Abstract

The popularity of the so-called Market Models has led researchers and practitioners to ask two important questions about modelling interest-rate derivatives. The first (and highly contentious) question is, how many stochastic drivers are needed to value accurately any given derivative? The second, which arises because of the high dimensionality of Market Models, even those with a small number of stochastic drivers, is how can callable products be valued using Monte Carlo simulation?

In this paper we consider the Longstaff-Schwartz algorithm, an effective algorithm developed to answer the second of these questions, and in so doing we shed light on the first of these questions. We show that the success of the Longstaff-Schwartz algorithm for high-dimensional models demonstrates that, in a way we make precise, low-dimensional models are sufficient, but that in another sense the higher dimensionality still plays a part.

Using the insight gained from this analysis we go on to develop models which have these desirable properties - high dimensionality and accurate calibration properties on the one hand, but the ability to collapse the models onto a lower-dimensional ‘core’ models for the purposes of valuing callable derivatives. The core models that we develop are Markovian and can thus be implemented efficiently using lattice methods, avoiding the need for more costly Monte Carlo simulation.

Contents

1 Introduction

2 Longstaff-Schwartz

2.1 Review of Longstaff-Schwartz

2.2 Idealised Longstaff-Schwartz

2.3 Optimality of Idealised Longstaff-Schwartz

2.4 Dimensionality of Idealised Longstaff-Schwartz

2.5 Approximate Markov property of Longstaff-Schwartz

*Citigroup Centre, Canada Square, Canary Wharf, London, E14 5LB
†Department of Statistics, University of Warwick, Coventry, CV4 7AL
3 Reducible Markov-functional models

3.1 Recap of standard Markov-functional approach

3.1.1 One-dimensional case

3.1.2 Multi-dimensional case

3.2 Multiplicative DF reducible Markov-functional model

3.2.1 Calibration

3.2.2 Implementation

3.2.3 Examples

3.3 Multiplicative LIBOR reducible Markov-functional model

3.3.1 Calibration

3.3.2 Implementation

3.4 Example product valuations

3.4.1 Callable inverse floater

3.4.2 Callable ratchet swap

4 Extensions

4.1 Approximate multiplicative model

4.2 Calibration and fitting

4.3 The core model and index calibration

4.4 Product valuation

4.5 Other extensions
1 Introduction

The introduction of so-called ‘Market Models’ by Brace, Gatarek and Musiela [6] and independently by Miltersen, Sandmann and Sondermann [18], has had dramatic effects on how practitioners think about and implement interest rate models. Prior to Market Models, models were formulated either in terms of a short rate ([5], [12]) or in terms of instantaneous forward rates [11]. As a result the properties of the model were hard to understand and the effect of model parameters on the behaviour of the models was usually rather opaque.

Market Models (which we will often refer to as BGM models) are formulated directly in terms of market observable rates, such as LIBORs, their volatilities and correlations. As a result it is much easier to understand these models and to implement a version with realistic properties. The models have been extended in many ways, including the addition of stochastic volatility, skews and jumps, and a large number of numerical studies have been carried out. Much of this can be found summarised in the books and articles by Rebonato [21], [22].

Whereas the Market Models provide an excellent framework for selecting and understanding a model, they do have one significant drawback. An accurate implementation of a Market Model can only be done by simulation because of the high dimensionality of the model. This is true even when there is only one stochastic driver for the model. This is a problem for non-callable, path-dependent products because it is time consuming to generate enough Monte Carlo paths to get a price with sufficient accuracy that the ‘Greeks’ (risk sensitivities) will be accurate enough to use in practice. For callable products the problem is even more acute since simulation is a poor tool for performing a calculation which needs to be performed by working backwards in time.

The first of these difficulties, calculation speed even for non-callable products, can be alleviated in several ways – see Piterbarg [20] for a discussion of some of them. One significant step which does help is to use parallel computation when calculating risk, and this is something many banks are now doing. But currently it is still the case, we believe, that most banks do not use Market Models when calculating risk. Instead they fall back on one of the earlier short-rate models, such as Hull-White [12].

The second problem, of how to value callable products via simulation, has also received a lot of attention. Several different algorithms have been developed. Glasserman [10] provides a summary of these techniques, which he categorises as parametric (see [1] for this applied to Bermudan swaption pricing), random tree [7], statespace partitioning [2] and stochastic mesh/regression [8], [17].

One of the most successful algorithms, and the one adopted most widely by practitioners, is the Longstaff-Schwartz algorithm [17]. This is a regression-based approach and it is the starting point for this paper. Its success is due to its general applicability combined with its ability to deal with both high-dimensional models and multiple exercise times. All the other techniques fail on at least one of these criteria. The accuracy of the Longstaff-Schwartz algorithm [16] has made it feasible to implement Market Models for the pricing (if not the risk calculations) of callable products. We will describe the algorithm in detail in the next section.

The successful use of the Longstaff-Schwartz algorithm with Market Models leads to an interesting question. Many practitioners are now arguing that Market Models must be of high, or even full, rank. By full rank we mean that there should be a separate stochastic driver for each underlying LIBOR. Because implementation is done using simulation and the Longstaff-Schwartz algorithm, it turns out that the marginal cost in terms of implementation speed for adding extra factors is low (unlike when one uses a lattice approach to pricing). So if this is indeed the case, it presents no additional practical difficulties to add more factors. On the other hand, part of the Longstaff-Schwartz algorithm is to approximate the exercise region, and all the studies we have seen show that only a low number of factors are needed to estimate this exercise region with enough accuracy to get an acceptable valuation for the product. Fur-
thermore, banks have successfully run their trading operations to date using only low-factor models, and continue to do so for the purposes of risk. So where does the truth lie?

In this paper we examine this question by analysing the Longstaff-Schwartz algorithm. We present an improved version of the algorithm, the Idealised Longstaff-Schwartz algorithm of Section 2.2. This improved algorithm is not one that could be implemented in practice, but it is what the standard Longstaff-Schwartz algorithm is aspiring to achieve. We show that using this idealised algorithm corresponds, in all but one important respect, to using a model with the dimension of the decision variables and, furthermore, we present a proof that this Idealised Longstaff-Schwartz algorithm is optimal amongst all such algorithms.

The only place in which the full model dimensionality is used is in the calculation of the intrinsic value of the option being valued, the calculation of the value of the underlying non-callable product. Once this intrinsic value has been evaluated in the full model, the algorithm immediately calculates the expected value of this intrinsic given the decision variables. It is only this conditional expectation that is used by the algorithm. So the algorithm only uses a low-dimensional summary of the intrinsic value, but the full model is used to generate this summary.

This analysis of the Idealised Longstaff-Schwartz algorithm shows that, with the exception of calculating the initial intrinsic value summary, when using the Longstaff-Schwartz algorithm most of the extra dimensions added to the original Market Model are redundant. The extensive work done that shows the Longstaff-Schwartz algorithm is successful has demonstrated that this conclusion is not specific to this algorithm but that it applies to modelling in general: with the exception of the calculation of the products intrinsic value, a low-dimensional (Market) model should suffice for pricing.

This conclusion is very much in line with the way practitioners have used models for many years, and explains why low factor models are still accurate enough to be used in practice. To value a callable interest rate product accurately you first need to capture the intrinsic value accurately. Having done this, you need a model which captures the main market dynamics - and the first two factors will suffice in a single currency interest-rate model.

The adequacy of low factor models, a desire to avoid Monte Carlo simulation, combined with the need to capture the intrinsic value of a product correctly, leads us to ask whether it would be possible to develop models which are of high dimension for the purposes of calculating intrinsic values, but which can quickly be collapsed to a low-dimensional Markovian model. In Section 3 we present some models of this type. The ideas we use in constructing these models are extensions of those used to develop Markov-functional models [14]. The techniques are powerful because they allow a model to be built which is both Markovian and has the desired marginal distributions – the important calibration property which makes Market Models so appealing.

The ability to formulate a `core ’ model which captures the important dynamics in the market, but which can be extended to more factors to capture intrinsic values correctly is practically important. The question then arises as to whether this idea can be taken further. Is it possible to build a model around the core model with even better calibration properties without distorting the model dynamics? Can this be done without overfitting, which would result in a model that accurately matches the prices of vanilla instruments but which is then a poor model for valuing more complex callable products. In Section 4 we present some models, inspired by the ideas in Section 3, which have very accurate calibration properties but do not adversely affect the model’s dynamics. The price we pay for this is that the models are not arbitrage-free. However the degree to which the models admit arbitrage is limited by a number of factors, as we explain in Section 4, which makes them potentially very useful in practice.
2 Longstaff-Schwartz

The Longstaff-Schwartz algorithm for pricing multi-callable products via simulation has been widely adopted in the interest rate derivatives market (see, for example, [10], [15] and [20] amongst others). It has the advantage that it can easily be adapted to apply to a range of exotic products, and numerical studies ([15], [20]) have shown it to be effective. Here we present this algorithm and perform an analysis upon it. We shall always couch the discussion in interest rate nomenclature, but the arguments apply equally well to multi-asset and multi-currency models.

First some notation which will be needed later. We suppose that the state of the economy can be summarised by some Markov process \((x, \varepsilon)\) defined on the complete filtered probability space \((\Omega, \{\mathcal{F}_t\}, \mathbb{P})\), where \(\mathcal{F}_0\) is assumed to be trivial. We impose no constraint on the dimension of this Markov process, thus we are including all models that are used in practice. In particular, multi-currency and multi-asset models are included, all BGM models are included, as are models which incorporate stochastic volatility. The reason for us splitting the Markov process in two parts, \(x\) and \(\varepsilon\), will become clear in Section 3 but, roughly speaking, \(x\) relates to the major variability in the model, perhaps the first few principal components of price movements and \(\varepsilon\) relates to the rest.

Because our prime interest is in interest-rate models, we will present our discussion of the Longstaff-Schwartz algorithm in terms of Bermudan-callable products rather than more general American-callable products. Throughout we will denote by \(T_1, \ldots, T_n\) the dates on which the product can be exercised. We will consider a product that allows the option holder the right, at any \(T_i\), to enter into some underlying exotic trade. We will often write \(x_i\) and \(\varepsilon_i\) when we more accurately mean \(x_{T_i}\) and \(\varepsilon_{T_i}\), and similarly for other variables.

Throughout this section we will be working in a single equivalent martingale measure. To reduce the notational burden we will then absorb the numeraire into our notation. So we will let \(V_i(x_i, \varepsilon_i)\) represent the numeraire-rebased value, at time \(T_i\), of the product. The intrinsic value \(V_i(x_i, \varepsilon_i)\) of the option at \(T_i\) is its value if we choose to exercise at \(T_i\) and enter the underlying trade. Similarly the extrinsic value at \(T_i\) is the value of the option if we decide not to exercise at \(T_i\) but instead to wait until a later time. We will denote by \(I_i(x_i, \varepsilon_i)\) and \(E_i(x_i, \varepsilon_i)\) respectively the numeraire-rebased intrinsic and extrinsic values. Note that, implicit in the above, is the assumption that we follow the optimal strategy when making exercise decisions. Later we will want to deal with the option value under a given, possibly suboptimal, strategy \(S\). When we do so, to avoid overburdening the notation, we will when the context is clear keep to this same notation and not explicitly introduce the dependence on \(S\).

2.1 Review of Longstaff-Schwartz

Full pricing of a multi-callable product by simulation would be too onerous to carry out. To get around this, the Longstaff-Schwartz algorithm makes an approximation in order to make its exercise decision. Recall the usual pricing relationships that hold for a multi-callable product:

\[
V_i(x_i, \varepsilon_i) = \max \{ I_i(x_i, \varepsilon_i), E_i(x_i, \varepsilon_i) \},
\]

\[
E_i(x_i, \varepsilon_i) = \mathbb{E} \left[ V_{i+1}(x_{i+1}, \varepsilon_{i+1}) \mid \mathcal{F}_i \right]
= \mathbb{E} \left[ I_{i+1}(x_{i+1}, \varepsilon_{i+1}) I_{i+1}(x_{i+1}, \varepsilon_{i+1}) \geq E_{i+1}(x_{i+1}, \varepsilon_{i+1}) \right] \mid \mathcal{F}_i \right] + \mathbb{E} \left[ E_{i+1}(x_{i+1}, \varepsilon_{i+1}) I_{i+1}(x_{i+1}, \varepsilon_{i+1}) < E_{i+1}(x_{i+1}, \varepsilon_{i+1}) \right] \mid \mathcal{F}_i \right] \ (1)
\]

Note that we have implicitly assumed that the underlying is not path dependent in these equations. In many cases where this is the case this limitation can be removed by augmenting the statespace.

Exact evaluation of these expectations is not feasible when using Monte Carlo simulation. As a result the Longstaff-Schwartz algorithm approximates the exercise regions

\[
R_i := \{ \omega \in \Omega : E_i((x_i, \varepsilon_i)(\omega)) \leq I_i((x_i, \varepsilon_i)(\omega)) \}
\]
for each $i = 1, \ldots, n$ (if $\omega \in R_i$, then we exercise at $T_i$, if we have not already done so). This approximation is performed using explanatory variables and basis functions, as we now describe.

The idea is as follows. For each $i = 1, \ldots, n$ we use an approximation of the form

$$
\begin{align*}
I_i(x_i, \varepsilon_i) &\approx \bar{I}^b_i := \sum_j \hat{\alpha}_{ij} f_j^b(\eta_i(x_i, \varepsilon_i)), \\
E_i(x_i, \varepsilon_i) &\approx \bar{E}^b_i := \sum_j \hat{\beta}_{ij} g_j^b(\eta_i(x_i, \varepsilon_i)).
\end{align*}
$$

(2)

The (vector) variable $\eta_i$ is the explanatory variable. The idea is that $\eta_i$ will contain most of the relevant information which determines the exercise decision at $T_i$. So, typically, $\eta_i$ might be of dimension two, with one component being a LIBOR and the other being some longer swap rate (stochastic volatility might also be included). The basis functions $f_j^b$ and $g_j^b$ are now chosen with the objective that, for suitable constants $\hat{\alpha}_{ij}$ and $\hat{\beta}_{ij}$, the approximations in (2) will be accurate.

The parameters $\hat{\alpha}_i := (\hat{\alpha}_{ij})$ and $\hat{\beta}_i := (\hat{\beta}_{ij})$ are chosen by least-squares regression using a set of Monte-Carlo paths $\omega_k$ pre-simulated for this purpose:

$$
\begin{align*}
\hat{\alpha}_i &= \arg \min_{\alpha_i} \frac{1}{n_{\text{sim}}} \sum_{k=1}^{n_{\text{sim}}} \left( \bar{I}_i(\omega_k) - \sum_j \alpha_{ij} f_j^b(\eta_i(x_i, \varepsilon_i)(\omega_k)) \right)^2, \\
\hat{\beta}_i &= \arg \min_{\beta_i} \frac{1}{n_{\text{sim}}} \sum_{k=1}^{n_{\text{sim}}} \left( \bar{E}_i(\omega_k) - \sum_j \beta_{ij} g_j^b(\eta_i(x_i, \varepsilon_i)(\omega_k)) \right)^2.
\end{align*}
$$

(3)

Here $n_{\text{sim}}$ is the total number of simulation paths, the $k$th being denoted by $\omega_k$, and, for each random variable $X$ appearing in (3), $X(\omega_k)$ is its $k$th simulated value.

There are a number of variants for the definition of the quantities $\bar{E}$ and $\bar{I}$, as we now describe. Most straightforward to deal with is $\bar{I}_i(\omega)$. If the numeraire-rebased intrinsic value at $T_i$ for the underlying product is known explicitly as a function of $(x_i, \varepsilon_i)$, then this is the value assigned to $\bar{I}_i(\omega) \equiv \bar{I}_i((x_i, \varepsilon_i)(\omega))$. This would, for example, be the case for a Bermudan swaption since in this case the underlying value is a function of the known discount and forward LIBOR curves at time $T_i$. For other products, such as callable ratchet swaps, the intrinsic is itself a highly complex function of interest rates, volatilities and correlations. In this latter case $\bar{I}_i(\omega)$ is taken to be the sum of all future numeraire-rebased payments (for this sample path $\omega$) that would be made if the decision is taken to exercise at $T_i$.

The extrinsic $\bar{E}_i(\omega)$ is less straightforward. This has to be found by working backwards in time. The simplest approach, having already calculated $\bar{E}_{i+1}$ and $\bar{I}_{i+1}$, is to set

$$
\bar{E}_i(\omega) := \max \{ \bar{E}_{i+1}(\omega), \bar{I}_{i+1}(\omega) \}.
$$

A second alternative is the let $\tilde{E}_i(\omega)$ be the sum of all future numeraire-rebased payments (for this sample path $\omega$) if the decision is taken not to exercise at $T_i$ but then to follow the appropriate strategy – exercise only when $\bar{E}_k(\omega) \leq \bar{I}_k(\omega)$ for $k > i$.

Remark 1: What we have just described is one variant of the Longstaff-Schwartz algorithm. Note that we have chosen to fit the basis functions to approximate the numeraire-rebased intrinsic and extrinsic values. This is slightly non-standard. It is more common, we believe, to fit the intrinsic and extrinsic directly. As we shall see in Section 2.3, the approach presented here is slightly better. All our conclusions remain valid whichever choice is made at this point.
Once the exercise rule in Longstaff-Schwartz has been determined, the final valuation is carried out by forward simulation using a completely new set of random paths. The total numeraire-rebased payment on each sample path is calculated, with exercise occurring at \( \hat{\tau}_b(\omega) \) := min \( \{ i \geq 1 : \hat{E}_b^i((x_i, \varepsilon_i)(\omega)) \leq \hat{I}_b^i((x_i, \varepsilon_i)(\omega)) \} \). This total numeraire-rebased payment is then averaged over all the sample paths. Using new paths ensures that there is no bias in the algorithm, which will then generate a lower bound on the price (since the exercise strategy is suboptimal). Thus the valuation process does not involve the explicit evaluation by backwards induction of an equation such as (1) (but such a relationship does of course remain true).

Research on the Longstaff-Schwartz algorithm has shown it to be an effective scheme if used with care. The quality of the results can be assessed by comparison with upper bounds on the value of the same options. See, for example, the work by Joshi and Theis [16] on Bermudan swaptions. The following general properties have been observed, as is well summarised in [20]:

(i) For single currency products, typically one or at most two explanatory variables are required.

(ii) These two factors need to capture the level and tilt of the curve. Beyond that, the exact choice of the variables selected is less important.

(iii) The choice of basis functions is important, and they should be chosen with the particular pricing problem in mind.

### 2.2 Idealised Longstaff-Schwartz

We will now extend and improve the Longstaff-Schwartz algorithm just described in a number of ways. In doing so we will see what the algorithm is ‘aspiring to achieve’. It is only the practicalities of simulation that prevent this ideal from being attained.

As a first extension, suppose that we could use an arbitrary number of sample paths in the least-squares regression, equation (3). Then, by the strong law of large numbers, the estimators \( \hat{\alpha}_i \) and \( \hat{\beta}_i \) would be the solutions to

\[
\hat{\alpha}_i = \arg \min_{\alpha_i} \mathbb{E} \left[ \left( \hat{I}_i - \sum_j \alpha_{ij} f_j(\eta_i(x_i, \varepsilon_i)) \right)^2 \right],
\]

\[
\hat{\beta}_i = \arg \min_{\beta_i} \mathbb{E} \left[ \left( \hat{E}_i - \sum_j \beta_{ij} g_j(\eta_i(x_i, \varepsilon_i)) \right)^2 \right].
\]

A further extension, which would yield a further (significant) improvement, would be to extend the random variables over which we minimise. Recall that we are trying, in equation (2), to find random variables \( \hat{E}_i^b \) and \( \hat{I}_i^b \) to approximate \( E_i^b \) and \( I_i^b \). If these random variables are closer to their targets (in a least squares sense), one would expect (but could not guarantee, of course) the result to be more accurate – a tighter lower bound since the strategy will be closer to optimal. A natural extension would be to remove the dependence on the \( \eta_i \) through the basis functions and allow any \( \sigma(\eta_i) \)-measurable random variable. Even better would be to allow any \( \mathcal{G}_i \)-measurable random variable, where

\[
\mathcal{G}_i := \sigma(\eta_1, \ldots, \eta_i) .
\]

We denote the approximating random variables in this case by \( \hat{E}_i^\mathcal{G} \) and \( \hat{I}_i^\mathcal{G} \). The problem of solving (4) is now replaced by having to find \( \mathcal{G}_i \)-measurable \( \hat{E}_i^\mathcal{G} \) and \( \hat{I}_i^\mathcal{G} \) that minimise...
The solution to this is standard and well-known:

\[
\hat{I}_i^c = E[I_i | G_i], \quad \hat{E}_i^c = E[E_i | G_i].
\]

These estimates can be used to specify an exercise strategy in exactly the same way as discussed earlier for the usual Longstaff-Schwartz algorithm. Note, before moving on, that \(\hat{I}_i^c\) and \(\hat{E}_i^c\) (in the case where \(E_i\) is calculated from future payments made when following the strategy) can be related directly to the intrinsic and extrinsic values

\[
\hat{I}_i^c = E[I_i | G_i] = E[E[I_i | F_i] | G_i] = E[I_i(x_i, e_i) | G_i],
\]

\[
\hat{E}_i^c = E[E_i | G_i] = E[E[E_i | F_i] | G_i] = E[E_i(x_i, e_i) | G_i].
\] (5)

**Remark 2:** Equation (5) has an important consequence. Considering the decision at \(T_{i+1}\) yields

\[
E_i = E[I_{i+1} \mathbf{1}\{\hat{I}_{i+1}^c \geq \hat{E}_{i+1}^c\} + E_{i+1} \mathbf{1}\{\hat{I}_{i+1}^c < \hat{E}_{i+1}^c\}|F_i]
\]

and thus

\[
\hat{E}_i^c = E[I_{i+1} \mathbf{1}\{\hat{I}_{i+1}^c \geq \hat{E}_{i+1}^c\} + E_{i+1} \mathbf{1}\{\hat{I}_{i+1}^c < \hat{E}_{i+1}^c\}|G_i]
\]

\[
= E[I_{i+1} \mathbf{1}\{\hat{I}_{i+1}^c \geq \hat{E}_{i+1}^c\} + E_{i+1} \mathbf{1}\{\hat{I}_{i+1}^c < \hat{E}_{i+1}^c\}|G_i]
\]

\[
= E[I_{i+1} \mathbf{1}\{\hat{I}_{i+1}^c \geq \hat{E}_{i+1}^c\} + \hat{E}_{i+1} \mathbf{1}\{\hat{I}_{i+1}^c < \hat{E}_{i+1}^c\}|G_i]
\]

\[
= \max\left\{\hat{I}_{i+1}^c, \hat{E}_{i+1}^c\right\}|G_i.\] (6)

This gives a way to determine \(\hat{E}_i^c\) inductively from knowing only the values \(\hat{I}_i^c\).

**Remark 3:** Note the similarity between (1) and (6). However it is not, in general, true that \(\hat{E}_i^c = E[E_i^{opt} | G_i]\) where \(E_i^{opt}\) denotes the numeraire-rebased extrinsic value for the optimal strategy. The extrinsic value in equation (5) is the one arising from following the particular strategy under consideration, and not the optimal one. In particular, \(E[E_i^{opt} | G_i]\) does not solve an equation such as (6).

### 2.3 Optimality of Idealised Longstaff-Schwartz

Longstaff-Schwartz generates an exercise strategy via the sequence of sets \(\{R^b_1, \ldots, R^b_n\}\) where

\[
R^b_i := \left\{\omega \in \Omega : \hat{E}_i^b((x_i, e_i)(\omega)) \leq \hat{I}_i^b((x_i, e_i)(\omega))\right\}
\]

For this strategy recall that the exercise time is defined by

\[
\hat{\tau}_i^b(\omega) := \min\left\{i \geq 1 : \omega \in R_i^b\right\}.
\]

Note that any exercise strategy can be defined in a similar way, but with a different choice of sets replacing the \(R_i^b\).
We can show that the Idealised Longstaff-Schwartz algorithm is optimal amongst all \( \{G_i\} \)-measurable exercise strategies. To see this, consider any other \( \{G_i\} \)-measurable strategy \( S \) specified by the sequence of sets \( \{A_i, i = 1, \ldots, n\} \) where \( A_i \in G_i \). Let \( V_0^S \) denote the numeraire-rebased time-zero value of the option under this strategy, and let \( V_0^G \) denote the same quantity but under the Idealised Longstaff-Schwartz algorithm.

**Lemma:** Let \( E_i^S \) denote the numeraire-rebased extrinsic value corresponding to the strategy \( S \). Then, for \( i = 1, 2, \ldots, n \),
\[
E_i^S \leq \hat{E}_i^G.
\]

It follows that
\[
V_0^S \leq V_0^G.
\]

**Proof:** Note first the standard backwards induction relationship for the extrinsic value of the strategy \( S \),
\[
E_i^S = \mathbb{E} \left[ V_{i+1}^S | \mathcal{F}_i \right] = \mathbb{E} \left[ I_{i+1} \mathbf{1}\{A_{i+1}\} + E_{i+1}^S \mathbf{1}\{A_{i+1}'\} | \mathcal{F}_i \right].
\]

It follows from the tower property that
\[
\mathbb{E} \left[ E_i^S | G_i \right] = \mathbb{E} \left[ \mathbb{E} \left[ I_{i+1} \mathbf{1}\{A_{i+1}\} + E_{i+1}^S \mathbf{1}\{A_{i+1}'\} | \mathcal{F}_i \right] | G_i \right] = \mathbb{E} \left[ I_{i+1} \mathbf{1}\{A_{i+1}\} + E_{i+1}^S \mathbf{1}\{A_{i+1}'\} | G_i \right] = \mathbb{E} \left[ \hat{E}_{i+1}^G \mathbf{1}\{A_{i+1}\} + \mathbb{E} \left[ E_{i+1}^G | G_{i+1} \right] \mathbf{1}\{A_{i+1}'\} | G_i \right].
\]

This yields the values \( \mathbb{E}[E_i^S | G_i] \) by backwards induction. To establish the first inequality in the lemma we also proceed by backwards induction. Observe that \( E_n^S = \hat{E}_n^G = 0 \) and assume that \( \mathbb{E}[E_j^S | G_j] \leq \hat{E}_j^G \) holds for \( j = i + 1, \ldots, n \). Using (7), the inductive hypothesis and equation (6) it follows that
\[
\mathbb{E} \left[ E_i^S | G_i \right] = \mathbb{E} \left[ \hat{E}_{i+1}^G \mathbf{1}\{A_{i+1}\} + \mathbb{E} \left[ E_{i+1}^G | G_{i+1} \right] \mathbf{1}\{A_{i+1}'\} | G_i \right] \leq \mathbb{E} \max \left\{ \hat{E}_{i+1}^G, \mathbb{E} \left[ E_{i+1}^G | G_{i+1} \right] \right\} \mathbf{1}\{A_{i+1}'\} | G_i \] = \hat{E}_i^G,
\]

as required.

The final result in the lemma follows on recalling that \( \mathcal{F}_0 \) is trivial and on applying the tower property at \( T_1 \),
\[
V_0^S = \mathbb{E} \left[ I_1 \mathbf{1}\{A_1\} + E_1^S \mathbf{1}\{A_1'\} | \mathcal{F}_0 \right] = \mathbb{E} \left[ I_1 \mathbf{1}\{A_1\} + E_1^S \mathbf{1}\{A_1'\} | G_0 \right] = \mathbb{E} \left[ \hat{E}_1^G \mathbf{1}\{A_1\} + \mathbb{E} \left[ E_1^G | G_1 \right] \mathbf{1}\{A_1'\} | G_0 \right] \leq \mathbb{E} \max \left\{ \hat{E}_1^G, \mathbb{E} \left[ E_1^G | G_1 \right] \right\} | G_0 \] = V_0^G,
\]
as required.
Remark 4: The analysis in this section appears to be dependent on the measure and numeraire used – the quantities $\hat{I}_i^G$ and $\hat{E}_i^G$ are conditional expectations in a given equivalent martingale measure. By contrast, the optimal exercise decision is independent of the measure and numeraire used. The resolution of this apparent paradox is the fact that, under a change of measure, both the measure and the variable being conditioned change (the numeraire being used to divide cashflows changes when the measure changes). These two effects offset each other perfectly.

Remark 5: We have shown that the Idealised Longstaff-Schwartz algorithm is optimal amongst all $\{G_i\}$-measurable exercise strategies. That the estimates $\hat{E}_i^G$ and $\hat{I}_i^G$ upon which the approximations for the exercise regions of this algorithm are based are derived using a least-squares criterion, which then turns out to be optimal, is coincidental and fortuitous. Had some other criterion been used in the Longstaff-Schwartz algorithm, such as minimising the absolute difference, then this optimality would no longer be achieved.

Remark 6: We noted in Remark 1 that our definition of the Longstaff-Schwartz algorithm is slightly non-standard, but better than the standard approach. This optimality result is the justification for this statement.

2.4 Dimensionality of Idealised Longstaff-Schwartz

Associated with any Longstaff-Schwartz implementation is an embedded decision variable process, the dimension of which is usually much less than the original Markov process that defines the economy. We have shown that the Idealised Longstaff-Schwartz algorithm, which improves upon the Longstaff-Schwartz algorithm, is optimal amongst all exercise strategies that are measurable with respect to $\{G_i\}$, the filtration generated by the decision variables. There is one further observation to make at this point, and this relates back to equation (6),

$$\hat{E}_i^G = \mathbb{E} \left[ \max \left\{ \hat{I}_{i+1}^G, \hat{E}_{i+1}^G \right\} | G_i \right].$$

The key point to note is that, although the original model was possibly of a high dimension, once the $\hat{I}_i^G$ have been evaluated the $\hat{E}_i^G$ can be found by backward induction and the whole pricing problem can be solved by working only with the process $\eta$, which is potentially much easier to handle because of its low dimension. This would be easier still if $\eta$ were Markovian. We consider this point in the next section.

2.5 Approximate Markov property of Longstaff-Schwartz

The process $\eta$ as defined above is not necessarily Markovian, indeed usually it would not be. However, in this section we present an heuristic argument which shows that, when the Longstaff-Schwartz algorithm is effective, $\eta$ will be ‘nearly Markovian’.

Suppose $\eta$ is not Markovian. Then, for some $i$, the conditional distribution of $\eta_{i+1}$ given $\eta_i$ is not the same as the conditional distribution of $\eta_{i+1}$ given $\mathcal{F}_i$. Thus (and this is how the Markov property is usually characterised), there exist bounded continuous functions $f$ such that

$$\mathbb{E} \left[ f (\eta_{i+1}) | \eta_i \right] \neq \mathbb{E} \left[ f (\eta_{i+1}) | \mathcal{F}_i \right].$$

When we say that $\eta$ is ‘nearly Markovian’ we mean that, even though (8) does not hold exactly, across a broad range of functions $f$ we have

$$\left| \mathbb{E} \left[ f (\eta_{i+1}) | \eta_i \right] - \mathbb{E} \left[ f (\eta_{i+1}) | \mathcal{F}_i \right] \right| \ll \left| \mathbb{E} \left[ f (\eta_{i+1}) | \mathcal{F}_i \right] \right|.$$
That is, the change in the conditional expectation caused by augmenting \( \sigma(\eta_i) \) with the extra information contained in the whole of \( \mathcal{F}_i \) is small compared with the ‘correct’ conditional expectation given the whole history \( \mathcal{F}_i \).

Consider now the Longstaff-Schwartz algorithm. The exercise decision at \( T_i \) is based on the random variables \( \bar{E}_i^p \) and \( \bar{P}_i^p \). Just as in Section 2.2 we expected to improve the accuracy of the algorithm by using \( E_i^p \) and \( P_i^p \) instead, we also expect, for exactly the same reasons but to a lesser degree, to improve the accuracy by basing the exercise decision on \( \hat{E}_i^p(\eta_i) \) and \( \hat{P}_i^p(\eta_i) \) defined by

\[
\hat{P}_i^p(\eta_i) := \mathbb{E}[I_i|\eta_i], \\
\hat{E}_i^p(\eta_i) := \mathbb{E}[E_i|\eta_i].
\]

Suppose that the algorithm, implemented based on \( \hat{P}_i^p(\eta_i) \) and \( \hat{E}_i^p(\eta_i) \) has performed well back to \( T_{i+1} \) and we are now about to calculate the numeraire-rebased extrinsic value at \( T_i \). The accuracy of the Longstaff-Schwartz algorithm, across a broad parameter range, means that there is some connected domain \( D_i \), near the centre of the probability distribution, and some small \( \delta_i \) such that, for all \( (x_i, \varepsilon_i) \in D_i \),

\[
|E_i - \hat{E}_i^p| < \delta_i, \\
|I_i - \hat{P}_i^p| < \delta_i.
\]

(See Remark 9 below for a fuller discussion on this point.) Elsewhere there is no guarantee the approximation will be so accurate since there is no need for accuracy in the region where it is clear which of \( E_i \) and \( I_i \) is larger.

Let \( V_i^p \) denote the numeraire-rebased option value at \( T_i \), as calculated using the Longstaff-Schwartz algorithm, and note the usual backwards induction relationship

\[
V_{i+1}^p = I_{i+1}\hat{1}_{i+1} + E_{i+1}\hat{1}_{i+1}, \\
E_i = \mathbb{E}[V_{i+1}^p|\mathcal{F}_i],
\]

where

\[
\hat{1}_{i+1} := 1\{\hat{P}_i^p(\eta_{i+1}) \geq \hat{E}_i^p(\eta_{i+1})\}, \\
\hat{1}_c := 1\{\hat{P}_i^p(\eta_{i+1}) < \hat{E}_i^p(\eta_{i+1})\}.
\]

We can now write

\[
|E_i - \hat{E}_i^p| = \left| \mathbb{E}[V_{i+1}^p|\mathcal{F}_i] - \mathbb{E}[V_{i+1}^p|\eta_i] \right| \\
= \left| \mathbb{E}[V_{i+1}^p 1\{D_{i+1}\} |\mathcal{F}_i] - \mathbb{E}[V_{i+1}^p 1\{D_{i+1}\} |\eta_i] \right| \\
+ \left| \mathbb{E}[V_{i+1}^p 1\{D_{i+1}^c\} |\mathcal{F}_i] - \mathbb{E}[V_{i+1}^p 1\{D_{i+1}^c\} |\eta_i] \right| \\
\leq \left| \mathbb{E}[V_{i+1}^p 1\{D_{i+1}\} |\mathcal{F}_i] - \mathbb{E}[V_{i+1}^p 1\{D_{i+1}\} |\eta_i] \right| \\
+ \left| \mathbb{E}[V_{i+1}^p 1\{D_{i+1}^c\} |\mathcal{F}_i] - \mathbb{E}[V_{i+1}^p 1\{D_{i+1}^c\} |\eta_i] \right|. \tag{10}
\]

We know that, for \( (x_i, \varepsilon_i) \in D_i \), the difference between the two extrinsic value terms (the left-hand side of (10)) is small, less that \( \delta_i \). This could, in theory, be the case even if both of the last two terms in (10) are large – the terms inside the moduli could be large but opposite in sign. However one would not expect this to be true for this problem as there is no structural reason why the two terms should have any close association with each other. As a result, both the last two terms will be small, being of a similar size to \( \delta_i \).

Our interest lies particularly with the first of these two terms, the magnitude of which we will now denote by \( \delta_i \), which is similar in magnitude to \( \delta_i \). Decomposing \( \hat{E}_i^p(\eta_{i+1}) \) as \( (\hat{E}_i^p(\eta_{i+1}) - E_{i+1}) + E_{i+1} \), and similarly for \( \bar{P}_i^p(\eta_{i+1}) \), we can now write

\[
\mathbb{E}\left[ (\hat{E}_{i+1}(\eta_{i+1})\hat{1}_{i+1} + \hat{E}_{i+1}(\eta_{i+1})\hat{1}_{c,i+1}) 1\{D_{i+1}\} |\mathcal{F}_i \right]
\]
- \mathbb{E} \left[ (\hat{p}_{i+1}^p (\eta_{i+1}) \mathbb{1}_{i+1}^c + \hat{E}_{i+1}^p (\eta_{i+1}) \hat{\mathbb{1}}_{i+1}^c) \mathbb{1}\{D_{i+1}\} | \eta_i \right] \\
\leq \mathbb{E} \left[ \mathbb{1}\{D_{i+1}\} | \mathcal{F}_i \right] - \mathbb{E} \left[ \mathbb{1}\{D_{i+1}\} | \eta_i \right] \\
+ \mathbb{E} \left[ (\hat{p}_{i+1}^p (\eta_{i+1}) - I_{i+1}) \mathbb{1}_{i+1} + (\hat{E}_{i+1}^p (\eta_{i+1}) - E_{i+1}) \hat{\mathbb{1}}_{i+1} \right] \mathbb{1}\{D_{i+1}\} | \mathcal{F}_i \\
+ \mathbb{E} \left[ (\hat{p}_{i+1}^p (\eta_{i+1}) - I_{i+1}) \hat{\mathbb{1}}_{i+1} + (\hat{E}_{i+1}^p (\eta_{i+1}) - E_{i+1}) \mathbb{1}_{i+1} \right] \mathbb{1}\{D_{i+1}\} | \eta_i \\
\leq \delta + 4 \delta_i. \tag{11}

Ignore for a moment the term \mathbb{1}\{D_{i+1}\} and consider the implications of (11). It shows that the first expectation on the left-hand side of (11) is hardly altered by adding all the information in \mathcal{F}_i to that already contained in \sigma(\eta_i). This is the required ‘approximate Markov’ property, as captured by (9), but only for one function \( f \),

\[
f(\eta_{i+1}) := \hat{p}_{i+1}^p (\eta_{i+1}) \mathbb{1}\{\hat{p}_{i+1}^p (\eta_{i+1}) \geq \hat{E}_{i+1}^p (\eta_{i+1})\} + \hat{E}_{i+1}^p (\eta_{i+1}) \mathbb{1}\{\hat{p}_{i+1}^p (\eta_{i+1}) < \hat{E}_{i+1}^p (\eta_{i+1})\}.
\]

Note, however, that different market inputs and trade details will generate a different function \( f \), albeit of the same general form. For these changed inputs the Longstaff-Schwartz algorithm is still effective. As a result, this robustness of the algorithm ensures that (11) will be valid for a range of similar functions. This will ensure the approximate Markov structure.

We return now to the presence of the term \mathbb{1}\{D_{i+1}\} in (11). If \( D_{i+1} \) depends on \( (\mathcal{X}_{i+1}, \mathcal{E}_{i+1}) \) only through \( \eta_{i+1} \) then this has no effect on the argument above. In general this will not be the case. Note, however, that even in general it does not adversely affect the argument. It’s role is to focus attention on a large domain near the centre of the probability distribution. Any dependence on \( (\mathcal{X}_{i+1}, \mathcal{E}_{i+1}) \) not captured by \( \eta_{i+1} \) would only increase the dependence of the expectations on the particular filtration used, even if \( \eta \) were Markovian. Thus we can still draw the same conclusions.

**Remark 7**: Note that the discussion above argues that \( \eta \) is approximately Markov. The argument follows from the accuracy of the Longstaff-Schwartz algorithm for a given product, but the conclusion (that \( \eta \) is approximately Markov) is not product specific. This is consistent with the empirical observation that a low-dimensional process \( \eta \) (usually two, capturing level and tilt) is adequate across a wide variety of products.

**Remark 8**: Consider the error introduced in the Longstaff-Schwartz algorithm at time \( T_{i+1} \) assuming no error at \( T_{i+1} \). It is readily seen that this takes the form

\[
\text{error} = \mathbb{E} \left[ (I_{i+1} - E_{i+1}) (1 - \hat{I}_{1}) \right], \tag{12}
\]

where

\[
\hat{I}_{1} := \mathbb{1}\{\hat{p}_{i+1}^p (\eta_{i+1}) \geq \hat{E}_{i+1}^p (\eta_{i+1})\}
\]

\[
I_{1} := \mathbb{1}\{I_{i+1} \geq E_{i+1}\}.
\]

Consider now the form of this error. There will only be a contribution when the decision rule incorrectly suggests exercise when the option should be held, or vice versa. But when this error is made one would expect the true intrinsic and extrinsic value functions to be close, and so the contribution from the term \( I_{i+1} - E_{i+1} \) is small. This is one of the things the Longstaff-Schwartz algorithm has in its favour – errors are only made when the effect of the error is small. Note, however, that an equation such as (12) holds for any exercise rule, and so this effect in itself does not explain the reasons why the Longstaff-Schwartz algorithm is effective.

**Remark 9**: Consider again the error in equation (12). The approximating functions \( \hat{E}_{i+1}^p \) and \( \hat{p}_{i+1}^p \) only enter through the indicator \( \hat{I}_{1} \), so all that matters is not the accuracy of the functions...
Longstaff-Schwartz, Effective Model Dimensionality and ...

$\hat{E}$ and $\hat{I}$, but merely the accuracy of the indicator $1\{\hat{I}^p_{i+1} (\eta_{i+1}) \geq \hat{E}^0_{i+1} (\eta_{i+1})\}$. So it appears that it is not necessary, for example, that $\hat{E}^p$ be a good representation of $E$. This, however, is not the case. For the algorithm to work reliably and robustly in practice, it must dictate exercise appropriately across a wide range of market input parameters. Changes in market volatilities, interest rates, trade details, etc. will change when exercise should occur, and the algorithm must assess/predict this accurately. The only way it can do so robustly is if $\hat{E}^p$ accurately represents $E$ and $\hat{I}^p$ accurately represents $I$. One exception to this statement would be when all the probability mass is located well away from the exercise boundary. But in this case there will be very little time value and many algorithms would perform well – the important thing for the Longstaff-Schwartz algorithm is that it work well precisely when there is a lot of time value and when the probability mass is centred near the exercise boundary.

3 Reducible Markov-functional models

The analysis of Section 2, as summarised in Section 2.4, has demonstrated that an implementation of the Longstaff-Schwartz algorithm results in a model that is, in effect, of the same dimension as the decision variable process. In a single currency interest rate model this is usually one or two dimensions, with the first variable capturing the overall level of interest rates and the second capturing the ‘tilt’ in the interest rate curve. The one sense in which the model is of high dimension is through the calculation of the conditional expectations in equation (5) to evaluate the conditioned intrinsic values $\hat{I}^p_i$. We argued in Section 2.5 that the decision variable process $\eta$, although not exactly Markovian should be close to Markovian. If, in fact, $\eta$ were Markovian then the valuation problem reduces further. Suppose this is indeed the case. We will now identify $\eta$ with the components $x_i$ of the original Markov process $(x_i, \varepsilon_i)$ that defines the economy – this was the reason we originally wrote this process in this way. We are here making the assumption that $x$ is itself Markovian, even with $\varepsilon$ removed from the statespace. We achieve this by making $x$ and $\varepsilon$ mutually independent, something that would hold if, for example, these were taken from a principal component decomposition of the sources of noise within the economy. Notice now that (5) and (6) reduce to

\[
\begin{align*}
\hat{I}^0_i & := \mathbb{E} [I_i(x_i, \varepsilon_i)|G_i] = \mathbb{E} [I_i(x_i, \varepsilon_i)|x_i] =: \hat{I}^p_i(x_i) \\
\hat{E}^0_i & := \mathbb{E} [E_i(x_i, \varepsilon_i)|G_i] = \mathbb{E} [E_i(x_i, \varepsilon_i)|x_i] =: \hat{E}^p_i(x_i) \\
\hat{E}^0_i(x_i) & = \mathbb{E} \max \left\{ \hat{I}^p_{i+1}(x_{i+1}), \hat{E}^0_{i+1}(x_{i+1}) \right\} |G_i \\
& = \mathbb{E} \max \left\{ \hat{I}^p_{i+1}(x_{i+1}), \hat{E}^0_{i+1}(x_{i+1}) \right\} |x_i .
\end{align*}
\]

As a result, if the process $\eta$ can indeed be identified with a Markov process $x$, then the pricing algorithm can be implemented on a low-dimensional lattice. This would get around some of the obvious drawbacks of simulation. Simulation is, by its nature, one of the slower numerical methods and one has to work extremely hard when calculating risk numbers, the ‘Greeks’ (by comparison, a well-designed three-dimensional lattice can, for example, produce a price accurate enough for pricing and risk purposes on a 30 year annual exercise trade in about one second, far better than simulation can achieve). For this reason most banks choose not to use simulation pricing for the purposes of risk management. A second drawback of using Longstaff-Schwartz and simulation is the care needed in selecting the basis functions.

The question, therefore, arises as to whether it is possible to generate models which can obtain the full benefits of the Idealised Longstaff-Schwartz algorithm and yet which have a Markovian structure that enables them to be implemented in a lattice-type algorithm. The benefits of this are clear, but note in particular that the Greeks will, in a careful implementation, be extremely stable.
In this and the next section we present some models which achieve these objectives, and we call them Reducible Markov-functional Models. In this section we present some models which are truly arbitrage-free and have BGM-like calibration properties. In Section 4 we present some models that are approximately arbitrage-free. We expect the latter models to be very close to arbitrage-free, for reasons stated later, and they have very nice implementation properties. For this reason we expect these latter models will ultimately prove more useful in practice, although we have as yet carried out no numerical study to back up these claims. We hope to present something on this in due course.

3.1 Recap of standard Markov-functional approach

Before introducing the more general Markov-functional models we will first summarise the standard case in which the driving process, denoted by \( x \), is already of low dimension. In our discussion we will restrict to the LIBOR form of the models, which is the focus of the reducible models we develop in later sections.

The basic idea is to choose some Markov process \( x \) (calibrated in some way to market covariances) and then to develop a model in which, for all \( 0 \leq t \leq T \), \( D_{tT} \equiv D_{tT}(x_t) \) for some functional form \( D_{tT}(\cdot) \). In fact the model will actually only be specified on a grid. That is, we specify the functional forms \( D_{TT}(x_T) \) for \( 1 \leq i < j \leq n + 1 \), since this is all that is (typically) needed in practice. The functional forms are derived numerically from market prices and the martingale properties necessary to make the model arbitrage-free.

As in a LIBOR Market Model we will assume we have a set of contiguous forward LIBORs denoted by \( L^i \) for \( i = 1, \ldots, n \), corresponding to tenor structure \( T_1, \ldots, T_{n+1} \). We will write \( S_i := T_{i+1} \) for \( i = 1, \ldots, n \) and so we have that \( L^i \) is the LIBOR corresponding to the period \([T_i, S_i] \). From now on we will use \( \mathbb{P} \) to denote the terminal measure, corresponding to taking the bond \( D_{Sn} \) as numeraire, and \( \mathbb{E} \) will be used for expectations in this measure. When we take, instead, \( D_{Si} \) as numeraire we will denote the corresponding probability measure and expectation by \( \mathbb{P}_i \) and \( \mathbb{E}_i \) respectively. Note that, henceforth, our variables will no longer be numeraire-rebased within the notion – when we want to divide by any particular numeraire, we will do so explicitly.

For ease of exposition we begin with the case in which \( x \) is one-dimensional.

3.1.1 One-dimensional case

In setting up our model we make the assumption that the \( i \)th forward LIBOR at time \( T_i \), \( L^i_{T_i} \), is a monotonic increasing function of the variable \( x_{T_i} \). Further define

\[
V^i(K) = D_0 S_i \mathbb{E}_i \left[ 1\{L^i_{T_i} > K\} \right]
\]

\[
= D_0 S_n \mathbb{E}_n \left[ D_{T_i S_i} \mathbb{E}_i \left[ 1\{L^i_{T_i} > K\} \right] \right]
\]

\[
= D_0 S_n \mathbb{E}_n \left[ \prod_{j=1+1}^{n} \left( 1 + \alpha_j L^j_{T_i}(x_{T_i}) \right) 1\{L^i_{T_i} > K\} \right], \quad (13)
\]

the value of a digital caplet with strike \( K \) setting at \( T_i \), paying at \( S_i \). The algorithm for finding the functional forms works back from \( T_n \). Suppose we have reached \( T_i \), having already found the functional forms \( D_{T_j T_i} (\cdot) \), \( i < j \leq k \leq n + 1 \) (so trivially, at time \( T_n \) this is true because there is nothing to know). For \( j > i \) we can recover the functional forms \( \frac{D_{T_j T_i}}{D_{T_i S_n}} (x_{T_i}) \) numerically from the martingale property of numeraire-rebased assets

\[
\frac{D_{T_j T_i}}{D_{T_i S_n}} (x_{T_i}) = \mathbb{E} \left[ \frac{D_{T_{i+1} T_i}}{D_{T_{i+1} S_n}} (x_{T_{i+1}}) | x_{T_i} \right].
\]
(Note again that this is trivial at $T_i$). We already know that $D_{T_i T_i}(x_{T_i}) \equiv 1$, so we will have recovered all the required functional forms if we can determine $L_{T_i}^j(x_{T_i})$. To do this, choose a grid of values $x^*$ and for each $x^*$ calculate

$$J_0^i(x^*) := D_{0S_n}\left[\frac{D_{T_i S_i}(x_{T_i})1\{x_{T_i} > x^*\}}{D_{T_i S_i}}\right]. \quad (14)$$

Now find the value $K(x^*)$ such that $V'(K(x^*)) = J_0^i(x^*)$. Equating equations (13) and (14) we find that

$$D_{0S_n}\left[\frac{D_{T_i S_i}(x_{T_i})1\{L_{T_i}^j(x_{T_i}) > K(x^*)\}}{D_{T_i S_i}}\right] = V'(K(x^*)) = J_0^i(x^*)$$

$$= D_{0S_n}\left[\frac{D_{T_i S_i}(x_{T_i})1\{x_{T_i} > x^*\}}{D_{T_i S_i}}\right]$$

$$= D_{0S_n}\left[\frac{D_{T_i S_i}(x_{T_i})1\{L_{T_i}^j(x_{T_i}) > L_{T_i}^j(x^*)\}}{D_{T_i S_i}}\right]. \quad (15)$$

Under the assumption that $L_{T_i}^j(x)$ is increasing in $x$ we can now conclude that $L_{T_i}^j(x^*) = K(x^*)$, thus we have derived the required functional form.

### 3.1.2 Multi-dimensional case

This is a straightforward generalisation of the one-factor case. The key to this extension is to ensure in the generalisation that we:

(i) Retain the univariate and monotonicity properties which were required to make the functional fitting efficient.

(ii) Capture the desired correlation/covariance structure.

To do this we introduce the idea of a pre-model. The rough idea is as follows. Suppose we have in mind some model and correlation structure. So, for example, suppose we intended to develop something similar to a $k$-factor BGM model. A first order approximation to this model could, for example, be obtained by taking the usual BGM SDE and replacing the time-dependent drift with its time-zero value. This would result in a model with something very close to the desired correlation structure, for which all LIBORs are lognormally distributed, but for which there is a significant arbitrage. This makes the approximation too poor for use as a BGM model in practice, but it remains adequate as a starting point for an arbitrage-free Markov-functional model. (We note at this point that Bennett and Kennedy [3],[4] show that Markov-functional and separable BGM models are very similar across a broad range of parameter values.)

The pre-model expresses each LIBOR as a function of the driving Markov process (which is now of dimension $k$). This is what gives the desired covariance structure. The approach now is to regard the Markov-functional sweep as a (small) perturbation of this pre-model. In particular we assume that the functional dependence of $L_{T_i}^j$ on the multi-dimensional $x_{T_i}$ is only via the pre-model LIBOR $\tilde{L}_{T_i}^j$. Thus

$$L_{T_i}^j(x_{T_i}) = f_i \left(\tilde{L}_{T_i}^j(x_{T_i})\right),$$

for some monotonic function $f_i$. It is this specialisation that enables us to achieve the univariate and monotonicity properties in this higher dimensional setting.
The last step is the derivation of the functional forms $f_i$. This is almost identical to the one-factor case, with small changes. One difference is that we now define $J_i^0(x^*)$ to be

$$J_i^0(x^*) := D_{0S_n} \mathbb{E} \left[ \frac{D_{T_i,S_n}(x_{T_i})}{D_{T_i,S_n}} 1 \{ \tilde{L}_i^i(x_{T_i}) > x^* \} \right].$$

We again solve $V^i(K(x^*)) = J_i^0(x^*)$, and now (15) becomes

$$D_{0S_n} \mathbb{E} \left[ \frac{D_{T_i,S_n}(x_{T_i})}{D_{T_i,S_n}} 1 \{ \tilde{L}_i^i(x_{T_i}) > x^* \} \right] = J_i^0(x^*) = D_{0S_n} \mathbb{E} \left[ \frac{D_{T_i,S_n}(x_{T_i})}{D_{T_i,S_n}} 1 \{ \tilde{L}_i^i(x_{T_i}) > x^* \} \right] = D_{0S_n} \mathbb{E} \left[ \frac{D_{T_i,S_n}(x_{T_i})}{D_{T_i,S_n}} 1 \{ f_i(\tilde{L}_i^i(x_{T_i})) > f_i(x^*) \} \right].$$

We conclude that $f_i(x^*) = K(x^*)$ which yields our required functional form.

### 3.2 Multiplicative DF reducible Markov-functional model

We now develop our first reducible Markov-functional model. Here the discount factors take a particular multiplicative form, namely:

$$\frac{D_{iS}}{D_{iT_i}}(x_i, \varepsilon_i) := M^{1}_{iS}(x_i) M^{2}_{iS}(\varepsilon_i)$$

where $M^1$ and $M^2$ are independent under the terminal measure $\mathbb{P}$. More precisely we will set

$$M^{1}_{iT_i}(x_i) := \prod_{j=1}^{n} \left( 1 + \alpha_j L^j_i (x_i) \right),$$

$$M^{2}_{iT_i}(\varepsilon_i) := \prod_{j=1}^{n} M^j_i (\varepsilon_i),$$

which can be written, equivalently, as

$$\left( 1 + \alpha_i L^i_i \right) = \left( 1 + \alpha_i \tilde{L}^i_i (x_i) \right) M^i_i (\varepsilon_i).$$

Note, here, that the $\{L^j_i\}$ and $\{M^j_i\}$ are mutually independent families but there is no restriction on the correlation structure between the $L^i_i$ or between the $M^j_i$. Note also that both $M^1_i$ and $M^2_i$ are martingales under $\mathbb{P}$ since they are independent and the product $M^1_i M^2_i$ is a martingale. Similarly, both $\tilde{L}^1_i$ and $\tilde{M}^i_i$ are martingales under $\mathbb{P}_i$.

As we shall see in Section 3.2.2, we shall use Markov-functional techniques to find the precise functional forms $L^i_i(x_i)$. We will introduce a standard Markov-functional pre-model $\tilde{L}^i_i(x_{T_i})$ and then define

$$\tilde{L}^i_i(x_{T_i}) = f_i \left( \tilde{L}^i_i(x_{T_i}) \right).$$

The functions $f_i$ will be found numerically.

The independence assumption is important for the analysis that follows. Note, however, that it is not particularly restrictive. To see this, consider a general $n$-dimensional Hull-White model. It can be shown that, under this model, discount factors take the form

$$\frac{D_{iS}}{D_{iT_i}} = \frac{D_{iS}}{D_{iT_i}} \prod_{k=1}^{n} \exp \left( \int_{0}^{t} \sigma_{a,S}^k dZ^k_u - \frac{1}{2} \exp \left( \int_{0}^{t} \sigma_{a,S}^k \right)^2 du \right).$$

©2004 Phil Hunt & Joanne Kennedy
http://www.kennedy-hunt.com

16 Version 1.1 (25 January 2005)
Longstaff-Schwartz, Effective Model Dimensionality and 

\[ D_{0S} = \prod_{k=1}^{2} \exp \left( \int_{0}^{t} \sigma_{uS}^k dZ_u^k - \frac{1}{2} \exp \left( \int_{0}^{t} (\sigma_{uS}^k)^2 du \right) \right) \times \prod_{k=3}^{n} \exp \left( \int_{0}^{t} \sigma_{uS}^k dZ_u^k - \frac{1}{2} \exp \left( \int_{0}^{t} (\sigma_{uS}^k)^2 du \right) \right) \]  

(19)

where \((Z_1, \ldots, Z_n)\) is a standard Brownian motion with \(dZ_i dZ_j = \delta_{ij} dt\). This represents discount factors as a product of independent martingales in the terminal measure \(P\).

3.2.1 Calibration

Model calibration for these models proceeds as it would for any other multi-dimensional model. As a first step, a model parameterisation is chosen. For BGM, for example, this could mean specifying a functional form for correlations between different LIBORs and for the term structure of volatility for each LIBOR. In our case, part of that parameterisation is the decision to make the model take the product form given in (17). In addition to this, assumptions might be made on the functional form of correlations between the \(M_i\), and similarly between the \(\hat{L}_i\) introduced in equation (18). This latter correlation is, as we shall see, controlled by specifying a correlation structure for the process \(x\) and by selection of an appropriate pre-model \(\tilde{L}_i\) used in relating the \(\hat{L}_i\) to the process \(x\) in equation (18).

Once this parameterisation is chosen, the next step is to estimate the free parameters, in particular those that determine correlations between the different LIBORs. This can be done by fitting the model to the swaption matrix. At this step, just as is done when calibrating BGM, approximations should be used to recover swaption volatilities from the model parameters. See \cite{19} for a discussion of this in the case of a BGM model. For BGM, part of this approximation procedure is not to worry about the model drifts and to concentrate only on the diffusion part of the model. In our case we work only with the pre-model at this stage and act as if \(f_i \equiv 1\), all \(i\). This is a reasonable thing to do because the functions \(f_i\) are typically small perturbations of the pre-model that remove the arbitrage that is present in the pre-model.

At the end of these two steps the distributions of \(x\) and of the \(M_i\) will have been determined. Our calibration approximations have not been specific about which measure we were working in, so we at this point assume that the distribution of \(x\) is known in the terminal measure \(P\) but \(M_i\) is known in the measure \(P_i\) (for reasons which become clear below). What we do not yet have is an arbitrage-free model specified in a single measure. The analogy with BGM would be to say that we have decided that all LIBORs should be lognormal in their own measure, with a given correlation structure, but we have not yet determined the appropriate LIBOR drifts to construct a BGM model in the terminal measure with all these properties. What it remains for us to do now is to construct the model in the terminal measure.

3.2.2 Implementation

One of the nice properties of this model is that the implementation is almost identical to that of a standard Markov-functional model, as described in Section 3.1. This is again done working backwards through time, starting on the final exercise date \(T_n\).

We will specify a LIBOR based model. This does not mean we are restricting to LIBOR based products, any more that a LIBOR Market Model is used only for non-swap based products. It is just more convenient here to use LIBORs as the basic building blocks for the model.

Someone building a LIBOR Market Model might typically decide to make all the LIBORs \(L_i^t\) lognormal in their own measure \(P_i\). This is equivalent to specifying \(P_i(L_i^{T_i} > x)\), all \(x\). We assume to begin with that we know, for each \(i\),

\[ G_i(x) := P_i(L_i^{T_i} > x) . \]
These probabilities are an output of the calibration described in Section 3.2.1 above. For example we might, in the approximations used there, have assumed that each $\hat{L}_T^i$ is lognormal in its own measure $\mathbb{P}_i$, or we might have assumed that each LIBOR $L_T^i$ is lognormal in this measure (these are very similar assumptions qualitatively since most of the model variability is captured by the $\{L_i^j\}$ and relatively little by the $\{M^j\}$). In this latter case there is now a little work to do to determine $G_i(x)$. Denoting the (known) $\mathbb{P}_i$-density of $\log M^j$ by $\phi \log M$, we have that
\[
H_i(K) := \mathbb{P}_i(\log(1 + \alpha_i L_T^i) > K) = \int_{-\infty}^{\infty} \phi \log M(x) \hat{H}_i(K - x) \, dx.
\]
where
\[
\hat{H}_i(x) := \mathbb{P}_i(\log(1 + \alpha_i \hat{L}_T^i) > x).
\]
This convolution suggests using Fourier transforms to solve for $H_i$, and thus $G_i$, yielding
\[
\hat{H}_i = \mathcal{F}^{-1} \left( \frac{\mathcal{F}(H_i)}{\mathcal{F}(\phi \log M)} \right),
\]
where $\mathcal{F}(f)$ denotes the Fourier transform of the function $f$.

At this point we know, from the calibration process and the calculation just described, the distribution of each $M_T^n$ and $\hat{L}_T^n$ in their own measure $\mathbb{P}_i$, the functional forms $\hat{L}_T^n$, and also the distribution of $x$ under the terminal measure $\mathbb{P}_i$. It remains to find the functions $f_i$ in equation (18) which map the pre-model functional forms into the functional forms $\hat{L}_T^n(x_T)$.

In the case of a standard Markov-functional model, the functional form $L_T^n(x_i)$ at time $T_i$ is calculated from equation (16). There we find functions $f_i$ such that
\[
D_{0S_n} \mathbb{E} \left[ \frac{D_{T,S_n}}{D_{T,S_n}}(x_T) 1\{L^n_T(x_T) > K\} \right] = D_{0S_n} \mathbb{E} \left[ \frac{D_{T,S_n}}{D_{T,S_n}}(x_T) 1\{f_i(L^n_T(x_T)) > K\} \right] = V^n(K).
\]
(20)
Here that expression is replaced by the following. For every $A > 1$, the functional form $f_i$ (and hence the functional form $\hat{L}_T^n(x_i)$) is chosen to satisfy
\[
\mathbb{E} \left[ \prod_{j=i+1}^{n} \left( 1 + \alpha_j \hat{L}_{T_j}^i(x_i) \right) 1\{\hat{L}_T^n(x_i) > A\} \right] = \mathbb{E} \left[ \prod_{j=i+1}^{n} \left( 1 + \alpha_j L_{T_j}^i(x_i) \right) 1\{f_i\left(L_T^n(x_i)\right) > A\} \right] = \left( \prod_{j=i+1}^{n} \left( 1 + \alpha_j L_0^j \right) \right) G_i(A)
\]
(21)
Note that the left-hand side of (20) is of the same form as the left-hand side of equation (21). In both cases the right-hand side is some known function, so the numerical procedure is identical in both cases.

We need to verify that this model is consistent with the distributions $G_i$ to which we are fitting the model. To see that this is true, note first that $\rho_i^j := \frac{\partial}{\partial x^j} |_{x^j | \hat{x}}$ is given by
\[
\rho_i^j = \frac{D_{iS_n}/D_{0S_n}}{D_{iS_n}/D_{0S_n}}
\]
Longstaff-Schwartz, Effective Model Dimensionality and ...
where $\tilde{L}^i$ is a Markov-functional pre-model and the function $f_i$ is derived to fit the desired distributions, as described later. Next set $\mathcal{M}^{2,i} := M^i$. We suppose, without further loss of generality, that $M^i_0 = 1, \forall i$. As in the case of multiplicative discount factors the $\{L^i\}$ and $\{M^i\}$ are mutually independent families under the terminal measure $\mathbb{P}$ (and hence also under $\mathbb{P}_i$, as the reader can check). As in that case, there is no restriction on the correlation structure between the $\tilde{L}^i$. However we must here insist that the martingales $\mathcal{M}^{2,i}$ are mutually independent under the terminal measure $\mathbb{P}$.

The independence assumption between $\mathcal{M}^{1,i}$ and $\mathcal{M}^{2,i}$ is again important for the analysis that follows. Note, however, that it is not particularly restrictive. Consideration of a general $n$-dimensional BGM model demonstrates this. Under this model, LIBORs are of the form

$$
L^i_t = L^0_t \prod_{k=1}^{\infty} \exp \left( \int_0^t \sigma^i_k dZ^k_u - \frac{1}{2} \int_0^t \left( \sigma^i_k \right)^2 du \right)
$$

$$
= L^0_t \left\{ \prod_{k=1}^{n} \exp \left( \int_0^t \sigma^i_k dZ^k_u - \frac{1}{2} \int_0^t \left( \sigma^i_k \right)^2 du \right) \right\},
$$

\begin{align*}
&\times \left\{ \prod_{k=3}^{\infty} \exp \left( \int_0^t \sigma^i_k dZ^k_u - \frac{1}{2} \int_0^t \left( \sigma^i_k \right)^2 du \right) \right\},
\end{align*}

where $(Z_1, \ldots, Z_n)$ is a standard Brownian motion under the measure $\mathbb{P}_i$ with $dZ_i dZ_j = \delta_{ij} dt$. This represents the LIBOR $L^i$ as a product of independent martingales under the measure $\mathbb{P}_i$.

The assumption that the $\mathcal{M}^{2,i}$ are mutually independent is a clear restriction. Nonetheless the presence of the term $\mathcal{M}^{2,i}$ in the model (as opposed to setting $\mathcal{M}^{2,i} \equiv 1$ and having a model based only on the low number of factors in the variable $x$) does provide a significant extension to the type of correlation matrices that are achievable. In particular, it allows a better fit to correlation matrices which decay rapidly from one as a function of LIBOR separation.

### 3.3.1 Calibration

This follows exactly the same lines as the calibration of the multiplicative discount factor model of Section 3.2. The only difference, of course, is the initial form of the model.

### 3.3.2 Implementation

This also follows very similar lines to the multiplicative discount factor model. A starting point once more is the law of $x$ under $\mathbb{P}$, the distribution of the $\{M^i\}$ under $\mathbb{P}_i$, and the distribution $G_i(x) = \mathbb{P}_i(\tilde{L}^i(x_1) > x)$. The Markov-functional fitting step, equation (21), is unaltered. What does change slightly is the proof that the resultant functional forms are consistent with the given distributions $G_i$. Now we have $\rho_i := \frac{d\mathbb{P}_i}{d\mathbb{P}} |_{\mathbb{P}}$, is given by

$$
\rho_i = \frac{D_{s_i} / D_{0s_i}}{D_{s_i} / D_{0s_i}} = \prod_{j=i+1}^{n} \frac{1 + \alpha_j \tilde{L}^j_i M^j_0}{1 + \alpha_j \tilde{L}^j_0 M^j_0}
$$

and thus

$$
G_i(x) = \mathbb{E}_i \left[1\{\tilde{L}^i_{T_i} > x\}\right]
$$
\[ E \left[ \rho_{T_i} \mathbf{1}\{\hat{L}_{T_i} > x\} \right] = E \left[ \prod_{j=i+1}^{n} \frac{1 + \alpha_j \hat{L}_{T_i}^j M_{T_i}^j}{1 + \alpha_j \hat{L}_{0_i}^j} \mathbf{1}\{\hat{L}_{T_i} > x\} \right] = \left( \prod_{j=i+1}^{n} \left( 1 + \alpha_j \hat{L}_{0_i}^j \right) \right)^{-1} E \left[ \prod_{j=i+1}^{n} \left( 1 + \alpha_j \hat{L}_{T_i}^j \right) \mathbf{1}\{\hat{L}_{T_i} > x\} \right]. \] (22)

as required.

### 3.4 Example product valuations

To illustrate the use of these models we consider two important practical examples, the callable inverse floater and callable ratchet swap.

#### 3.4.1 Callable inverse floater

We consider valuation in the multiplicative discount factor model. An inverse floater swap comprises two legs. The first, receive leg say, is a standard LIBOR leg. The second pay leg is the inverse floater leg. The \(i\)th payment on this leg, paid at \(S_i\), is of the form

\[ C_{i}^{\text{pay}} := \alpha_i \left( K - g_i \hat{L}_{T_i}^i \right)_+, \]

where \(K\) is some constant and \(g_i\) is some gearing factor. Note that the coupon is constrained to be positive.

A callable inverse floater is just an inverse floater which the option holder can cancel on any coupon date. This can be decomposed into a standard inverse floater plus the right to enter an inverse floater. The inverse floater that is entered on option exercise is precisely the one above but with the pay and receive legs reversed.

We will here consider valuation of the right to enter the reversed inverse floater. Note that the underlying inverse floater looks very much like a series of caplets (for which caplet dynamics and volatilities are paramount). The call feature is more like a Bermudan swaption (when \(g_i \equiv 0\) it is exactly a Bermudan swaption), for which swap dynamics and volatilities are most important. It is this combination of features that makes the product difficult to value.

Note that the inverse floater is an example of a product for which the net \(i\)th coupon is some non-linear function of the underlying LIBOR

\[ C_i := h_i(\hat{L}_{T_i}^i). \]

We shall treat this general case in the following discussion.

Valuation by backwards induction through a lattice is standard. So the main task facing us is to find \(I_i^G(x_i)\), the conditional value of the intrinsic at \(T_i\) given \(x_i\). To avoid unnecessary notation we will now drop the superscript \(G\) and write this as \(\hat{I}_i(x_i)\). To find \(\hat{I}_i(x_i)\), note that we can first represent the intrinsic value at \(T_i\) as the value of the next coupon, to be paid at \(S_i\), plus the value of all later coupons. Adopting the convention that \(\hat{I}_{n+1} \equiv 0\) and using a standard change of measure, we then have

\[ \hat{I}_i(x_i) = E \left[ \frac{D_{T_i,S_i}}{D_{T_{i+1},S_n}} h_i(L_{T_i}^i) | x_i \right] + E \left[ \hat{I}_{i+1}(x_{i+1}) | x_i \right] = E \left[ \frac{D_{T_i,S_i}}{D_{T_{i+1},S_n}} | x_i \right] E_i \left[ h_i(L_{T_i}^i) | x_i \right] + E \left[ \hat{I}_{i+1}(x_{i+1}) | x_i \right] \]
Longstaff-Schwartz, Effective Model Dimensionality and ...

\[
\begin{align*}
&= \left( \prod_{j=1}^{n} (1 + \alpha_j \hat{L}_{T_j}^i) \right) \left( \prod_{j=1}^{n} M_{0j}^i \right) \hat{E}_i \left[ h_i(L_{T_j}^i) | x_i \right] + \hat{E}_i \left[ I_{i+1}(x_{i+1}) | x_i \right] \\
&= \left( \prod_{j=1}^{n} (1 + \alpha_j \hat{L}_{T_j}^i) \right) \left( \prod_{j=1}^{n} M_{0j}^i \right) \hat{h}_i(\hat{L}_{T_j}^i) + \hat{E}_i \left[ I_{i+1}(x_{i+1}) | x_i \right]
\end{align*}
\]

where

\[
\hat{h}_i(\hat{L}_{T_j}^i) := \hat{E}_i \left[ h_i(L_{T_j}^i) | x_i \right] = \hat{E}_i \left[ h_i \left( \frac{(1 + \alpha_j \hat{L}_{T_j}^i) M_{0j}^i - 1}{\alpha_i} \right) | \hat{L}_{T_j}^i \right].
\]

The evaluation of \( \hat{h} \) is a straightforward numerical calculation since \( \hat{L}^i \) and \( M^i \) are independent under \( \mathbb{P}_i \).

### 3.4.2 Callable ratchet swap

Now we consider valuation in the multiplicative LIBOR model. A ratchet swap comprises two legs, one (receive leg, say) being a standard LIBOR leg, the other (pay leg) being the ratchet leg. On the ratchet leg each coupon is based on LIBOR, but the coupon is capped and floored. The level of the \( i \)th cap is set either at the start of the trade (a lifetime cap) or with reference to the previous coupon payment. Typically the cap rate might be 50 basis points higher than the previous coupon rate. The floor behaves similarly.

Let \( R_i \) denote the coupon rate that applies to the \( i \)th coupon on the ratchet leg, so the actual payment made is given by \( C^\text{pay}_i = \alpha_i R_i \). In general \( R_i \) can be related to \( R_{i-1} \) by an expression of the form

\[
R_i = \min \left\{ \max \left\{ L_{T_j}^i, a_i R_{i-1} + b_i \right\}, c_i R_{i-1} + d_i \right\},
\]

for suitable constants \( a_i, b_i, c_i, d_i \). More generally, we will here treat the case where the net coupon paid at \( S_i \) can be written in the form

\[
C_i = h_i(L_{T_j}^i, R_{i-1}).
\]

To value this product on a lattice we need to extend the statespace to ensure that the value process is Markovian. Thus we take \( \langle x_i, \varepsilon_i, R_{i-1} \rangle \) as the statespace. Now instead of finding \( \hat{I}_i(x_i) \) we need to find

\[
\hat{I}_i^D := \hat{E}_i \left[ I(x_i, \varepsilon_i, R_{i-1}) | \mathcal{G}_i \right],
\]

where \( \mathcal{G}_i := \sigma(x_1, R_0, \ldots, x_i, R_{i-1}) \). Note that the discrete-time process \( (x_i, R_{i-1}) \) is Markovian as a result of the mutual independence of the \( M^i \) in this model. This, combined with the fact that future coupons depend on the past only through \( R_{i-1} \), ensures that

\[
\hat{I}_i(x_i, R_{i-1}) := \hat{E}_i \left[ I(x_i, \varepsilon_i, R_{i-1}) | x_i, R_{i-1} \right].
\]

Note that we have again suppressed the \( \mathcal{G} \) from the notation.

Valuation can again be carried out on a lattice which now represents \( (x_i, R_{i-1}) \). It remains only for us to calculate \( \hat{I}_i(x_i, R_{i-1}) \). Adopting the convention that \( \hat{I}_{n+1} \equiv 0 \), we can work backwards and obtain

\[
\hat{I}_i(x_i, R_{i-1}) = \hat{E}_i \left[ \frac{D_{T_i} S_i}{D_{T_i} S_n} h_i(L_{T_j}^i, R_{i-1}) | \mathcal{G}_i \right] + \hat{E}_i \left[ \hat{I}_{i+1}(x_{i+1}, R_i) | x_i, R_{i-1} \right]
\]

\[\text{(1)}\]
where

\[ h_i(\hat{L}_t^i, R_{t-1}) := E_i \left[ h_i(L_T^i, R_{t-1}) | x_i, R_{t-1} \right] = E_i \left[ h_i(\hat{L}_T^i, M_T^i, R_{t-1}) | x_i, R_{t-1} \right]. \]

4 Extensions

4.1 Approximate multiplicative model

The models presented in Section 3 are arbitrage-free and, because of the high dimensionality of \((x, \varepsilon)\), they are able to calibrate well to both cap and swaption prices. They do, however, have two drawbacks. The first arises when pricing something like a Bermudan swaption. This product requires knowledge of swap rates on a lattice and for both models presented in Section 3 these are not straightforward to calculate.

The second drawback, one shared by most models currently in use, is that it is not possible to fit exactly the prices of a wide range of underlying cap and swaption prices. In a sense this is not such a drawback since it is widely agreed that to fit a model exactly to all swaption prices is inadvisable. To do so one would need to distort the local dynamics of the model to a large degree, and so doing would adversely affect the model’s effectiveness as a tool for pricing more exotic products. This is exacerbated because the input market data, the swaption volatility matrix, is in parts illiquid and volatilities are often rounded. On the other hand, it is very useful to be able to calibrate a model very accurately to those underlying instruments which will become a hedge, and those which are liquid and reliable.

In this section we present a model which gets around both of these difficulties. We develop a model that is able to fit a large range of market prices exactly without adversely affecting the model’s dynamics. To do this we introduce a model which is very similar to the one introduced in Section 3.3. The model here is more general than the one given earlier but it is only an approximation in that the model admits arbitrage. However, as we discuss below, we expect this arbitrage to be negligible for practical purposes.

4.2 Calibration and fitting

For both of the (arbitrage-free) models presented in Section 3 calibration was performed by first choosing the general functional form for the model (including the pre-model) and then fitting the model parameters. We do the same here. As in Section 3.3 we model LIBORs to be of the form

\[ L_t^i := M_t^{1,i}(x_t) M_t^{2,i}(\varepsilon_t) \]

where \(M^1\) and \(M^2\) are independent under the terminal measure \(\mathbb{P}\). Unlike in Section 3.3 we remove the restriction that the \(\{M_t^{2,i}\}\) should be mutually independent.
Many practitioners nowadays like to base their models on BGM, either directly implementing BGM or implementing a model which in some close sense emulates it. This is why we are here presenting a model for which LIBORs (and later swap rates also) have a lognormal form. This restriction can easily be dropped, and incorporating other effects such as stochastic volatility and skews is a simple matter.

This more general model is clearly richer than the multiplicative LIBOR model already considered. However the calibration of this model is still straightforward – the standard approximations used at this stage apply just as well in this more general setting. Indeed this is equivalent to a full BGM calibration, which is well understood.

Output from the calibration process is the law of $x$ in the terminal measure $P$ and the functions $G_i(x) = P_i(L > x)$. The next step in the construction of the multiplicative LIBOR model of Section 3.3 was to find the functional forms $\hat{L}_T(x_i)$ by solving equation (21). We do exactly the same for this new model.

The extra generality introduced here (and the presence of the terms $\{M_{2,i}\}$) means that equation (20) no longer applies, the functional forms $\{\hat{L}_T(x_i)\}$ will no longer ensure that each $\hat{L}_T$ has the target law $G_i$ in the measure $P_i$, and the model will admit arbitrage. To overcome these problems we now view the modelling we have done so far in a slightly different light.

### 4.3 The core model and index calibration

Suppose we have developed some model for interest rates. This model will capture the dynamics of all discount factors. Discount factors can in turn be used to calculate values for forward LIBOR. Or can they?

It is well-known that LIBOR, which is a market index, trades at a basis to the value that would come from a discount curve. What this means is that if discount factors are used to calculate LIBOR,

$$L_0 S_{[0, S]} = 1 - \frac{D_{0S}}{\alpha D_{0S}},$$

then the resulting value will differ from the one which can be observed in the market. There are a number of reasons for this, including credit and liquidity considerations. LIBOR represents a deposit rate that would apply when depositing money with high quality banks. A discount curve, on the other hand, relates more to a particular counterparty. To overcome this issue of basis it is common when modelling to separate the concept of LIBOR as an index from LIBOR derived from a discount curve. We adopt this idea, and extend it, in formulating our model.

We now adopt the model, defined by $x$ and just the $\{\hat{L}^i\}$ as our model of the discount curve. We call this the core model. Note that, by virtue of the Markov-functional sweep (22) this is a valid model, one which values all discount factors correctly. It also captures most of the variability in the market through the process $x$, which for a single currency interest rate model would usually be of dimension two. However it will systematically underestimate LIBOR volatilities and swaption volatilities since we have discarded the independent terms $\{M_{2,i}\}$ from the model (if we had intended to develop a standard two-factor model we would reduce this problem by calibrating initially to the two factors without including the $\{M_{2,i}\}$). We compensate for this by now treating each LIBOR, and indeed all swap rates, as a basis to this core model. In doing so we not only recover the ‘lost volatility’, we are also able to calibrate to any swaption prices without distorting the core model dynamics.

So we now complete the model description by showing how to derive a LIBOR (and general swap) index within the model. Let $y^i$ denote some market index, the swap rate that sets at $T_i$ and ends at $S_j$. Let $\hat{y}^i_j$ denote the value of this swap rate were it to be calculated in the core...
model. Note that $L^i \equiv y^i_1$, so we are capturing the LIBORs with this notation. We now suppose we can write the index $y^i_1$ in the form

$$y^i_1 = f_{ij}(\hat{y}^j_1, \lambda^j_1).$$

Here $f_{ij}$ is some functional form, to be calculated, and $\lambda^j_1$ is a unit mean lognormal random variable, independent of $x$.

The choice of the lognormal distribution for the $\lambda^j_1$ is again driven by our desire in this exposition to produce a lognormal BGM-type model. More general models would perhaps change this choice. The full description of the $\{\lambda^j_1\}$ and their covariance is another output from the initial calibration to market swaption volatilities. More precisely, an output from the calibration process is a set of (model-derived) covariances for various swap-rates $\{y^i_1\}$. The covariance structure for the $\{\lambda^j_1\}$ is chosen to be consistent with this, under the simplifying assumption that $f_{ij}$ is the identity map, for all $i, j$.

It remains for us to explain how to fit the functional form $f_{ij}$. This follows familiar lines, with $f_{ij}$ being chosen so that the model correctly values swaptions based on this swap rate.

The value, at $T_i$, of a digital payers swaption with strike $K$ is

$$V^{ij}_{T_i}(K) = \hat{P}^{ij}_{T_i} 1\{y^j_1 > K\},$$

where $\hat{P}^{ij}_{T_i}$ is the PVBP (in the core model) for a swap running from $T_i$ to $S_j$. Hence the value of this swaption at time zero is given by

$$V^{ij}_0(K) = D_0 S_n E \left[ \hat{P}^{ij}_{T_i} 1\{y^j_1 > K\} \right] = D_0 S_n E \left[ \frac{\hat{P}^{ij}_{T_i}}{D_{T_i} S_n} 1\{f_{ij}(\hat{y}^j_1, \lambda^j_1) > K\} \right].$$

We evaluate the functions $f_{ij}$ efficiently as follows. First evaluate

$$h_{ij}(A) := D_0 S_n E \left[ \frac{\hat{P}^{ij}_{T_i}}{D_{T_i} S_n} 1\{y^j_1 > A\} \right],$$

for a grid of values $A$. This can be done completely working within the core model. Now, for a grid of values $x^*$ calculate

$$J^{ij}_0(x^*) := E \left[ h_{ij}(x^*/\lambda^j_1) \right],$$

and solve to find $K(x^*)$ such that

$$V^{ij}_0(K(x^*)) = J^{ij}_0(x^*).$$

Putting this all together we obtain

$$D_0 S_n E \left[ \frac{\hat{P}^{ij}_{T_i}}{D_{T_i} S_n} 1\{f_{ij}(\hat{y}^j_1, \lambda^j_1) > K(x^*)\} \right] = V^{ij}_0(K(x^*)) = J^{ij}_0(x^*) = E \left[ h_{ij}(x^*/\lambda^j_1) \right] = D_0 S_n E \left[ \frac{\hat{P}^{ij}_{T_i}}{D_{T_i} S_n} 1\{\hat{y}^j_1 > x^*/\lambda^j_1\} \right].$$
Thus, assuming \( f_{ij} \) is monotone, \( f_{ij}(x^*) = K(x^*) \) and we have the desired functional form.

**Remark 10:** Even had we chosen to take both \( x \) and \( \varepsilon \) into the core model, this basis idea could still be used. It gives a way to make small changes to the distribution of individual index swap rates without adversely affecting the dynamics of the model.

**Remark 11:** The core model, as a model of the discount curve, is arbitrage-free. The inclusion of the \( \{M^2, i\} \) into the core model would, as we have already mentioned, introduce arbitrage. This led us to introduce the concept of the core model and to build up indices separately. If we only apply this approach to generate LIBOR indices, then the resultant model will be arbitrage-free. However, if we also simultaneously apply it to swap indices then the model will again admit arbitrage, because we can now either derive a swap index as described above, or we could calculate it by bringing the relevant LIBOR indices back through the tree.

These two approaches will give different answers, and this inconsistency generates the arbitrage. Note, however, two important details of this approach. First, the arbitrage is caused by an inconsistency at the level of the \( \lambda_i^j \). These capture a relatively small amount of the stochastic nature of the model – most of it resides in the process \( x \). So the effect should, by comparison, be much smaller than the drift errors in standard approximations to BGM models since these approximations are not working only on lower order terms. The second thing to note is the fact that the inconsistency is caused by the two approaches giving different swap-rate distributions. However, because market prices are being fitted directly, the approach will not introduce errors in the mean values of numeraire-rebased discount factors. So the effect is in this sense, also, of lesser consequence.

### 4.4 Product valuation

This follows the same approach as for the other models that we have introduced. The first step is to identify the underlying product and then to calculate \( \hat{I}_i(x_i) \), the conditioned numeraire-rebased intrinsic value. The rest then follows by backwards induction on a lattice. The only point we need to cover here is how to calculate \( \hat{I}_i(x_i) \).

Typically the intrinsic value will be some function of a market swap rate, as is the case for a Bermudan swaption. So we suppose we can write this intrinsic value at \( T_i \) in the form \( \hat{P}_T g(y_i^T) \) for some annuity stream \( \hat{P} \) and some function \( g \). It then follows that

\[
\hat{I}_i(x_i) = \mathbb{E} \left[ \frac{\hat{P}_{T_i}}{D_{T_i} S_n} g(y_i^T) \mid x_i \right] = \frac{\hat{P}_{T_i}}{D_{T_i} S_n} \mathbb{E} \left[ g(y_i^T) \mid x_i \right] = \frac{\hat{P}_{T_i}}{D_{T_i} S_n} \hat{g}(\hat{y}_i^T),
\]

where

\[
\hat{g}(\hat{y}_i^T) := \mathbb{E} \left[ g(y_i^T) \mid \hat{y}_i^T \right] = \mathbb{E} \left[ g(f_{ij}(\hat{y}_i^T, \lambda_i^j)) \mid \hat{y}_i^T \right].
\]

Evaluation of \( \hat{g} \) can be carried our efficiently using standard integration techniques.
4.5 Other extensions

There are a number of other extensions possible beyond what has been covered here. We hope to cover these in detail elsewhere, but for now we just note two of the possibilities.

- Stochastic volatility can readily be incorporated within these and any standard Markov-functional models. All one needs to do is to generate some stochastic volatility process, assume that market rates are a function of this process, and then apply a Markov-functional sweep to the state variable.

- Multi-currency versions of all the models can be implemented. The basic approach to this, in the terminal measure, is covered in [13]. For a Markov-functional model developed using a rolling discount bond numeraire, see [9].
References


