Projet C2 - Barrier Option in Libor Market Model

Yayrale AKIBODE

Professeur: V. LEMAIRE

Master 2 Probabilité et Finance, Sorbonne Université

10 mai 2024

Implémentation de l'algorithme par marche aléatoire similaire à la méthode du pont Brownien pour le pricing des options barrières dans un modèle de diffusion , Pricing de barrière Cap et de Swaption et Comparaison à une méthode de Monte Carlo classique

---

Introduction

L'algorithme de marche aléatoire permet de résoudre le problème de Dirichlet [a] sur un domaine $\mathcal{D}$. Sur le domaine l'EDS caractéristique est approximée par un schéma d'Euler faible. Tandis que proche de la fontière, soit la solution est arrêtée sur les bords, soit elle retourne sur le domaine avec une probabilité modélisée par une interpolation linéaire. Cet algorithme est similaire à la méthode du pont de diffusion. Cependant il a l'avantage d'être flexible. Ainsi il peut être appliqué à plusieurs structures notamment les contrats impliquant plusieurs sous-jacents corrélés. Dans notre travail, nous appliquerons cet algorithme au pricing d'options barrière. A titre d'illustration, nous pricerons un cap et un swaption avec barrière. Ainsi une étude numérique nous permettra de discuter la robustesse de cet algorithme suite à des simulations issues d'une implémentaion.

[a] Résolution d'une EDP sur un domaine en spécifiant les valeurs de la solution sur la frontière.

1 Présentation mathématique

1.1 Algorithme de marche aléatoire pour les diffusions avec temps d'arrêt

1.1.1 Définition du problème

Nous travaillons sur l'espace de probabilité $(\Omega, \mathcal{F}, \mathbb{P})$. Nous considérerons le temps t $(0 \leq t \leq T)$ et un domaine $G \subset \mathbb{R}^d$ tq $Q = [t_0, T[\times G$ est un cylindre dans $\mathbb{R}^{d+1}$. $\Gamma = \overline{Q} \setminus Q$ est la fontière du domaine Q. Le processus $(w_t, \mathcal{F}_t)$ est un processus de Wiener standard de dimension r. Le prix $u(t,x)$ solution de l'EDP s'écrit comme ci-dessous.

$$u(t,x) = \mathbb{E}\left[\varphi\left(\tau, X_{t,x}(\tau)\right) Y_{t,x,1}(\tau) + Z_{t,x,1,0}(\tau)\right] \tag{1}$$

où $X_{t,x}(s), Y_{t,x,y}(s), Z_{t,x,y,z}(s), s \geq t$, sont des solutions de Cauchy du système suivant :

$$\begin{aligned}
dX &= (b(s,X) - \sigma(s,X)\mu(s,X))ds + \sigma(s,X)dw(s), \quad X(t) = x,\\
dY &= c(s,X)Yds + \mu^\top(s,X)Ydw(s), \quad Y(t) = y,\\
dZ &= g(s,X)Yds + F^\top(s,X)Ydw(s), \quad Z(t) = z,
\end{aligned}$$

$(t,x) \in Q$, $\tau = \tau_{t,x}$ est le premier temps de sortie de $(s, X_{t,x}(s))$ de Γ.
Dans les EDS, $b(s,x)$ est un vecteur de dimension d, $\sigma(s,x)$ est une matrice de dimension $d \times r$, $\mu(s,x)$ et $F(s,x)$ sont des vecteurs de dimension r, $Y(s)$, $Z(s)$, $c(s,X)$ et $g(s,X)$ sont des scalaires. On admettra que ces coefficients et la fonction $\varphi(t,x)$ définie sur Γ et ∂G satisfont les conditions régulières .
Nous pourrons remarquer que $u(t,x)$ ne dépend pas de $\mu(s,x)$ et de $F(s,x)$. Ainsi cette flexibilité nous permettra de réduire la variance lors du calcul de l'espérance avec la méthode de Monte Carlo. Par exemple si on a :

$$\sum_{i=1}^{d} \sigma^{ij} \frac{\partial u}{\partial x^i} + u\mu^j + F^j = 0, j = 1, \ldots, r, \tag{2}$$

alors $\mathrm{Var}\left[\varphi\left(\tau, X_{t,x}(\tau)\right) Y_{t,x,1}(\tau) + Z_{t,x,1,0}(\tau)\right] = 0$ et $\varphi\left(\tau, X_{t,x}(\tau)\right) Y_{t,x,1}(\tau) + Z_{t,x,1,0}(\tau) \equiv u(t,x)$. Cependant, le choix de μ et F optimaux dépend de $u(t,x)$, ce qui n'est pas pratique. Donc dans la suite nous allons utiliser des proxy de $u(t,x)$ pour pouvoir déterminer les valeurs optimales.

1.1.2 Construction de la solution

► **Solution sur le domaine** Q : on approxime le système d'EDS par un schéma d'Euler faible.

$$\begin{aligned}
X_{t,x}(t+h) &\approx X = x + h(b(t,x) - \sigma(t,x)\mu(t,x)) + h^{1/2}\sigma(t,x)\xi,\\
Y_{t,x,y}(t+h) &\approx Y = y + hc(t,x)y + h^{1/2}\mu^\top(t,x)y\xi,\\
Z_{t,x,y,z}(t+h) &\approx Z = z + hg(t,x)y + h^{1/2}F^\top(t,x)y\xi,
\end{aligned}$$

$h > 0$ est le pas de discrétisation, $\xi = (\xi^1, \ldots, \xi^r)^\top$, $\xi^i, i = 1, \ldots, r$, sont des variables aléatoires i.i.d prenant les valeurs ± 1 avec une probabilité $p = \frac{1}{2}$. Cela implique que X prend 2^r valeurs différentes.

► **Solution sur les bords de** Q : commençons par construire l'ensemble $S_{t,h}$ comme zone proche de la fontière de G donc $S_{t,h} \subset \overline{G}$. $S_{t,h}$ est l'ensemble des $x \in G$ tq au moins une des 2^r de X n'appartienne pas à $\overline{G}$. $\overline{Q}$ étant compact, $\exists \lambda > 0$ tq la distance de x à ∂G est supérieure ou égale à $\lambda\sqrt{h}$ alors $x \notin S_{t,h}$. En d'autres termes, si $x \in \overline{G} \setminus S_{t,h}$ alors $X \in \overline{G}$.

Pour construire la solution sur les bords on a deux algorithmes de convergence différente :

Algorithme 1 de convergence $\mathcal{O}(h)$:

Soient $x \in S_{t,h}$ et x^π la projection de x sur ∂G (x^π est unique si on admet que h est assez petit et ∂G est régulière), $n(x^\pi)$ le vecteur unitaire directeur de x à x^π, $p = p_{x,h}$, $q = q_{x,h} = 1 - p_{x,h}$ tq :

$$p_{x,h} = \frac{h^{1/2}\lambda}{\left|x + h^{1/2}\lambda n\left(x^\pi\right) - x^\pi\right|},$$

$$X^\pi_{x,h} = \begin{cases} x^\pi & \text{avec probabilité } p \\ x + h^{1/2}\lambda n\left(x^\pi\right) & \text{avec probabilité } q \end{cases}$$

La seconde valeur $x + h^{1/2}\lambda n\left(x^\pi\right) \notin S_{t,h}$. D'autre part p est toujours supérieure à $\frac{1}{2}$.

Cela peut s'expliquer par l'approximation d'une fonction v de C^2 sur $\overline{G}$ par une interpolation linéaire.

$$v(x) = \mathbb{E}v\left(X^\pi_{x,h}\right) + O(h) = pv\left(x^\pi\right) + qv\left(x + h^{1/2}\lambda n\left(x^\pi\right)\right) + O(h).$$

Algorithme 2 de convergence $\mathcal{O}(\sqrt{h})$:

Il s'agit d'une version simplifiée de l'algorithme 1. Si $x \in S_{t,h}$ alors $X^\pi_{x,h}$ prend uniquement la valeur x^π (i.e $X^\pi_{x,h} = x^\pi$ avec $p = 1$).

1.2 Modèle LMM (Libor Market Model)

L'idée de ce modèle est de modéliser directement le taux forward qui est le sous-jacent des caps et des floors. Le LIBOR [1] est le taux forward de référence. Comme notation, nous allons considérer des maturités équidistantes $T_0 < \cdots < T_N = T, \quad T_i = i\delta, \quad i = 0, \ldots, N$ avec $\delta = \frac{T - T_0}{N}$.
Le taux LIBOR $L^i(t) = L\left(t, T_i, T_{i+1}\right) = \frac{1}{\delta}\left(\frac{P(t,T_i)}{P(t,T_{i+1})} - 1\right)$ est le taux forward à t ($t_0 \leq t \leq t \wedge T_i$) qui emprunte 1 à T_i et paye $1 + \delta L(t, T_i)$ à T_{i+1}.

[1] Suite à des manipulations le taux LIBOR a été remplacé par des taux journaliers. La logique des modèles restent la même. Cependant les taux sont maintenant postcomptés au lieu d'être précomptés.

La dynamique des LIBOR sous la probabilité forward est défini comme ci-dessous.

$$\frac{dL^i(t)}{L^i(t)} = \begin{cases} \sigma_i(t)\sum_{j=k+1}^{i} \frac{\delta L^j(t)}{1+\delta L^j(t)}\rho_{i,j}\sigma_j(t)dt + \sigma_i(t)dW_i^{T_{k+1}}(t), i > k, t \leq T_k, \\ \sigma_i(t)dW_i^{T_{k+1}}(t), i = k, t \leq T_i, \\ -\sigma_i(t)\sum_{j=i+1}^{k} \frac{\delta L^j(t)}{1+\delta L^j(t)}\rho_{i,j}\sigma_j(t)dt + \sigma_i(t)dW_i^{T_{k+1}}(t), i < k, t \leq T_i, \end{cases} \tag{3}$$

où $W^{T_{k+1}} = \left(W_0^{T_{k+1}}, \ldots, W_{N-1}^{T_{k+1}}\right)^\top$ est un mouvement brownien standard de dimension N sous la probabilité $\mathrm{Q}^{T_{k+1}}$ avec comme numéraire $P(t, T_{k+1})$. La fonction de corrélation instantanée que nous utiliserons dans la suite est donnée par :

$$\mathbb{E}\left[W_i^{T_{k+1}}(t)W_j^{T_{k+1}}(t)\right] = \rho_{i,j} = \exp\left(-\beta\left|T_i - T_j\right|\right), \quad i, j = 0, \ldots, N-1$$

Les $\sigma_i(t), i = 0, \ldots, N-1$, sont les volatilités instantanées. On admettra qu'elles sont déterministes.

1.3 Application à un cap

Comme application, nous allons pricer un cap avec barrière sur un seul sous-jacent avec l'algorithme de marche aléatoire. En effet, un cap est une option sur taux qui permet de se prévenir contre la hausse des taux. Il est généralement valorisé comme la somme des parties positives des caplets. Un caplet est une option sur taux sur un intervalle unitaire $[T_i, T_{i+1}[$. Donc pricer un cap peut revenir tout simplement à pricer un caplet.
Un caplet avec barrière ou "knock-out caplet" est un caplet qui paye le payoff d'un caplet simple tant qu'une certaine barrière H n'est pas franchie. Lorsque la barrière est franchie le produit n'existe plus donc sa valeur est nulle. La valeur d'un "knock-out caplet" est donnée par l'expression suivante :

$$V_{\text{caplet}}(t) = \delta P(t, T_{i+1})\,\mathbb{E}^{\mathrm{Q}^{T_{i+1}}}\left[\left(L^i(T_i) - K\right)_+ \mathbb{1}_{(\theta > T_i)} \mid \mathcal{F}_t\right]$$

Le terme $\delta P(t, T_{i+1})$ vient du changement de numéraire où $P(t, T_{i+1})$ est le prix zéro-coupon à t de maturité T_{i+1}. Pour ajouter la barrière, on ajoute une indicatrice indiquant que la valeur du caplet est nulle lorsqu'on franchit la barrière. De fait, θ est le premier temps de $L^i(s), s \geq t$, de $G = (0, H)$. Si τ est le premier temps de sortie de la trajectoire $(s, L^i(s))$ de $Q = [t, T_i) \times (0, H)$ alors $\tau = \theta \wedge T_i$. Le caplet portant sur un intervalle unitaire, la corrélation instantanée est nulle donc la dynamique du LIBOR s'écrit :

$$\frac{dL^i(s)}{L^i(s)} = \sigma_i(s)dW_i^{T_{i+1}}(s), s \leq T_i$$

Avec cette dynamique, le caplet admet une formule fermée. Cette formule fermée va nous permettre d'étudier la véracité de notre algorithme. Mais aussi elle pourra être utile pour faire une réduction de variance.

$$\begin{aligned} V_{\text{caplet}}(t) =& V_{\text{caplet}}\left(t, L^i(t)\right) \\ =& \delta P(t, T_{i+1})\left\{L^i(t)\left[\Phi\left(\delta_+\left(L^i(t)/K, v_i\right)\right) - \Phi\left(\delta_+\left(L^i(t)/H, v_i\right)\right)\right]\right. \\ & - K\left[\Phi\left(\delta_-\left(L^i(t)/K, v_i\right)\right) - \Phi\left(\delta_-\left(L^i(t)/H, v_i\right)\right)\right] \\ & - H\left[\Phi\left(\delta_+\left(H^2/\left(KL^i(t)\right), v_i\right)\right) - \Phi\left(\delta_+\left(H/L^i(t), v_i\right)\right)\right] \\ & \left. + KL^i(t)\left[\Phi\left(\delta_-\left(H^2/\left(KL^i(t)\right), v_i\right)\right) - \Phi\left(\delta_-\left(H/L^i(t), v_i\right)\right)\right]/H\right\}, \end{aligned} \tag{4.3}$$

Φ est la fonction de répartition de la loi normale, $\delta_+(x,v) = \frac{ln(x)+\frac{v^2}{2}}{v}$ et $\delta_-(x,v) = \frac{ln(x)-\frac{v^2}{2}}{v}$ avec $v_i^2 = \int_t^{T_i}(\sigma_i(s))^2 ds$.

Dans la suite, au lieu de calculer $V_{\text{caplet}}(t)$, nous calculerons :

$$\tilde{V}_{\text{caplet}}(t) = \frac{V_{\text{caplet}}(t)}{\delta P(t, T_{i+1})}$$

1.4 Application à un swaption

L'application à un swaption s'apparente à celle du cap. Cependant la fonction de corrélation instantanée de la dynamique des LIBOR n'est plus nulle car le taux swap dépend à chaque instant de tous les LIBOR futurs.
Nous allons considérer un swap receveur "knock-out". L'option sur le swap a pour maturité T_0. Ainsi on a :

$$V_{\text{swaption}}(0) = P(0,T_0)\, E^{\mathrm{Q}^{T_0}}\left[\delta\left(R_{\text{swap}}(T_0) - K\right)_+ \sum_{j=1}^{N} P(T_0,T_j)\,\chi(\theta > T_0)\right]$$

Le sous-jacent ici est le taux swap [1]. θ est le premier temps de sortie de $R_{\text{swap}}(s), s \geq 0$ de $(0, R_{up})$. Le taux swap et les prix zéro-coupons $P(T_0,T_j)$ s'écrivent en fonction des taux LIBOR.

$$R_{\text{swap}}(s) = \frac{1 - 1/\prod_{j=0}^{N-1}(1+\delta L^j(s))}{\delta \sum_{i=0}^{N-1} 1/\prod_{j=0}^{i}(1+\delta L^j(s))}$$

La loi jointe des taux LIBOR est déterminée par la dynamique ci-dessous.

$$\frac{dL^i(t)}{L^i(t)} = \sigma_i(t)\sum_{j=0}^{i}\frac{\delta L^j(t)}{1+\delta L^j(t)}\rho_{i,j}\sigma_j(t)dt + \sigma_i(t)dW_i^{T_0}(t), i = 0,\ldots,N-1$$

De même que le "knock-out caplet", le swaption admet une formule fermée proxy qui va nous permettre d'avoir une idée sur l'exactitude de nos simulations.

$$\begin{aligned}
V_{\text{swaption}}(0) = \delta\sum_{j=1}^{N} P(0,T_j)\Big\{ & R_{\text{swap}}(0)\left[\Phi\left(\delta_+\left(R_{\text{swap}}(0)/K, v_{R_{\text{swap}}}\right)\right) - \Phi\left(\delta_+\left(R_{\text{swap}}(0)/R_{up}, v_{R_{\text{swap}}}\right)\right)\right] \\
& - K\left[\Phi\left(\delta_-\left(R_{\text{swap}}(0)/K, v_{R_{\text{swap}}}\right)\right) - \Phi\left(\delta_-\left(R_{\text{swap}}(0)/R_{up}, v_{R_{\text{swap}}}\right)\right)\right] \\
& - R_{up}\left[\Phi\left(\delta_+\left(R_{\text{up}}^2/\left(KR_{\text{swap}}(0)\right), v_{R_{\text{swap}}}\right)\right) - \Phi\left(\delta_+\left(R_{\text{up}}/R_{\text{swap}}(0), v_{R_{\text{swap}}}\right)\right)\right] \\
& + KR_{\text{swap}}(0)\left[\Phi\left(\delta_-\left({R_{up}}^2/\left(KR_{\text{swap}}(0)\right), v_{R_{\text{swap}}}\right)\right) - \Phi\left(\delta_-\left(R_{\text{up}}/R_{\text{swap}}(0), v_{R_{\text{swap}}}\right)\right]/R_{\text{up}}\right\},
\end{aligned}$$

où $\delta_\pm$ reste les mêmes que précédemment et,

$$v_{R_{swap}}^2 = \int_0^{T_i}\left(\sigma_{R_{swap}}(s)\right)^2 ds$$

[1] C'est le taux fixe qui rend nulle la valeur du swap à T_0

avec $\sigma_{R_{swap}}(s)$ la volatilité instantanée de la dynamique log-normale du taux swap. Comme approximation , nous pouvons utiliser la formule suivante :

$$v_{R_{swap}}^{LMM} = \sum_{i,j=0}^{N-1} \frac{\omega_i(0)\omega_j(0)L^i(0)L^j(0)\rho_{ij}}{\left(R_{\text{swap}}(0)\right)^2} \int_0^{T_0} \sigma_i(s)\sigma_j(s)ds$$

où

$$\omega_i(0) = \frac{1/\prod_{j=0}^{i-1}(1+\delta L^j(0))}{\sum_{k=0}^{N-1} 1/\prod_{j=0}^{k}(1+\delta L^j(0))}$$

2 Implémentation du modèle

2.1 Application à un caplet avec barrière

Pour garantir la positivité des taux, nous allons simuler lnL_k^i au lieu de L_k^i ainsi lorsqu'on passe à l'exponentielle on est sûr d'avoir une valeur positive.

► **Solution sur le domaine** $Q = [t, T_i[\times]0, H[$: le schéma d'Euler faible du système d'EDS s'écrit :

$$\begin{aligned} lnL_{k+1}^i &= lnL_k^i - \frac{1}{2}(\sigma_i(t_k))^2 h + \sigma_i(t_k)\sqrt{h}\xi_{k+1} \\ Y_{k+1} &= 1 \\ Z_{k+1} &= Z_k + F(s, L_k^i)\sqrt{h}\xi_{k+1}, \end{aligned}$$

Soit $M = \frac{T_i}{h}$ tel que M soit un entier $\xi_i, i = 1, \ldots, M$, sont des variables aléatoires i.i.d prenant les valeurs ± 1 avec une probabilité $p = \frac{1}{2}$.

► **Solution sur les bords de** Q :
Commençons par construire la zone frontière $S_{t,h}$:

$$S_{t,h} = \{L_k^i : lnL_{k+1}^i \geq lnH - \lambda_k\sqrt{h}\}$$

Nous prendrons $\lambda_k\sqrt{h} = \sigma_i(t_k)\sqrt{h}$ afin d'avoir une zone frontière assez large.

Algorithme 1

$$p = \frac{\lambda_k\sqrt{h}}{\left|lnH - lnL_k^i + \lambda_k\sqrt{h}\right|}.$$

Donc si $L_k^i \in S_{t,h}$ alors :

$$lnL_{k+1}^i = \begin{cases} lnH & \text{avec probabilité } p \\ lnL_k^i - \lambda_k\sqrt{h} - \frac{1}{2}(\sigma_i(t_k))^2 h + \sigma_i(t_k)\sqrt{h}\xi_{k+1} & \text{avec probabilité } q \end{cases}$$

Si $lnL^i_{k+1} = lnH$ alors la trajectoire $(\theta = k+1, lnL^i_\theta, Z_\theta)$ est la trajectoire finale. Sinon on continue jusqu'à M et $(\theta = M, lnL^i_\theta, Z_\theta)$ sera la trajectoire finale.

Algorithme 2

Si $lnL^i_k \in S_{t,h}$ alors $lnL^i_{k+1} = lnH$ et la trajectoire $(\theta = k+1, lnL^i_\theta, Z_\theta)$ est la trajectoire finale. Sinon on continue jusqu'à M et $(\theta = M, lnL^i_\theta, Z_\theta)$ sera la trajectoire finale.

Le prix du caplet s'écrit finalement :

$$\tilde{V}_{\text{caplet}}(t) = \mathbb{E}^{\mathbb{Q}^{T_{i+1}}}\left[\left(exp(lnL^i_\theta) - K\right)_+ \mathbb{1}_{(\theta=M)} + Z_\theta\right]$$

► **Réduction de variance**

La dernière question qui se pose est celle du choix de F. Pour cela nous allons juste exploiter l'expression (2). Ainsi :

$$\begin{aligned} F(s, L^i) &= -\sigma_i(s)\frac{\partial}{\partial L^i}\tilde{V}_{\text{caplet}}(s, L^i(s)) \\ &= \left[\Phi\left(\delta_+\left(L^i(t)/K, v_i\right)\right) - \Phi\left(\delta_+\left(L^i(t)/H, v_i\right)\right)\right] \\ &+ K\left[\Phi\left(\delta_-\left(H^2/\left(KL^i(t)\right), v_i\right)\right) - \Phi\left(\delta_-\left(H/L^i(t), v_i\right)\right)\right]/H \end{aligned}$$

Nous allons combiner cette méthode avec une méthode antithétique afin d'avoir une meilleure réduction de variance.
Soit :

$$\bar{X} = \frac{X + X'}{2}$$

tq X et X' soient de même variance. La variance de cette estimation est donnée par :

$$\text{Var}(\bar{X}) = \frac{\text{Var}(X) + \text{Cov}(X, X')}{2}$$

Donc il suffit que $\text{Cov}(X, X') < 0$ pour que $\text{Var}(\bar{X}) < \frac{\text{Var}(X)}{2}$. De façon pratique nous prendrons $X' = -X$.

2.2 Algorithme de l'application à un swaption avec barrière

Nous calculerons la somme suivante comme ci-dessous.

$$\sum_{j=1}^{N} P(T_0, T_j) = \sum_{i=0}^{N-1} 1/\prod_{j=0}^{i}\left(1 + \delta L^j(t)\right)$$

Et pour $P(t, T_0)$, si nous avons une courbe de taux plate, nous utiliserons la relation :

$$P(t, T_o) = \frac{1}{(1 + L(t))^{(T_0 - t)}}$$

► **Solution sur le domaine** $Q = [t, T_0[\times G$ où G est l'espace des taux LIBOR correspondant aux taux swap sur $]0, R_{up}[$: le schéma d'Euler faible de l'EDS s'écrit :

$$\begin{aligned}\ln L_{k+1}^i =& \ln L_k^i + \sigma_i(t_k) \sum_{j=0}^{i} \frac{\delta L_k^j}{1+\delta L_k^j} \rho_{ij} \sigma_j(t_k) h \\ & - \frac{1}{2}(\sigma_i(t_k))^2 h + \sigma_i(t_k) \sqrt{h} \sum_{j=0}^{N-1} U_{i,j} \xi_{j,k+1}, \\ i =& 0, \ldots, N-1,\end{aligned}$$

Soit $M = \frac{T_i}{h}$ tel que M soit un entier $\xi_i, i = 1, \ldots, M$, sont des variables aléatoires i.i.d prenant les valeurs ± 1 avec une probabilité $p = \frac{1}{2}$. U est la matrice tel que $\rho = UU^\top$.

► **Solution sur les bords de** Q :
Commençons par construire la zone frontière $S_{t,h}$. Considérons :

$$\ln \hat{L}_{k+1} = \max_i \ln L_k^i + \sigma_{\text{Max}}^2 hN - \frac{1}{2}\sigma_{\text{Max}}^2 h + \sigma_{Max}\sqrt{hN},$$

$$\begin{aligned}\tilde{R}_{\text{swap}} =& R_{\text{swap}} \left(L_k^0 \left(1 + \sigma_0(t_k)\sigma_{Max,k} h + \sigma_0(t_k)\sqrt{Nh}\right), L_k^1 (1 + 2\sigma_1(t_k)\sigma_{Max,k} h \right. \\ & \left. + \sigma_1(t_k)\sqrt{(N-1)h}\right), \ldots, L_k^{N-1}\left(1 + N\sigma_{N-1}(t_k)\sigma_{Max,k} h + \sigma_{N-1}(t_k)\sqrt{h}\right)\Big),\end{aligned}$$

avec $\sigma_{Max} = \max_{i,k} \sigma_i(t_k)$ alors :

$$S_{t,h} = \{L_k : \ln \hat{L}_{k+1} \geq \ln R_{up} \quad \text{et} \quad \tilde{R}_{\text{swap}} \geq R_{up}\}$$

Pour le swaption nous allons implémenter que l'algorithme 1.

Algorithme 1

$$p = \frac{\lambda\sqrt{h}}{|\ln L_k^\pi - \ln L_k| + \lambda\sqrt{h}},$$

$$\lambda\sqrt{h} = \sqrt{N}\left(\sigma_{\text{Max}}^2 hN - \frac{1}{2}\sigma_{Max}^2 h + \sigma_{Max}\sqrt{hN}\right);$$

Construction de $\ln L_k^\pi := \left(L_k^{\pi,0}, L_k^{\pi,1}, \ldots, L_k^{\pi,N-1}\right)^\top$ la projection de $\ln L_k$ sur la frontière en résolvant le problème de minimisation suivante :

$$\begin{cases} min \quad |\ln L_k^\pi - \ln L_k|^2 = \left(\ln L_k^{\pi,0} - \ln L_k^0\right)^2 + \cdots + \left(L_k^{\pi,N-1} - \ln L_k^{N-1}\right)^2 \\ tq \ln L_k^{\pi,0} = \ln\left(\frac{R_{up}\cdot\left(1+\sum_{i=0}^{N-2}\prod_{j=i+1}^{N-1}\left(1+\delta L_k^{\pi,j}\right)\right)+1}{\prod_{j=1}^{N-1}\left(1+\delta L_k^{\pi,j}\right)} - \frac{1}{\delta}\right) \end{cases}$$

Donc si $L_k \in S_{t,h}$ et $u \sim \mathcal{U}([0,1])$ alors :

$$lnL_{k+1} = \begin{cases} \ln L_k^{\pi} & \text{pour} \quad u < p \\ \ln L_k + \lambda\sqrt{h}\frac{\overrightarrow{\ln L_k^{\pi}} \ln L_k}{|\ln L_k^{\pi} - \ln L_k|} & \text{sinon}\,; \end{cases}$$

Si $lnL_{k+1} = lnL_k^{\pi}$ alors la trajectoire $(\theta = k+1, lnL_\theta)$ est la trajectoire finale. Sinon on continue jusqu'à M et $(\theta = M, lnL_\theta)$ sera la trajectoire finale.

► Réduction de variance

Pour la réduction de variance, nous utiliserons uniquement la méthode antithétique définie auparavant.

2.3 Choix d'implémentation informatique

Après avoir évalué les différents aspects de notre projet, nous avons pris en considération les avantages et les inconvénients de l'implémentation en Python et en C++. Voici nos conclusions :

2.3.1 Python

- **Avantages :**
  - Facilité de lecture et d'écriture du code.
  - Grande disponibilité de bibliothèques pour diverses tâches.
  - Rapidité de développement.
- **Inconvénients :**
  - Performances potentiellement plus faibles par rapport à C++.
  - Moins adapté pour les applications nécessitant un traitement intensif ou une vitesse optimale.

2.3.2 C++

- **Avantages :**
  - Performances élevées, idéal pour les applications nécessitant une exécution rapide.
  - Contrôle précis de la gestion de la mémoire.
- **Inconvénients :**
  - Syntaxe plus complexe et développement potentiellement plus lent.
  - Moins de bibliothèques disponibles par rapport à Python.

Après avoir pris en compte ces facteurs, nous avons décidé d'implémenter notre projet en **Python** en raison de sa facilité de développement et de sa grande disponibilité de bibliothèques pour notre application spécifique.

3 Etude numérique

3.1 Résultats numériques du caplet

Pour le caplet, nous allons comparer les résultats de l'algorithme de marche aléatoire 1 et 2. En effet, les résultats sont similaires. Cependant, l'algorithme 1 est coûteux en temps. Donc pour l'étude numérique, nous allons commenter les résultats de l'algorithme 2. Le nombre de simulation de Monte Carlo des résultats ci-dessous est de 10^5. Les autres paramètres ont pour valeur $\sigma = 0.25$, $H = 0.28$, $i = 9$, $K = 0.01$, $L_0^i = 0.13$ et $Z_0 = 0$.

► Algorithme 2 avec $F = 0$ et $F \neq 0$
Nous pouvons remarquer qu'il y a un grand biais lorsque $F = 0$ (i.e. que nous n'ajoutons pas le terme en Z lors du calcul de l'espérance). Cependant lorsque $F \neq 0$, le biais est réduit. Evidemment, la précision du prix diminue au fur et à mesure que le pas h devient très grand.

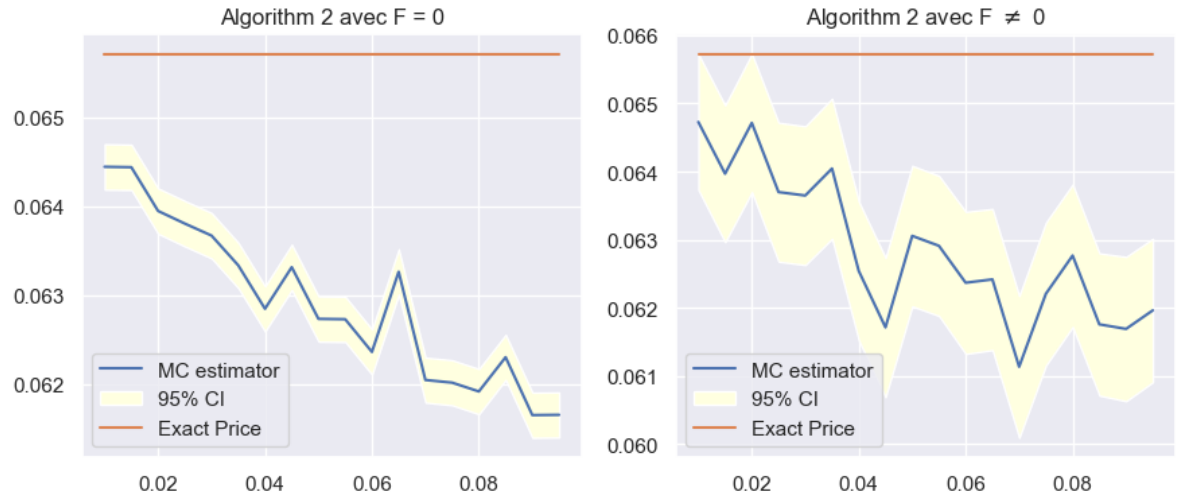

FIGURE 1 : Visualisation du biais du prix du caplet pour $F = 0$ et $F \neq 0$. Les deux figures retracent l'évolution du prix du caplet en fonction du pas h. A gauche lorsque F=0, nous pouvons remarquer que le biais entre le prix exact et l'estimation de Monte Carlo est assez grande. A droite lorsque $F \neq 0$, le biais est plus moindre.

► Extrapolation de Richardson
Cette extrapolation consiste à faire une combinaison linéaire entre un schéma d'Euler de pas $\frac{T}{n}$ et $\frac{T}{2n}$. Si $\bar{X}_T^{n,x}$ est le schéma d'Euler :

$$\mathbb{E}[f(X_T^x)] = \frac{1}{M}\sum_{k=1}^{M} 2f(\bar{X}_T^{2n,x}) - f(\bar{X}_T^{n,x})$$

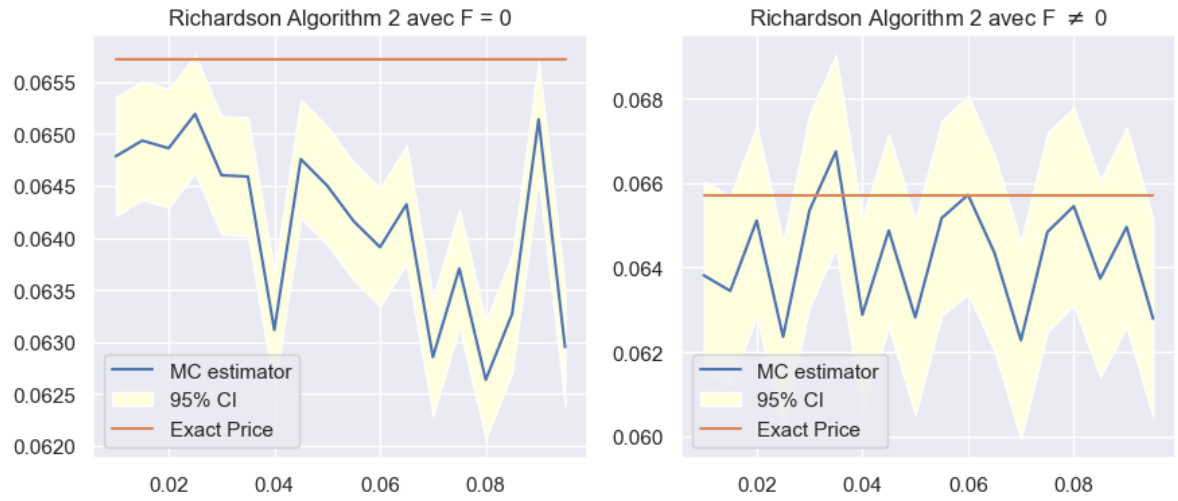

FIGURE 2 : A gauche, lorsque $F = 0$, nous pouvons remarquer une réduction générale du biais par rapport au pas h. A droite, le prix exact est maintenant compris dans l'intervalle de confiance de l'estimation de Monte Carlo.

► Comparaison avec une méthode de pont brownien

Pour la comparaison nous considérons le cas où $F = 0$. La formule suivante permet de pricer les options barrières "Up and Out". ([3])

$$\mathbb{E}\left(f(\bar{X}_T^n)\mathbf{1}_{\left\{\sup_{t\in[0,T]}\bar{X}_t^n \leq L\right\}}\right) = \mathbb{E}\left[f(\bar{X}_T^n)\mathbf{1}_{\{\max_{0\leq t\leq n}\bar{X}_{t_k}^n \leq L\}}\prod_{k=0}^{n-1}\left(1 - e^{-\frac{2n}{T}\frac{(\bar{X}_{t_k}^n - L)(\bar{X}_{t_k}^n - L)}{\sigma^2(\bar{X}_{t_k}^n, \bar{X}_{t_k}^n)}}\right)\right].$$

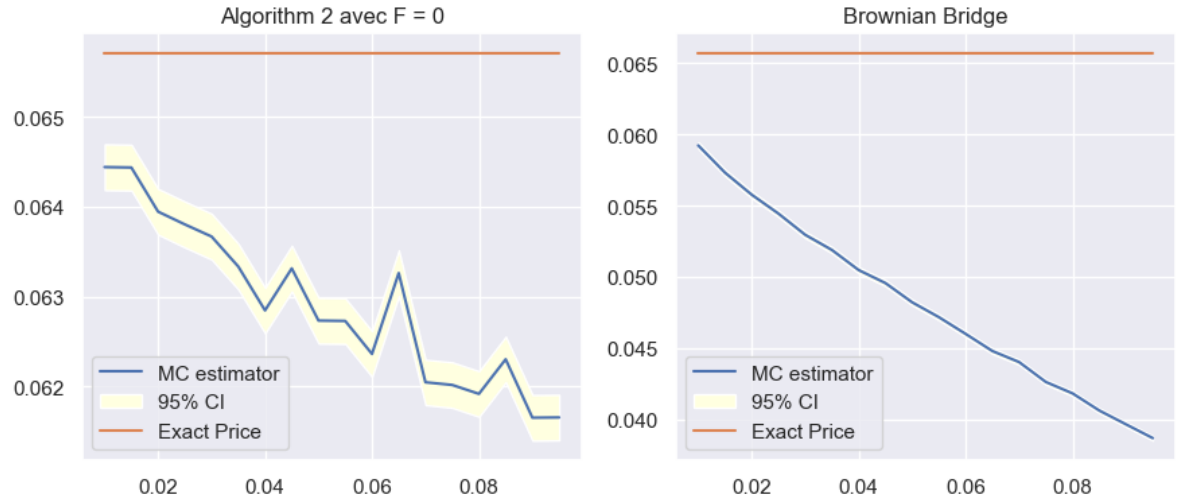

FIGURE 3 : La comparaison des deux méthodes montrent que la méthode du pont brownien sous-évalue le caplet car le biais est très grand par rapport à l'algorithme de la marche aléatoire. Cependant en terme de variance, la variance de la méthode du pont brownien est presqu'invisible dont très petite.

3.2 Résultats numériques du swaption

Les paramètres du swaption ont pour valeur : $\sigma = 0.1$, $R_{up} = 0.075$, $T_0 = 10$, $T^* = 20$, $K = 0.01$, $\beta = 0.1$, $\delta = 1$ et une courbe de taux plate de valeur 5%. Avec ces valeurs la valeur du proxy du swaption définie auparavant est de 0.15556.

► **Proposition** : Lorsqu'on est sur la fontière, afin de déterminer si on arrête l'algorithme ou si on retourne à l'intérieiur du domaine on utilise une variable de loi uniforme ($\mathcal{U}([0,1])$. Cependant, nous savons que p la probabilité calculée est toujours supérieur à $\frac{1}{2}$. Donc nous pouvons utiliser une variable de loi uniforme ($\mathcal{U}([\frac{1}{2},1])$. En effet, si nous maintenons la proposition faite par l'article, l'algorithme sera souvent arrêté et donc le prix du swaption sous évalué.

D'autre part, les simulations pour calculer le prix du swaption sont en 3 dimensions : la dimension de Monte Carlo, la dimension du temps t et la dimension des maturités T. Cela rend l'exécution de l'algorithme très lente. Pour un nombre de simulation de Monte Carlo de 10^3 , un pas $h = 0.01$ et en tenant compte de la proposition, la valeur simulée est de 0.063108. Donc l'algorithme de la marche aléatoire sous-évalue le prix du swaption.

3.3 Limites

► **Caplet**
Pour citer quelques limites de ces résultats numériques, nous pouvons parler de la variance lorsque $F \neq 0$. En effet, le choix de F est fait de façon optimal afin de réduire la variance. Mais nous pouvons remarquer un effet contraire sur les résultats numériques. Cela n'a pas forcément une explication théorique mais peut peut-être s'expliquer par la manière de coder.

► **Swaption**
L'algorithme du swaption est assez complexe car elle requiert la résolution d'un problème d'optimisation avec contrainte. Assurément, la solution optimale dépend du point initial, ce qui n'est pas toujours évident à déterminer. D'autre part, utiliser une variable uniforme pour déterminer si l'algorithme s'arrête ou pas n'est pas particulièrement objectif. Et cela mène à une sous évaluation du prix.

Conclusion

Somme toute, ce travail s'est avéré très intéressant de part les performances de l'algorithme de la marche aléatoire. De fait, nous avons pu évaluer un swaption barrière en tenant compte de la corrélation entre les taux LIBOR de différentes maturités. Ce qui n'est pas évident à implémenter avec la méthode du pont brownien. D'autre part, les résultats sur la partie du caplet se sont avérés très intéressants. Par ailleurs, nous avons pu également étudier le contraste entre le biais et la variance. Assurément, lorsque le biais est réduit, la variance augmente. Il est donc judicieux de choisir lequel négliger selon la nature du problème. En ce qui concerne, la comparaison entre l'algorithme de la marche aléatoire et la méthode du pont brownien, nous pouvons affirmer que le premier est plus pointu et donc plus précis. Cependant, la méthode classique reste plus facile à implémenter et plus intuitive.

Références

[1] M. Krivko et M.V. Tretyakov. “Application of simplest random walk algorithms for pricing barrier options” (2012).

[2] G. N. MILSTEIN et M.V. Tretyakov. “The simplest random walk for the DIRICHLET problem” (2002).

[3] G. PAGES. *Numerical Probability, An Introduction with Applications to Finance*. 2018.