The One Factor Libor Market Model Using Monte Carlo Simulation: An Empirical Investigation

Alonso Pena[*], Antonio di Sabatino[†], Simone Ligato,[‡]
Simone Ventura[§], Andrea Bertagna[¶]

December 15, 2010

Abstract

The Libor Market Model (LMM) is an advanced mathematical model used to price interest rate derivatives. Also known as the BGM model after its authors (Brace, Gatarek, Musiela, 1997), the LMM has become hegemonic in the financial markets worldwide. The LMM in reality is not a single model, but rather as a large family of models (Rebonato 2000, Brigo and Mercurio, 2006). Its many variants include: the number of factors considered, the type of volatility modelling used, the type of correlation modelling used, if stochastic volatility or SABR are used, if forward libor rates or swap rates are used, if semi-analytical or numerical solution methods are used, among others. The many faces of the LMM offer the disadvantage of making it difficult to understand for beginners. It also makes it difficult to clearly see what is the best version to use in practice. Our aim in this contribution will be to construct the simplest possible version of the LMM, implement it in C++ and investigate its accuracy to price real market-quoted interest rate derivatives. We consider three examples: a plain vanilla interest rate swap (IRS), and IRS with a CAP and an IRS with a CORRIDOR feature. Our results show that in these cases our implementation of the LMM1F captures quite well the market prices of these products, as compared with Bloomberg and Sungard Monis.

Keywords: LIBOR market model, Monte Carlo simulation, interest rate swaps, C++ programming language.

[*] SDA Bocconi School of Management and Thomson Reuters (alonso.pena@thomsonreuters.com, alonso.pena@sdabocconi.it)

[†] Thomson Reuters (antonio.disabatino@thomsonreuters.com)

[‡] Banca Carige (simone.ligato@carige.it)

[§] Banca Carige (simone.ventura@carige.it)

[¶] Unicredit Group (andrea.bertagna@unicredit.eu)

1 Introduction

The Libor Market Model (LMM) is an advanced mathematical model used to price interest rate derivatives. Also known as the BGM model after its authors (Brace, Gatarek, Musiela, 1997), the LMM has become hegemonic in the financial markets worldwide. The literature offers a wide range of publications about the LMM, mostly about its many variants and its complex advanced issues. As of December 2010 the academic SSRN Database (http://www.ssrn.com/) contained 180 articles matching for the words "libor market model, while the commercial site Amazon.com (http://www.amazon.com) contained 40 books.

The LMM is in fact not a single model, but rather as a large family of models (Rebonato 2000, Brigo and Mercurio, 2006). Its many variants include: the number of factors considered, the type of volatility modelling used, the type of correlation modelling used, if stochastic volatility or SABR are used, if forward libor rates or swap rates are used, if semi-analytical or numerical solution methods are used, among others. This rich and varied literature has the advantage of offering many avenues of research and development for the LMM, mostly from a theoretical point of view. However, it has the disadvantage of making it difficult for beginners to understand, as well as to form an opinion as to what version to use from practical point of view. Our aim in this contributionis to construct the simplest possible version of the LMM, implement it in C++, Mathematica and Excel and investigate its accuracy to price real market-quoted interest rate derivatives.

In the following sections we describe the methodology followed, the application to three real-world structured interest rate derivative products, and discuss the significance of our results.

2 Methodology

Our methodology and notation follows closely that of Pelsser (2000) which, even though succinct, provides a clear introduction to the Libor Market Model (LMM). From all the possible variations of the LMM, in this work we chose the simplest implementation: embodied in the use of Lognormal stochastic differential equations (Geometric Brownian Motion) for the forward rates and a single Wiener process driving the volatility in all rates (i.e. a one factor case). Under these conditions, we further explore the use of two forms of volatility: flat and piecewise. In the next paragraphs we develop the main steps used in the methodology.

2.1 The Libor Market Model

Following Brace, Gatarek, and Musiela (1997), we start by assuming that the marketed assets in an interest rate economy are zero-coupon bonds (or discount bonds) with different maturities. Let $D(t, T)$ denote the value at time t of a discount bond which pays 1 at maturity T. We further assume that if you invest your money in a money-market account for a given period, the interest earned over this period is quoted as a LIBOR rate. At the end of a period of length ΔT, one receives an interest equal to αL, where L denotes the LIBOR rate and $\alpha = \Delta T$ denotes the accrual factor or daycount fraction. Hence, we obtain the relationship $(1 + \alpha L)D(0, \Delta T) = 1$, which states that the present value (i.e. the value today) of the notional plus the interest earned at the end of a ΔT period is equal to the notional.

In addition, a forward LIBOR rate $L_{TS(t)}$ is the interest rate one can contract for at time t to put money in a money-market account for the time period $[T, S]$ in the future. We define the forward LIBOR rate using the relation

$$D(t,T) = [1 + \alpha_{TS} L_{TS}(t)]D(t,S) \tag{1}$$

where α_{TS} denotes the daycount fraction for the period $[T, S]$. Solving for L yields

$$L_{TS}(t) = \frac{1}{\alpha_{TS}}(\frac{D(t,T) - D(t,S)}{D(t,S)}) \tag{2}$$

The time T is known as the maturity of the forward LIBOR rate and $(S-T)$ is called the tenor.

At time T the forward LIBOR rate L_{TS} is fixed or set and is then called a spot LIBOR rate. Note, that the spot LIBOR rate is fixed at the beginning of the period at T, but is paid at the end of the period at S.

2.2 The LIBOR Process

In most markets the quoted forward LIBOR rates are associated with specific tenor ΔT, for example in the case of EURIBOR 1M, 2M, 3M, etc. This tenor could be for example 3 months (e.g., for USD) or 6 months (e.g., for EUR). Therefore, we assume there are N forward LIBOR rates with this specific tenor, which we denote by $L_i(t) = L_{T_i T_{i+1}}(t)$ and $T_i = i\Delta T$ for $i = 1, ..., N$, with daycount fractions $\alpha_i = \alpha_{T_i, T_{i+1}}$. For this set of LIBOR rates, we denote the associated discount factors by $D_i(t) = D(t, T_i)$.

At this point we note that the process

$$\alpha_i L_i(t) = \frac{D_i(t) - D_{i+1}(t)}{D_{i+1}(t)} \tag{3}$$

is a ratio of marketed assets. So, if we take the discount bond D_{i+1} as a numeraire and assuming that the economy is arbitrage-free, under the martingale measure Q^{i+1} associated with the numeraire D_{i+1} the process $\alpha_i L_i$ will be a martingale. As α is a constant, we can then deduct that the process $L_i(t)$ must also be a martingale under Q^{i+1}.

Given this crucial observation, Brace, Gatarek, and Musiela (1997), made the assumption that L_i can be modeled by the following stochastic differential equation:

$$dL_i(t) = \sigma_i(t) L_i(t) dW^{i+1} \qquad (4)$$

where W^{i+1} denotes a Wiener process under Q^{i+1}.

2.3 The Terminal Measure

In the classic LMM model, thus, the key rate to consider is the last rate. Using the last discount bond D_{N+1} as numeraire we will attempt to bring all the LIBOR rates under the same measure. The first step is to consider the application of the change of measure dQ^i/dQ^{i+1}. By repeated application of this change of measure for different i, we can bring all forward LIBOR processes under the same measure. Using the Change of Numeraire Theorem we know that the Radon-Nikodym derivative $\rho(t)$ is given by

$$\frac{dQ^i}{dQ^{i+1}} = \rho(t) = \frac{D_i(t)/D_i(0)}{D_{i+1}(t)/D_{i+1}(0)} = \frac{D_{i+1}(0)}{D_i(0)}(1 + \alpha_i L_i(t)) \qquad (5)$$

To apply Girsanov's Theorem we need to find the process $\kappa(t)$ such that

$$\rho(t) = \exp\left\{ \int_0^t \kappa(s) dW^{1+1}(s) - \frac{1}{2}\int_0^t \kappa(s)^2 ds \right\} \qquad (6)$$

An application of Ito's Lemma shows that $d\rho(t) = \rho(t)\kappa(t)dW^{i+1}$. Hence, $\kappa(t)$ is the "volatility" of the Radon-Nikodym derivative $\rho(t)$. Using the above we find

$$d\rho(t) = \frac{\alpha_i \sigma_i(t) L_i(t)}{1 + \alpha_i L_i(t)} \rho(t) dW^{i+1} \qquad (7)$$

and thus

$$dW^i = dW^{i+1} - \frac{\alpha_i \sigma_i(t) L_i(t)}{1 + \alpha_i L_i(t)} dt \qquad (8)$$

Under the terminal measure theorem, the terminal LIBOR rate L_N is a martingale. Therefore the process

$$dL_{N-1}(t) = \sigma_{N-1}(t) L_{N-1}(t) (dW^{N+1} - \frac{\alpha_N \sigma_N(t) L_N(t)}{1 + \alpha_N L_N(t)} dt) dL_{N-1}(t) \qquad (9)$$

dL$_{N-1}(t) = -\frac{\alpha_N \sigma_N(t) L_N(t)}{1+\alpha_N L_N(t)} \sigma_{N-1}(t) L_{N-1}(t) dt \sigma_{N-1}(t) L_{N-1}(t) dW^{N+1}$

If we apply this procedure repeatedly for all L back to front, we can derive that under the terminal measure the process

$$dL_i(t) = -\sum_{k=i+1}^{N} \frac{\alpha_k \sigma_k(t) L_k(t)}{1 + \alpha_k L_k(t)} \sigma_i(t) L_i(t) dt + \sigma_i(t) L_i(t) dW^{N+1} \qquad (10)$$

In this last equation we can see that, apart from the terminal LIBOR rate L_N, all LIBOR rates are no longer martingales under the terminal measure, but have a drift term that depends on the forward LIBOR rates with longer maturities. These equations therefore describe the dynamics of the evolution of the initial set of forward rates $L_i(0, T)$ into the future rates $L_i(t, T)$.

2.4 Monte Carlo Simulation

Given the complexity of the processes in the LIBOR market model, it is not possible to obtain a closed-form solution for all the forward rates. This is not a problem, however, as robust numerical methods can be used to solve. One important method that is widely used for market models is Monte Carlo simulation (see Glasserman (2003)).

To implement the LIBOR market model, we work under the terminal measure Q^{N+1}. We start by first, draw a path for the Brownian motion W^{N+1} at the reset dates T_n using the formula

$$W^{N+1}(T_{n+1}) = W^{N+1}(T_n) + \epsilon_n \sqrt{T_{n+1} - T_n} \tag{11}$$

where the ϵ_n are drawings from a standard normal distribution. Based on this path for the Brownian motion we can calculate all future forward LIBOR rates.

The forward LIBOR rates in the first column are the forward rates observed at time 0. In the subsequent columns we see for each reset date T_n the forward rates that depend on the Brownian motion $W^{N+1}(T_n)$. The forward rates $L_n(T_n)$ are the realisations of the spot LIBOR rates. In each column, the forward rate $L_i(T_{n+1})$ is updated using the discretisation

$$L_i(T_{n+1}) = L_i(T_n) - \sum_{k=i+1}^{N} \frac{\alpha_k \sigma_k(T_n) L_k(T_n)}{1 + \alpha_k L_k(T_n)} \sigma_k(T_n) L_i(T_n)(T_{n+1} - T_n) \tag{12}$$

$$+ \sigma_i(T_n)(W^{N+1}(T_{n+1}) - W^{N+1}(T_n))$$

Alternatively, we can use a discretisation based on the stochastic differential equation of $Log(L_i)$ which yields

$$L_i(T_{n+1}) = L_i(T_n) \exp\left\{[A](T_{n+1} - T_n) + \sigma_i(T_n)(W^{N+1}(T_{n+1}) - W^{N+1}(T_n))\right\} \tag{13}$$

and

$$A = -\sum_{k=i+1}^{N} \frac{\alpha_k \sigma_k(T_n) L_k(T_n)}{1 + \alpha_k L_k(T_n)} \sigma_i(T_n) - \frac{1}{2}\sigma_i(T_n)^2$$

Given a set of LIBOR rates realised on this path of the Brownian motion, we can calculate for any time point T_n the value of the discount bonds $D_i(T_n)$ for $i = n, ..., N + 1$ as

$$D_i(T_n) = \prod_{k=n}^{i-1} \left(1 + \alpha_k L_k(T_n)\right)^{-1} \tag{14}$$

All the LIBOR rates $L_i(t)$ and the discount factors $D_i(t)$ that we have just calculated can be conveniently organized in a matrix format that we will call the LIBOR TABLEAU and the DISCOUNTING TABLEU, respectively.

T_0	$W^{N+1}(T_1)$	$W^{N+1}(T_2)$	$W^{N+1}(T_3)$	...	$W^{N+1}(T_N)$
$L_1(0)$	$L_1(T_1)$				
$L_2(0)$	$L_2(T_1)$	$L_2(T_2)$			
$L_3(0)$	$L_3(T_1)$	$L_3(T_2)$	$L_3(T_3)$		
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	
$L_N(0)$	$L_N(T_1)$	$L_N(T_2)$	$L_N(T_3)$	...	$L_N(T_N)$

Table 1: Libor rates tableau

T_0	$W^{N+1}(T_1)$	$W^{N+1}(T_2)$	$W^{N+1}(T_3)$	...	$W^{N+1}(T_N)$
$D_1(0)$	$D_1(T_1)$				
$D_2(0)$	$D_2(T_1)$	$D_2(T_2)$			
$D_3(0)$	$D_3(T_1)$	$D_3(T_2)$	$D_3(T_3)$		
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	
$D_N(0)$	$D_N(T_1)$	$D_N(T_2)$	$D_N(T_3)$	...	$D_N(T_N)$

Table 2: Discount factors tableau

Now that once we have calculated the LIBOR rates and the discount factors for a particular path, *i.e.* a single Monte Carlo simulation. We are in a position to determine the payoff of a particular derivative. For example, a plain vanilla interest rate swap based on a LIBOR rate with a 3 month tenor (*e.g.* EURIBOR3M, will require the forward rates defined in the diagonal of the LIBOR TABLEAU, as these correspond to the 3M forward rates as seen every 3M in the future. Once these rates are identified we need to compte the relevant cashflows and discount each of them using the appropriate discount factors, for this individual Monte Carlo realisation, as defined in the DISCOUNTING TABLEAU. The volatilities $\sigma_k(T_n)$ to be used are defined in the VOLATILITY TABLEAU.

	T_1	T_2	T_3	...	T_N
$L_1(0)$	$\sigma(t,T)$				
$L_2(0)$	$\sigma(t,T)$	$\sigma(t,T)$			
$L_3(0)$	$\sigma(t,T)$	$\sigma(t,T)$	$\sigma(t,T)$		
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	
$L_N(0)$	$\sigma(t,T)$	$\sigma(t,T)$	$\sigma(t,T)$	...	$\sigma(t,T)$

Table 3: Volatility tableau

The final step in the pricing procedure is to use again the numeraire, to "bring back" the price under the natural measure. This a kind of discounting we must apply to the payoff and is called deflation (Pelsser, 2000). Let $V'(t)$ denote the value at time t of the payoff $V(t)$ divided by the numeraire.The payoff $V'(T_{n+1})$ is determined at time T_n but is not received until time T_{n+1}. Hence, we can calculate the numeraire rebased payoff at time T_{n+1} as

$$V^{'}(T_{n+1}) = \frac{V(T_{n+1})}{D_{N+1}(T_{n+1})}(15)$$

2.5 Volatility Assumptions

The volatility function $\sigma_k(t)$ can take many functional forms (see Brigo and Mercurio (2006). The particular form to be used is a choice of the analyst. In line with our intention to use the simplest version of the LMM one factor, in this paper we use the two simples forms: (a) a constant volatility denoted FLAT, and (b) a time-homogeneous form. These are illustrated in the tables below.

	T_1	T_2	T_3	...	T_N
$L_1(0)$	σ				
$L_2(0)$	σ	σ			
$L_3(0)$	σ	σ	σ		
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	
$L_N(0)$	σ	σ	σ	...	σ

Table 4: FLAT volatility tableau

	T_1	T_2	T_3	...	T_N
$L_1(0)$	σ_1				
$L_2(0)$	σ_2	σ_1			
$L_3(0)$	σ_3	σ_2	σ_1		
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	
$L_N(0)$	σ_N	σ_{N-1}	σ_{N-2}	...	σ_1

Table 5: Piecewise volatility tableau

2.6 Implementation

The above methodology has been implemented in the programming language ANSI C++. We created an external library to the financial pricing software Reuters Kondor+ Structured Products (Version 1.2) under the operating system Sun Solaris (also compiled under Windows). These form part of a larger library consisting of 400 classes and *circa* 24,000 lines of code.

Different random generators used for the Monte Carlo engine include the Merseney-Twister algorithm and the Sobol quasi-random sequence. Also, our Monte Carlo engine used an antithetic approach. An example of the headers written for the C++ implementation is shown in Appendix 1.

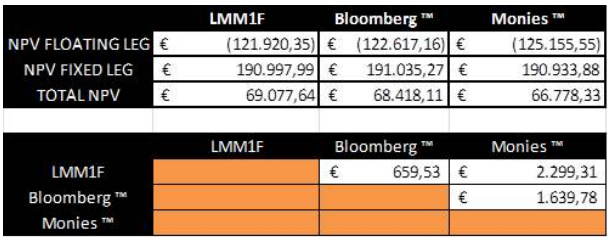
We have also implemented parts of the LMM formulas using the software Wolfram Mathematica and Microsoft Excel.

3 Results

3.1 Example 1: Plain Vanilla Interest Rate Swap

As a first example of the application of our methodology we price a plain vanilla interest rate swap. The terms of the contract are as follows: Bank pays counterparty: EURIBOR6M+20 bps, semi-annual payments (every 6 months), ACT/360. Counterparty pays bank: 4%, annual payments (every year), ACT/360. Notional: one million EUR. Start date: 1 Dec 2010. End date: 1 Dec 2015.

The results of the pricing are shown below in Table 6 using the RAN2 random generator and 5000 Monte Carlo simulations. Our results LMM1F are presented for the FLAT volatility assumption defined in the TABLEUX described in the Methodology. We compare our results against Bloomberg and Sungard Monies.



	LMM1F	Bloomberg ™	Monies ™
NPV FLOATING LEG	€ (121.920,35)	€ (122.617,16)	€ (125.155,55)
NPV FIXED LEG	€ 190.997,99	€ 191.035,27	€ 190.933,88
TOTAL NPV	€ 69.077,64	€ 68.418,11	€ 66.778,33

	LMM1F	Bloomberg ™	Monies ™
LMM1F		€ 659,53	€ 2.299,31
Bloomberg ™			€ 1.639,78
Monies ™			

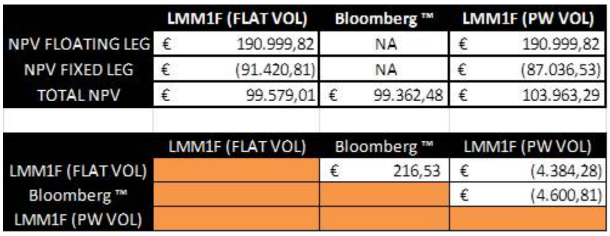
Table 6: Example 1 results

We further present a detailed comparison of the cashflows between the three methods in Figures 1, 2, 3, 4, 5. As we can see the prices of LMM1F and Bloomberg are very close indeed. There is a difference of only 650 EUR. The difference the LMM1F is about 2200 EUR, which is similar to the difference between Bloomberg and Monies. All of these differences are still in the range of about 1000-2200 EUR, which in a notional of 1 million EUR are acceptable.

3.2 Example 2: Interest Rate Swap With CAP

As a second example of the application of our methodology we price a capped interest rate swap. The terms of the contract are as follows: Bank pays counterparty: EURIBOR6M+20 bps, semi-annual payments (every 6 months), ACT/360, with a max(EURIBOR6M,2.30%). Counterparty pays bank: 4%, annual payments (every year), ACT/360. Notional: one million EUR. Start date: 1 Dec 2010. End date: 1 Dec 2015.

The results of the pricing are shown below in Table 7 using the RAN2 random generator and 5000 Monte Carlo simulations. Our results LMM1F are presented for two volatility assumptions, FLAT and PIECEWISE, as defined in the TABLEUX described in the Methodology. We compare against Bloomberg.



	LMM1F (FLAT VOL)	Bloomberg ™	LMM1F (PW VOL)
NPV FLOATING LEG	€ 190.999,82	NA	€ 190.999,82
NPV FIXED LEG	€ (91.420,81)	NA	€ (87.036,53)
TOTAL NPV	€ 99.579,01	€ 99.362,48	€ 103.963,29

	LMM1F (FLAT VOL)	Bloomberg ™	LMM1F (PW VOL)
LMM1F (FLAT VOL)		€ 216,53	€ (4.384,28)
Bloomberg ™			€ (4.600,81)
LMM1F (PW VOL)			

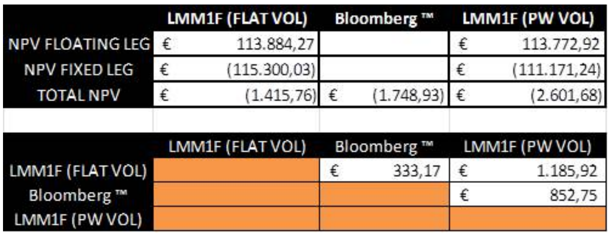
Table 7: Example 2 results

As we can see the prices of LMM1F(FLAT) and Bloomberg are very close indeed. LMM1F and Bloomberg both use the same constant volatility of 38.32%. There is a difference of only 216 EUR. The difference of the price using the LMM1F(PW) is larger but still in the range of about 4000-5000 EUR, which in a notional of 1 million EUR are acceptable. We show some further redetails of the results in the screenshots presented in Figures 7 and 8.

3.3 Example 3: Interest Rate Swap CORRIDOR

As a third example of the application of our methodology we price a structured interest rate swap known as CORRIDOR. The terms of the contract are as follows: Bank pays counterparty: EURIBOR6M-20 bps, semi-annual payments (every 6 months), ACT/360, as long as the benchmark rate EURIBOR6M is outside the corridor defined by the rates 3% and 4%. If the benchmark rate EURIBOR6M is inside the corridor, then the bank pays counterparty 5%.Counterparty pays bank EURIBOR6M every six months, ACT/360. Notional: one million EUR. Start date: 1 Dec 2010. End date: 1 Dec 2015.

The results of the pricing are shown below in Table 8using the RAN2 random generator and 5000 Monte Carlo simulations. Our results LMM1F are presented for two volatility assumptions, FLAT and PIECEWISE, as defined in the TABLEUX described in the Methodology. We compare against Bloomberg.



	LMM1F (FLAT VOL)	Bloomberg ™	LMM1F (PW VOL)
NPV FLOATING LEG	€ 113.884,27		€ 113.772,92
NPV FIXED LEG	€ (115.300,03)		€ (111.171,24)
TOTAL NPV	€ (1.415,76)	€ (1.748,93)	€ (2.601,68)

	LMM1F (FLAT VOL)	Bloomberg ™	LMM1F (PW VOL)
LMM1F (FLAT VOL)		€ 333,17	€ 1.185,92
Bloomberg ™			€ 852,75
LMM1F (PW VOL)			

Table 8: Example 3 results

As we can see the prices of LMM1F(FLAT) and Bloomberg are very close indeed. LMM1F and Bloomberg both use the same constant volatility of 20%. There is a difference of only 300 EUR. The difference of the price using the LMM1F(PW) is larger but still in the range of about 1000 EUR, which in a notional of 1 million EUR are acceptable. Note that this example is deep in the money and thus more likely to be sensitive to volatility. We show further details in the screenshots presented in Figures 9 and 10.

4 Conclusions

Our results show that there is a good agreement between the LMM1F and two standard derivatives pricing libraries (Bloomberg and Sungard Monis). Even though we focused on a limited sample of deals, consisting of a plain vanilla interest rate swap, an interest rate swap with a cap and a corridor interest rate swap, our results show consistently convergent results. The prices calculated using LMM1F, Bloomberg and Monis, are in all cases only a few thousand euros apart in deals with a notional of one million euros, corresponding to differences of only a few basis points in terms of notional-weighted errors. Our results also show that the simplest version of the Libor Market Model is capable of capturing the prices of some of the most common interest rate derivatives in the market, namely structured swap.

We also show that the effect of various volatility assumptions is limited. When using flat and piecewise volatility the price of the instruments we studied changed only marginally. The role of volatility however is likely to be more prominent when considering a multifactor LMM. We also show that we can achieve reasonable convergence with a relatively small number of Monte Carlo simulations (5,000 in all examples). Taken together these results support the notion that when pricing interest rate derivatives based on LIBOR or EURIBOR rates, a one factor LMM could provide a reasonable enough approximation. A one factor LMM is simple to implement and fast enough to compute (all our examples ran in a less than ten seconds).

The present methodology has certainly limitations. The LMM1F is surely not a wise choice to price more complex interest rate derivatives, such as those based on constant maturity swap (CMS) rates or CMS spreads. An example could be an interest rate swaps in which a floating leg could pay a factor of a EUR CMS 10Y rate. Another example could be an interest rate swaps in which a floating leg pays a factor of the difference between a EUR CMS 10Y rate minus a EUR CMS 2Y rate. These derivatives are likely to require a multifactor LMM in order to be able to consider the full evolution dynamics of the forward rates and the convexity involved in the CMS rates.

Future work includes the ideas of an extension to a "plain" multifactor LMM model, particularly in terms of volatility and correlation modelling. We also intend to use the LMM1F to price a number of other more complex interest rate derivatives based on LIBOR or EURIBOR rates.

5 Commercial Products

- Reuters Kondor+ Structured Products is a trademark of Thomson Reuters Corporation.
- Bloomberg Terminal is a trademark of Bloomberg L. P..
- Sungard Monis is a trademark of SunGard Data Systems Inc..

6 Acknowledgements

The authors would like to express their gratitude for the financial support of CAREFIN - Centre for Applied Research in Finance, Bocconi University, Milan, Italy, and his director Prof Andrea Resti. Also, we would like to thank some colleagues for their support during the preparation of this work: Fabrice Aresu, Paolo Raviola, Gianluca Pedulla, Cedric Roumilhac and Irens Ibro.

References

[1] Antoon Pelsser, Efficient Methods for Valuing Interest Rate Derivatives, Springer-Verlag, 1st Edition, 2000.

[2] Brace A, Gatarek D, Musiela M, *The Market Model of Interest Rate Dynamics*, Mathematical Finance 7, 1997.

[3] Paul Glasserman, *Monte Carlo Methods in Financial Engineering*, Springer Verlag, 2003.

[4] Riccardo Rebonato, *Interest-Rate Option Models*, Second Edition, Wiley, 2000.

[5] Mark S. Joshi, *The Concepts and Practice of Mathematical Finance*, Cambridge University Press, 2003.

[6] Damiano Brigo and Fabio Mercurio, *Interest Rate Models - Theory and Practice: With Smile, Inflation and Credit*, Springer Verlag, 2006.

7 APPENDIX 1: C++ Implementation

Below we show an example of a header file which implements the LMM1F, in this case the constructor that sends the input parameters to the BGM.

```
BGMRateModel(

const std::vector<double>& p_LiborRates,// initial LiborRate at time=zero
const std::vector<double>& p_discount,  // initial dfs at time =zero
const std::vector<double>& p_sigma,     // flat volatility for each LiborRates
const std::vector<double>& deltaTime,   // between resetDate for each Libor
const std::vector<double>& dcf,         // between AccrualDate for each Libor
RandomNumbers* p_rand,
Payoff* p_payoff=NULL,
const MCparams* p_mc=new MCparams()

);
```