Fast Monte Carlo Bermudan Greeks

In recent years, much effort has been devoted to improving the efficiency of the Libor market model. Matthias Leclerc, Qian Liang and Ingo Schneider extend the pioneering work of Giles & Glasserman (2006) and show how fast calculations of Monte Carlo Greeks are feasible even within the framework of Bermudan-style derivatives. The authors demonstrate the efficiency gains in detail

Fast pricing and calculating hedging parameters are still a challenge in the framework of the Libor market model (LMM), which has become the fundamental pricing model in the fixed-income environment. Traditionally, for fixed-income securities, Greeks are calculated by the so-called bump and revalue method: each initial forward rate is perturbed by a basis-point shift and then the security is valued again. Besides the simplicity, there is no further advantage. The LMM is usually implemented with Monte Carlo methods and this can be rather slow, especially using the perturbation described before. The simulation procedure in the LMM is done in a forward measure and so the natural way to calculate Greeks is to do them on the fly. Giles & Glasserman (2006) have shown that under specific circumstances, their adjoint method can be suitable to get the Greeks a lot faster and save a considerable amount of computation time.

The rest of this article is structured as follows. We describe the dynamics of the LMM and fix notations. We review the basic forward calculations of pathwise Greeks (delta and vega) and then describe the fundamental principles of the adjoint method, both for European-style derivatives only. Next, we describe how the usual forward framework can be extended to value Bermudan options: after a forward procedure we need to work backwards to calculate the optimal exercise times. Based on this idea, we develop the modifications necessary for the adjoint method. Numerical applications demonstrate the extended adjoint method. Interestingly, both adjoint extensions are based on the originally developed pathwise forward method.

LMM and its implementation

We consider the basic assumptions of the LMM of Brace, Gatarek & Musiela (1997). The LMM consists of M assets, namely M bonds. For these bonds, a set of M maturities $\{T_i\}_{i=1}^{M}$ is given with $T_0 := 0 < T_1 < \cdots < T_M$, which are referred to as the tenor structure, and the maturity of the ith bond is T_i. The horizon of the LMM is defined as the maturity T_{M-1}. The time steps $\delta_i := T_{i+1} - T_i$ for $i = 0, \ldots, M-1$ are referred to as the tenor distances.

Jamshidian (1997) developed an approach to model forward Libors with respect to the spot measure. Let $\tilde{L}_i(t)$ denote the forward Libor fixed at time $t \leq T_i$ for the interval $[T_i, T_{i+1})$ with $i = 1, \ldots, M-1$. In addition, we assume each forward volatility $\sigma_i(t)$ (d-dimensional) to be deterministic. For practical purposes, it often suffices to restrict the volatility dependence on time to piecewise constant functions that change value only at the T_i. Further, we define $\eta(t)$ to describe the index of the next tenor date after t. The arbitrage-free dynamics of the forward Libors under the spot measure is governed by the stochastic differential equation (SDE):

$$\frac{d\tilde{L}_i(t)}{\tilde{L}_i(t)} = \mu_i\left(t, \tilde{L}(t)\right)dt + \sigma_i^{\top}(t)\,dW(t), \qquad (1)$$
$$t \in [0, T_i], \quad i = 1, \ldots, M-1$$

where W is a d-dimensional standard Brownian motion under the spot measure and the drift is given by:

$$\mu_i\left(t, \tilde{L}(t)\right) := \sum_{j=\eta(t)}^{i} \frac{\delta_j \tilde{L}_j(t)\sigma_i^{\top}(t)\sigma_j(t)}{1 + \delta_j \tilde{L}_j(t)}$$

To simulate the forward Libors, we fix a time grid $0 = t_0 < t_1 < \cdots < t_N = T_{M'}$ with time steps $h_n := t_{n+1} - t_n$ for $n = 0, \ldots, N-1$, where $M' < M$ and $\{T_0, \ldots, T_{M'}\} \subseteq \{t_0, \ldots, t_N\}$. Using Itô's lemma and the Euler scheme applied to the logarithms of the forward Libors yields:

$$L_i(n+1)$$
$$= L_i(n)\exp\left(\left(\mu_i(n) - \frac{1}{2}\left\|\sigma_i(n)\right\|^2\right)h_n + \sqrt{h_n}\,\sigma_i^{\top}(n)Z(n+1)\right) \qquad (2)$$
$$n = 0, \ldots, N-1, \quad i = \eta(n), \ldots, M-1$$

with new notations $L_i(n) := L_i(t_n)$, $\mu_i(n) := \mu_i(t_n, L(t_n))$, $\sigma_i(n) := \sigma_i(t_n)$ and $\eta(n) := \eta(t_n)$, where $Z(1), \ldots, Z(N)$ are independent standard normal random vectors in $\mathbb{R}^d$. For efficiency, we calculate first:

$$S_i(n) := \sum_{j=\eta(n)}^{i} \frac{\delta_j L_j(n)\sigma_j(n)}{1+\delta_j L_j(n)}, \quad i = \eta(n), \ldots, M-1$$

and then calculate $\mu_i(n) = \sigma_i^{\top}(n) S_i(n)$, hence the total computational cost is $O(M)$ per time step. Furthermore, we use the convention that $L_i(n+1) = L_i(n)$ if $i < \eta(n)$.

More details on how to simulate the LMM under the spot measure can be found in Glasserman (2004), section 3.7.

Next, we consider the expected value of a discounted payout $g(\tilde{L}(T_M))$ of an interest rate derivative in LMM, where $\tilde{L}(t)$ is d-dimensional and satisfies the SDE (1). The simulated form is denoted as $g(L(N))$, and $L(N)$ is given through the equation (2). We can write $\partial g(L(N))/\partial L(0)$ for the vector of deltas of $g(L(N))$ and write also $\partial g(L(N))/\partial\theta$ for the vega of $g(L(N))$, where θ denotes a parameter of the forward volatilities. A sufficient condition for the correctness of the pathwise estimates of the deltas and vegas is that the discounted payout function g is smooth enough.

Forward method: pathwise deltas and vegas

Glasserman & Zhao (1999) developed the application of the forward method to estimate the deltas and vegas. Based on their notation, we recap the estimation of deltas. Let $\Delta(n) := \partial L(n)/\partial L(0) \in \mathbb{R}^{M\times M}$:

$$\Delta_{ij}(n) := \frac{\partial L_i(n)}{\partial L_j(0)}, \quad n = 0,\ldots,N, \quad i = 0,\ldots,M-1, \quad j = 0,\ldots,i$$

We thus arrive at the forward estimative formula of the vector of the deltas:

$$\begin{aligned} &\boldsymbol{\Delta}\big(g(L(N))\big) \\ &:= \frac{\partial g(L(N))}{\partial L(0)} = \frac{\partial g(L(N))}{\partial L(N)}\frac{\partial L(N)}{\partial L(0)} = \frac{\partial g(L(N))}{\partial L(N)}\Delta(N) \end{aligned} \tag{3}$$

To implement $\Delta(N)$ accurately, we calculate the partial derivatives with respect to the vector $L(0)$ on both sides of the equation (2). So we obtain the recursion $\Delta_{ij}(n+1) = \Delta_{ij}(n)$ for $i < \eta(n)$ and $i \geq \eta(n)$:

$$\begin{aligned} &\Delta_{ij}(n+1) \\ &= \frac{L(n+1)}{L_i(n)}\Delta_{ij}(n) + L_i(n+1)h_n\sigma_i^{\top}(n)\sum_{k=\eta(n)}^{i}\frac{\delta_k\sigma_k(n)\Delta_{kj}(n)}{\big(1+\delta_k L_k(n)\big)^2} \end{aligned} \tag{4}$$

with the initial matrix $\Delta(0) = I_M$. The computational cost per recursive step of implementing (4) is $O(M^2)$ for a fixed index i, since there is an $O(M)$ cost in computing the summations for each j. The recursion (4) can also be written as:

$$\Delta(n+1) = D(n)\Delta(n), \quad D(n) \in \mathbb{R}^{M\times M} \tag{5}$$

where for $i = j$:

$$D_{ii}(n) := \begin{cases} 1 & i < \eta(n) \\ \frac{L_i(n+1)}{L_i(n)} + \frac{L_i(n+1)\|\sigma_i(n)\|^2\delta_i h_n}{(1+\delta_i L_i(n))^2} & i \geq \eta(n) \end{cases}$$

and for $i \neq j$:

$$D_{ij}(n) := \begin{cases} \frac{L_i(n+1)\sigma_i^{\top}(n)\sigma_j(n)\delta_j h_n}{(1+\delta_j L_j(n))^2} & i > j \geq \eta(n) \\ 0 & \textit{otherwise} \end{cases}$$

Next, we address the estimation of vegas through the pathwise forward method. Let $\Theta(n) := \partial L(n)/\partial\theta \in \mathbb{R}^M$. We then calculate the vegas by means of:

$$\begin{aligned} \boldsymbol{\Theta}\big(g(L(N))\big) &:= \frac{\partial g(L(N))}{\partial\theta} \\ &= \frac{\partial g(L(N))}{\partial L(N)}\frac{\partial L(N)}{\partial\theta} = \frac{\partial g(L(N))}{\partial L(N)}\Theta(N) \end{aligned} \tag{6}$$

where $\Theta(N)$ is computed by the recursion:

$$\Theta(n+1) = D(n)\Theta(n) + B(n) \tag{7}$$

and the initialisation $\Theta(0) = \mathbf{0}$. To be specific, $B(n)$ is given as:

$$\begin{aligned} B_i(n) &= L_i(n+1)h_n\sigma_i^{\top}(n)\sum_{k=\eta(n)}^{i}\frac{\delta_k L_k(n)}{1+\delta_k L_k(n)}\frac{\partial\sigma_k(n)}{\partial\theta} \\ &+ L_i(n+1)\Big(h_n S_i^{\top}(n) - h_n\sigma_i^{\top}(n) + \sqrt{h_n}Z^{\top}(n+1)\Big)\frac{\partial\sigma_i(n)}{\partial\theta} \end{aligned} \tag{8}$$

for $i = \eta(n), \ldots, M-1$ and $B_i(n) = 0$ otherwise.

Adjoint method: pathwise deltas and vegas

Giles & Glasserman (2006) introduced an alternative to the forward pathwise approach and called it the adjoint method. To estimate the deltas and vegas they calculate the Greeks in a backward direction. The idea for the adjoint deltas can be shown as follows:

$$\begin{aligned} \boldsymbol{\Delta}\big(g(L(N))\big) &\overset{(3)}{=} \frac{\partial g(L(N))}{\partial L(N)}\Delta(N) \\ &\overset{(5)}{=} \frac{\partial g(L(N))}{\partial L(N)}D(N-1)D(N-2)\cdots D(0)\Delta(0) \\ &=: V^{\top}(0) \end{aligned} \tag{9}$$

where $V(0)$ can be calculated through the following backward recursion:

$$V(n) = D^{\top}(n)V(n+1), \quad V(N) = \left(\frac{\partial g(L(N))}{\partial L(N)}\right)^{\top} \tag{10}$$

and we note that $\Delta(0) = I_M$.

The precise formula to compute $V(0)$ after initialising $V(N)$ is therefore:

$$\begin{aligned} V_i(n) &= \frac{L_i(n+1)}{L_i(n)}V_i(n+1) \\ &+ \frac{\sigma_i^{\top}(n)\delta_i h_n}{\big(1+\delta_i L_i(n)\big)^2}\sum_{j=i}^{M-1}L_j(n+1)V_j(n+1)\sigma_j(n) \end{aligned} \tag{11}$$

for $i \geq \eta(n)$ and $V_i(n) = V_i(n+1)$ for $i < \eta(n)$. The summation can be calculated at a cost $O(M)$, hence for a fixed index i there is an $O(M)$ cost in updating each recursive step of (11). Similarly, we can write the principle of the adjoint method to estimate the vegas:

$$\begin{aligned}
\mathbf{\Theta}\big(g\big(L(N)\big)\big) &\overset{(6)}{=} \frac{\partial g\big(L(N)\big)}{\partial L(N)}\Theta(N) \\
&\overset{(7)}{=} \frac{\partial g\big(L(N)\big)}{\partial L(N)}\big(B(N-1)+D(N-1)B(N-2)+D(N-1)\cdots \\
&\quad +\cdots + D(N-1)D(N-2)\cdots D(1)B(0)\big) \\
&\overset{(10)}{=} V^{\top}(N)B(N-1)+V^{\top}(N-1)B(N-2)+\cdots+V^{\top}(1)B(0) \\
&= \sum_{n=0}^{N-1} V^{\top}(n+1)B(n)
\end{aligned}$$

From the analysis above, we conclude that the strength of the adjoint method is based on a vector-vector recursion compared with the vector-matrix recursion of the forward method. Now we try to develop this idea for Bermudan-style derivatives.

Deltas and vegas of Bermuda swaptions

A Bermudan swaption refers to a modified American style of swaption. It grants the holder the right to enter into the underlying instrument, for instance a fixed-for-floating interest rate swap, at each prearranged exercise date in a schedule, provided that the holder has not exercised the right at any previous time.

Let us define a Bermudan swaption formally. An $H \times M$ Bermudan swaption is defined on the tenor structure $0 = T_0 < T_1 < \cdots < T_M$ with tenor distances $\delta_n := T_{n+1} - T_n$ for $n = 0, \ldots, M-1$. It can enter the underlying instrument on any of the dates $\{T_n\}_{n=H}^{M-1}$. If the Bermudan swaption is exercised at time $T_r \geq T_H$, that is, T_r is the optimal exercise time, then the underlying instrument is a stream of payments $\{X_n\}_{n=r}^{M-1}$. Because the underlying instrument is actually a fixed-for-floating interest rate swap, under the LMM these payments are:

$$X_n := \phi\mathcal{N}\delta_n\big(L_n(n)-R\big)$$

for $n = r, \ldots, M-1$, where R is the fixed rate and $\mathcal{N}$ is the nominal value. In addition, it is a payer swap if $\phi = 1$ and a receiver swap if $\phi = -1$, which respectively correspond to the payer and receiver Bermudan swaption. Furthermore, notice that each X_n is determined at time T_n, but the payment is made at time T_{n+1}. Hence the discount factor of the payment X_n at time T_{n+1} is:

$$PV_{n+1} := \prod_{i=0}^{n}\frac{1}{1+\delta_i L_i(i)}$$

Then the expected value of the $H \times M$ Bermudan swaption in the LMM under the spot measure at time T_0 is given as:

$$P^{BS}(0) := \mathbb{E}\left(\sum_{n=r}^{M-1} PV_{n+1}X_n\right) \qquad (12)$$

The essential problem in valuing a Bermudan swaption via the Monte Carlo method is according to (12) the determination of the optimal exercise time in each simulated path, where the exercise value is maximised. A general approach to choosing the optimal exercise time and value of a Bermudan swaption is the so-called least-squares Monte Carlo (LSM) algorithm, which is addressed in Longstaff & Schwartz (2001). They propose the following procedure: define regression variables for each possible exercise time and then regress the continuation value and compare them with the exercise value in a backward simulation. We consider only a single explanatory variable, the swap net present value. Three regression functions were employed, a constant, a linear and a quadratic term.

Adjoint estimation of deltas and vegas

The pathwise deltas of an $H \times M$ Bermudan swaption can be approximated as follows:

$$\mathbf{\Delta}\big(P^{BS}(0)\big) \approx \mathbb{E}\left(\sum_{n=r}^{M-1}\mathbf{\Delta}\big(PV_{n+1}X_n\big)\right) \qquad (13)$$

where r is an estimate of the optimal exercise time index calculated during the LSM algorithm. This elegant formula was developed by Piterbarg (2004). It allows the deltas of Bermudan swaptions through the Monte Carlo method to be determined.

Now we develop a new fast Monte Carlo algorithm to estimate the deltas and vegas of Bermudan swaptions, which is based on the formula (13) and the adjoint method described above. First, we suppose $V_n^{\top}(N_n) := \partial(PV_{n+1}X_n)/\partial L(n)$ [1] for $n = r, \ldots, M-1$. Then we define a new vector $\mathbf{V}$ for further analysis. Let:

$$\mathbf{V}\big(N_{M-1}\big) := V_{M-1}\big(N_{M-1}\big)$$

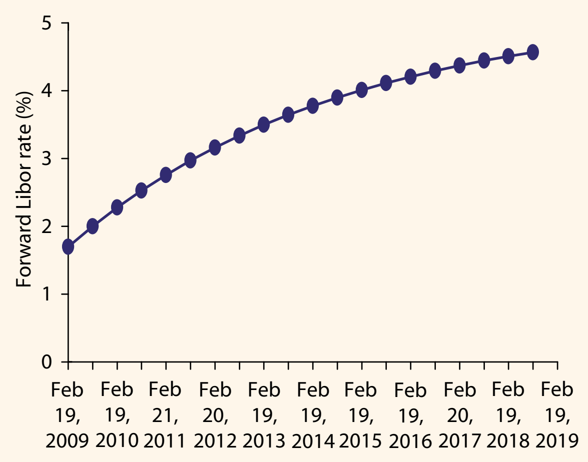
and the following backward recursion for $m = N_{M-1}-1, \ldots, 0$:

$$\mathbf{V}(m) := \begin{cases} D^{\top}(N_l)\mathbf{V}(N_l+1)+V_l(N_l) & m \in \{N_l \mid l = r,\ldots,M-2\} \\ D^{\top}(m)\cdots D^{\top}(N_r-1)\mathbf{V}(N_r) & m < N_r \\ D^{\top}(m)\cdots D^{\top}\big(N_{\eta(m)}-1\big)\mathbf{V}\big(N_{\eta(m)}\big) & otherwise \end{cases}$$

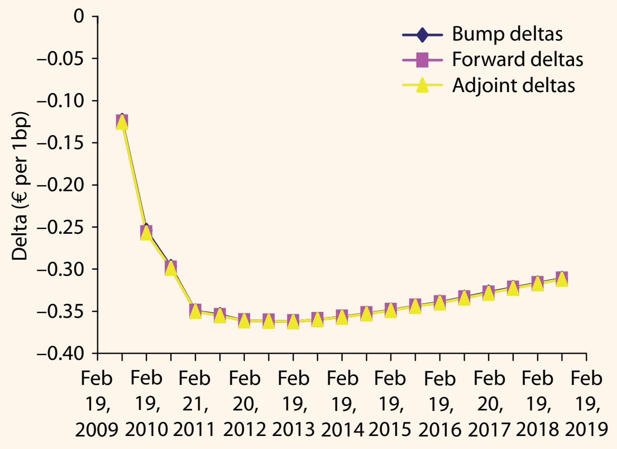
The important equation, which presents the principle for the fast estimation of adjoint deltas, is:

$$\begin{aligned}
&\sum_{n=r}^{M-1}\mathbf{\Delta}\big(PV_{n+1}X_n\big) = \sum_{n=r}^{M-1} V_n(0) \\
&\overset{(9)}{=} \sum_{n=r}^{M-1} D^{\top}(0)D^{\top}(1)\cdots D^{\top}(N_n-1)V_n(N_n) \\
&= D^{\top}(0)D^{\top}(1)\cdots D^{T}(N_r-1) \\
&\quad \times\big[V_r(N_r)+D^{\top}(N_r)\cdots D^{\top}(N_{r+1}-1) \\
&\quad \times\big[V_{r+1}(N_{r+1})+\cdots+D^{\top}(N_{n-1})\cdots D^{\top}(N_n-1) \\
&\quad \times\big[V_n(N_n)+\cdots D^{\top}(N_{M-3})\cdots D^{\top}(N_{M-2}-1) \\
&\quad \big[V_{M-2}(N_{M-2})+D^{\top}(N_{M-2})\cdots D^{\top}(N_{M-1}-1)\times V_{M-1}(N_{M-1})\big] \\
&\quad \cdots\big]\cdots\big]\big] \\
&\overset{(14)}{=} D^{\top}(0)D^{\top}(1)\cdots D^{\top}(N_{M-1}-1)\mathbf{V}(N_{M-1})
\end{aligned}$$

1. Market data of forward Libors at Feb 19, 2009



2. Bermuda swaption deltas calculated using adjoint method versus those calculated by forward method and bump method



Note: bump sizes 0.0001 × 1bp, for all relevant Libors. Bump deltas are calculated with 65,536 paths. Forward deltas and adjoint deltas are calculated with 32,768 paths

The corresponding adjoint vega is as a matter of course the summation of scalar products:

$$\sum_{m=0}^{N_{M-1}-1} \mathbf{V}^{\top}(m+1)B(m)$$

since the scalar product is a bilinear operation.

Our new efficient algorithm extends the Monte Carlo method originally proposed by Giles & Glasserman (2006).

■ **Algorithm 1** (adjoint deltas and vegas of an H x M Bermudan swaption):

■ (1) Running a forward simulation of the LMM on the tenor structure $\{T_0, \dots, T_M\}$ under the spot measure with p paths.

■ (2) Performing the LSM algorithm, for each path ω determining the optimal exercise time $T_{r(\omega)}$.

■ (3) For each path ω:

(a) calculate $\mathbf{V}^{\top}(N_{M-1}) \leftarrow V^{\top}_{M-1}(N_{M-1})$.

(b) for n from $M-2$ to $r(\omega)$ do

(i) Calculate $\mathbf{V}(m)$ backwards from $m = N_{n+1}$ to $m = N_n$ in accordance with the recursive formula (11).

(ii) Calculate $\mathbf{V}(N_n) \leftarrow \mathbf{V}(N_n) + V_n(N_n)$

end for

(c) Computing $\mathbf{V}(m)$ backwards from $m = N_r(\omega)$ to $m = 0$ according to the recursive formula (11).

(d) Calculating the delta vector as $\mathbf{V}^{\top}(0)$ or vega as:

$$\sum_{m=0}^{N_{M-1}-1} \mathbf{V}^{\top}(m+1)B(m)$$

for the $H \times M$ Bermudan swaption regarding the path ω.

■ (4) Average the result over all paths.

Calibration and numerical results

We first look at a 1 × 20 Bermudan swaption with half-year constant tenor distances (in financial terms: a 0.5 × 10 swaption), whose underlying is a 4.5% receiver swap, that is, we receive a 4.5% fixed rate and pay Libor at each tenor date. The nominal value is €10,000.

Our set-up is a one-factor model. For the tests, we used the interest rate curve of forward Libors (figure 1) and the at-the-money swaption volatilities of February 19, 2009. We transformed the at-the-money swaption volatilities into forward volatilities (see Brigo & Mercurio, 2001, for further details).

To check the consistency of our method, we also calculate the deltas and vegas with the bump and revalue method mentioned in the introduction and the forward method. The forward method is essentially the method developed by Piterbarg (2004).

We then estimate the deltas and vegas of the above 1 × 20 Bermudan swaption for all the three methods (bump method with 65,536 paths and bump sizes 0.0001 × 1bp, forward method with 32,768 paths and adjoint method with 32,768 paths), which are displayed in figures 2 and 3. They show that the adjoint method has the same accuracy as the forward method. Both methods are much faster than the traditional bump method, which is shown for comparison.

Test for efficiency

Next we compare the efficiency of the forward method and our improved adjoint method for the deltas and vegas of Bermudan

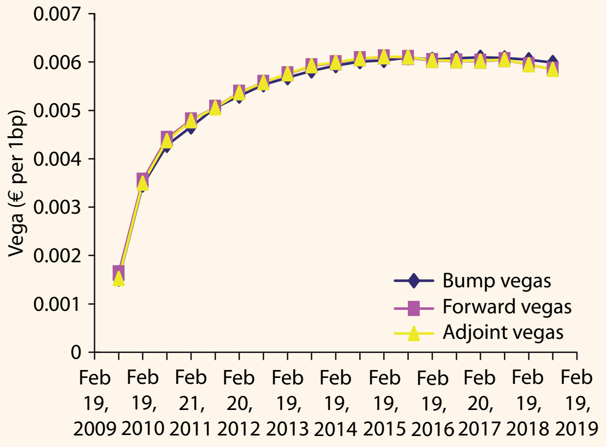
[1] *The equation implies automatically that:*

$$V_n^{\top}(0) = V_n^{\top}(N_0) = \frac{\partial PV_{n+1}X_n}{\partial L(0)} = \Delta(PV_{n+1}X_n)$$

Here, N_n come from $t_{N_n} = T_n$ for $n = 0, \dots, M-1$ and in more detail:

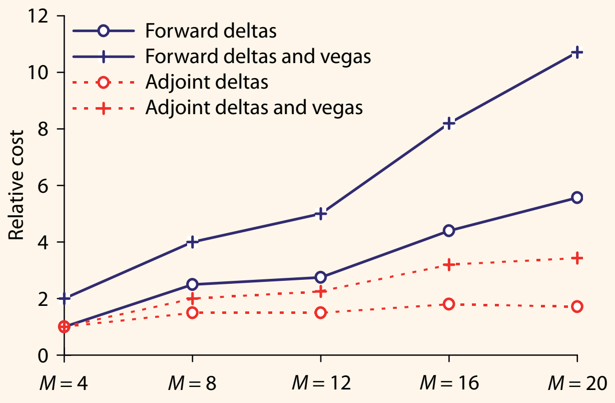
$$V_n^{\top}(N_n)_j$$
$$= -\frac{\phi N \delta_j \delta_n (L_n(n) - R)}{1 + \delta_j L_j(j)} \prod_{i=0}^{n} \frac{1}{1 + \delta_i L_i(i)} \textit{ for } j = 1, \dots, n-1$$
$$V_n^{\top}(N_n)_j$$
$$= \phi N \delta_n \left(-\frac{\delta_j (L_n(n) - R)}{1 + \delta_j L_j(j)} + 1 \right) \prod_{i=0}^{n} \frac{1}{1 + \delta_i L_i(i)} \textit{ for } j = n$$
$$V_n^{\top}(N_n)_j$$
$$= 0 \textit{ for } j = n+1, \dots, M-1$$

3. Bermuda swaption vegas calculated using adjoint method versus those calculated by forward method and bump method



Note: bump sizes 0.0001 × 1bp for all relevant Libors. Bump vegas are calculated with 65,536 paths. Forward vegas and adjoint vegas calculated with 32,768 paths

4. Relative cost of the evaluations for forward and adjoint deltas and vegas of five 1 × *M* receiver Bermuda swaptions



swaptions. For this purpose, we test five $1 \times M$ receiver Bermudan swaptions ($M = 4, 8, 12, 16, 20$) with half-year constant tenor distances. The fixed rate is 4.5% and the nominal value is €10,000. We use the same LMM and the same calibrated data from the market as above. Furthermore, we simulate each method with 32,768 paths and use the same sequence of random numbers.

Figure 4 plots the relative cost[2] for the forward and adjoint method to calculate the deltas and vegas of the above Bermudan swaptions. The two curves with circles compare the forward and adjoint evaluations of deltas. The other two curves with crosses compare the evaluations of deltas and vegas. Figure 4 shows that the relative cost of the adjoint method is roughly constant, whereas the relative cost of the forward method increases nearly linearly with M. Compared with the relative cost figure in the original work by Giles & Glasserman (2006), we see a slight increase in the calculation of the adjoint delta and vega. This is a result of the LSM algorithm. So the adjoint method is much faster than the forward method when calculating deltas and vegas of Bermudan swaptions, especially for big M, that is, long maturities.

Conclusions

We have introduced an innovative adjoint algorithm to calculate the deltas and vegas of Bermudan swaptions, which is a very fast technique within simulation of the LMM. Bermudan swaptions belong to callable Libor exotics and are by far the most prevalent and important type of those. The other callable Libor exotics (for example, callable capped floaters and callable inverse floaters) differ from the Bermudan swaptions for the most part only in the computation of the coupons X_n. Thus we can also calculate the deltas and vegas for these callable Libor exotics through the same adjoint procedure. ●

[2]
$$\text{Relative cost} = \frac{\text{Time cost of computing deltas/vegas of the Bermuda swaption}}{\text{Time cost of valuing the Bermuda swaption}}$$

Matthias Leclerc is executive director at Value&Risk, Frankfurt and professor of mathematics at the University of Augsburg. Qian Liang is a graduate student in mathematics at the University of Kaiserslautern & Fraunhofer ITWM. Ingo Schneider is head of financial engineering at DekaBank. The result of this article is based on the diploma thesis of the second author at the University of Augsburg. Email: ingo.schneider@deka.de

REFERENCES

Brace A, D Gatarek and M Musiela, 1997
The market model of interest rate dynamics
Mathematical Finance 7, pages 127–155

Brigo D and F Mercurio, 2001
Interest rate models, theory and practice
Springer Verlag

Giles M and P Glasserman, 2006
Smoking adjoints: fast Monte Carlo Greeks
Risk January, pages 88–92

Glasserman P, 2004
Monte Carlo methods in financial engineering
Springer Verlag

Glasserman P and X Zhao, 1999
Fast Greeks by simulation in forward Libor models
Journal of Computational Finance 3, pages 5–39

Jamshidian F, 1997
Libor and swap market models and measures
Finance and Stochastics 1, pages 293–330

Longstaff F and E Schwartz, 2001
Valuing American options by simulation: a simple least-squares approach
Review of Financial Studies 14, pages 113–147

Piterbarg V, 2004
Computing deltas of callable Libor exotics in forward Libor models
Journal of Computational Finance 7(3), pages 107–144