Options américaines par l'approche duale

Mohammed FELLAJI, Zakaria IRAQI

15 avril 2019

Introduction

Le but principal de ce projet est d'étudier l'approche duale pour valoriser les options américaines ou bermudéennes. Il s'agit des options qui peuvent s'exercer à n'importe quel instant (dans le cas américain) ou à des instants précis (dans le cas bermudéeen).
L'approche standard pour valoriser ce type d'options consiste à chercher le temps d'arrêt optimal à travers une approximation de l'enveloppe de Snell :

$$P_0 = \sup_{\tau t.a \in [0,T]} \mathbb{E}[Z_t]$$

Tel que $(Z_t)_t$ est l'obstacle que nous allons définir dans les sections suivantes. À la différence de l'approche standard, l'approche duale consiste à chercher des martingales à la place des temps d'arrêts.

$$P_0 = \inf_{M \in \mathcal{M}_0} \mathbb{E}\left[\sup_{t \in [0,T]} (Z_t - M_t)\right]$$

Dans ce rapport, nous allons présenter la méthode proposée par Rogers dans son article [64]" Monte Carlo valuation of American options-2002." ainsi que les résultats numériques de cet approche dans le cas du put américain dont le payoff est $(K - X_t)_+$ et dans le cas d'une option sur moyenne géométriques de payoff $\left(K - \sqrt{X_t^1 X_t^2}\right)_+$. Finalement, nous allons présenter l'algorithme Multilevel introduit par D. Belomestny, J. Schoenmakers, et F. Dickmann dans leur article [13]" Multilevel dual approach for pricing americanstyle derivatives.Finance and Stochastics - 2013." .

1 Approche duale

1.1 Description générale

Nous travaillons dans un espace de probabilité filtré $(\Omega, \mathcal{F}, (\mathcal{F}_t)_t, \mathbb{P})$ dans lequel nous considérons un processus adapté à la filtration $(\mathcal{F}_t)_t$. Cela représente la valeur actualisée du payoff de l'option américaine.
Par absence d'opportunités d'arbitrage, le prix de l'option américaine à l'instant 0 est donnée par :

$$P_0 = \sup_{\tau t.a \in [0,T]} \mathbb{E}[Z_\tau]$$

Nous appelons cela le problème primal.
La détermination du temps optimal de ce problème permet d'avoir des bornes inférieures ce qui est peut être réalisée par des algorithmes d'approximation d'espérances conditionnelles comme Longstaff-Schwartz.
La méthode duale se base principalement sur le théorème suivant :

Théorème

Notons $\mathcal{M}_0$ l'espace des martingales nulles en 0. Alors l'enveloppe de Snell peut s'écrire :

$$P_0 = \inf_{M \in \mathcal{M}_0} \mathbb{E}\left[\sup_{t \in [0,T]} (Z_t - M_t)\right]$$

Démonstration :
Pour tout $M \in \mathcal{M}_0$ nous avons :

$$P_0 = \sup_{\tau t.a \in [0,T]} \mathbb{E}[Z_t - M_t] \leq \mathbb{E}\left[\sup_{t \in [0,T]} (Z_t - M_t)\right]$$

Or, nous savons que l'enveloppe de Snell est une sur-martingale. Donc d'après la décomposition de Doob-Meyer, nous avons :

$$P_t = P_0 + M_t^\star - A_t$$

Avec $(M_t^\star)_t$ une martingale nulle en 0 et $(A_t)_t$ un processus prévisible croissant partant de 0. Nous avons :

$$\inf_{M \in \mathcal{M}_0} \mathbb{E}\left[\sup_{t \in [0,T]} (Z_t - M_t)\right] \leq \mathbb{E}\left[\sup_{t \in [0,T]} (Z_t - M_t^*)\right]$$

Or nous savons que l'enveloppe de Snell domine Z donc :

$$\inf_{M \in \mathcal{M}_0} \mathbb{E}[\sup_{t \in [0,T]} (_t - M_t)] \leq \mathbb{E}[\sup_{t \in [0,T]} (P_t - M_t^*)] = \mathbb{E}[\sup_{t \in [0,T]} (P_0 - A_t^*)] = P_0$$

Donc nous avons montré que l'inf est atteint par la martingale de Doob. L'objectif dans la suite sera donc d'approcher cette martingale afin d'avoir une borne supérieure du prix la plus proche possible de la vraie valeur.

2 Méthode de Rogers

"The selection of the martingales appears to be more art than science"
Le choix de la martingale dans cette méthode se fait de manière intuitive. Par exemple, si le processus $(Z_t)_t$ est une semi-martingale, alors choisir la partie martingale peut donner de très bons résultats. Cela marche très bien dans le cas où Z n'est pas à variation finie. De plus, la partie martingale de la valeur de l'option européenne constitue un très bon candidat.
Afin d'améliorer la borne choisie par cette méthode, nous considérons le problème de minimisation suivant :

$$\inf_{\lambda} \mathbb{E}\left[\sup_{t\in[0,T]} (Z_t - \lambda M_t^*)\right] \tag{1}$$

Dans le cas où plusieurs choix de martingales sont possible, nous considérons le vecteur des martingales et le problème de minimisation précédent devient un problème de minimisation vectorielle.

$$\inf_{\lambda} \mathbb{E}\left[\sup_{t\in[0,T]} (Z_t - \lambda.M_t^*)\right]$$

2.1 Put américain

Nous considérons un actif dont la dynamique sous la probabilité risque neutre est la suivante :

$$\frac{dS_t}{S_t} = rdt + \sigma dW_t \qquad r \in \mathbb{R}_+, \sigma \in R_+^* \tag{2}$$

Le choix de la martingale, comme indiqué dans l'article sera la valeur actualisée du put européen quand l'option est dans la monnaie(c-à-d quand la valeur de l'actif devient inférieure au strike).

$$dM_t^* = \mathbb{1}_{\{t\geq t^*\}} d\widetilde{P_{BS}}(t, S_t) \tag{3}$$

avec $t^* = \inf\{t \geq 0, S_t \leq K\}$ et $\widetilde{P_{BS}}(t, S_t)$ le prix actualisé du put européen. Dans la suite nous allons prendre la forme générale de :

$$t^* = \inf\{t \geq 0,\ \phi(S_t) > 0\}.$$

Où ϕ est le payoff de l'option.
Dans ce modèle le prix du put européen est donnée par :

$$P_{BS}(t, S_t) = Ke^{-r(T-t)}\mathcal{N}(-d_0) - S_t\mathcal{N}(-d_1)$$

avec $d_1 = \frac{\ln \frac{S_t}{K} + \left(\frac{\sigma^2}{2} + r\right)(T-t)}{\sigma\sqrt{T-t}}$ et $d_0 = d_1 - \sigma\sqrt{T-t}$

Algorithme

Nous considèrons les instants $0 = t_0 < t_1 < \ldots < t_N = T$: Pour simuler le sous-jaçent, on génére des nombres aléatoire de loi normale $\xi_k \sim N(0,1)$ à chaque instant et on utilise la solution exacte de la dynamique précédente :

$$S_{t_k} = S_{t_{k-1}} e^{(r-\frac{\sigma^2}{2})(t_k - t_{k-1}) + \sigma\sqrt{t_k - t_{k-1}}\xi_k} \qquad k > 0$$

Après nous construisons notre martingale à partir de l'équation (3) avec $M_0^* = 0$

$$M_{t_k}^* = M_{t_{k-1}}^* + \mathbb{1}_{\{t_k \geq t^*\}} \left(\widetilde{P_{BS}}(t_k, S_{t_k}) - \widetilde{P_{BS}}(t_{k-1}, S_{t_{k-1}}) \right)$$

Finalement, nous résolvons le problème de minimisation en considérant l'algorithme du gradient stochastique suivant :

$$\lambda_n = \lambda_{n-1} - \frac{1}{n+1} \nabla P_0(\lambda) \tag{4}$$

Avec $P_0(\lambda) = \mathbb{E}\left[\max_{k \in \{0,\ldots,N\}} \left(e^{-rt_k}(K - S_{t_k})_+ - \lambda M_{t_k}^* \right) \right]$ et $\lambda_0 = 1$
Remarquons que la fonction que nous cherchons à minimiser est convexe et que le pas choisi vérifie bien $\sum \frac{1}{n+1} = \infty$ et $\sum \frac{1}{(n+1)^2} < \infty$.

2.2 Put sur moyenne géométrique

Nous travaillons dans un cadre multi-dimensionnel du modèle de Black-Scholes. Pour cela, nous considérons n actifs corrélés entre eux dont la dynamique s'écrit :

$$\forall d \in \{1, \ldots, n\}, \frac{dS_{t,d}}{S_{t,d}} = rdt + \sigma_d dB_{t,d} \tag{5}$$

où r est le taux d'intérêt, (σ_1, ... , σ_n) sont les volatilités de chacun des actifs et $B = (B_1, ..., B_n)$ est un vecteur de n mouvements browniens standards et réels de matrice de corrélation $Cor(B_{t,i}, B_{t,j}) = \rho_{i,j}$

Pour pouvoir simuler les trajectoires de ces actifs, nous considérons un mouvement brownien multidimensionnel W. Dans ce cas, nous posons $(B_t, t \geq 0) = (LW_t, t \geq 0)$ pour tout $t \geq 0$ où L est la factorisée de Cholesky de la matrice de corrélation ρ, donc $\rho = L'L$ et $W = (W^1, ..., W^D)$ est un mouvement brownien standard à valeurs dans $\mathbb{R}^n$. Nous obtenons au final l'équation permettant de simuler la trajectoire de n actifs :

$$\forall d \in \{1, \ldots, n\}, S_{t_k,d} = S_{t_{k-1},d} e^{(r-(\sigma_d)^2/2)(t_k - t_{k-1}) + \sigma_d \sqrt{(t_k - t_{k-1})} L_d \xi_k} \tag{6}$$

où L_d est la ligne d de la matrice L et $(\xi_k)_{k \in \mathbb{N}^*}$ est une suite i.i.d. de vecteurs gaussiens centrés réduits de matrice de covariance identité à valeurs dans $\mathbb{R}^n$. D'après la formule d'Itô ainsi que le critère de Lévy pour les martingales de crochet égale à t, nous pouvons démontrer que :

$$\frac{d\left(\prod_{d=1}^n S_{t,d}\right)}{\prod_{d=1}^n S_{t,d}} = \left(r + \frac{\sigma^2}{2} - \frac{1}{2n}\sum_{d=1}^n \sigma_d^2\right) dt + \sigma dW_t \tag{7}$$

avec

$$dW_t = \frac{1}{n\sigma}\sum_{d=1}^n \sigma_d dW_t^d$$

et

$$\sigma = \frac{1}{n}\sqrt{\sum_{d=1}^n \sigma_d^2 + 2\sum_{i<j} \rho_{i,j}\sigma_i\sigma_j}$$

Nous pouvons donc utiliser la formule de Black Scholes pour calculer le prix du put européen sur la moyenne géométrique dont le payoff est $\left(K - \prod_{d=1}^n S_{t,d}\right)_+$. Il est équivalent au prix du put d'un actif de volatilité σ et de dividende $q = \frac{1}{2n}\sum_{d=1}^n \sigma_d^2 - \frac{\sigma^2}{2}$

$$P_{BS}(t, S_{t,1}, \ldots, S_{t,n}) = Ke^{-r(T-t)}\mathcal{N}(-d_0) - S_t e^{-q(T-t)}\mathcal{N}(-d_1)$$

avec $d_1 = \frac{\ln \frac{S_t}{K} + \left(\frac{\sigma^2}{2} + r - q\right)(T-t)}{\sigma\sqrt{T-t}}$ et $d_0 = d_1 - \sigma\sqrt{T-t}$
De manière équivalente à ce qui est fait dans le cas du put sur un actif, nous pouvons donc prendre comme martingale la valeur actualisée de ce prix européen. Dans ce cas, l'algorithme précédent restera le même.

2.3 Réduction de variance

Pour réduire la variance, nous avons choisi d'utiliser la méthode de variables antithétiques. Il s'agit de calculer le prix de cette façon :

$$prix = \mathbb{E}\left[\frac{\phi(\xi) + \phi(-\xi)}{2}\right]$$

où ξ un vecteur gaussien, $\phi(\xi) = \max_{k\in\{0,\dots,N\}} \left(e^{-rt_k}(K - S_{t_k})_+ - \lambda M^*_{t_k}\right)$ et $\phi(-\xi) = \max_{k\in\{0,\dots,N\}} \left(e^{-rt_k}(K - S'_{t_k})_+ - \lambda M^*_{t_k}\right)$ pour le cas du Put. Cela permet de réduire la variance sans modifier le prix puisque ξ et $-\xi$ ont la même loi.
Pour pouvoir faire cela nous récupérons dans notre simulation des sous-jaçents S et S' tel que :

$$S_{t_k} = S_{t_{k-1}} e^{(r-\frac{\sigma^2}{2})(t_k - t_{k-1}) + \sigma\sqrt{t_k - t_{k-1}}\xi_k} \qquad k > 0$$

$$S'_{t_k} = S'_{t_{k-1}} e^{(r-\frac{\sigma^2}{2})(t_k - t_{k-1}) - \sigma\sqrt{t_k - t_{k-1}}\xi_k} \qquad k > 0$$

2.4 Résultats numériques

2.4.1 Put américain

Nous prenons les mêmes valeurs numériques que celles de l'article de Rogers : $\sigma = 0.4$, $K = 100$, $r = 0.06$, $T = 0.5$.
Quand l'option est à la monnaie, le calcul est fait avec 5000 trajectoires, par contre quand ce n'est pas le cas (c-à-d quand $S_0 > 100$) ce dernier se fait avec 10 000.

S_0	**Prix européen**	**Prix Américain** (vrai)	**Prix Américain** (Article Rogers)	**Prix obtenu**	**Intervalle de confiance**	λ
80	20.6893	21.6059	21.6953	21.6592	21.65509 - 21.66331	1.05246
85	17.3530	18.0374	18.1008	18.0748	18.0707 - 18.0789	1.04678
90	14.4085	14.9187	14.9692	14.9439	14.93954 - 14.94826	1.0416
95	11.8516	12.2314	12.2685	12.2486	12.24479 - 12.25241	1.03774
100	9.6642	9.9458	9.9703	9.9651	9.9280 - 10.002	1.03388
105	7.8183	8.0281	8.0439	8.0257	7.9812 - 8.0702	1.03069
110	6.2797	6.4352	6.4757	6.446	6.406 - 6.4856	1.0279
115	5.0113	5.1265	5.1363	5.1075	5.0724 - 5.1425	1.0251
120	3.9759	4.0611	4.0761	4.0413	4.0039 - 4.0786	1.02281

Nous remarquons que les valeurs obtenues quand l'option est à la monnaie sont très proches des vraies valeurs avec de très bons intervalles de confiance. Par contre, quand l'option ne l'est pas, nous obtenons des intervalles de confiance plus large.
Pour régler ce problème nous pouvons utiliser une autre méthode de réduction de variance comme l'Importance Sampling en réalisant une translation vers des valeurs à la monnaie.

2.4.2 Put sur moyenne géométrique

Pour cette partie, nous prenons les valeurs numériques $n = 2$, $\sigma_1 = 0.4$, $\sigma_2 = 0.1$, $\rho = 0.5$, $K = 100$, $r = 0.06$ et $T = 0.5$.

De manière similaire au pricing du Put, nous faisons le calcul avec 5000 trajectoires quand l'option est à la monnaie et 10 000 quand ce n'est pas le cas.
Pour tester notre algorithme, nous comparons les valeurs obtenues avec les valeurs du put américain de volatilité $\sigma = \frac{1}{2}\sqrt{\sigma_1^2 + \sigma_2^2 + 2\rho\sigma_1\sigma_2} = 0.22913$ et dividende $q = \frac{1}{4}\left(\sigma_1^2 + \sigma_2^2\right) - \frac{\sigma^2}{2} = 0.01625$ dans un simulateur en ligne.
La même remarque pour le put s'applique ici : la variance de notre estimateur

$S_{0,1}$	$S_{1,1}$	**Prix Européen**	**Prix Américain** (Simulateur en ligne)	**Prix obtenu**	**Intervalle de confiance**	λ
80	80	18.4212	20	20.0201	20.0198 - 20.02038	1.0859
80	90	14.404	15.4067	15.4728	15.4707 - 15.47488	1.07958
85	95	10.7726	11.3922	11.4342	11.4310 - 11.4373	1.06558
100	90	7.75359	8.1152	8.15313	8.14998 - 8.15628	1.05414
105	95	5.37102	5.5668	5.6031	5.60040 - 5.6058	1.04436
105	100	4.37538	4.5815	4.53989	4.51480 - 4.56497	1.04018
110	85	6.81128	7.1586	7.13574	7.13364 - 7.13783	1.04974
115	105	2.30911	2.4178	2.36607	2.34456 - 2.38758	1.03153
120	110	1.43886	1.4942	1.46387	1.44041 - 1.48733	1.02759

Monte-Carlo augmente quand l'option n'est pas à la monnaie, c'est pourquoi il est intéressant de mettre en œuvre une méthode de réduction de variance comme l'Importance sampling.

3 Multilevel pour l'approche duale

3.1 Principe de la méthode Multilevel

Pour une suite décroissante fixée $(n_i)_{i\in[\![1;n]\!]}$ telle que $\forall i \in [\![1;n]\!],\ n_i \geq 1$, et une suite de pas de temps croissante donnée $(t_k)_{k\in[\![1;L]\!]}$ de même longueur. On souhaite pouvoir simuler un processus de dimension $\mathcal{J}$, ainsi que son obstacle Z.

Initialement, on simule les premières trajectoires :

$$\{(Z_j^{(i)}, M_j^{k_0,(i))}),\ i = 1, ..., n_0, j = 1, ..., \mathcal{J}\}$$

.

Ensuite, on simule , à niveau l fixé($l \in [\![1;L]\!]$), indépendamment, des trajectoires :

$$\{(Z_j^{(i)}, M_j^{k_{l-1},(i))}, M_j^{k_l,(i))}),\ i = 1, ..., n_l, j = 1, ..., \mathcal{J}\}$$

.

On sait approximer l'enveloppe de Snell par :

$$\begin{aligned} \mathrm{Y}(\mathrm{M}^{k_L}) &= Y(M^{k_0}) + \sum_{l=1}^{L}(Y(M^{k_l}) - Y(M^{k_{l-1}}) \\ &= \mathbb{E}[\mathcal{Z}(M_0^k)] + \sum_{l=1}^{L}\mathbb{E}[\mathcal{Z}(M_l^k) - \mathcal{Z}(M_{l-1}^k)]] \end{aligned} \tag{8}$$

De ce fait, l'estimation de cette enveloppe par la méthode multilevel est donnée comme suit :

$$Y^{n,k} := \frac{1}{n_0}\sum_{i=1}^{n_0}\mathcal{Z}^(i)(M^{k_0}) + \sum_{l=1}^{L}\frac{1}{n_l}\sum_{i=1}^{n_l}[\mathcal{Z}^{(i)}(M^{k_l}) - \mathcal{Z}^{(i)}(M^{k_{l-1}})]$$

Avec $\mathcal{Z}^i(M^k) := max_{j=1,..,\mathcal{J}}(Z_j^{(i)} - M_j^{k,(i)});\ i \in [\![1;n_l]\!], k \in \mathbf{N}$, et $M^{k,(i)}$ la i-ème trajectoire simulée de la martingale M^k.

Dans la suite, $n_0, n_1, n_2, ..., n_l$ et $k_0, k_1, ..., k_l$ sont choisis de façon à traiter différemment n_0 par rapport aux autres, et lier le reste des paramètres : $k_l = k_0\kappa^l$ pour $l \in [\![0;L]\!]$, $n_l = n_1\kappa^{1-l}$ pour $l \in [\![1;L]\!]$. En réutilisant les simulations à un instant dans les instants suivant, la compléxité de l'algorithme est de la forme :

$$\mathcal{C}_{ML} = n_0k_0 + \sum_{l=1}^{L} n_lk_l = n_0k_0 + n_1k_0\kappa L$$

.

La constante de départ k_0 est choisie à priori et par expérience. Par exemple, $k_0 = 100$ s'avère être un résultat robuste pour le reste.

D'un autre côté, la variance de $Y^{k,n}$ vérifie :

$$Var[Y^{k,n}] = \frac{v^2(0,k_0)}{n_0} + \sum_{l=1}^{L} \frac{1}{n_l} v^2(k_{l-1}, k_l)$$

.

Où $v^2(k_{l-1}, k_l) := Var[\mathcal{Z}^{(r)}(M^{k_l}) - \mathcal{Z}^{(r)}(M^{k_l-1})]$ et $M^0 = Z$

De suite, en cherchant une minimisation de la variance, à biais et complexité fixée. En suposant l'existence de $\mathcal{V}_\infty$ et de σ_∞ tels que

$$\forall l : v(k_{l-1}, k_l) = \frac{\mathcal{V}_\infty}{\sqrt{k_l}} \; et \; v(0,k_0) = \sigma_\infty$$

pour des valeurs k_0 grandes, on peut alors choisir n_0^* et n_1^* solution de ce problème, définis de la manière suivante :

$$n_0^* = \frac{N\kappa^L}{1 + L\sqrt{v^2(k_{L-1}, K)\kappa^L/\sigma_\infty^2}}$$

$$n_1^* = n_0^* \sqrt{\frac{v^2(k_{L-1}, K)\kappa^{L-2}}{\sigma_\infty * 2}}$$

. Avec N le nombre de pas équivalent, trouvé depuis le système d'équations :

$$NK = n_0 k_0 + n_1 k_0 \kappa L \; et \; k_L = k_0 \kappa^L = K$$

.

3.2 Pricing des swaptions bermudiennes dans le Libor Market Model

Pour implémenter la méthode Multilevel introduite dans l'article [13] Multilevel dual approach for pricing american style derivatives, nous avons décidé comme dans l'article de pricer des swaptions dans le LMM.

On se place dans un espace de probabilité filtré $(\Omega, \mathcal{F}, (\mathcal{F})_t, \mathbb{P})$. La dynamique sous la probabilité $\mathbb{P}*$ permettant de donner le cours des taux forward dans le LMM telle qu'elle est définie dans l'article est la suivante :

$$\frac{dL_i}{L_i} = \sum_{j=\kappa(t)}^{i} \frac{\delta_j L_j \gamma_i . \gamma_j}{1 + \delta_j L_j} dt + \gamma_i . dW \qquad t \in [0, T_i], \qquad i \in 1, \ldots, n$$

Où $\delta_i = T_{i+1} - T_i$, $\gamma_i(t) = (\gamma_{i,1}(t), \ldots, \gamma_{i,n}(t))$, $\kappa(t) = \min\{m, T_m \geq t\}$ et W est un d-mouvement brownien sous $\mathbb{P}*$
La probabilité $\mathbb{P}*$ est celle qui corresponde au changement de numéraire suivant :

$$B_*(t) = \frac{B_{\kappa(t)}}{B_1(0)} \prod_{i=1}^{\kappa(t)-1} (1 + \delta_i L_i(T_i))$$

Tel que $B_i(t)$ est le zéro coupon d'échéance T_i.
Pour simuler les taux forwards, nous utilisons un schéma d'Euler sur le log de taux forward avec un pas de $\delta/5$.
En ce qui concerne les valeurs numériques, nous prenons les mêmes valeurs numériques définies dans l'article : $|\gamma_i(t)| = cg(T_i - t)$, tel que $g(s) = g_\infty + (1-g_\infty+as)e^{-bs}$, avec $c = 0.2$, $a = 1.5$, $b = 3.5$, $g_\infty = 0.5$. $\gamma_i(t) = cg(T_i-t)e_i$. Notre matrice de corrélations sera définie donc par : $\rho_{i,j} = e_i.e_j = e^{(-\phi|i-j|)}$, pour tout $i, j = 1, \ldots, n-1$, avec $\phi = 0.0413$.

$$\Delta \ln L_i(t_k) = \left(\sum_{j=\kappa(t_k)}^{i} \frac{\delta_j L_j \gamma_i.\gamma_j}{1 + \delta_j L_j} - \frac{|\gamma_i|^2}{2} \right) \Delta_{t_k} + \gamma_i.(W_{t_{k+1}} - W_{t_k})$$

Swaption bermudienne

Une swaption bermudienne donne à son acheteur le droit de rentrer dans un swap pour un strike donné dans une des dates d'exercices définies $\{\eta_1, \ldots, \eta_J\} \subset \{T_1, \ldots, T_n\}$. Le payoff correspondant à cette option est le suivant :

$$S(T_i) = \left(\sum_{j=i}^{n-1} B_{j+1}(T_i)\delta_j(L_j(T_i) - \theta) \right)_+$$

Avec ces paramètres, notre obstacle sera définie par :

$$Z_j = \frac{S(\eta_j)}{B_*(\eta_j)}$$

Pour construire notre martingale nous considérons la stratégie d'exercice suivante :

$$\tau_i = \inf \left\{ j \in \{i, \ldots, J\} \max_{k \in \{j,\ldots,n\}} \mathbb{E}[Z_k|\mathcal{F}_j] \leq Z_j \right\}$$

Comme dans l'article, nous calculons $\mathbb{E}[Z_k|\mathcal{F}_j]$ en utilisant l'approximation de Rebonato :

$$\mathbb{E}[Z_k|\mathcal{F}_j] = C_{j,n}(T_j) \left(S_{j,n}(T_j)\mathcal{N}(d_1) - \theta\mathcal{N}(d_0) \right)$$

Tel que $C_{j,n}(T_j) = \sum_{k=j+1}^{n} \delta_k P(T_j, T_k)$, $S_{j,n}(T_j) = \frac{1-\prod_{k=j+1}^{n} \frac{1}{(1+\delta_k L_k(T_j))}}{C_{j,n}(T_j)}$, $d_1 = \frac{\ln \frac{S_{j,n}(T_j)}{\theta} + \frac{\sigma_{j,n}^2}{2}}{\sigma_{j,n}\sqrt{T_j - t}}$, et $d_0 = d_1 - \sigma_{i,j}\sqrt{T_k - T_j}$.
L'approximation de Rebonato consiste à approcher la volatilité du taux swap S vue comme un processus log-normal de Black. La formule correspondante est la suivante :

$$\sigma_{j,n}^2 = \sum_{k,l}^{n} \frac{w_k(0) w_l(0) L_k(0) L_l(0) \rho_{k,l}}{T_j S_{j,n}(0)^2} \int_0^{T_j} |\gamma_l(t)||\gamma_k(t)| dt$$

Avec $w_k(T_j) = \frac{\delta_k P(t,T_k)}{\sum_{l=k+1}^{n} \delta_l P(t,T_l)}$
La martingale optimale sera donc définie pour tout $i \in \{1, \ldots, J\}$ par :

$$M_i = \sum_{k=1}^{i} \Delta_k$$

Tel que $\Delta_k = \mathbb{E}[Z_{\tau_i}|\mathcal{F}_i] - \mathbb{E}[Z_{\tau_i}|\mathcal{F}_{i-1}]$ sont calculés par une méthode de Monte-Carlo avec un nombre de tirages égale à k qui varie en fonction du Level.

4 Architecture du moteur de calcul

Nous avons raisonné en adoptant une vision top/down. Génériquement, un produit financier verse un ou des flux et peut avoir une formule fermé donnant son prix comme le cas du put dans le modèle de Black Scholes. Nous avons donc créé une classe abstraite `Product` contenant les fonctions `Payoff(int maturityIndex, gsl_matrix *path)` et `TheoriticalPrice(double t, double T, gsl_matrix* asset= nullptr)`. Les classes Put, GeometricMeanPut ainsi que Swaption dérivants de `Product` et implémentent ces deux fonctions.
En ce qui concerne les prix théoriques, les classes Put et GeometricMeanPut utilisent le modèle de Black Scholes et donc les prix donnés sont issus des formules fermés dans ce modèle comme nous l'avons présenté avant. En ce qui concerne la classe Swaption, nous utilisons Libor Market Model et par suite le prix donné est une approximation donnée par Rebunato.
Le même raisonnement était adopté pour la partie Modèle. Nous avons choisi de définir une classe abstraite Model contenant les fonctions `Simul_Path(gsl_matrix* path, double T, int nbTimeSteps, gsl_rng* rng, gsl_matrix* path_minus= nullptr)` permettant d'avoir une trajectoire partante de l'instant 0 et `Simul_Path(gsl_matrix* path, double t, double T, int nbTimeSteps, gsl_matrix* past, gsl_rng*`

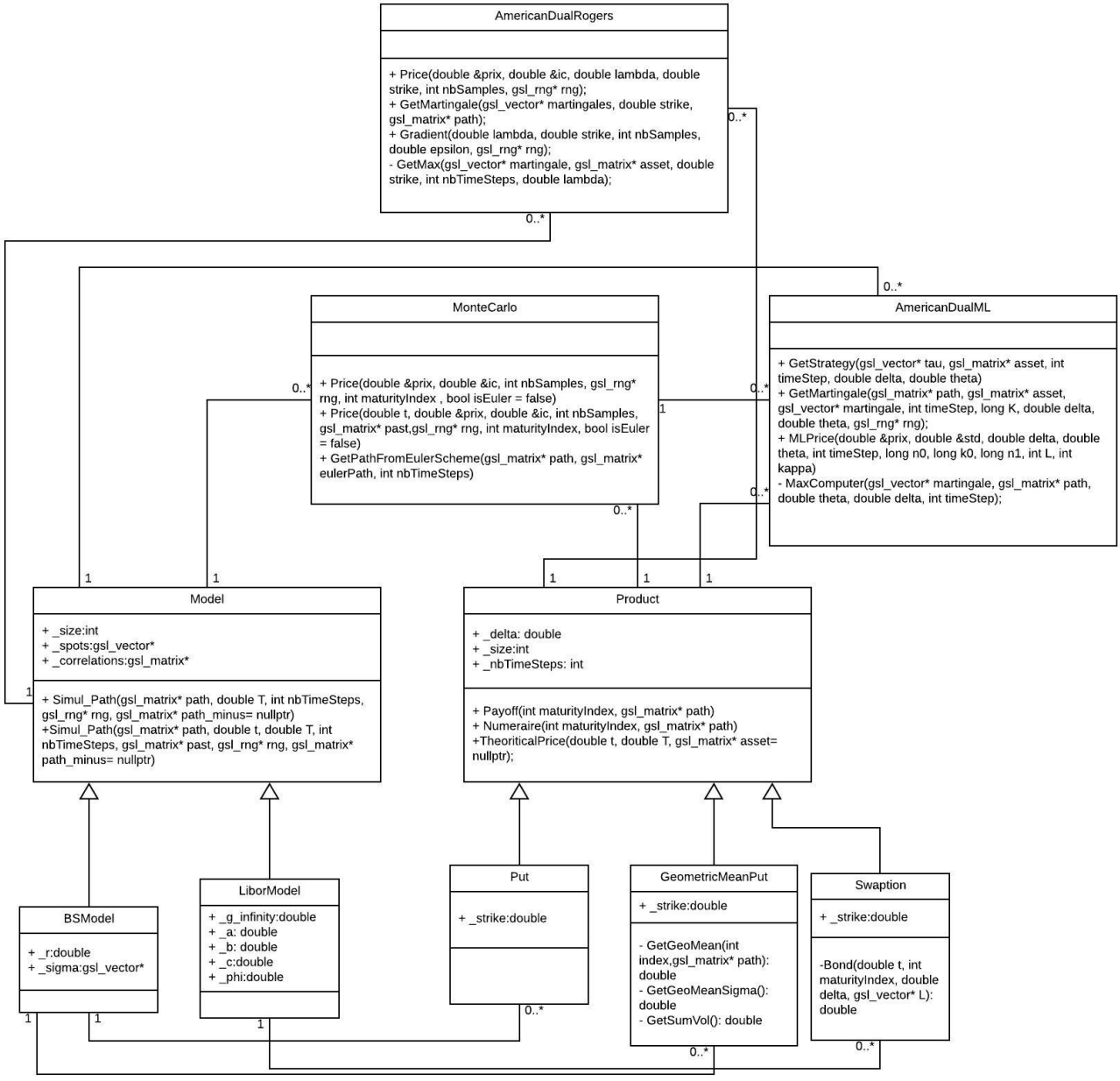

FIGURE 1 – Architecture du moteur de calcul

`rng, gsl_matrix* path_minus= nullptr)` pour une trajectoire partante de l'instant t. Les classes BSModel et LiborModel dérivent de la classe Modèle et implémentent donc ces deux fonctions. Dans le premier cas nous utilisons le modèle de Black et Scholes multi-dimensionnel comme nous l'avons

expliqué précédemment et dans le deuxième nous implémentons le schéma d'Euler du LMM tel qu'il est défini dans l'article du MultiLevel.
Les classes MonteCarlo, AmericanDualML et AmericanDualRogers utilisent de façon générique les classes Model et Product pour réaliser les calculs demandés :
Dans MonteCarlo, nous définissons `Price(double &prix, double &ic, int nbSamples, gsl_rng* rng, int maturityIndex , bool isEuler = false)` ainsi que `Price(double t, double &prix, double &ic, int nbSamples, gsl_matrix* past,gsl_rng* rng, int maturityIndex, bool isEuler = false)` qui permettent le calcul du prix d'un produit avec la moyenne de Monte Carlo à l'instant initial ou à un instant quelconque.
En ce qui concerne la classe AmericanDualRogers, nous définissons la fonction `GetMartingale(gsl_vector* martingales, double strike, gsl_matrix* path)` qui va calculer la martingale optimale selon l'algorithme que nous avons définie avant. Nous définissons également la fonction `Gradient(double lambda, double strike, int nbSamples, double epsilon, gsl_rng* rng)` qui va être utilisé dans l'algorithme de gradient stochastique. Finalement, nous calculons le prix de l'option américaine grâce à notre fonction `Price(double &prix, double &ic, double lambda, double strike, int nbSamples, gsl_rng* rng)`

Remarque

Nous utilisons dans tous ce qui concerne le calcul scientifique (Factorisée de Cholesky, génération des nombres aléatoires, calcul d'intégrales ...) la bibliothèque GSL. Il s'agit d'une très bonne bibliothèque de calcul scientifique permettant de réduire la complexité de notre moteur de calcul.

Conclusion

Le but de cette étude était de vérifier l'impact des méthodes duales et d'une implémentation en multilevel pour la valorisation des options américaines et bermudéennes pour différentes options. Nous nous sommes intéressés au cas du put américain de payoff $(K-X_t)_+$, et d'une option géométrique de payoff $(K-\sqrt{X_t^1 X_t^2})_+$, avant de nous intéresser à la méthode multilevel qui permet de réduire la variance de la méthode d'Anderson et Broadie.