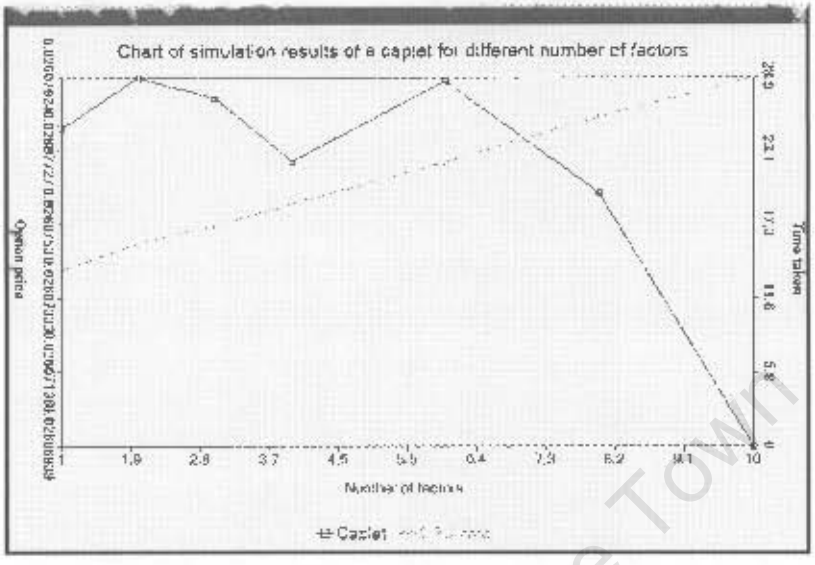

Figure 24: The effect of the number of factors on the price of a caplet. There is only a 0.01% difference in price across simulations. The computation time (in seconds) is almost linear in the number of factors.

shows the effect of factor reduction on the caplet price. There is hardly any change in the caplet price (less than 0.01%) as we move from one to ten factors. We should expect almost no change in the caplet price for even one factor because factor reduction affects only the correlation between the forward rates: a caplet price is not influenced by correlation, only the volatility of the forward rate prevailing over the lifetime of the option. Notice though that the change in computation time is almost linear in the number of factors.

Figure 25 shows the effect of factor reduction on the swaption price. There is a more significant change in the swaption price (around 0.1%) as we move from one to ten factors, with the most dramatic change arising from one to four factors. After five factors there is little change to the swaption price. We would expect this larger influence of the number of factors on the swaption price because a swaption is sensitive to the correlation between the forward rates prevailing over its entire lifetime. Again, the change in computation time is almost linear in the number of factors.

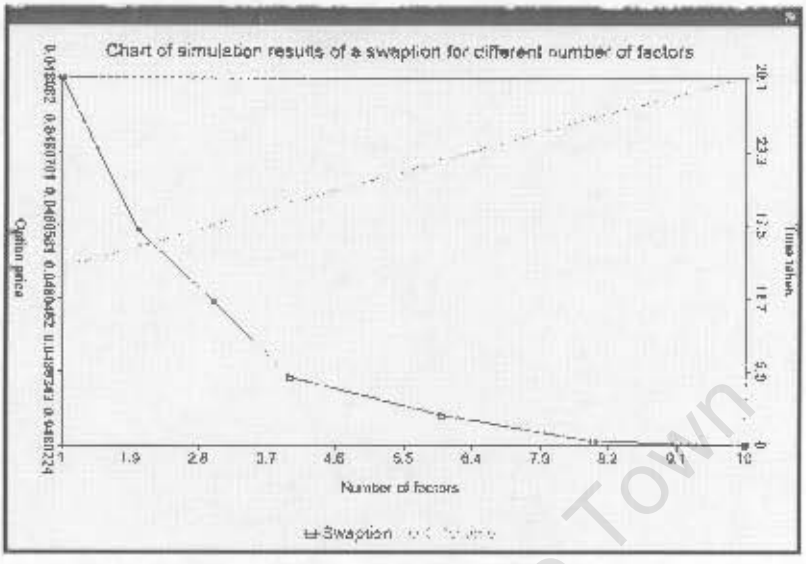

Figure 25: The effect of the number of factors on the price of a swaption. There is only a 0.1% difference in price across simulations. The computation time (in seconds) is almost linear in the number of factors.

7.5 Effect of different random number sources and antithetics

Let us now turn to examining the effect that the different random number generators have on the pricing of derivatives. We have at our disposal three sources of random numbers:

- pseudo-random numbers generated by the Mersenne Twister algorithm (presented in Section 6.2.1);
- pseudo-random numbers generated by the Mersenne Twister algorithm with antithetic sampling (presented in Section 6.2.3); and
- quasi-random numbers generated by the Sobol sequences (presented in Section 6.2.2)

Figure 26 shows the results of pricing a caplet using the three different sources of random numbers. Notice how the Sobol-driven simulation converges very quickly, needing as few as 4096 iterations to have settled down

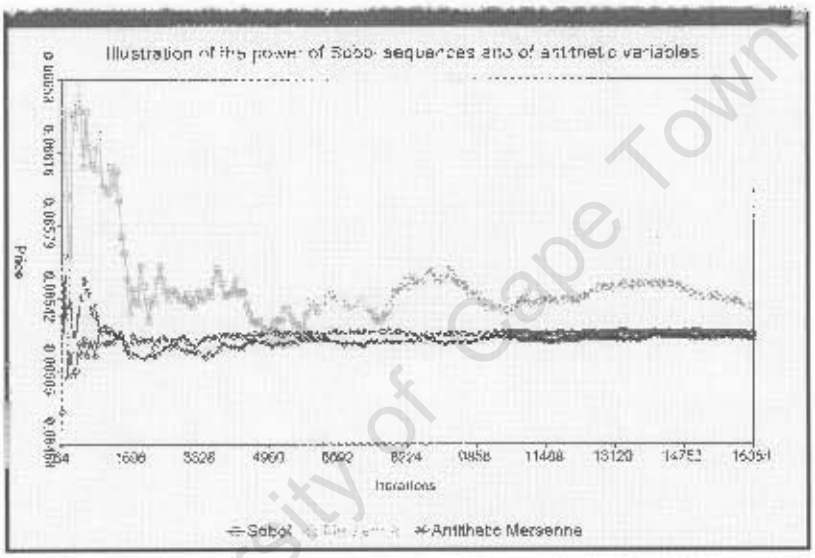

Figure 26: Illustration of the power of Sobol sequences and of antithetic variables. Notice how the Sobol-driven simulation converges very quickly. The antithetic-driven simulation seems to converge almost as quickly although it exhibits a slow drift upwards. The vanilla Mersenne-driven simulation has terrible convergence characteristics.

around the option price. The antithetic-driven simulation seems to converge almost as quickly although it exhibits a slow drift upwards: it has almost coincided with the Sobol-driven simulation by 16384 simulations. The vanilla Mersenne-driven simulation has terrible convergence characteristics.

These results speak of the power of Sobol-driven Monte Carlo techniques: although not truly random, their ability to quickly probe the broadest cross-section of the simulation space ensures that they produce prices that take into account most of the features of a derivative product. Obviously the finer features, e.g. the discontinuities caused by barrier options, would be detected only as more iterations are performed and the Sobol number mesh becomes small enough to approximate these features. Pseudo-random techniques might uncover these finer features sooner as they randomly sample the simulation space, but in a probabilistic sense they are unlikely to find them consistently sooner.

7.6 The Greeks

This experiment calculates the Greeks of three options: a caplet expiring at T_1 with payoff at T_2; a caplet expiring at T_3 with payoff at T_4; and a swaption expiring at T_4 with payoffs from times T_5 to T_{10}. The Greeks were calculated using the different "bump" factors in the underlying forward rates as discussed in Section 4.6: $\Delta L_i = \sqrt[4]{\epsilon} L_i$ (as recommended by [29]); $\Delta L_i = 0.01 L_i$ (1% recommended by [49]) and just for good measure $\Delta L_i = 0.001 L_i$ (0.1%). I included the third factor to see just how sensitive our estimates of the Greeks are to the choice of bump factors.

All three options were calculated using the T_9 discount bond as numeraire so that the caplets might be priced in the same framework as a 5-year swaption starting in year 4.

Figure 27 shows the calculations for delta. Notice that the results using any of the bump factors are all but identical. As we would expect, the $caplet_2$ option is sensitive primarily to L_2 since its payoff is directly dependent on that forward rate. Obviously the rate L_1 does not play an important enough role in the discounting of the payoff to warrant a significant sensitivity. However

the $caplet_4$ option does show slight sensitivity to the rates L_1 through L_3: they are used to discount the payment to the valuation time. As expected the $caplet_4$ option is very sensitive to L_4. The $swaption_4$ option is the most interesting. It generates cashflows from time T_5 to T_9 dependent on the forward rates L_5 to L_9 and so is obviously going to have significant sensitivity to those forward rates. It is also rather sensitive to the forward rates before its expiry, probably because of the discounting of the cashflows, but also because of the role they play in the calculation of the discount bonds used in the valuation of the par swap rate in Equation (3).

Figure 28 shows the calculations for gamma. Here we start seeing instabilities in the calculated values depending on the size of the bump factor. Indeed, using the $\sqrt[4]{\varepsilon}$ bump factor yields results that are completely unstable (in the order of 10^6) and so have been omitted. There is a slight difference between the values of gamma calculated using 1% and 0.1% bump factors and it is not clear which values might be the correct ones to use when hedging. It might be reasonable for the hedging trader to estimate the size of jump in underlying forward price that she would expect before hedging and use that number to calculate gamma. We see sensitivities to the underlying forward rates similar to those of delta.

7.7 Effect of different numeraires on accuracy

This experiment tests that our change of numeraire mathematics is correct: Equation (13) gives us the dynamics of our forward rates under any choice of discount bond as numeraire. This means that we should calculate the same price for a derivative no matter which discount bond we use as numeraire.

Figure 29 shows us the calculated price of a caplet with expiry at the end of year 1 and payment at the end of year 2 when priced under different discount bond numeraires with expiries ranging from 4 years to 20 years. Notice that the price does indeed remain relatively constant (at least to within 0.1%). What we also expect is an increase in the variance of this price: as we push the numeraire further out in time we have to carry our caplet cashflow at year 3 forward through many more simulated forward

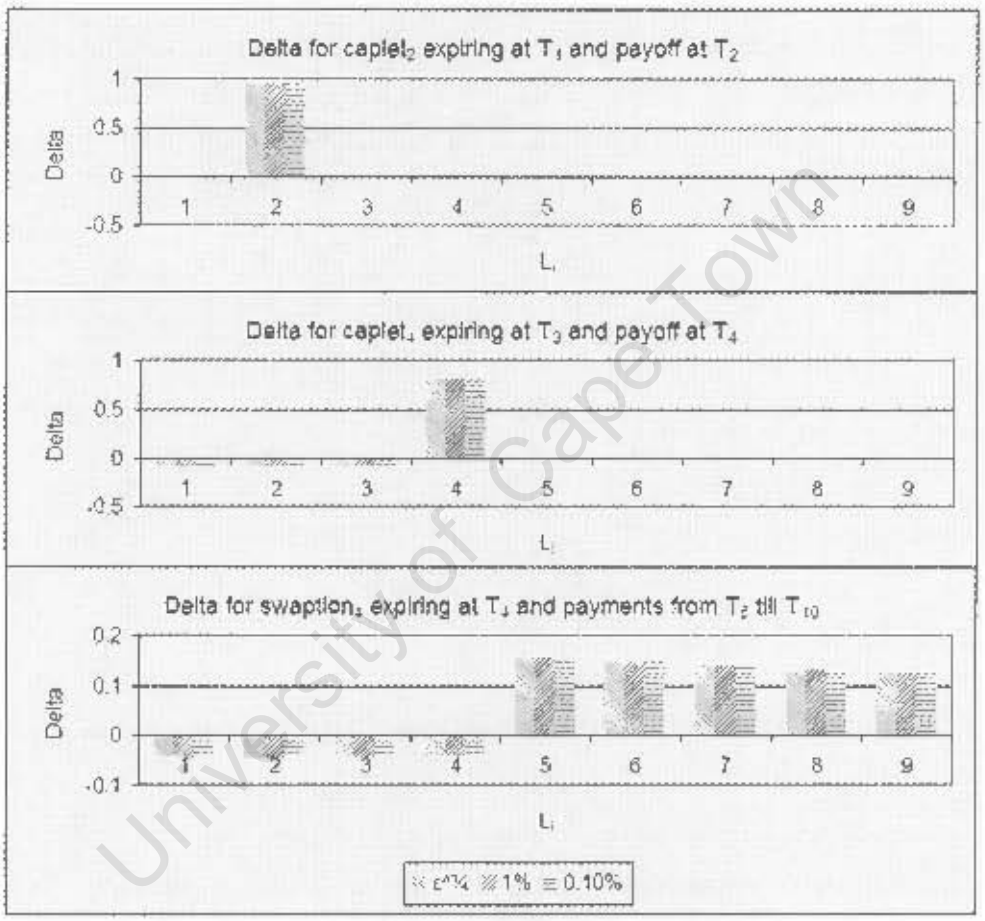

Figure 27: This chart shows the delta for three different options calculated using the three "bump" amounts in the underlying.

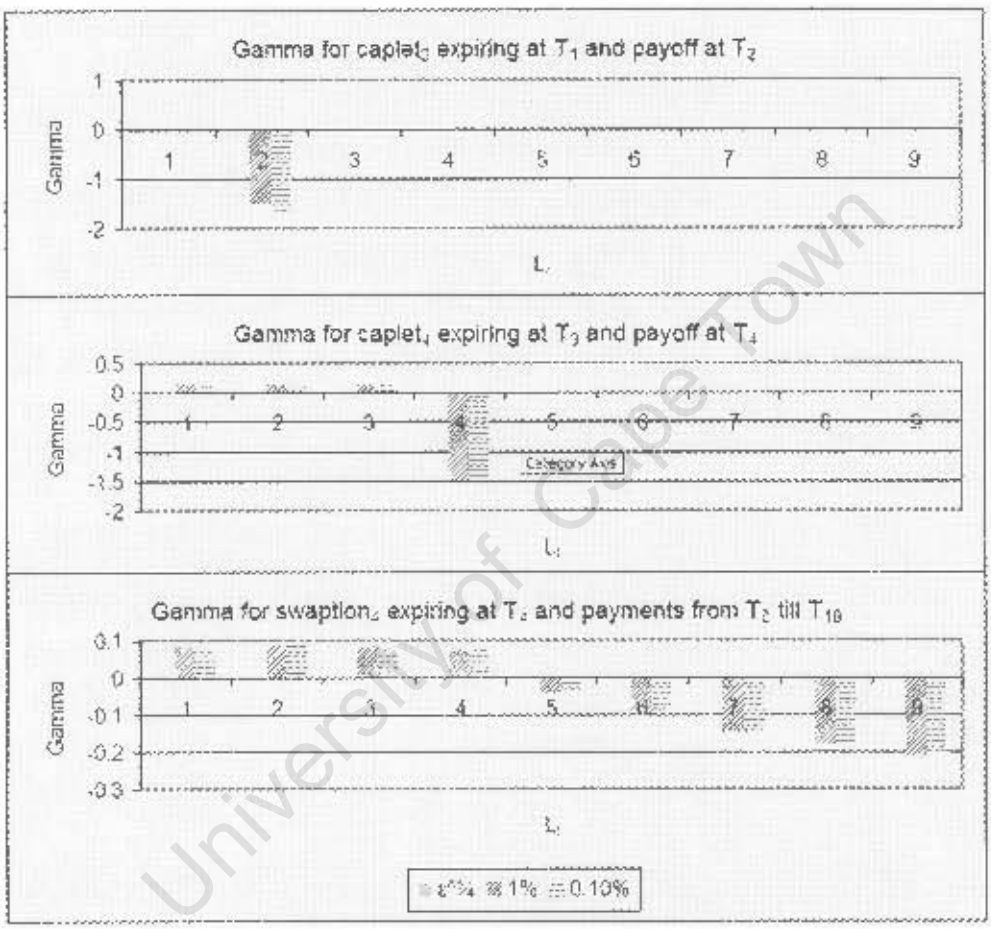

Figure 28: This chart shows the gamma for three different options calculated using the three "bump" amounts in the underlying. Note that the values calculated using $\sqrt[4]{\varepsilon}$ as the bump amount were completely unstable (in the order of 10^6) and so have been omitted.

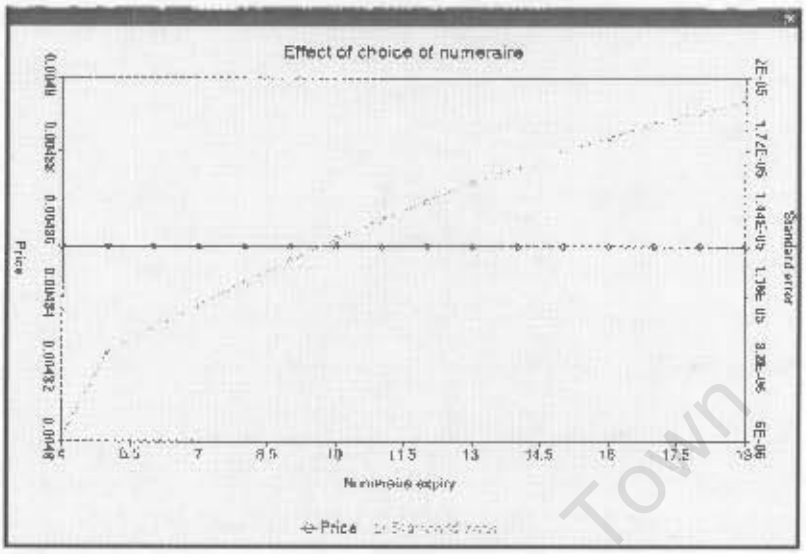

Figure 29: Notice that the choice of numeraire has little impact on the price of the derivative, although there is an increase in the variance of our simulation estimate.

rates to our terminal measure and then discount it back to today. The effect of the additional variance of each simulated forward rate should cause the derivative variance to increase too. This is evident in Figure 29.

7.8 Pricing various options

For completeness, Figure 30 shows the prices and Greeks of various interest rate derivatives priced using the Libor Market Model. Although we done go into much detail about the properties of these derivative prices and have not even shown the initial forward curve nor the forward volatilities that got us to these prices, it is instructive to do some brief inspection of the results:

- Notice that the down-and-out caplet is cheaper than the corresponding caplet, as we would expect from an option that might become worthless.
- Notice that the limit cap is cheaper than the corresponding cap as it provides only 2 out of the 6 payoffs that does the ordinary cap.

- The two caplet types are sensitive only to the underlying forward rate at their expiry (on which their payoff is contingent), whereas the caplets are (obviously) sensitive to all the forward rates they span. Interesting is the fact that the limit cap with two payoffs is sensitive to the first two forward rates it spans and hardly sensitive to the third: this must mean that the comprising caplets are rather in the money and will likely be exercised.

- The swaption is sensitive to all the forward rates it spans while the trigger swap is sensitive only to the first few. It is negatively correlated to movements in the latter forward curves: these must be high enough to trigger the fixed swap payment and so their value is reflected only in the discounting of the later cashflows.

		Caplet with strike 0.085 and maturity T4	Down-and-out caplet with strike 0.085, barrier 0.075 and maturity T4	Cap with strike 0.005 and maturities from T2 to T7	Limit cap with strike 0.005, limit 2 and maturities from T2 to T7	Swaption with strike 0.005 and first payment time T2	Trigger swap with ceiling 0.02 and first payment time T2
Price		0.011550079	0.011067611	0.248805977	0.019149046	0.046046105	0.089500171
Delta	T2	-0.011323607	-0.010850599	-0.24373135	-0.018773575	-0.04514324	-0.087745266
	T3	-0.011435722	-0.010958031	0.724277865	0.842793932	-0.045590203	0.881497802
	T4	-0.011323607	-0.010850599	0.712622309	0.937580084	0.17585047	0.347442076
	T5	0.541218687	0.577440303	0.662387348	0.000245616	0.154879501	-0.057616009
	T6	-3.98E-06	0.000159375	0.663834288	-9.45E-06	0.14748016	-0.041944451
	T7	-5.15E-06	0.000130994	0.665056739	-7.24E-06	0.14063288	-0.027184996
	T8	-4.75E-06	0.000109504	0.666062559	-5.61E-06	0.134215554	-0.013124856
Gamma	T2	0.022203159	0.021275685	0.47790473	0.036810936	0.088516156	0.172049527
	T3	0.022645011	0.021699066	-0.763112351	21732.54962	0.090277646	-1.745534273
	T4	0.022203151	0.021275677	-1.39730609	-1.838398296	-0.049519711	-67.88122332
	T5	11.96071378	-227.9488113	-1.215398255	-0.000459642	-0.10343251	0.105659327
	T6	-0.000420299	-0.000682259	-1.228230831	1.17E-05	-0.149532365	0.077600507
	T7	-0.000259195	-0.000488234	-1.239803595	9.24E-06	-0.186692071	0.05089886
	T8	-0.000165425	-0.0003644	-1.25018829	7.02E-06	-0.217279252	0.024649222

Figure 30: The price and Greeks of various options priced using the Libor Market Model.

8 Conclusion

The intention of this dissertation was to draw from the literature to bring into one body of work a treatment of the Libor Market Model from start to finish so that the reader might understand the basics of the Libor Market Model and implement the model without having to piece together information from other sources. Has this been achieved? Certainly the derivation of the mathematical model in Section 4 has distilled work from several sources into a consistent notational framework. The sandy areas in the mathematical model where it is left to the implementor to choose specific specifications for volatility, correlation, choices of factors, Monte Carlo details were set in concrete in Sections 5 and 6. And finally, Section 7 showed that the model, as implemented, works as hypothesised without any nasty surprises.

The achievement of the programming effort of several thousand lines of code is the simple interface that one needs to implement to price a new interest rate product. One needs simply to describe the cashflows of the derivative to price it. The calibration, factor reduction, Monte Carlo simulations, calculation of Greeks and everything else that goes into the implementation of the Libor Market Model is done behind the scenes.

The model has shown itself to be extremely flexible in its ability to calculate the prices and Greeks of a wide variety of options with minimal effort. The beautiful simplicity of the model ensures that it can be easily understood and quickly and correctly implemented. As computers become faster, the time it takes to perform the Monte Carlo simulations will drop, allowing even more complex derivatives to be priced without having to make significant changes to the model or code.

There are obviously some shortcomings of the model, but they are by no means insurmountable. The pricing of American or Bermudan options poses challenges to any model relying on Monte Carlo techniques: significant progress has been made in improving this state of affairs, especially in the case of Bermudan options. The calculation of the Greeks using a general finite difference approach is risky when there are discontinuities in the option payoffs - especially those of path dependent options: techniques like

the pathwise method and the likelihood ratio method help to resolve this, although in their current form they can not be applied to general options without individual changes to their code. Computation time can become a problem, especially when calculating the sensitivities of long-term options to each of the many forward rates they span: distributed computing techniques (to which Monte Carlo methods are particularly suited) can help alleviate this problem by sharing the load across many computers.

In addition to the active research being done where the Libor Market Model has its shortcomings, there is significant work being done to advance the Libor Market Model to take into account skews and smiles in the interest rate volatilities; to take advantage of developments in Levy and jump processes; and into using vast computing grids to use the Libor Market Model to risk manage portfolios of thousands of derivatives. All these areas provide scope for some very interesting further research.

9 Bibliography

References

[1] Alexander, C.; Common Correlation and Calibrating the Lognormal Forward Rate Model; Wilmott Magazine; March 2003 pp68-78; 2003

[2] Andersen, L.; A simple approach to the pricing of Bermudan swaptions in the multifactor LIBOR market model; Journal of Computational Finance; 3.2 pp5-32; 2000

[3] Björk, T.; Arbitrage Theory in Continuous Time; Second Edition; Oxford University Press; 2004

[4] Brace, A., Gatarek, D. & Musiela, M.; The market model of interest rate dynamics; Mathematical Finance; 7 pp127-155; 1997

[5] Black, F.; The Pricing of Commodity Contracts; Journal of Financial Econometrics; 3 pp167-179; 1976

[6] Black, F., Derman, E. & Toy, W.; A one-factor model of interest rates and its application to treasury bond options; Financial Analysts Journal; pp33-39; 1990

[7] Boyle, P., Broadie, M. & Glasserman, P.; Monte Carlo methods for security pricing; Journal of Economic Dynamics and Control; 21 pp1267-1321; 1997

[8] Brigo, D.; A note on correlation and rank reduction; Banca IMI Working Paper; www.damianobrigo.it; 2002

[9] Brigo, D. & Mercurio, F.; Calibrating LIBOR; Risk Magazine; January issue pp117-121; 2002

[10] Brigo, D., Mercurio, F. & Morini, M.; Different Covariance Parameterizations of the Libor Market Model and Joint Caps/Swaptions Calibration; Product and Business Development Group Banca IMI working paper; 2003

[11] Brigo, D., Mercurio, F. & Morini, M.; The Libor model dynamics: Approximations, calibration and diagnostics; European Journal of Operational Research; 163 pp30–51; 2005

[12] Broadie, M.; Glasserman, P.; Estimating Security Price Derivatives Using Simulation; Management Science; 42.2 pp269-285; 1996

[13] Cairns, A.J.G.; Interest Rate Models: An Introduction; Princeton University Press; 2004

[14] Coffey, B. & Schönmakers, J.; Stable implied calibration of a multi-factor Libor model via a semi-parametric correlation structure; Working paper 611; Weierstrass-Institut für Angewandte Analysis und Stochastik; 2000

[15] Coffey, B. & Schönmakers, J.; Systematic generation of parametric correlation structures for the Libor Market Model; International Journal of Theoretical and Applied Finance; 6.5 pp507-519; 2003

[16] Cox, J.C., Ingersoll, J.e. & Ross, S.A.; An intertemporal general equilibrium model of asset prices; Econometrica; 53 pp363-384; 1985

[17] Derman, E.; My Life as a Quant: Reflections on Physics and Finance; John Wiley and Sons; 2004

[18] Glasserman, P.; Monte Carlo Methods in Financial Engineering; Springer; 2004

[19] Glasserman, P. & Zhao, X.; Fast Greeks by simulation in forward LIBOR models; Journal of Computational Finance; 3 pp5-39; 1999

[20] Glasserman, P. & Zhao, X.; Arbitrage-free discretization of lognormal forward Libor and swap rate models; Finance and Stochastics; 4 pp35-68; 2000

[21] Grubišić, I. & Pietersz, R.; Efficient rank reduction of correlation matrices; Working paper; Utrecht University; www.few.eur.nl/few/people/pietersz; 2004

[22] Hagan, P. & West, G.; Interpolation Methods for Curve Construction; Working paper, to appear in Applied Mathematical Finance; 2005

[23] Harrison, J.M. & Pliska, S.; Martingales and stochastic integrals in the theory of continuous trading; Stochastic Processes and their Applications; 11 pp215-260; 1981

[24] Heath, D., Jarrow, R. & Morton, A.; Bond pricing and the term structure of interest rates: A new methodology for contingent claims valuation; Econometrica; 60 pp77-105; 1992

[25] Ho, T.S.Y. & Lee, S.B.; Term structure movements and pricing interest rate contingent claims; Journal of Finance; 41; 1986

[26] Hull, J.C.; Options, Futures, and Other Derivatives; Fifth Edition; Prentice Hall; 2003

[27] Hull, J.C. & White, A.; Pricing interest rate derivative securities; Review of Financial Studies; 3; 1990

[28] Hunt, P.J. & Kennedy, J.E.; Financial Derivatives in Theory and Practice; Revised Edition; John Wiley and Sons; 2004

[29] Jäckel, P.; Monte Carlo Methods in Finance; John Wiley and Sons; 2002

[30] Jamshidian, F.; Libor and swap market models and measures; Finance and Stochastics; 1 pp293–330; 1997

[31] Joshi, M.S.; A short note on exponential correlation functions; QUARC Royal Bank of Scotland working paper; 2000

[32] Joshi, M.S.; The Concepts and Practice of Mathematical Finance; Cambridge University Press; 2003

[33] Jäckel P. & Rebonato R.; Accurate and optimal calibration to co-terminal European swaptions in a FRA-based BGM framework; QUARC Royal Bank of Scotland working paper; 2000

[34] Jäckel P. & Rebonato R.; Linking Caplet and Swaption Volatilities in a BGM/J Framework: Approximate Solutions; QUARC Royal Bank of Scotland working paper; 2000

[35] Jäckel P. & Rebonato R.; The Link between Caplet and Swaption Volatilities in a BGM/J Framework; Journal of Computational Finance; 6.4 pp41-59; 2003

[36] Joshi, M.S. & Theis, J.; Bounding Bermudan swaptions in a swap-rate market model; Quantitative Finance; 2 pp370-377; 2002

[37] Klooden, P.E. & Platen, E.; Numerical Solution of Stochastic Differential Equations; Springer; 1992

[38] Kolodko, A. & Schoenmakers, J.; Iterative construction of the optimal Bermudan stopping time; Financial Engineering and Applications; 2004

[39] Longstaff, F.A. & Schwartz, E.S.; Interest rate volatility and the term structure: A two-factor general equilibrium model; Journal of Finance; 47; 1992

[40] Longstaff, F., Santa-Clara, P. & Schwartz, E.; The Relative Valuation of Caps and Swaptions: Theory and Empirical Evidence; Journal of Finance; 56.6 pp2067–2109; 2000

[41] Matsumoto, M. & Nishimura, T.; Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator; ACM Transactions on Modeling and Computer Simulation (TOMACS); 8:1 pp3-30; 1998

[42] Merton, R.C.; Theory of Rational Option Pricing; Bell Journal of Economics and Management Science; 4 pp141-183; 1973

[43] Meyer, M.J.; The Martingale Project; http://martingale.berlios.de; 2003

[44] Mikosch, T.; Elementary Stochastic Calculus with Finance in View; Advanced Series on Statistical Science & Applied Probability Vol. 6; World Scientific; 1998

[45] Milterstein, M., Sandmann, K. & Sondermann, D.; Closed-form solutions for term structure derivatives with lognormal interest rates; Journal of Finance; pp409-430; 1997

[46] Musiela, M. & Rutkowski, M.; Continuous-time term structure models: Forward measure approach; Finance and Stochastics; 1 pp261-291; 1997

[47] Niederreiter, H.; Low-discrepancy sequences and global function fields with many rational places; Finite Fields and their Applications; 2 pp241-273; 1996

[48] Øksendal, B.; Stochastic Differential Equations: An Introduction with Applications; Fifth Edition; Springer; 2000

[49] Panday, S.; Efficient Simulation within the LIBOR Market Model; Master's Degree in Econometrics and Operations Research, Vrije Universiteit Amsterdam; 2003

[50] Pelsser, A.; Efficient Methods for Valuing Interest Rate Derivatives; Springer; 2000

[51] Piterbarg, V.; A Practitioner's Guide to Pricing and Hedging Callable Libor Exotics in Forward Libor Models; http://ssrn.com/abstract=427084; 2003

[52] Pietersz, R. & Pelsser, A.; Risk Managing Bermudan Swaptions in the Libor BGM Model; Journal of Derivatives; 11.3 pp51-62; 2004

[53] Pietersz, R. & Pelsser, A.; Swap Vega in BGM: Pitfalls and Alternatives; Risk 17 (March04) pp91-93; 2004

[54] Press, W.H., Teukolsky, S.A., Vetterling, W.T. & Flannery, B.P.; Numerical Recipes in C: The Art of Scientific Computing; Second Edition; Cambridge University Press; 1992

[55] Rebonato R.; Interest-rate option models; Second Edition; John Wiley and Sons; 1996

[56] Rebonato R.; On the pricing implications of the joint log-normality assumption for the cap and swaption markets; Journal of Computational Finance; 2.3 pp30-52; 1999

[57] Rebonato R.; Volatility and correlation; John Wiley; 1999

[58] Rebonato R.; Accurate and optimal calibration to co-terminal European swaptions in a FRA-based BGM framework; QUARC Royal Bank of Scotland working paper; 2000

[59] Rebonato, R.; Modern Pricing of Interest Rate Derivatives; Princeton University Press; 2002

[60] Sidenius, J.; Libor Market Models in practice; Journal of Computational Finance; 3.3 pp75-99; 2000

[61] Sobol, I.M.; On the distribution of points in a cube and the approximate evaluation of integrals; USSR Computational Mathematics and Mathematical Physics; 7 pp86-112; 1967

[62] Svoboda, S.; An Investigation of Various Interest Rate Models and their Calibration in the SA Market; MSc Dissertation; University of the Witwatersrand; 2002

[63] Vasicek, O.; An Equilibrium Characterisation of the Term Structure; Journal of Financial Econometrics; 5 pp177-188; 1977

[64] West, G.; Better Approximations to Cumulative Normal Functions; Wilmott Magazine; May 2005 pp70-76; 2005

[65] Wilmott, P.; Derivatives: The Theory and Practise of Financial Engineering; John Wiley and Sons; 1998

[66] Wilmott, P.; Paul Wilmott on Quantitative Finance; John Wiley and Sons; 2000

10 Code Appendix

The source code implemented during the development of this dissertation is available online at http://www.jimme.net. Including the thousands of lines of source code as an appendix seemed somewhat counterproductive. Instead, included are a few snippets of code that are useful enough to warrant being published in print, not readily available in the literature, and concise enough to make a reasonable appendix.

10.1 Integrating $\int_s^t \sigma_i(u)\sigma_j(u)du$

To calculate $\int_s^t \sigma_i(u)\sigma_j(u)du$ use:

```
public double getIntegralDefinite(double s, double t, double Ti, double Tj)
{
  return getIntegralIndefinite(t, Ti, Tj) - getIntegralIndefinite(s, Ti, Tj);
}

public double getIntegralIndefinite(double t, double Ti, double Tj)
{
  double ctmTi=c*(t-Ti);
  double ctmTj=c*(t-Tj);
  double q=ctmTi+ctmTj;
  double ac=a*c;
  double cd=c*d;
  double f=1.0/(c*c*c);
  double A=ac*cd*(exp(ctmTj)+exp(ctmTi))+c*cd*cd*t;
  double B=b*cd*(exp(ctmTi)*(ctmTi-1)+exp(ctmTj)*(ctmTj-1));
  double C=exp(q)*(ac*(ac+b*(1-q))+b*b*(0.5*(1-q)+ctmTi*ctmTj))/2;
  return f*(A-B+C);
}
```

10.2 Moro's Normal inverse transform

Moro's algorithm is used to convert a uniformly distributed number in the range $[0, 1]$ to a Normally distributed number in the range $[-\infty, \infty]$. It is purportedly accurate to 14 decimal places, although I would be a little sceptical out in the tails of the distribution.

```
public static double InvDist_SN_Moro(double y)
{
  double zz;
  double z = y - 0.5;

  if (Math.Abs(z) < 0.42)
  {
    zz = Math.Pow(z,2);
    zz = z * (((-25.44106049637 * zz + 41.39119773534)
          * zz + -18.61500062529) * zz + 2.50662823884) /
          ((((3.13082909833 * zz + -21.06224101826) * zz + 23.08336743743)
          * zz + -8.4735109309) * zz + 1);
  }

  else
  {
    if (z > 0) zz = Math.Log(-Math.Log(1 - y));
    else zz = Math.Log(-Math.Log(y));

    double build = 2.888167364E-07 + zz * 3.960315187E-07;
    build = 3.21767881768E-05 + zz * build;
    build = 3.951896511919E-04 + zz * build;
    build = 3.8405729373609E-03 + zz * build;
    build = 2.76438810333863E-02 + zz * build;
    build = 0.160797971491821 + zz * build;
    build = 0.976169019091719 + zz * build;
    zz = 0.337475482272615 + zz * build;
```

```
    if (z <= 0) zz = -zz;
  }

  return zz;
}
```

10.3 The Nelder-Mead algorithm

The Nelder-Mead algorithm is an optimisation algorithm for finding a global minimum in a highly non-linear and non-differentiable search space. It comes in two parts: the `ObjectiveFunction` interface, which must be implemented to evaluate and constrain the objective function at the point p; and the `NelderMead` class, which performs the actual optimisation.

```
using System;

namespace Utilities.Mathematics.Optimisation.NelderMead
{
  public interface ObjectiveFunction
  {
    double evaluate(double[] p);
    void constrainSearch(ref double[] p);
  }
}
using System;

namespace Utilities.Mathematics.Optimisation.NelderMead
{
  public class NelderMead
  {
    const double GROW_FACTOR = 2.0;
    const double REFLECT_FACTOR = -1.0;
    const double CONTRACT_FACTOR = 0.5;
```

```
const double SHRINK_FACTOR = 0.5;

public double MAX_ITERATIONS = 10000;
public double TOLERANCE = 1E-12;

int dimensions;
ObjectiveFunction objective_function;
double[][] points;
double[] scores;

public NelderMead(int adimensions)
{
  dimensions = adimensions;
  points = new double[dimensions+1][];
  for (int i = 0; i < dimensions+1; ++i)
  {
    points[i] = new double[dimensions];
  }
  scores = new double[dimensions+1];
}

public double[][] generateEmptyStartupParameters(double[] scales)
{
  Utilities.Random.RandomAugmented ra =
    Utilities.Random.RandomAugmented.getSeededRandomAugmented();

  double[][] initial_points = new double[dimensions+1][];
  for (int i = 0; i < dimensions+1; ++i)
  {
    initial_points[i] = new double[dimensions];
    for (int j = 0; j < dimensions; ++j)
    {
      initial_points[i][j] = ra.NextDoubleBalanced(scales[j]);
```

```
    }
  }

  return initial_points;
}

public void initialiseSearch(ObjectiveFunction aobjective_function,
  double[][] starting_points)
{
  objective_function = aobjective_function;

  // Store our initial conditions
  for (int i = 0; i < dimensions+1; ++i)
  {
    for (int j = 0; j < dimensions; ++j)
    {
      points[i][j] = starting_points[i][j];
    }
  }

  // Evaluate our initial conditions
  for (int i = 0; i < dimensions+1; ++i)
  {
    scores[i] = objective_function.evaluate(points[i]);
  }
  sortPoints();
}

public double[] search(out string error_message, out double optimum_score)
{
  // Check that we have been initialised
  if (null == objective_function)
  {
```

```
    throw new
      GenericException("Objective function not set.  Can not search.");
  }

  // Initialise the error message
  error_message = "";

  // Keeps track of how many iterations we have done
  int num_iterations = 0;

  // The latest midpoint and newpoint we have for reflections, etc.
  // Defined here for memory allocation efficiency
  double[] mid_point = new double[dimensions];
  double[] new_point = new double[dimensions];

  while (true)
  {
    // Check our tolerance.  If we are within it, return
    double tolerance = 0.0;
    for (int j = 0; j < dimensions; ++j)
    {
      tolerance += Math.Abs(points[0][j] - points[dimensions][j]);
    }
    tolerance /= dimensions;
    if (tolerance < TOLERANCE)
    {
      optimum_score = scores[0];
      return points[0];
    }

    // Check that we have not exceeded the number of iterations
    ++num_iterations;
    if (num_iterations > MAX_ITERATIONS)
```

```
{
  error_message =
    "Exceeded number of iterations in the Nelder-Mead search.  Results m
  optimum_score = scores[0];
  return points[0];
}

// Find the mid point of the best points (note n points, not n+1,
// since we leave out the worst)
for (int j = 0; j < dimensions; ++j)
{
  mid_point[j] = 0.0;
}
for (int i = 0; i < dimensions; ++i)
{
  for (int j = 0; j < dimensions; ++j)
  {
    mid_point[j] += points[i][j];
  }
}
for (int j = 0; j < dimensions; ++j)
{
  mid_point[j] /= dimensions;
}

// Do a reflection test
if (testOnePointMove(REFLECT_FACTOR, ref mid_point, ref new_point))
{
  // If we managed a reflection, test a little further out
  testOnePointMove(GROW_FACTOR, ref mid_point, ref new_point);
}

// If the reflection failed, then try a contraction
```

```
    else if (testOnePointMove(CONTRACT_FACTOR, ref mid_point, ref new_point)
    {
    }

    // If reflection and contraction failed, then shrink towards
    // our best point
    else
    {
      doManyPointMove();
    }
  }
}

void doManyPointMove()
{
  for (int i = 1; i < dimensions+1; ++i)
  {
    for (int j = 1; j < dimensions; ++j)
    {
      points[i][j] = SHRINK_FACTOR * (points[i][j] + points[0][j]);
    }
    objective_function.constrainSearch(ref points[i]);
    scores[i] = objective_function.evaluate(points[i]);
  }

  sortPoints();
}

bool testOnePointMove(double scale, ref double[] mid_point,
  ref double[] new_point)
{
  // Calculate our new point dimensions
  for (int j = 0; j < dimensions; ++j)
```

```
    {
      new_point[j] = mid_point[j] +
        scale * (points[dimensions][j] - mid_point[j]);
    }

    // Make sure our new point obeys the constraints
    objective_function.constrainSearch(ref new_point);

    // See if our new point is better
    double new_score = objective_function.evaluate(new_point);

    // If it is, replace our old point and return true
    if (new_score < scores[dimensions])
    {
      for (int j = 0; j < dimensions; ++j)
      {
        points[dimensions][j] = new_point[j];
      }
      scores[dimensions] = new_score;

      sortPoints();

      return true;
    }

    // If it is not smaller, return false
    else
    {
      return false;
    }
  }

  void sortPoints()
```

```
    {
      // Lets do a simple bubble sort
      // This can be improved since the only unsorted one is usually at
      // the end of the array.  But its probably not worth it cos there
      // are always so few dimensions
      for (int i = 0; i < dimensions; ++i)
      {
        for (int j=i+1; j < dimensions+1; ++j)
        {
          if (scores[i] > scores[j])
          {
            swapPoints(i, j);
          }
        }
      }
    }

    void swapPoints(int a, int b)
    {
      Utilities.Swap.swap(ref scores[a], ref scores[b]);
      for (int j = 0; j < dimensions; ++j)
      {
        Utilities.Swap.swap(ref points[a][j], ref points[b][j]);
      }
    }
  }
}
```

10.4 The Mersenne Twister algorithm

The Mersenne Twister algorithm is useful for generating pseudo-random uniformly distributed numbers with excellent "random" properties in a machine independent manner.

```
using System;
using Utilities.GUI.Charting;
using Utilities.Mathematics.LinearAlgebra;

namespace Utilities.Random
{
  /**
  * This class is an implementation of the paper "Mersenne Twister:
  * A 623-Dimensionally Equidistributed Uniform Pseudo-Random Number Generator"
  * by Matsumoto
  */
  public class MersenneTwister : IUniformRandomSource
  {
    const int N = 624;
    const int M = 397;
    const uint MATRIX_A = 0x9908b0df;
    const uint UPPER_MASK = 0x80000000;
    const uint LOWER_MASK = 0x7fffffff;
    const uint TEMPERING_MASK_B = 0x9d2c5680;
    const uint TEMPERING_MASK_C = 0xefc60000;
    ulong[] mt = new ulong[N];
    int mti = N+1;
    ulong[] mag01 = {0, MATRIX_A};

    const ulong DEFAULT_SEED = 4357;
    ulong seed_used;

    public MersenneTwister()
    {
      SeedMT(DEFAULT_SEED);
    }

    public MersenneTwister(ulong seed)
```

```
{
  if (0 == seed)
  {
    throw new Utilities.GenericException("Seed can not be zero.");
  }

  SeedMT(seed);
}

public void reset()
{
  SeedMT(seed_used);
}

private void SeedMT(ulong seed)
{
  seed_used = seed;

  mt[0] = seed & 0xffffffff;
  for (mti = 1; mti < N; ++mti)
  {
    mt[mti] = (69069 * mt[mti-1]) & 0xffffffff;
  }
}

public ulong RandomInt()
{
  ulong y;
  if (mti >= N)
  {
    int kk;

    for (kk = 0; kk < N-M; ++kk)
```

```
    {
      y = (mt[kk] & UPPER_MASK) | (mt[kk+1] & LOWER_MASK);
      mt[kk] = mt[kk+M] ^(y >> 1) ^mag01[y&0x1];
    }

    for (;kk < N-1; ++kk)
    {
      y = (mt[kk] & UPPER_MASK) | (mt[kk+1] & LOWER_MASK);
      mt[kk] = mt[kk+(M-N)] ^(y >> 1) ^mag01[y&0x1];
    }

    y = (mt[N-1] & UPPER_MASK) | (mt[0] & LOWER_MASK);
    mt[N-1] = mt[M-1] ^(y >> 1) ^mag01[y&0x1];

    mti = 0;
  }

  y = mt[mti++];
  y ^= (y >> 11);
  y ^= (y << 7) & TEMPERING_MASK_B;
  y ^= (y << 15) & TEMPERING_MASK_C;
  y ^= (y >> 18);

  return y;
}

public double RandomDouble()
{
  return ((double) RandomInt() / (double) 0xffffffff);
}

public void RandomUniformVector(Vector vector)
{
```

```
      for (int i = 0; i < vector.cols; ++i)
      {
        vector[i] = RandomDouble();
      }
    }

    public double nextRandomDouble()
    {
      return RandomDouble();
    }
  }
}
```