École Nationale Supérieure des Mines de Saint-Étienne

Calcul Haute Performance

Finance Quantitative

Résolution de l'équation à dérivées partielles de Black-Scholes, Dupire en CUDA

Matthieu Pinard, M.Sc.
matthieu.pinard@etu.emse.fr

Dr. Asdin Aoufi
aoufi@emse.fr

13 août 2017

Table des matières

1 Introduction

En 1973, Black, Scholes et Merton proposent un modèle pour la valorisation d'options dans un contexte de volatilité et de taux constants puis dépendants du temps.
Ce modèle, toujours utilisé aujourd'hui, a été donné sous la forme d'une équation aux dérivées partielles, et a permis de démocratiser les activités de trading sur dérivés.
La résolution de cette EDP permet de donner des prix aux dérivés traités par les banques de manière relativement simple, bien qu'aujourd'hui, du fait de l'augmentation de la puissance de calcul, les algorithmes de Monte Carlo sont préférés parce qu'ils peuvent être très facilement parallélisés.
Cette étude aura pour but d'analyser la résolution numérique de l'équation à dérivées partielles de Black-Scholes pour des options européennes et américaines – sous un contexte de volatilité constante ou de volatilité locale.
Pour se faire, nous implémenterons un Pricer : un programme informatique – développé par les équipes de Recherche Quantitative – qui, en utilisant les données de marché, permet de donner un prix à un produit dérivé, ainsi que d'autres indicateurs si nécessaire (par exemple, les Grecs pour le Trading afin qu'ils puissent réaliser une couverture sur leur book, et des indicateurs de risques).
L'accent sera mis sur les différents schémas de résolution (explicite et implicite en temps) ainsi que sur la performance du Pricer, qui sera développé en CUDA C.

2 CUDA

CUDA (Compute Unified Device Architecture) est une plateforme logicielle de programmation parallèle développée par nVidia en 2007, pour utiliser les nombreux cœurs de ses GPU (Graphics Processing Unit). Cette technologie sert à délocaliser les calculs du CPU vers le GPU – ce qui s'avère être intéressant pour les problèmes de grande dimension rencontrés en finance, tels que les simulations de Monte Carlo et la résolution d'EDP.
Elle est rendue accessible au programmeur grâce à une API en C/C++ et au compilateur NVCC, qui peut compiler et optimiser le code en fonction de la génération du GPU sur lequel il va être lancé.
En pratique, l'utilisateur de CUDA va écrire des kernels – des routines en C exécutées sur le processeur graphique - qu'il va lancer en parallèle sur plusieurs cœurs de son GPU. A chaque cœur est attribué un thread, les threads sont regroupés en blocs de taille identique. Il est intéressant de noter que la mémoire GPU et la mémoire CPU sont disjointes : aussi, tenter de lire la mémoire GPU à partir du processeur central (ou inversement) résultera en une violation d'accès.
La plateforme CUDA contient également diverses librairies, dont CUBLAS (routines d'algèbre linéaire : BLAS niveaux 1 à 3), CURAND (génération de nombres pseudo-aléatoires) et CUSOLVER (résolution de systèmes linéaires, décompositions).
CUDA présente cependant plusieurs inconvénients :

- La synchronisation entre GPU et CPU est lente, car elle utilise le bus PCI-Express : il faut alors éviter au maximum les interactions entre ces deux entités.
- Le problème doit être suffisamment grand en nombre d'opérations flottantes pour justifier l'utilisation du GPU : une unité GPU est plus lente qu'une unité CPU (1.5 GHz contre 4.0 GHz) et la création de threads (lancement de kernels) a un coût.
- CUDA ne peut être utilisé que sur plateforme nVidia.
- Les GPU « grand-public » n'autorisent que le calcul simple précision à pleine puissance (C++ : float), seuls les GPU professionnels permettent le calcul double précision à demie puissance.

Une des fonctionnalités de CUDA est la mémoire partagée, indiquée par __shared__. Cette dernière permet aux threads d'un même bloc de partager la mémoire allouée au bloc, qui est plus rapide que la mémoire globale. Cette fonctionnalité est particulièrement utile lorsque plusieurs threads lisent une même valeur, comme c'est le cas pour les schémas de résolution implicite et explicite.

3 Option européenne

3.1 Introduction

Une option est un contrat qui donne à l'acheteur le droit – et non l'obligation - d'acheter (pour un Call) ou de vendre (pour un Put) un actif sous-jacent (ou un dérivé) à un prix fixé à l'avance (le Strike). À ce droit est associé un prix, qu'on appelle la Prime (ou le Premium).
Une option européenne permet d'exercer ce droit à une date fixée, la Maturité. Il existe d'autres types d'option (américaine, bermudienne, asiatique, etc.) qui modifient les conditions d'exercice de ce droit.
Le prix des options européennes sur Equity/FX peut être déterminé analytiquement en utilisant la formule de Black-Scholes (1973) ou de Black (1976). Ce prix analytique sera un élément de comparaison pour contrôler les prix issus de la résolution de l'équation à dérivées partielles de Black-Scholes (Fig.1).

$$Call = Se^{-qT}N\left(d1\right) - Ke^{-rT}N\left(d2\right) = e^{-rT}\left(FN\left(d1\right) - KN\left(d2\right)\right)$$
$$Put = Ke^{-rT}N\left(-d2\right) - Se^{-qT}N\left(-d1\right) = e^{-rT}\left(KN\left(-d2\right) - FN\left(-d1\right)\right)$$
$$d_{1,2} = \frac{\ln\left(\frac{S}{K}\right) + \left(r - q \pm \frac{\sigma^2}{2}\right)T}{\sigma\sqrt{T}} = \frac{\ln\left(\frac{F}{K}\right) \pm \frac{\sigma^2}{2}T}{\sigma\sqrt{T}}$$

Cependant, un des paramètres du modèle, la volatilité implicite, dépend aujourd'hui du temps et du Strike (Fig.2). Ce n'était pas le cas avant le crash boursier de 1987, où seule une dépendance temporelle était considérée. En effet, une des hypothèses du modèle de Black-Scholes était une distribution log-normale du sous-jacent à maturité, mais les données historiques tendent à infirmer cette hypothèse (« fat tails » et crashs plus fréquents que prévus).
Divers modèles ont alors été développés :

- Modèles de saut : Ajout d'un processus de Poisson dans l'équation différentielle stochastique de Black-Scholes pour modéliser les sauts (ex : Merton, 1976)
- Volatilité stochastique : la volatilité est corrélée avec le sous-jacent et possède une nature aléatoire (ex : Heston en 1993, SABR)
- Volatilité locale : la volatilité est déterministe, dépend du temps et du niveau du sous-jacent (Dupire, 1994)

Ces modèles induisent ce qu'on appelle un « smile » de volatilité. L'avantage du modèle à volatilité locale est qu'il permet de reconstituer tous les prix des options européennes cotées sur le marché, contrairement aux modèles à volatilité stochastique.
Les modèles à volatilité constante sont inadaptés au pricing de dérivés complexes (autres que des combinaisons d'options « vanille »), et on utilisera volontiers une volatilité stochastique pour les produits qui dépendent du caractère aléatoire de la volatilité ou dépendant de la distribution du sous-jacent en plusieurs instants. Pour les autres produits, une volatilité locale donne de bons résultats.
Dans la suite de notre étude, nous considèrerons d'abord une volatilité constante (en nous limitant alors à un Strike précis) puis une volatilité locale.

3.2 EDP de Black-Scholes

Le modèle de Black-Scholes repose sur nombre d'hypothèses plus ou moins fortes, qui sont :

- Il existe un actif risqué S (l'action) et un actif sans risque (par exemple, le Money Market ou les obligations).
- Le taux d'intérêt de l'actif sans risque est constant égal à r.
  - Merton : ce taux peut éventuellement dépendre du temps.
- L'actif risqué suit un mouvement géométrique Brownien, dont le drift μ et la volatilité σ sont constants.
  - Merton : Une dépendance temporelle reste possible. Cette hypothèse est la plus problématique, notamment au niveau de la volatilité, puisque cette dernière est corrélée à l'actif sous-jacent.
- L'actif risqué ne verse pas de dividendes.
  - En pratique, le modèle Black utilise les niveaux des Futures – ces derniers ne versent pas de dividendes. Nous modifierons ensuite l'EDP originelle pour y inclure des dividendes continus et proportionnels.
- Il n'y a pas d'opportunités d'arbitrage.
- Il est possible d'emprunter/prêter n'importe quel montant de l'actif sans risque.
- Il est possible d'acheter/vendre n'importe quelle fraction de l'actif risqué.
  - Cette hypothèse peut paraître trop forte, mais les books d'options sont en général assez importants pour qu'un arrondi n'ait que peu d'influence.
- Pas de frais de transaction.
  - En pratique, il y a des frais et un taux de repo – ie. le taux auquel les actions sont converties en cash.

Sous ce modèle, on peut écrire une équation à dérivées partielles qui décrit le prix de l'option européenne, disons $V = V(t,\ S)$ où t est le temps et S le sous-jacent.
L'équation de diffusion de l'actif risqué est :

$$dS = \mu S dt + \sigma S dW$$

En appliquant le lemme d'Itô à $V(t,\ S)$ nous obtenons :

$$dV = \left(\frac{\partial V}{\partial t} + \mu S\frac{\partial V}{\partial S} + \frac{\sigma^2 S^2}{2}\frac{\partial^2 V}{\partial S^2}\right)dt + \sigma S\frac{\partial V}{\partial S}dW$$

Soit :

$$-\frac{\partial V}{\partial S}dS = -\mu S\frac{\partial V}{\partial S}dt - \sigma S\frac{\partial V}{\partial S}dW$$

$$dV - \frac{\partial V}{\partial S}dS = \left(\frac{\partial V}{\partial t} + \frac{\sigma^2 S^2}{2}\frac{\partial^2 V}{\partial S^2}\right)dt$$

En considérant $\Pi = V - \frac{\partial V}{\partial S}S$, nous constatons que ce portefeuille est sans risque, donc par la deuxième hypothèse du modèle Black-Scholes, nous avons :

$$d\Pi = r\Pi dt$$

$$d\left(V - \frac{\partial V}{\partial S}S\right) = r\left(V - \frac{\partial V}{\partial S}S\right)dt$$

D'où, par simplification par dt :

$$\frac{\partial V}{\partial t} + \frac{\sigma^2 S^2}{2}\frac{\partial^2 V}{\partial S^2} = rV - rS\frac{\partial V}{\partial S}$$

Cette EDP devient alors :

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV = 0$$

Dans le cas d'un sous-jacent versant des dividendes, on reprend la preuve précédente où q est le taux de dividendes.

$$dS = (\mu - q)\, Sdt + \sigma S dW$$

Soit :

$$dV - \frac{\partial V}{\partial S}dS = \left(\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2}\right) dt - qS\frac{\partial V}{\partial S}dt = r\left(V - \frac{\partial V}{\partial S}S\right) dt$$

D'où :

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + (r - q)\, S\frac{\partial V}{\partial S} - rV = 0$$

On reconnait ici une équation dite d'advection-diffusion-réaction.

3.3 Modèle à volatilité locale (Dupire)

L'apport de Dupire a été de modifier l'équation de diffusion en utilisant une surface de volatilité $\sigma(t,\ S)$:

$$dS = \mu(t)\, S dt + \sigma(t,\ S)\, S dW$$

Le problème réside dans la construction d'une telle surface de volatilité. Appliquons l'équation de Fokker-Planck à cette équation de diffusion, en notant $p(x,\ t)$ la densité de probabilité de x à l'instant t, nous avons :

$$\frac{dp}{dt}(x,\ t) = -\frac{d}{dx}\left(x\mu(t)\, p(x,t)\right) + \frac{1}{2}\frac{d^2}{dx^2}\left(x^2\sigma^2(x,\ t)\, p(x,\ t)\right) \tag{1}$$

En effectuant une intégration par parties, on note que :

$$\int_K^\infty (x-K)\frac{d}{dx}\left(x\mu(T)\, p(x,T)\right)dx = \Big[(x-K)\, x\mu(T)\, p(x,T)\Big]_K^\infty - \int_K^\infty (x-K+K)\,\mu(T)\, p(x,T)\, dx$$

On fait l'hypothèse que $\lim_{x\to\infty} xp(x,T) = 0$.

$$-\int_K^\infty (x-K)\frac{d}{dx}\left(x\mu(T)\, p(x,T)\right)dx = \mu(T)\left(\int_K^\infty (x-K)\, p(x,T)\, dx + K\int_K^\infty p(x,T)\, dx\right)$$

$$-\int_K^\infty (x-K)\frac{d}{dx}\left(x\mu(T)\, p(x,T)\right)dx = \mu(T)\, e^{rT}\left(C(K,T) + K\frac{dC}{dK}(K,T)\right) \tag{2}$$

Nous avons également, par la formule de Carr-Madan :

$$e^{rT}\frac{\partial^2 C}{\partial K^2}(K,T) = \int_0^\infty \frac{\partial^2 (x-K)^+}{\partial K^2} p(x,T)dx = p(K,T)$$

Aussi, pour le terme de second ordre :

$$\int_K^\infty (x-K)\frac{1}{2}\frac{d^2}{dx^2}\left(x^2\sigma^2(x,\ T)\, p(x,\ T)\right)dx$$

$$= \left[(x-K)\frac{1}{2}\frac{d}{dx}\left(x^2\sigma^2(x,\ T)\, p(x,\ T)\right)\right]_K^\infty - \int_K^\infty \frac{1}{2}\frac{d}{dx}\left(x^2\sigma^2(x,\ T)\, p(x,\ T)\right)dx$$

$$\int_K^\infty (x-K)\frac{1}{2}\frac{\partial^2}{\partial x^2}\left(x^2\sigma^2(x,\ T)\, p(x,\ T)\right)dx = \frac{1}{2}K^2\sigma^2(K,T)p(K,T) = \frac{1}{2}K^2\sigma^2(K,T)e^{rT}\frac{\partial^2 C}{\partial K^2}(K,T) \tag{3}$$

Sous hypothèse $\lim_{x\to\infty} x\frac{d}{dx}\left(x^2\sigma^2(x,\ T)\, p(x,\ T)\right) = 0$.
Pour le terme temporel, on a :

$$\frac{d}{dt}\left(e^{-rt}\int_K^\infty (x-K)\, p(x,t)\, dx\right) = \frac{dC}{dt}$$

$$e^{-rt}\frac{d}{dt}\left(\int_K^\infty (x-K)\, p(x,t)\, dx\right) - re^{-rt}\int_K^\infty (x-K)\, p(x,t)\, dx = \frac{dC}{dt}$$

$$e^{-rt}\int_K^\infty (x-K)\frac{dp}{dt}(x,t)\, dx - rC = \frac{dC}{dt}$$

$$\int_K^{\infty} (x-K)\,\frac{dp}{dt}\,(x,t)\,dx = e^{rT}\left(\frac{dC}{dt} + rC\right) \tag{4}$$

Nous obtenons alors en réinjectant les termes précédents ((2), (3), (4)) dans l'équation de Fokker-Planck (1) :

$$\sigma(T,K) = \frac{1}{K}\sqrt{\frac{2\left(\frac{\partial C}{\partial t} + rC - \mu(T)\big(C + K\frac{\partial C}{\partial K}\big)\right)}{\frac{\partial^2 C}{\partial K^2}}}$$

Où $\frac{\partial C}{\partial t}$ (le Theta), $\frac{\partial C}{\partial K}$ (le Dual Delta) et $\frac{\partial^2 C}{\partial K^2}$ (le Dual Gamma) sont déduits des prix qui cotent sur le marché.
L'EDP devient alors :

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma(t,S)^2 S^2 \frac{\partial^2 V}{\partial S^2} + (r(t) - q(t))\,S\frac{\partial V}{\partial S} - r(t)V = 0$$

Il suffira de reprendre la preuve de l'EDP avec $\sigma = \sigma(t,\ S)$.
La condition initiale et les conditions aux limites seront décrites ultérieurement.

3.4 Méthode des différences finies

La méthode des différences finies permet d'approximer numériquement les équations différentielles (qu'elles soient ordinaires, à dérivées partielles ou stochastiques). Cette méthode consiste en la discrétisation du problème (création d'une grille – uniforme ou non - d'autant de dimensions que de paramètres du problème, 2 dans le cas Black-Scholes : t et S), et en l'approximation des dérivées de l'équation aux différents points de la grille. Une bonne règle pour la résolution d'options consiste à prendre $S \in \left[K - 6\sigma\sqrt{T}, K + 6\sigma\sqrt{T}\right]$ où K est le strike de l'option.
La convergence d'une résolution à l'aide de la méthode des différences finies dépend de plusieurs paramètres :

- Le choix de la méthode numérique d'intégration en temps : explicite, implicite, Crank-Nicolson... Ces méthodes sont conditionnellement ou inconditionnellement stables.
- Les erreurs de précision, dépendantes de la machine, calcul sur 32 bits (simple précision) ou 64 bits (double précision).
- L'approximation des dérivées (sensible au pas entre les points de la grille et à la méthode de calcul – décentrée amont, décentrée aval ou centrée : le développement de Taylor est alors utilisé pour fournir le terme d'erreur)

3.4.1 Approximation des dérivées

	Décentrée amont	Décentrée aval	Centrée
u'	$\frac{u(x+h)-u(x)}{h}$	$\frac{u(x)-u(x-h)}{h}$	$\frac{u(x+h)-u(x-h)}{2h}$
u''	$\frac{u(x+2h)-2u(x+h)+u(x)}{h^2}$	$\frac{u(x)-2u(x-h)+u(x-2h)}{h^2}$	$\frac{u(x+h)-2u(x)+u(x-h)}{h^2}$

Les erreurs, calculées à l'aide du développement de Taylor, sont les suivantes :

Erreurs	Décentrée amont	Décentrée aval	Centrée
Dérivée première	$\frac{hu''(\xi)}{2}$	$\frac{hu''(\xi)}{2}$	$\frac{h^2u^{(3)}(\xi)}{6}$
Dérivée seconde	$hu^{(3)}(\xi)$	$hu^{(3)}(\xi)$	$\frac{h^2u^{(4)}(\xi)}{12}$

Il en résulte que la méthode d'approximation centrée des dérivées offre les meilleurs résultats, c'est cette méthode qui sera utilisée pour la résolution de l'équation de Black-Scholes.

3.4.2 Méthode numérique

Nous utiliserons la méthode centrée pour l'approximation des dérivées, et comparerons ici les méthodes de résolution explicite et implicite en temps.
Par souci de simplicité, nous noterons $V(i,\, n) = V\left(S^i,\, t^n\right)$ où i est l'indice spatial et n l'indice temporel. Le problème est rétrograde en temps, avec une condition à la limite terminale.
Aussi, l'échelle du temps sera retournée pour transformer la condition à la limite en condition initiale : la maturité de l'option est en $n = 0$, et le changement de variable associé est $\tau = T - t$.
Nous rappelons alors l'équation différentielle de Black-Scholes à volatilité locale :

$$-\frac{\partial V}{\partial \tau} + \frac{1}{2}\sigma(T-\tau, S)^2 S^2 \frac{\partial^2 V}{\partial S^2} + (r(T-\tau) - q(T-\tau))\, S\frac{\partial V}{\partial S} - r(T-\tau)V = 0$$

Les conditions de Dirichlet aux limites sont les suivantes pour les options vanilles, elles sont tirées des formules de Black-Scholes :

$D(S,t)$	Call	Put
$S \to 0$	0	$Ke^{-\int_t^T r(u)du} - Se^{-\int_t^T q(u)du}$
$S \to +\infty$	$Se^{-\int_t^T q(u)du} - Ke^{-\int_t^T r(u)du}$	0

De cette équation différentielle, nous déduisons deux méthodes de résolution :

- La méthode explicite : Il s'agit de calculer $V(n+1)$ en fonction des $V(n)$ déjà connus.
  La discrétisation est alors :

$$-\frac{V(i,\,n+1)-V(i,\,n)}{\Delta\tau} + \frac{1}{2}\sigma^2(i,n)\,S^2(i)\left(\frac{V(i+1,n)-2V(i,n)+V(i-1,n)}{\Delta S^2}\right) + (r(n)-q(n))\,S(i)\left(\frac{V(i+1,n)-V(i-1,n)}{2\Delta S}\right) - r(n)\,V(i,n) = 0$$

Cela revient à effectuer à chaque pas de temps :

$$V(i,\,n+1) = \alpha(i)\,V(i-1,\,n) + \beta(i)\,V(i,\,n) + \gamma(i)\,V(i+1,n)$$

Soit en notation matricielle :

$$\begin{pmatrix} \beta(1) & \gamma(1) & 0 & \dots & \dots & 0 \\ \alpha(2) & \dots & \dots & & & \dots \\ 0 & \dots & \dots & \dots & & \dots \\ \dots & & \dots & \dots & \dots & 0 \\ \dots & & & \dots & \dots & \gamma(N-1) \\ 0 & \dots & \dots & 0 & \alpha(N) & \beta(N) \end{pmatrix} \begin{pmatrix} V(1,\,n) \\ \dots \\ \\ \\ \dots \\ V(N,n) \end{pmatrix} + \begin{pmatrix} \alpha(1)\,D(0,n) \\ 0 \\ \dots \\ \dots \\ 0 \\ \gamma(N)\,D(\infty,n) \end{pmatrix} = \begin{pmatrix} V(1,\,n+1) \\ \dots \\ \\ \\ \dots \\ V(N,n+1) \end{pmatrix}$$

La matrice ainsi introduite est tridiagonale.

- La méthode implicite : $V(n+1)$, ce qui conduit à la résolution d'un système linéaire tridiagonal.
  La discrétisation s'écrit :

$$-\frac{V(i,\,n+1)-V(i,\,n)}{\Delta\tau} + \frac{1}{2}\sigma^2(i,n+1)\,S^2(i)\left(\frac{V(i+1,n+1)-2V(i,n+1)+V(i-1,n+1)}{\Delta S^2}\right) + (r(n+1)-q(n+1))\,S(i)\left(\frac{V(i+1,n+1)-V(i-1,n+1)}{2\Delta S}\right) - r(n+1)\,V(i,n+1) = 0$$

Cela revient à effectuer à chaque pas de temps :

$$V(i,\,n) = \alpha^*(i)\,V(i-1,\,n+1) + \beta^*(i)\,V(i,\,n+1) + \gamma^*(i)\,V(i+1,n+1)$$

Soit, en notation matricielle :

$$\begin{pmatrix} \beta^*(1) & \gamma^*(1) & 0 & \dots & \dots & 0 \\ \alpha^*(2) & \dots & \dots & & & \dots \\ 0 & \dots & \dots & \dots & & \dots \\ \dots & & \dots & \dots & \dots & 0 \\ \dots & & & \dots & \dots & \gamma^*(N-1) \\ 0 & \dots & \dots & 0 & \alpha^*(N) & \beta^*(N) \end{pmatrix} \begin{pmatrix} V(1,\,n+1) \\ \dots \\ \\ \\ \dots \\ V(N,n+1) \end{pmatrix} + \begin{pmatrix} \alpha^*(1)\,D(0,n+1) \\ 0 \\ \dots \\ \dots \\ 0 \\ \gamma^*(N)\,D(\infty,n+1) \end{pmatrix} = \begin{pmatrix} V(1,\,n) \\ \dots \\ \\ \\ \dots \\ V(N,n) \end{pmatrix}$$

La méthode explicite a l'avantage d'être facile à implémenter et très rapide à calculer. En contrepartie, elle peut être instable numériquement, si les valeurs du pas de temps et du pas d'espace sont mal choisies.

La stabilité de la méthode explicite est atteinte si les $\alpha(i)$, $\beta(i)$ et $\gamma(i)$ définis plus haut sont positifs à chaque pas de temps.
Ceci impose alors les contraintes suivantes sur ΔS, puis sur $\Delta\tau$ si $\Delta S \geq \frac{\sigma(i,\,n)S(i)}{\sqrt{r(n)}} \geq \min_{i,\,n}\left(\frac{\sigma(i,\,n)S(i)}{\sqrt{r(n)}}\right)$.

$$\alpha(i),\gamma(i) \geq 0 \implies \Delta S(i,n) \leq \frac{\sigma^2(i,\ n)\,S(i)}{|r(n)-q(n)|} \leq \min_{i,\ n}\left(\frac{\sigma^2(i,\ n)\,S(i)}{|r(n)-q(n)|}\right)$$

$$\beta(i) \geq 0 \implies \Delta\tau \leq \frac{1}{\frac{\sigma(i,\ n)^2 S^2(i)}{\Delta S^2} - r} \leq \min_{i,\ n}\left(\frac{1}{\frac{\sigma(i,\ n)^2 S^2(i)}{\Delta S^2} - r}\right)$$

La somme des trois coefficients $\alpha(i)+\beta(i)+\ \gamma(i) = 1+r(n)\,\Delta\tau$. En considérant le terme d'erreur entre la solution analytique et la solution numérique (Fig.3, Fig.4), on remarque par ailleurs que :

$$|\epsilon(i,n+1)| = |\alpha(i)\epsilon(i-1,n) + \beta(i)\epsilon(i,n) + \gamma(i)\epsilon(i+1,n)| \leq |1+r(n)\Delta\tau| \max_{i-1\leq j\leq i+1} |\epsilon(j,n)|$$

D'où, en notant $E(n) = \max_{i-1\leq j\leq i+1} |\epsilon(j,n)|$, on a : $E(n+1) \leq |1+r(n)\,\Delta\tau|\,E(n)$ d'où :

$$E(N) \leq |1+R\Delta\tau|^N E(0) = E(0)\left(1+\frac{RT}{N}\right)^N \underset{N\to\infty}{\to} E(0)\,e^{RT}$$

Avec $R = \max_n r(n)$. On a alors une borne supérieure finie de l'erreur numérique, ce qui nous donne la stabilité du système – compte tenu des contraintes précédentes sur ΔS et $\Delta\tau$.
La méthode implicite est toujours stable numériquement, mais elle est également plus difficile à implémenter et plus lente par point de maillage, car un système linéaire tridiagonal doit être résolu à chaque pas de temps.
Aussi, l'erreur de consistance des deux méthodes est la même : $O\left(\Delta t + \Delta S^2\right)$. La complexité spatiale et temporelle est également la même pour les deux méthodes : $O(S*T)$.
Cela est lié au fait que la résolution du système linéaire de la méthode implicite revient à résoudre un système tridiagonal, ce qui a une complexité de $O(S)$. La librairie cuSPARSE résout ce système en utilisant les algorithmes de Parallel Cyclic Reduction et Cyclic Reduction. Résoudre le système linéaire en utilisant la méthode du pivot de Gauss aurait conduit à une complexité de $O\left(S^3\right)$.

4 Option américaine

Une option américaine est similaire à une option européenne, à la différence qu'elle est exerçable à tout moment entre l'émission et la maturité.
Les options traitées sur les échanges sont souvent de type américain, alors que les options OTC sont souvent européennes. Ces options américaines n'ont pas de formule analytique sous le modèle de Black-Scholes, bien que diverses approximations et méthodes numériques permettent d'approximer un prix (par exemple, l'arbre binomial de Cox-Ross-Rubinstein).
La différence en termes d'exercice de l'option impose certaines relations entre le prix d'une option européenne et le prix d'une option américaine (Fig.5).
Du fait de la possibilité d'exercice à n'importe quel moment, les conditions suivantes sont imposées par l'absence d'opportunité d'arbitrage :

$$C(K,T) \geq (S-K)^+$$

$$P(K,T) \geq (K-S)^+$$

$$C(K,T) \geq C^{EUR}(K,T)$$

$$P(K,T) \geq P^{EUR}(K,T)$$

En outre, nous pouvons étudier des majorants et minorants des prix d'options américaines : (nous supposons dans cette partie que l'actif ne verse pas de dividendes)

$$C(K,T) \leq S$$

$$P(K,T)$$

$$S - Ke^{-rT} \leq C(K,T) \rightarrow\ C(K,T) = C^{EUR}(K,T)$$

Pour prouver ce dernier point, il suffira de considérer un portefeuille contenant un long Call européen, un short actif, et Ke^{-rT} de cash déposé à un taux r. Si on a $S - Ke^{-rT} > C^{EUR}(K,T)$, le portefeuille aurait une valeur initiale négative et une valeur finale positive ou nulle, d'où l'arbitrage (raisonnement similaire pour les deux autres points).
On constate alors qu'en l'absence de dividendes, le Call américain et le Call européen sont identiques. Ce n'est cependant pas le cas pour les Puts.
On en déduit alors une parité Call-Put pour les options américaines : (sans dividendes)

$$S - K \leq C(K,T) - P(K,T) \leq\ S - Ke^{-rT}$$

Les options américaines ne sont alors pas soumises à la même équation que les options européennes.
Aussi, on va noter f la frontière d'exercice optimal. Par exemple, pour un Call américain, on va exercer l'option si $S(t) > Sf(t)$. Pour un Put américain, on exerce si $S(t) < Sf(t)$. Cette frontière n'est pas connue a priori, et transforme le problème en inégalité différentielle.
On déduit alors de nouvelles conditions pour les dérivés de type américain :
Si l'exercice est optimal, on a :

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2(t,S)S^2\frac{\partial^2 V}{\partial S^2} + (r-q)S\frac{\partial V}{\partial S} - rV < 0$$

Sinon :

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2(t,S)S^2\frac{\partial^2 V}{\partial S^2} + (r-q)S\frac{\partial V}{\partial S} - rV = 0$$

Aussi, on a continuité du dérivé et de son prix sur la frontière :

$$V(t,\ Sf(t)) = Payoff(Sf(t))$$

$$\frac{\partial V}{\partial S}(t,\ Sf(t)) = \frac{\partial Payoff}{\partial S}(Sf(t))$$

Ces conditions peuvent se réécrire sous la forme :

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2(t,S)\,S^2\frac{\partial^2 V}{\partial S^2} + (r-q)\,S\frac{\partial V}{\partial S} - rV \leq 0$$

$$V \geq Payoff$$

$$(V - Payoff)\left(\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2(t,S)\,S^2\frac{\partial^2 V}{\partial S^2} + (r-q)\,S\frac{\partial V}{\partial S} - rV\right) = 0$$

C'est un problème dit de complémentaire linéaire.
Les conditions de Dirichlet aux limites pour la résolution numérique sont différentes et reflètent la possibilité d'exercice :

	Call	Put
$S \to 0$	0	$K - S$
$S \to +\infty$	$S - K$	0

La résolution de cette EDP avec la méthode explicite est très simple. A chaque pas de temps, on applique la formule donnée précédemment pour les options européennes, puis on remplace les valeurs obtenues :

$$V(S,\ t) = \max\left(Payoff(S),\ V(S,\ t)\right) \tag{5}$$

Pour le schéma de résolution implicite, on utilise l'algorithme de Projected Successive Over Relaxation (PSOR). Il est similaire à l'algorithme SOR, auquel on applique la condition (5) à chaque itération.
Soient, en reprenant les notations précédentes et en posant :

$$A = \begin{pmatrix} \beta^*(1) & \gamma^*(1) & 0 & \dots & \dots & 0 \\ \alpha^*(2) & \dots & \dots & & & \dots \\ 0 & \dots & \dots & \dots & & \dots \\ \dots & & \dots & \dots & \dots & 0 \\ \dots & & & \dots & \dots & \gamma^*(N-1) \\ 0 & \dots & \dots & 0 & \alpha^*(N) & \beta^*(N) \end{pmatrix}, X = \begin{pmatrix} V(1,\ n+1) \\ \dots \\ \\ \\ \dots \\ V(N, n+1) \end{pmatrix}$$

$$B = \begin{pmatrix} V(1,\ n) \\ \dots \\ \\ \\ \dots \\ V(N,n) \end{pmatrix} - \begin{pmatrix} \alpha^*(1)\,Dirichlet(0, n+1) \\ 0 \\ \dots \\ \dots \\ 0 \\ \gamma^*(N)\,Dirichlet(\infty, n+1) \end{pmatrix}$$

Soit X^0 le vecteur X à l'état initial (on peut prendre la valeur du dérivé à la date précédente), une itération consiste à prendre à l'état $p+1$:

$$X_i^{p+1} = \max\left(Payoff,\ (1-\omega)\,X_i^p + \frac{\omega}{A_{ii}}\left(B_i - \sum_{j<i} A_{ij}X_j^{p+1} - \sum_{j>i} A_{ij}X_j^p\right)\right)$$

On itère alors jusqu'à convergence du système (on peut considérer $\left|\left|X_i^{p+1} - X_i^p\right|\right|$ comme condition d'arrêt).
Le choix du paramètre ω influe grandement sur la vitesse de convergence et la stabilité du système. Un $\omega < 1$ est dit de sous-relaxation, $1 < \omega < 2$ de sur-relaxation. Quand $\omega > 2$ ou $\omega < 0$, il y a divergence de la méthode.
De manière empirique, 6 itérations suffisent à la convergence du système avec $\omega = 1.3$, comme vu en annexe (Fig.6, Fig.7). Ce paramètre ω optimal varie cependant suivant les données de marché $\sigma(S,t), r, q$ ainsi que les paramétres de la grille $\Delta t, \Delta S, S$.

La littérature donne $\omega^{opt} = \frac{2}{1+\sqrt{1-\rho^2}}$ avec ρ rayon spectral de la matrice A suscitée. Le problème est que le calcul de ce rayon spectral doit se faire à chaque pas de temps - ce qui aurait un impact trop lourd sur les performances. On pourrait cependant utiliser ω^{opt} dans le cas du modèle Black-Scholes constant, calculé une seule fois au début du calcul.
On peut également remarquer que le calcul de X_i^{p+1} fait intervenir X_{i-1}^{p+1}, du fait de la nature itérative du SOR. En l'état, il n'est donc pas parallélisable sur un GPU.
Si on considère cependant $X_i^{p+1} \approx X_{i-1}^{p+1}$, on va diviser le problème en deux, suivant les indices pairs et impairs. Aussi, sous cette approximation, on peut calculer en parallèle les X_{2i}^{p+1} d'une part, et les X_{2i+1}^{p+1} d'autre part, puisque le calcul de X_{2i}^{p+1} ne fait pas intervenir X_{2j}^{p+1}, $j \neq i$ (même raisonnement pour les indices impairs).
Cette optimisation, implémentée dans le Pricer, est connue sous le nom de Red-Black Projected Successive Over Relaxation, et permet alors une grande parallélisation du problème. Cela reste cependant une hypothèse forte sur l'option - en particulier sur ses Grecs : le système peut diverger dans certains cas si θ ou Γ n'est pas borné. Par exemple, dans le cas d'une option vanille avec $t \to 0$ et $S = K$, on a $\theta \to \infty$, $\Gamma \to \infty$, et l'algorithme de PSOR diverge parfois.
Enfin, l'implémentation reste délicate et globalement lente par rapport au schéma explicite américain.

5 Conclusion

Lors de cette étude, nous avons pu implémenter trois techniques de résolution de l'EDP de Black-Scholes à l'aide de l'algorithme PSOR (résolution implicite pour les options américaines), l'algorithme de Parallel Cyclic Reduction (résolution implicite pour options européennes) et l'algorithme de résolution explicite. Nous avons alors pu étudier les conditions de stabilité, la convergence et la rapidité de tels algorithmes par rapport à un CPU.
Nous observons alors que l'utilisation d'un GPU pour les problèmes à deux dimensions (le sous-jacent et le temps) est difficilement justifiable.
En effet, pour la résolution explicite, le nombre de pas de sous-jacent est limité si on veut que le système reste stable : ce faible nombre ne permet pas au GPU de se démarquer suffisamment du CPU, notre nVidia GTX670 n'étant plus rapide qu'un code mono-thread executé sur l'AMD FX qu'à partir de 75 pas, deux fois plus rapide à partir de 150 pas, et trois fois plus rapide à partir de 225 pas. Or, le AMD FX possède 4 coeurs... et le schéma explicite ne permet pas de monter beaucoup plus haut en nombre de pas de sous-jacent.
Un autre problème réside dans le fait que notre implémentation GPU intègre la boucle temporelle à l'intérieur du kernel de résolution de l'EDP (sinon, c'est le CPU qui synchronise à chaque pas de temps). Or, la synchronisation du GPU ne peut être faite de manière simple qu'au niveau d'un bloc, d'où la limitation à un seul bloc de threads. Ceci peut avoir un impact pour la résolution implicite, on pourrait cependant penser à une synchronisation entre blocs en utilisant une variable globale atomique. CUDA 9 permettra nativement cette synchronisation au niveau du GPU.
Enfin, l'algorithme SOR/PSOR présente l'inconvénient d'être instable en cas de dérivées partielles infinies. C'est malheureusement ce qu'on observe avec une option vanille au voisinage du Strike et de la maturité : $\Gamma, \theta \to \infty$. On doit alors conserver des pas de temps petits, ou bien resserrer la grille autour de (K, t_0)
L'intérêt du GPU serait plus marqué dans le cas d'options avec plusieurs sous-jacents (Quanto, Rainbow, Basket Options par exemple), mais il faut cependant garder en tête que le temps de calcul est proportionnel au nombre de points où est discretisée l'EDP : au delà de 2 sous-jacents, on travaillera plutôt avec Monte-Carlo.
L'utilisation du GPU peut également être justifiée si on décide de pricer les options par batch (la GTX670 possédant 7 SM, et nos algorithmes ne s'exécutant que sur 1 SM, on pourrait calculer 7 options simultanément - et ainsi tirer un avantage significatif sur le CPU !)

Résolution explicite, option américaine, #S = 150			
Pas de temps	CPU (ms)	GPU (ms)	Gain
10000	21.06	10.78	1.95
20000	41.56	21.07	1.97
30000	62.26	31.47	1.98
40000	83.64	41.94	1.99
50000	104.76	52.22	2.01
60000	125.62	62.5	2.01
70000	146.3	72.86	2.01

Résolution explicite, option américaine, #T = 70000			
Pas d'action	CPU (ms)	GPU (ms)	Gain
50	50.2	68.62	0.73
75	72.84	68.98	1.06
100	95.4	67.44	1.41
125	118.26	71.76	1.65
150	141.44	70.58	2.00
200	185.06	67.72	2.73
250	230.94	71.3	3.24

Résolution implicite, option européenne, #S = 1024			
Pas de temps	CPU (ms)	GPU (ms)	Gain
1000	29.43	4.18	7.04
1500	44.05	6.08	7.25
2000	59.11	8.05	7.34
2500	74.43	9.93	7.50

Résolution implicite, option européenne, #T = 2500			
Pas d'action	CPU (ms)	GPU (ms)	Gain
614	44.65	8.58	5.20
717	51.96	8.62	6.03
819	59.48	8.98	6.62
1024	74.3	9.36	7.94

6 Annexe : Configuration technique

CPU	AMD FX-8320 (Piledriver) 4.55 GHz, 4 Cores/8 Threads, 2.2 GHz CPU/NB 8 MB L2, 8MB L3
RAM	8 GB DDR3 2200 MHz CL10-12-11
GPU	nVidia GTX670 (Kepler, GK104) 1015 MHz Core, 1702 MHz Memory, 1080 MHz Boost 2.73 – 2.90 TFLOPS 1344 CUDA Cores, 7 SM 2048 MB GDDR5, 256-bit bus
Versions Logicielles	Microsoft Windows 10 64 bits Microsoft Visual Studio 2015 Update 1 nVidia CUDA 8.0
Options du compilateur	–machine 64 sm=30 compute=30 -use_fast_math /Ox

7 Annexes : Graphiques

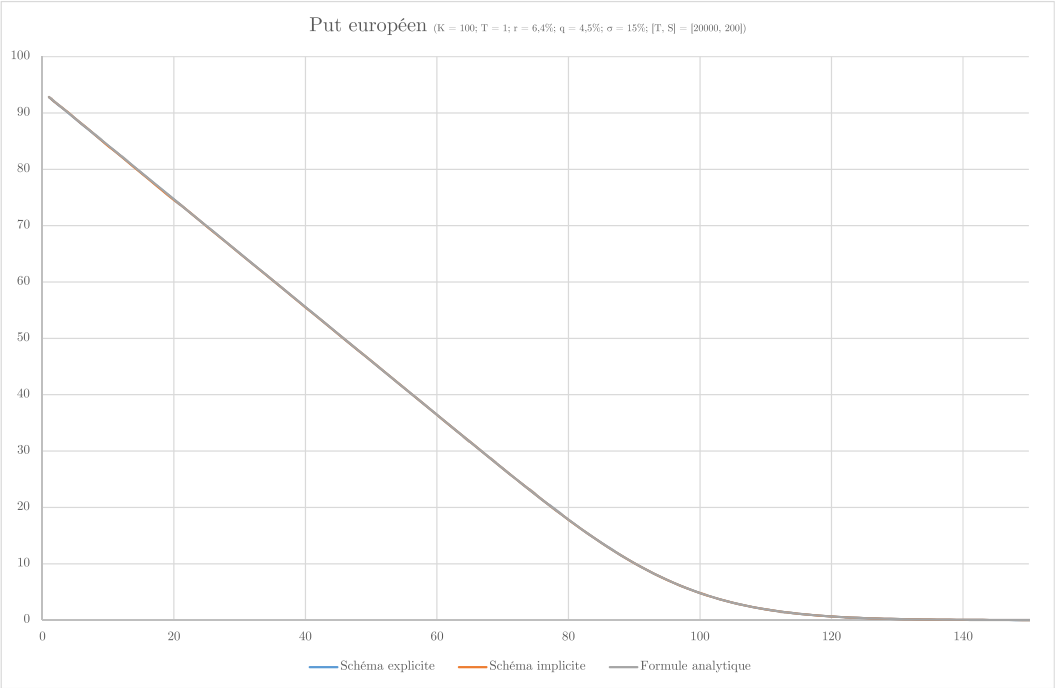

FIGURE 1 – Comparaison du prix d'un Put européen selon la méthode de résolution. L'abscisse représente le Spot, l'ordonnée le prix de l'option.

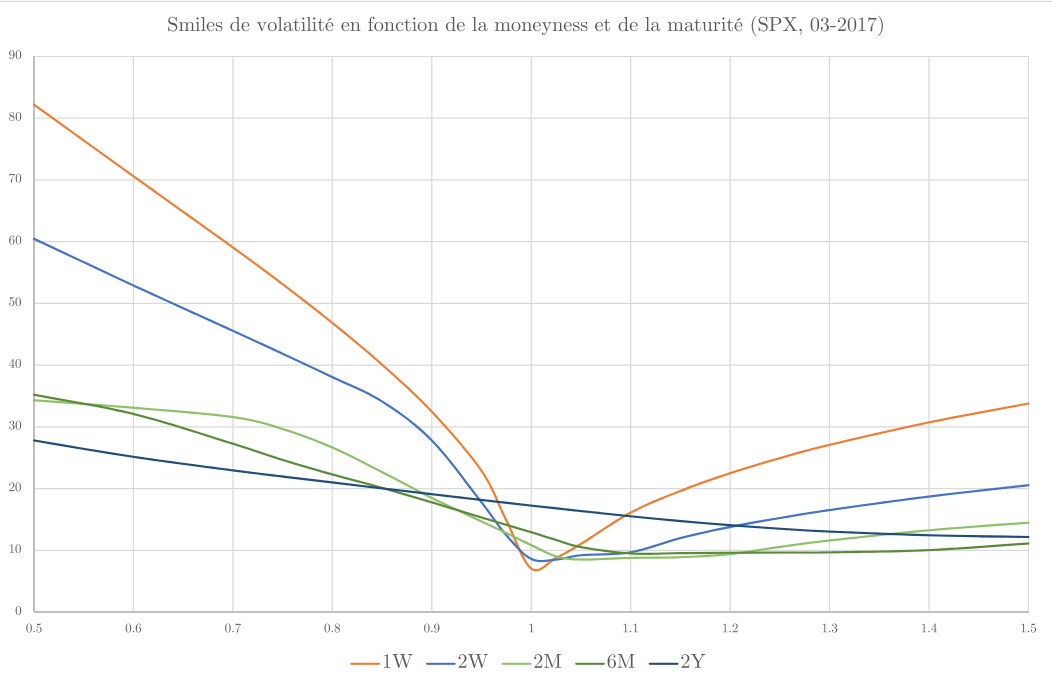

FIGURE 2 – Smiles de volatilité Black-Scholes pour plusieurs maturités sur l'indice SPX (S&P500). L'abscisse représente le Strike relatif (divisé par le Spot), l'ordonnée la volatilité Black-Scholes.

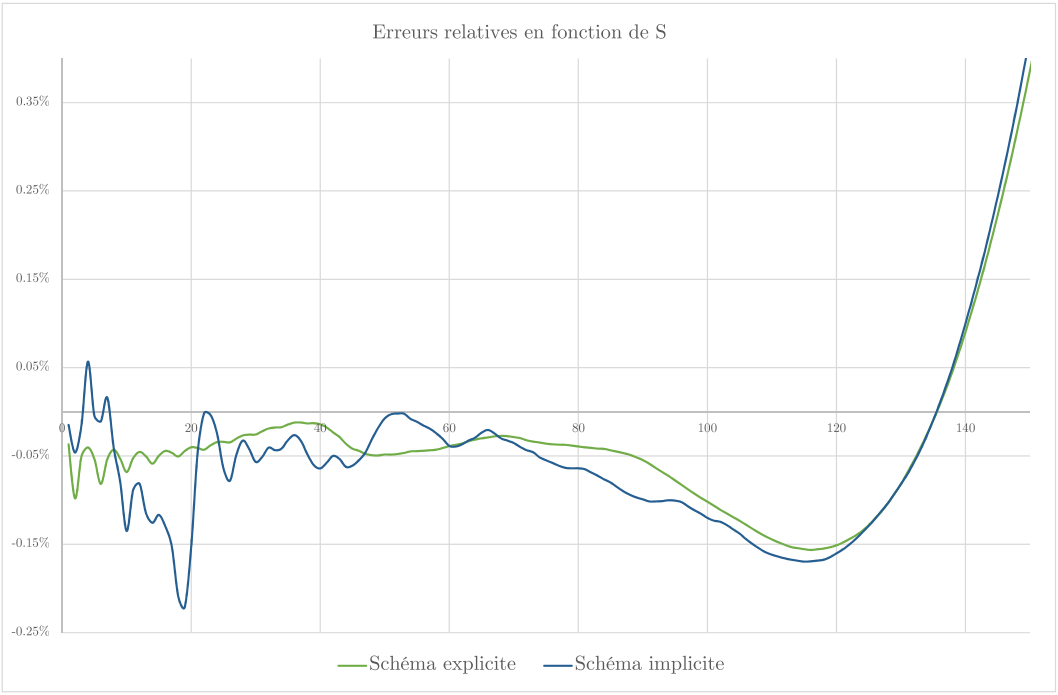

FIGURE 3 – Evolution de l'erreur relative d'une option européenne en fonction du Spot. L'erreur relative, en ordonnées, est : $\frac{P_{PDE}-P_{BS}}{P_{BS}}$ où P_{BS} est le prix obtenu en utilisant la formule de Black-Scholes.

FIGURE 4 – Evolution de l'erreur absolue d'une option européenne en fonction du Spot. L'erreur absolue, en ordonnées, est : $P_{PDE} - P_{BS}$.

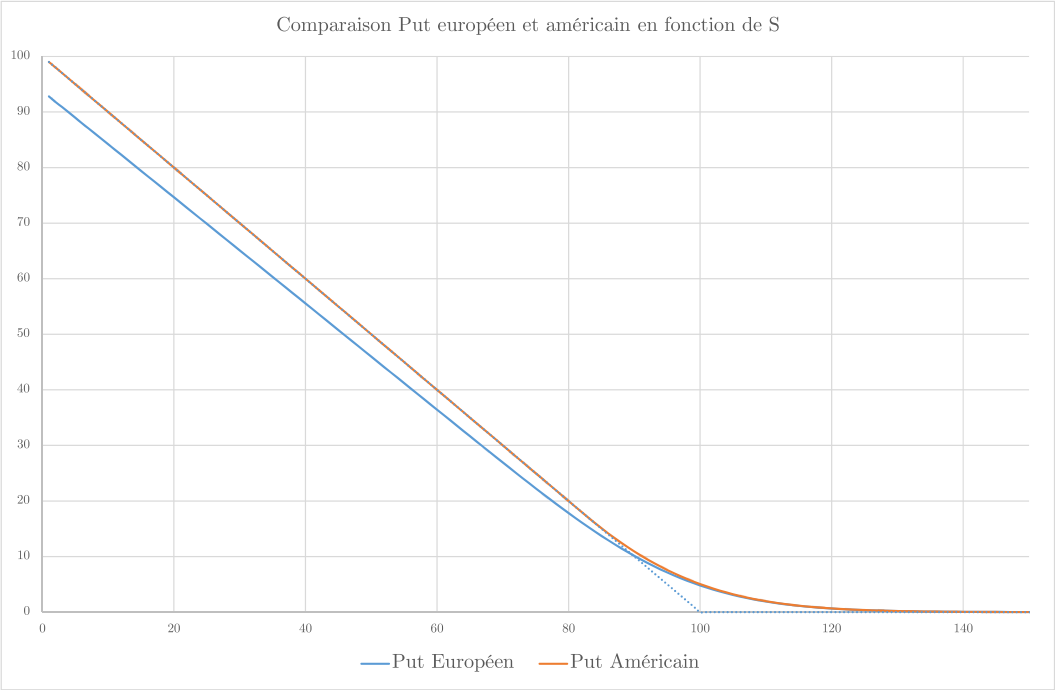

FIGURE 5 – Comparaison des prix du Put européen et américain en fonction du Spot. La valeur intrinsèque $(S-K)^+$ est tracée en pointillés, c'est un minorant du prix de l'option américaine, par non-arbitrage.

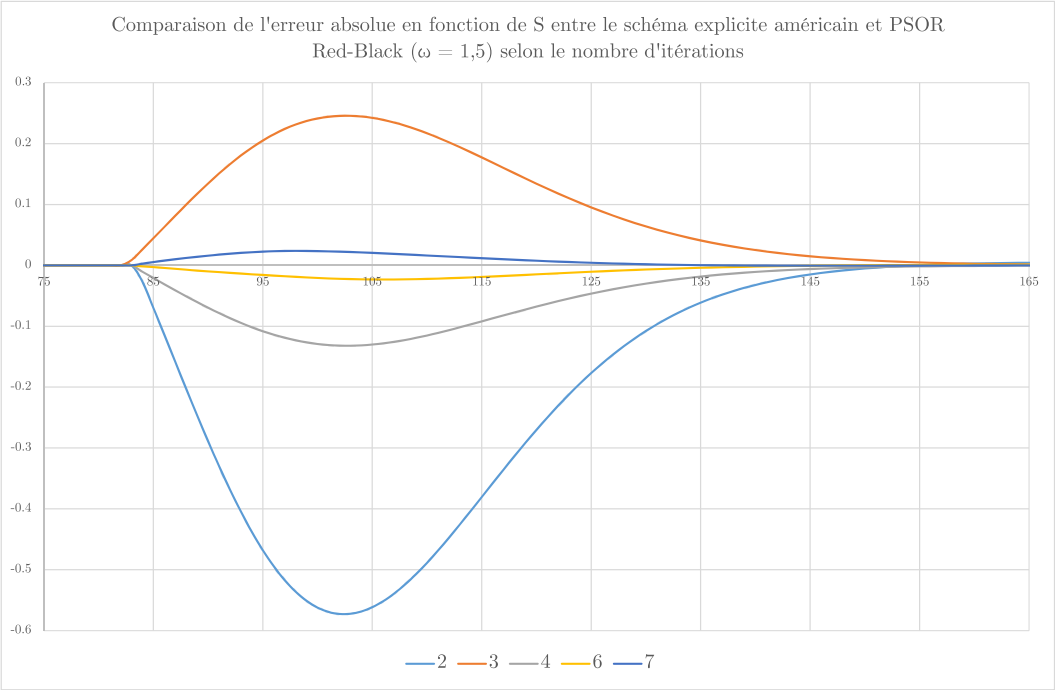

FIGURE 6 – Evolution de l'erreur absolue d'une option américaine en fonction du Spot entre le schéma implicite PSOR Red-Black et le schéma explicite. On fixe $\omega = 1.5$, on fait varier le nombre d'itérations entre 2 et 7.

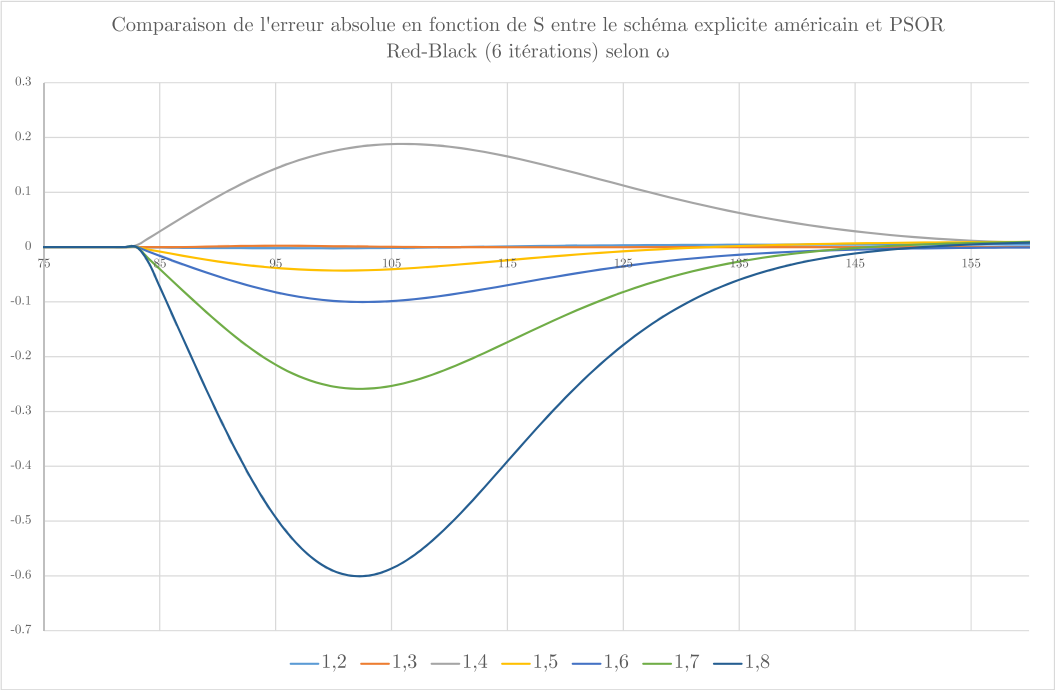

FIGURE 7 – Evolution de l'erreur absolue d'une option américaine en fonction du Spot entre le schéma implicite PSOR Red-Black et le schéma explicite. On fixe le nombre d'itérations à 6, on fait varier le paramètre de relaxation ω.