Computing Pricing Functions by Simulation

Stéphane CRÉPEY, Évry University, France
stephane.crepey@univ-evry.fr

April 10, 2008

Contents

I American and Game Options in a Local Volatility Model **3**

1 American Options **3**

2 Game Options **4**

2.1 No Call Protection 4

2.2 Standard Soft Call Protection 4

2.3 Highly Path dependent Soft Call Protection 5

2.4 Real-Life Call Protection 6

II CDO tranches in a Local Intensity Model **7**

3 Pricing **7**

4 Hedging **8**

Introduction

This note bears on the computation by simulation methods of financial derivatives pricing functions (pricing function and delta function). Note that we want to compute the pricing *function* for any possible values of its arguments (or, at least, on a fine grid), and not only the option price (and delta) at time 0. There are various situations in which it is important to compute the pricing function as a whole. The first one is of course when there is no other way for computing the price at 0 than computing the pricing function entirely, as it is the

case for options with control-theoretical features (typically: early exercise clauses, cf. part I). Another situation (illustrated in part II) is when one wants to assess the performance of a dynamic hedging scheme for the option, statistically on a great number of trajectories of the underlying. In this case one needs to know the option pricing function at every point of every simulated trajectory (in order to know by which amount one should rebalance the hedge at every point), hence basically everywhere.

In part I the set-up is a rather standard local volatility model (Black–Scholes-like model with local volatility, and, also, drift coefficient). Here the motivation for computing our pricing functions by simulation rather than by more standard (deterministic) numerical analysis approximation schemes comes from the fact that we want to be able to deal with highly path-dependent option payoffs, resulting in high-dimensional pricing functions. The use of deterministic approximation schemes is thus precluded by the curse of dimensionality. In part II deterministic approximation schemes might be considered as an alternative to our simulation method, but our motivation is accuracy, in a context of credit-risk where the information in the model consists of random times of jumps of a cumulative loss process on a credit portfolio. The interest of the simulation method proposed in this note is to be able to use this information exactly, whereas a deterministic scheme on a fixed time-grid would involve an approximation of jump times by projecting them in some way on the time-grid.

Part I

American and Game Options in a Local Volatility Model

We first consider the problem of computing by simulation the pricing function of American or more general Game options (like Convertible Bonds, with possibly highly path-dependent exotic clauses). To fix ideas, we consider the following (risk-neutral) *local volatility model*:

$$dS_t = S_t(\kappa(t, S_t)dt + \sigma(t, S_t)dW_t) \tag{1}$$

for a standard $\mathbb{P}$ – Brownian motion W, a local (risk-neutral, accounting for riskless interest-rates, dividend yields on S, credit-risk adjustment on S, etc.) drift coefficient $\kappa(t, S)$, and a local volatility function $\sigma(t, S)$. As is well-known, the diffusion (1) is well-posed in suitable space of solutions under relevant regularity and growth hypotheses (assumed henceforth) on the coefficients, and the process S defined in this way is a Markov process.

1 American Options

The generic multinomial (recombining) tree algorithm for pricing American options writes $\Pi_n(j) = \phi(S^j)$ for $j = 1 \dots m$, and then for $i = n-1, \dots, 0$, for $j = 1 \dots m$ (where i and j index the time and space step in the algorithm, respectively):

$$\Pi_i^j = \max\left(\phi\left(S^j\right), e^{-\kappa_i^j h} \sum_l p_l \Pi_{i+1}^{j+l}\right) . \tag{2}$$

For pricing an American option by Monte Carlo, a procedure consists in writing the generic dynamic programming equation (2) on a *stochastically generated (hence, non recombining) mesh* $(S_i^j)_{0 \le i \le n, 1 \le j \le m}$, using an appropriate discretization (Euler,..) scheme for the underlying diffusion. We thus get the following amendment to (2): $\Pi_n^j = \phi(S_n^j)$ for $j = 1 \dots m$, and then for $i = n-1, \dots, 0$, for $j = 1 \dots m$:

$$\Pi_i^j = \max\left(\phi\left(S_i^j\right), e^{-\kappa_i^j h} \mathbb{E}_{i,j} \Pi_{i+1}\right) \tag{3}$$

where $\mathbb{E}_{i,j}\Pi_{i+1}$ stands for the conditional expectation of Π_{i+1} given $S_i = S_i^j$. The problem thus reduces to computing conditional expectations (for $i \ge 1$, since for $i = 0$ the conditional expectation reduces to a simple expectation).

Recall that under mild conditions the conditional expectation $\mathbb{E}(X|Y)$ is equal to the Hilbert space projection $\mathbb{EL}(X|Y^0, Y^1, Y^2, \dots)$, or more generally $\mathbb{EL}(X|\varphi^0(Y), \varphi^1(Y), \varphi^2(Y), \dots)$, for a suitable basis $\varphi = (\varphi^l)_{l \in \mathbb{N}}$ of the set of the univariate real functions. At step $i \ge 1$ the involved conditional expectations may thus be computed in an elementary way by linear regression of the $(\Pi_{i+1}^j)_{1 \le j \le m}$ (which are already known at step i of the algorithm) against the $(\varphi^l(S_i^j))_{1 \le j \le m, 0 \le l \le q}$ where the integer q is a parameter in the method.

The computational cost of the regression is of the order of $O(m^2 q^2)$, hence an overall computational cost as $O(nm^2q^2)$. This is obviously too much for typical values of the parameters (e.g., $n = 100, m = 10^5, q = 7$). Fortunately this can be improved in a number of ways,

leading to methods perfectly amenable to a practical resolution, for problems of dimension up to 10 or more (whereas deterministic methods are out of scope for dimensions greater than 4).

The previous approach is the one developed in Longstaff and Schwartz [9]. For many alternative methods for computing the conditional expectations: more general *non-parametric regression* methods, Malliavin Calculus based methods, etc., we refer the reader to the literature (see Broadie and Glassermann [5, 6], Lions and Régnier [8], Pages and Bally [10], Tsitsiklis and VanRoy [12, 13], Bouchard et al. [2], among many).

Note that a *confidence interval* is not available in this case (it is only possible to compute a confidence interval *of the method* by randomizing the seed of the (pseudo-)generator which is used). One can however derive an upper bound on the price by resorting to a suitable dual Monte Carlo approach (see Rogers [11]). Since most pricing methods provide lower bounds, we thus end up with an interval.

Remark 1.1 It is also possible to compute likewise by non-parametric regression the *delta function*, cf. section 4.

2 Game Options

2.1 No Call Protection

In the case of a Game Option (e.g., a non-defaultable sero-coupon convertible bond) with no call protection, (3) simply needs to be amended as $\Pi_n^j = \phi(S_n^j)$ (with, e.g., $\phi(S) = \bar{N} \vee \kappa S$) for $j = 1 \dots m$, and then for $i = n-1, \dots, 0$, for $j = 1 \dots m$:

$$\Pi_i^j = \min\left(U_i(S_i^j)\ ,\ \max\left(L_i(S_i^j),\, e^{-\kappa_i^j h}\mathbb{E}_{i,j}\Pi_{i+1}\right)\right) \tag{4}$$

where L_i and U_i are the put and call payoff functions at step i (time ih) in the algorithme, e.g.

$$L_i(S) = \bar{P} \vee \kappa S\ ,\ U_i(S) = \bar{C} \vee \kappa S\ .$$

2.2 Standard Soft Call Protection

We now consider a soft call protection of the form $\{t \geq \bar{\tau}\}$ with

$$\bar{\tau} = \inf\{t \in \mathbb{R}_+\,;\, S_t \geq \bar{S}\} \wedge T$$

The related algorithm becomes:
• First compute the *no call protection pricing function* Π as in the previous subsection,
• Then, setting

$$\nu^j = \inf\{i \in \mathbb{N}_n\,;\, S_i^j \geq \bar{S}\} \wedge T\ ,$$

compute $\widetilde{\Pi}_i^j = \Pi_i^j$ on $\{(i,j); i \geq \nu^j\}$, and then for $i = n-1, \dots, 0$ for $j = 1 \dots m$, if $i < \nu^j$:

$$\widetilde{\Pi}_i^j = \max\left(L_i\left(S_i^j\right),\, e^{-\kappa_i^j h}\mathbb{E}_{i,j}\widetilde{\Pi}_{i+1}\right) \tag{5}$$

For $i \geq 1$ the conditional expectations in (6) may be computed by linear regression of the $(\widetilde{\Pi}^j_{i+1})_{1\leq j\leq m}$ (which are already known at step i of the algorithm) against the $(\varphi^l(S^j_i))_{1\leq j\leq m, 0\leq l\leq q}$, , for a suitable basis φ of the set of univariate real functions.

2.3 Highly Path dependent Soft Call Protection

Given a *decreasing* sequence of *monitoring dates* $T_0 = T, T_1, \ldots, T_k, \ldots$, $k \in \mathbb{N}$, we denote for $t < T$:

$$k_t = \max\{k \in \mathbb{N}^*; t < T_{k-1}\}\ ,\ \mathcal{S}_t = (\mathcal{S}^k_t)_{1\leq k\leq 30} = (S_{T_k})_{k_t\leq k\leq k_t+29}\ ,\ N_t = \#\mathcal{S}_t := \#\{1 \leq k \leq 30; \mathcal{S}^k_t \geq \bar{S}\}$$

We consider the call protection $t \geq \bar{\tau}$ with

$$\bar{\tau} = \inf\{t \in \mathbb{R}_+\,;\, N_t \geq 20\} \wedge T$$

The option upper payoff process at time t is now a function of time t, S_t and $\mathcal{S}_t$. Since $(S, \mathcal{S})$ is a Markov process, the option pricing function is thus in turn given as a function of 31 space variables (beyond time).

Given a time mesh $(t_i)_{0\leq i\leq n}$ *refining* $0, T_{k_0-1}, T_{k_0-2} \ldots, T_0$, on $[0, T]$, and the (observed) value of $(S_0, \mathcal{S}_0)$ at time 0, we then generate a grid $(S^j_i, \mathcal{S}^j_i)_{0\leq i\leq n, 1\leq j\leq m}$ in the obvious way, using past values of S to fill $\mathcal{S}$. Let also N^j_i stand for $\#\mathcal{S}^j_i$.

Example 2.1 Black–Scholes model with $r = 0$ and $\sigma = 20\%$. Having set $t_{i+1} - t_i = \frac{1}{4}$days:
• simulate, starting from S_0 given, $S_{6h} = S_0(1 + \sigma\sqrt{h}\varepsilon_1)$, $S_{12h} = S_{6h}(1 + \sigma\sqrt{h}\varepsilon_2)$, $S_{18h} = .., S_{24h}, S_{30h}, ..., S_{100days}$ for standard IID Gaussian r.v. ε_i;
• whenever t_i coincides with one the T_k's (i.e., one every fourth i), update the vector $\mathcal{S}$, so: $\mathcal{S}_0 = 30$ given values of S (representing past values of S corresponding to the 30 last monitoring dates T_k preceding the pricing time $t = 0$), $\mathcal{S}_{6h} = \mathcal{S}_0$, $\mathcal{S}_{12h} = \mathcal{S}_0$, $\mathcal{S}_{18h} = \mathcal{S}_0$, $\mathcal{S}_{1day} = (S_{1day}, \bar{\mathcal{S}}_0)$ where $\bar{\mathcal{S}}_0$ stands for the vector made of the components 1 to 29 of $\mathcal{S}_0$, $\mathcal{S}_{30h} = \mathcal{S}_{1day}$, etc..until $\mathcal{S}_{100days}$
• Redo this $m = 10^5$ times, hence 10^5 (pairs of) trajectories $(S_i, \mathcal{S}_i)_{0\leq i\leq n}$.

The pricing algorithm at time 0 then goes as follows:
• First compute the *no call protection pricing function* $(\Pi^j_i)_{0\leq i\leq n, 1\leq j\leq m}$ as in the previous subsection;
• Then, setting

$$\nu^j = \inf\{i \in \mathbb{N}_n\,;\, N^j_i \geq 20\} \wedge T\ ,$$

compute $\widetilde{\Pi}^j_i = \Pi^j_i$ on $\{(i, j); i \geq \nu^j\}$, and then for $i = n-1, \ldots, 0$ for $j = 1 \ldots m$, if $i < \nu^j$:

$$\widetilde{\Pi}^j_i = \max\left(L_i(S^j_i),\, e^{-\kappa^j_i h}\mathbb{E}_{i,j}\widetilde{\Pi}_{i+1}\right) \tag{6}$$

For $i \geq 1$ the conditional expectations in (6) may be computed by linear regression of the $(\widetilde{\Pi}^j_{i+1})_{1\leq j\leq m}$ (which are already known at step i of the algorithm) against the $(\varphi^l(S^j_i, \mathcal{S}^j_i))_{1\leq j\leq m, 0\leq l\leq q}$, for a suitable basis φ of the set of 30-variate real functions.

Remark 2.2 Here the requirement to work with a basis of 30-variate real functions represents of course a huge gap of complexity with respect to the previously considered (univariate) cases.

2.4 Real-Life Call Protection

A further difficulty is that real-life call protection clauses at time t (see, e.g., [1]) seem to be typically of the form $S_t \geq \bar{S}$, rather than $\sup_{[0,t]} S_u \geq \bar{S}$.

In the abstract set-up of Bielecki et al. [3], this involves considering generalized upper barriers of the form

$$\bar{U}_t = \mathbb{1}_{\Omega_t^c} \infty + \mathbb{1}_{\Omega_t} U_t \tag{7}$$

for a càdlàg event-process $(\Omega_t)_{t\in[0,T]}$, rather than

$$\bar{U}_t = \mathbb{1}_{\{t<\bar{\tau}\}} \infty + \mathbb{1}_{\{t\geq\bar{\tau}\}} U_t \tag{8}$$

for a stopping time $\bar{\tau}$ as before. So the cases considered in subsections 2.2 and 2.3 correspond to upper barriers of the form (8) with

$$\bar{\tau} = \inf\{t \in \mathbb{R}_+ \,;\, S_t \geq \bar{S}\} \wedge T, \text{ resp. } \bar{\tau} = \inf\{t \in \mathbb{R}_+ \,;\, N_t \geq 20\} \wedge T \,, \tag{9}$$

or, equivalently, to upper barriers of the form (7) with $\Omega_t = \{t \geq \bar{\tau}\}$ for $\bar{\tau}$ as in (9), whereas the related real cases of interest would rather correspond to upper barriers of the form (7) with

$$\widetilde{\Omega}_t = \{S_t \geq \bar{S}\}, \text{ resp. } \widetilde{\Omega}_t = \{N_t \geq 20\}.$$

This seems to pose no real problem in the abstract set-up of [3].

Specifying further the results to the Markovian set-up of the Black–Scholes model and assuming $\Omega_t = \Omega(t, S_t, \mathcal{S}_t)$ for a (finite-dimensional) Markovian factor process $(S, \mathcal{S})$, then the formally related algorithm writes, intuitively: $\Pi_n^j = \phi(S_n^j)$, and for $i = n-1, \ldots, 0$, for $j = 1 \ldots m$:

$$\Pi_i^j = \min\left(\bar{U}_i\left(S_i^j, \mathcal{S}_i^j\right) \,,\, \max\left(L_i\left(S_i^j\right), e^{-rh}\mathbb{E}_{i,j}\Pi_{i+1}\right)\right) \tag{10}$$

where the min plays no role on $\Omega_i^c\,(S, \mathcal{S})$, since $\bar{U}_i\,(S, \mathcal{S})$ is then equal to $+\infty$, and where the conditional expectations can (in theory..) be computed by regression. So the situation seems (slightly) simpler than previously, inasmuch as the pricing function may be computed in one row, without having to compute the no call protection pricing function in a first step as before.

But the problem in the Markovian set-up (even in the simple Black–Scholes model and in the simplest case $\widetilde{\Omega}_t = \{S_t \geq \bar{S}\}$) is that the related Markovian reflected BSDE error estimates seem out of reach, due to the fact that they involve continuous dependence of the whole sequence of successive times of exit from and entry to a domain by $(S, \mathcal{S})$, instead of continuous dependence of the *first exit time* from a domain by $(S, \mathcal{S})$ in the previous cases.

I spent some time thinking about this last aspect of the problem and at this stage I don't know what to do about it.

Part II

CDO tranches in a Local Intensity Model

We now consider the problem of pricing CDO tranches in the context of multi-name credit risk (see, e.g., [4]). For simplicity we only consider *protection branches of equity tranches* with (stylized) payoff

$$\phi(N_T) = \frac{(1-R)N_T}{n} \wedge k$$

at the maturity time T (we take interest rates as 0, for simplicity). Here N_t represents the number of firms defaulted at time t in a reference pool of n names (e.g., $n = 125$ on the DJ iTraxx market), R is a constant recovery (typically $R = 40\%$), and the "strike" (detachment point, in the context of CDOs) k belongs to $[0, 1]$.

We model the cumulative loss N as a (univariate) point process with (risk-neutral) *local intensity* $\lambda(t, N_t)$ (see, for instance, [7]), with, in order to ensure that N is stopped at level n (since there are n names in the pool), $\lambda(t, i) = 0$ for $i \geq n$. So $N_0 = 0$ and N jumps by one at some (increasing) $(0, +\infty)$-valued random times $\widetilde{t}_i$. (depending on the trajectory of N both in number and in location), Conditionally on the information available at time t, the probability of a jump in the next time interval $(t, t + dt)$ is $\lambda(t, N_t)dt$. Under standard hypotheses that we assume henceforth on the intensity function λ, N is a well-defined Markov process. Since we are only interested in the time horizon $[0, T]$, it is convenient to introduce, for $i = 1, \ldots, n$, $t_i = \widetilde{t}_i \wedge T$ (with the convention that $\widetilde{t}_i = +\infty$ and so $t_i = T$ in case the number of jumps is less than i on a given trajectory). Finally we set $t_0 = 0$.

As opposed to the situation of part I, there are no control-theoretical features involved here (our stylized CDO tranche is an European option with payoff function $\phi(i)$), so the related price process writes simply, for $t \in [0, T]$:

$$\Pi_t = \mathbb{E}_t \phi(N_T) = u(t, N_t) \ , \tag{11}$$

where $u(t, i)$ or $u_i(t)$ for $(t, i) \in [0, T] \times \{0, 1, \ldots, n\}$ is the *pricing function* (system of time-functionals u_i).

3 Pricing

Our aim in this section is to determine the pricing function u numerically by simulation. We refer the reader to the introduction (see also section 4) regarding our motivation for determining numerically the whole pricing function u rather than simply $u(0, 0) = \Pi_0 = \mathbb{E}\phi(N_T)$ (which could of course be done very simply and at a low computational cost by a standard "static" European Monte Carlo procedure).

Note that as expected given the nature of the model at hand, it is possible to devise a simple tree deterministic approximation scheme for the pricing function u (see [7]). However our intuition is that in order not to lose any information in this model, where the information here is represented by the exact default times on every trajectory, a simulation-based approximation method might be a better (or at least, an interesting) alternative to the previous tree method, in which jump times are discretized on a fixed time-grid (the same for any level i of the loss process N).

In view of (11) we have the following identity, for every $i = 0, \ldots, n-1$:

$$u(t_i, N_{t_i}) = \mathbb{E}_i u(t_{i+1}, N_{t_{i+1}}) \ , \text{ a.s.} \tag{12}$$

The above considerations motivate the following algorithm, based on (12), for computing the pricing function $u_i(t)$ by a backward recursion on the *loss levels* (rather than on time levels as in the tree method).

First observe that the time-functional u_n is known and constant, equal to $\phi(n)$. This is because, as the loss process N is stopped at the level n, we have that $N_T = n$ on $\{N_t = n\}$, almost surely for any $t \in [0, T]$. One then simulates $m = 10^5$ (say) trajectories of N over $[0, T]$ and the related payoff values $\phi(N_T)$, recorded as $(t_i^j)_{0 \leq i \leq n, 1 \leq j \leq m}$ and $(\phi^j)_{1 \leq j \leq m}$, respectively. Note that this can be done exactly by standard simulation methods (see, e.g., section 7 of [4] for a presentation in the context of a much more general model of credit risk), without any discretisation error (whereas in the tree method default times are discretised on a fixed time-grid). The main loop in the algorithm then goes as follows, for i decreasing from $n-1$ to 1 :

• For $j = 1 \ldots m$ set $\Pi_{i+1}^j = u_{i+1}(t_{i+1}^j)$ (where u_{i+1} is already known from step $i+1$ in the algorithm) if $t_{i+1}^j < T$ and $\Pi_{i+1}^j = \phi^j$ if $t_{i+1}^j = T$;

• Compute the time-functional u_i by *non-parametric regression* (cf. section 1) of the Π_{i+1}^j's wrt to the t_i^j's, where the non-parametric regression is performed over the *subset Ω_i of the indices j such that $t_i^j < T$* (so $u(t_i, N_{t_i}) = u_i(t_i)$ on Ω_i, and u_i can indeed be computed by regression of the Π_{i+1}^j's wrt to the t_i^j's over Ω_i).

Finally define the Π_1^j's as above with $i = 1$ therein and set Π_0^j equal, for any j, to the arithmetic average $\widehat{\Pi}_0$ of the Π_1^j's (no regression is possible nor needed here).

In practice at the first steps of the algorithm the set Ω_i will typically be very small, even possibly empty, so that the regression in the second point will be numerically ill-posed and work badly in practice. For such indices i the second point above should be replaced by:

• Set $u_i = \phi(i)$ (which is a known constant).

Note that, replaced in the context of deterministic numerical schemes, this amendment to the algorithm can be understood as a rather natural *boundary condition*.

4 Hedging

In our formalism the (stylized) *credit index* corresponds to the tranche with $k = 100\%$. In practice we are interested in hedging a given tranche (with $k < 100\%$, referred to as *the tranche* in what follows) with the index. Let u and v denote the pricing functions of the tranche and of the index, numerically determined by simulation using the method of section 3, run on a *common set of simulated trajectories* $(t_i^j)_{0 \leq i \leq n, 1 \leq j \leq m}$ of N. Let $(\Pi_i^j)_{0 \leq i \leq n, 1 \leq j \leq m}$ be defined as in section 3 (relative to the tranche to be hedged), and let $(\Theta_i^j)_{0 \leq i \leq n, 1 \leq j \leq m}$ denote the analog quantity relative to the index.

The related *delta-function* of the tranche wrt to the index, $\delta(t, i) = \delta_i(t)$ for $(t, i) \in [0, T] \times \{0, 1, \ldots, n-1\}$, can then be obtained by computing, for every $i \in \{0, \ldots, n-1\}$, $\delta_i(t)$ as $\frac{\delta_i^u(t)}{\delta_i^v(t)}$, where:

• $\delta_i^u(t)$ denotes the non-parametric regression of the $(\Pi_{i+1}^j - \Pi_i^j)(\Theta_{i+1}^j - \Theta_i^j)$'s wrt to the t_i^j's, and

• $\delta_i^v(t)$ denotes the non-parametric regression of the $(\Theta_{i+1}^j - \Theta_i^j)^2$'s wrt to the t_i^j's,

where again the regressions are performed over Ω_i. At least, this procedure defines $\delta_i(t)$ for i not too large—such that Ω_i is large enough in order for the regressions to be numerically well posed, say for $0 \leq i \leq \bar{i} (\leq n-1)$.

Note that the computations of u, v and δ can (and should better) be made within the same loop, where at step i in the algorithm one first computes u_i and v_i, which are then used for computing δ_i.

It is then interesting to assess numerically the performance of the discrete-time delta-hedging strategy defined by $\zeta_t = \delta_i(t_i)$ on $(t_i, t_{i+1}]$. Denoting

$$\nu^j = \max\{i \in \{0, \ldots, \bar{i}\}\,;\, t_i^j < T\}\ ,$$

the related *tracking error* (or *profit-and-loss*) e^j at time t_{ν^j+1} writes (from the point of view of the buyer of the tranche):

$$e^j = \sum_{0 \leq i \leq \nu^j} (\Pi_{i+1}^j - \Pi_i^j) - \delta_i(t_i^j)(\Theta_{i+1}^j - \Theta_i^j)\ .$$

It is expected that this tracking error should be small, given that our model for N is complete (see [7]).

Note finally that all these computations can be made using a set of trajectories $(t_i^j)_{0 \leq i \leq n, 1 \leq j \leq m}$ simulated in an arbitrary model of multi-name credit risk, like for instance the general model of [4], which includes the model N above as a (very) special case. In this extension of the method the pricing function u should probably be interpreted as some kind of 'projection' of a complex model, used as a proxy for the (risk-neutral) real market, on the class of local intensity models $\lambda(t, N_t)$. Since the knowledge of the pricing function u allows one to deduce λ in a local intensity model, our methodology allows one to 'calibrate' a 'tangent' local intensity model to a 'big' market model. A proxy of the 'calibration error' is given by the difference $\widehat{\Pi}_0 - \Pi_0$ where $\widehat{\Pi}_0$ is the price in the 'tangent' local intensity model thus calibrated (arithmetic average of the Π_1^j, see the final step of the algorithm in section 3) and Π_0 is the 'true' option price in the big model, or an accurate standard Monte Carlo approximation computed on the m trajectories $(t_i^j)_{0 \leq i \leq n, 1 \leq j \leq m}$.

References

[1] Boughanim N., Ouzou A.: IHG, *ADI Quantitative Research Internal Report*, 2006.

[2] Bouchard B., Ekeland I. and Touzi N. The Malliavin approach to Monte Carlo approximations to conditional expectations, *Preprint*, 2002.

[3] Bielecki, T.R., Crépey, S., Jeanblanc, M. and Rutkowski, M. Arbitrage pricing of defaultable game options with applications to convertible bonds, *Quantitative Finance*, Forthcoming.

[4] Bielecki, T.R., Crépey, S., Jeanblanc, M. and Rutkowski, M., Valuation of basket credit derivatives in the credit migrations environment. *Handbook of Financial Engineering*, J. Birge and V. Linetsky eds., Elsevier, 2006.

[5] M.Broadie P.Glassermann. Pricing American-style securities using simulation, *J. of Economic Dynamics and Control*, 21, 1323-1352, 1997.

[6] M.Broadie P.Glassermann. A Stochastic Mesh method for Pricing High-Dimensional American Options, *Journal of Computational Science* 7, pp. 35-72, 2004.

[7] J.-P. Laurent, A. Cousin and J.-D. Fermanian, Hedging default risks of CDOs in Markovian contagion models, *Preprint*.

[8] Lions, P.-L., Regnier, H.: Calcul du prix et des sensibilités d'une option américaine par une méthode de Monte Carlo, *Working Paper*, 2001.

[9] F.A.Longstaff E.S.Schwartz. Valuing American Options by simulations:A Simple Least-Squares Approach, *Review of Financial Studies*, Volume 14, Number 1, 2001, pp. 113-147(35).

[10] G.Pages V.Bally. A quantization method for the discretization of BSDE's and Reflected BSDE's, *Technical report 628, université Paris 6*, 2000.

[11] L.C.G. Rogers. Monte Carlo Valuation of American Options, *Mathematical Finance*, 32(4):1077–1088, 1995. Vol. 12, pp. 271-286, 2002.

[12] J.N.Tsitsiklis B.Van Roy. Regression methods for Pricing complex American-style Options, *Working Paper MIT*, 1-22, 2000.

[13] J.N.Tsitsiklis B.Van Roy. Optimal Stopping of Markov Processes: Hilbert Spaces theory, Approximations Algorithms and an application to pricing high-dimensional financial derivatives, *IEEE Transactions on Automatic Control*, 44, 10, 1840-1851, 1999.