

Rapport de Projet Programmation Avancée

Résolution de l'Équation de Black-Scholes par Différences Finies

MATHIAS LE BOUEDEC - LILOU MALFOY

3 janvier 2026

Table des matières

1	Introduction	2
2	Modélisation mathématique	2
2.1	La dynamique du sous-jacent (Modèle de Marché)	2
2.2	L'Équation de Black-Scholes Complète	2
2.3	L'Équation Réduite	2
2.4	Conditions initiales et aux bords des options	3
2.4.1	Modification de l'énoncé	3
2.4.2	Les conditions initiales et aux bords pour l'EDP réduite	4
3	Méthodes de Résolution Numérique	4
3.1	Discrétisation du domaine spatio-temporel	4
3.2	Résolution implicite pour l'équation de diffusion	5
3.3	Schéma Crank-Nicholson pour l'EDP complète	6
3.4	Résolution de $Ax = d$ avec A tridiagonale : l'algorithme de Thomas	7
4	Implémentation et Architecture	7
4.1	Structure des Classes et Modélisation UML	7
4.2	Difficultés rencontrées et solutions	9
4.2.1	Le drift dans le changement de variable \tilde{S}	9
4.2.2	Interpolation linéaire	9
4.2.3	Remarque : warning dû au mot-clé <code>override</code>	9
5	Affichage des courbes de $C(0, s)$ et conclusion	9
5.1	Analyse des courbes	10
5.1.1	Analyse du Call	10
5.1.2	Analyse du Put	11
5.2	Conclusion	11

1 Introduction

Ce projet de programmation en C++ a pour objectif de calculer le prix d'options (Call et Put) en résolvant l'équation de Black-Scholes. Une option financière est un contrat dérivé donnant à son détenteur le droit, sans obligation, d'acheter (Call) ou de vendre (Put) un actif sous-jacent à un prix d'exercice fixé à l'avance, à une date d'échéance donnée ou avant celle-ci.

2 Modélisation mathématique

2.1 La dynamique du sous-jacent (Modèle de Marché)

Le point de départ est l'hypothèse que le prix de l'actif sous-jacent S suit un **mouvement brownien géométrique**. Sous la probabilité historique, son évolution est décrite par l'Équation Différentielle Stochastique (EDS) suivante :

$$dS_t = \mu S_t dt + \sigma S_t dW_t \quad (1)$$

Où :

- S_t est le prix de l'actif à l'instant t .
- μ est la tendance du prix (drift) supposé constant.
- σ est la volatilité supposé constante.
- W_t est un processus de Wiener (processus aléatoire).

2.2 L'Équation de Black-Scholes Complète

On représente le prix de l'option par la fonction $C(t, S)$ définie sur $[0, T] \times [0, L]$.

En utilisant la formule d'Ito, on se ramène à un problème déterministe sous la forme d'une équation aux dérivées partielles.

$$\frac{\partial C}{\partial t} + rS \frac{\partial C}{\partial S} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 C}{\partial S^2} = rC \quad (2)$$

Avec :

- T : temps terminal (date d'échéance)
- r : taux d'intérêt sans risque

Contrairement aux problèmes d'évolution classiques où l'on connaît l'état initial (en $t = 0$) pour prédire le futur, la valeur de l'option n'est connue avec certitude qu'à sa date d'échéance T (c'est la condition terminale propre à l'option que l'on appelle le **payoff**). L'objectif est donc de remonter le temps de $t = T$ jusqu'à $t = 0$ qui indique le prix du contrat financier.

2.3 L'Équation Réduite

On peut également transformer l'équation de Black-Scholes en une équation de type diffusion, afin de faciliter sa résolution numérique. Cette transformation repose sur une suite de changements de variables.

En réorganisant l'équation (2), on obtient :

$$\frac{\partial C}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 C}{\partial S^2} + \left(r - \frac{1}{2}\sigma^2\right) S \frac{\partial C}{\partial S} - rC = 0$$

Afin d'éliminer la dépendance multiplicative en S dans les coefficients, on introduit la variable logarithmique

$$x = \ln(S),$$

qui donne :

$$S \frac{\partial}{\partial S} = \frac{\partial}{\partial x}.$$

$$S^2 \frac{\partial^2}{\partial S^2} = \frac{\partial^2}{\partial x^2}.$$

et on inverse le temps par

$$\tilde{t} = T - t.$$

On en déduit :

$$\frac{\partial}{\partial t} = -\frac{\partial}{\partial \tilde{t}}.$$

L'équation devient alors :

$$-\frac{\partial C}{\partial \tilde{t}} + \frac{1}{2}\sigma^2 \frac{\partial^2 C}{\partial x^2} + \left(r - \frac{1}{2}\sigma^2\right) \frac{\partial C}{\partial x} - rC = 0$$

Pour supprimer le terme proportionnel à C , on introduit la transformation :

$$\tilde{C} = e^{r\tilde{t}}C.$$

Après substitution, on obtient :

$$\frac{\partial \tilde{C}}{\partial \tilde{t}} - \frac{1}{2}\sigma^2 \frac{\partial^2 \tilde{C}}{\partial x^2} - \left(r - \frac{1}{2}\sigma^2\right) \frac{\partial \tilde{C}}{\partial x} = 0$$

On souhaite alors éliminer le terme de dérivée première pour se ramener à une équation de la chaleur, donc on pose :

$$\tilde{S} = x + \left(r - \frac{1}{2}\sigma^2\right) \tilde{t}.$$

Dans ce nouveau système de variables, l'équation se réduit à :

$$\frac{\partial \tilde{C}}{\partial \tilde{t}} = \frac{1}{2}\sigma^2 \frac{\partial^2 \tilde{C}}{\partial \tilde{S}^2} \quad (3)$$

Par identification, le coefficient de diffusion vaut :

$\mu = \frac{1}{2}\sigma^2$

2.4 Conditions initiales et aux bords des options

2.4.1 Modification de l'énoncé

Pour le call, il y a une erreur dans l'énoncé sur la condition en $s = L$.

En effet, $C(t, L) = Ke^{-r(t-T)}$ est incorrecte car lorsque le prix du sous-jacent s est très élevé (L), on considère que l'on va presque certainement exercer l'option. La valeur du Call tend alors vers $s - K$.

La condition correcte est :

$$C(t, L) = L - Ke^{-r(T-t)}$$

Les autres conditions sont correctes.

2.4.2 Les conditions initiales et aux bords pour l'EDP réduite

Pour l'EDP réduite, on utilise les changements de variables suivants :

$$\tilde{t} = T - t, \quad x = \ln(s) + \left(r - \frac{\sigma^2}{2}\right)\tilde{t}, \quad \tilde{C}(\tilde{t}, x) = e^{r\tilde{t}}C(t, s).$$

Condition initiale :

On pose $\tilde{t} = 0$, ce qui correspond à $t = T$. Le drift est nul, on a donc :

$$x = \ln(s), \quad s = e^x.$$

Pour une option **call** :

$$\tilde{C}(0, x) = \max(0, e^x - K).$$

Pour une option **put** :

$$\tilde{C}(0, x) = \max(0, K - e^x).$$

Conditions aux bords :

Comme la fonction \ln diverge en 0, pour résoudre numériquement le problème, on pose $s_{\min} = 10^{-5}$ et on réalise un maillage sur

$$x \in [x_{\min}, x_{\max}],$$

avec

$$x_{\min} = \ln(s_{\min}), \quad x_{\max} = \ln(L) + \left(r - \frac{\sigma^2}{2}\right)T.$$

— Pour une **option call**

$$\begin{aligned} \tilde{C}(\tilde{t}, x_{\min}) &= 0, & (s \rightarrow 0), \\ \tilde{C}(\tilde{t}, x_{\max}) &= e^{r\tilde{t}}(x_{\max} - Ke^{-r\tilde{t}}), & (s = L). \end{aligned}$$

— Pour une **option put**

$$\begin{aligned} \tilde{C}(\tilde{t}, x_{\min}) &= e^{r\tilde{t}}Ke^{-r\tilde{t}} = K, & (s \rightarrow 0), \\ \tilde{C}(\tilde{t}, x_{\max}) &= 0, & (s = L). \end{aligned}$$

3 Méthodes de Résolution Numérique

3.1 Discréétisation du domaine spatio-temporel

Pour résoudre numériquement les équations de Black-Scholes, il est nécessaire de discréétiser le domaine spatio-temporel. On découpe :

- le temps $[0, T]$ en M intervalles de pas Δt , avec l'indice i ;
- l'espace des prix $[0, L]$ en N intervalles de pas ΔS , avec l'indice j .

On note alors $C(t_i, S_j) = C_j^i$.

Les dérivées sont approchées à l'aide de différences finies, issues du développement de Taylor. Pour une fonction $C(t, S)$:

$$\begin{aligned} C_j^{i+1} &= C_j^i + \Delta t \frac{\partial C}{\partial t} \Big|_{(t_i, S_j)} + O(\Delta t^2) \\ C_{j+1}^i &= C_j^i + \Delta S \frac{\partial C}{\partial S} \Big|_{(t_i, S_j)} + \frac{(\Delta S)^2}{2} \frac{\partial^2 C}{\partial S^2} \Big|_{(t_i, S_j)} + O(\Delta S^3) \end{aligned}$$

On peut alors calculer la valeur future C_j^{i+1} uniquement en fonction des valeurs déjà connues (**schéma explicite**).

$$\begin{aligned}\frac{\partial C}{\partial t} &\approx \frac{C_j^{i+1} - C_j^i}{\Delta t} \\ \frac{\partial C}{\partial S} &\approx \frac{C_{j+1}^i - C_{j-1}^i}{2\Delta S} \\ \frac{\partial^2 C}{\partial S^2} &\approx \frac{C_{j+1}^i - 2C_j^i + C_{j-1}^i}{(\Delta S)^2}\end{aligned}$$

Ce schéma est conditionnellement stable. Si le pas de temps Δt est trop grand par rapport à ΔS , la solution peut diverger numériquement.

Alors que le schéma explicite calcule chaque point du futur indépendamment à partir du passé, le **schéma implicite** lie les points futurs entre eux, ce qui nécessite la résolution d'un système matriciel mais garantit une stabilité numérique totale.

Comparaison pour la dérivée seconde :

- **Explicite** : $\frac{\partial^2 C}{\partial S^2} \approx \frac{C_{j+1}^i - 2C_j^i + C_{j-1}^i}{(\Delta S)^2}$
- **Implicite** : $\frac{\partial^2 C}{\partial S^2} \approx \frac{C_{j+1}^{i+1} - 2C_j^{i+1} + C_{j-1}^{i+1}}{(\Delta S)^2}$

3.2 Résolution implicite pour l'équation de diffusion

Pour l'équation réduite (équation de diffusion) :

$$\frac{\partial \tilde{C}}{\partial \tilde{t}} = \frac{1}{2} \sigma^2 \frac{\partial^2 \tilde{C}}{\partial \tilde{S}^2}$$

On rappelle qu'on a effectué le changement de variable

$$\tilde{t} = T - t.$$

Dans ce nouveau référentiel \tilde{t} , on avance "normalement" de n (connu) vers $n + 1$ (inconnu).

Le schéma implicite s'écrit :

$$\frac{\tilde{C}_j^{i+1} - \tilde{C}_j^i}{\Delta \tilde{t}} = \frac{\sigma^2}{2} \frac{\tilde{C}_{j+1}^{i+1} - 2\tilde{C}_j^{i+1} + \tilde{C}_{j-1}^{i+1}}{(\Delta \tilde{S})^2}.$$

Ce qui donne sous forme matricielle :

$$\begin{pmatrix} 1 + 2\lambda & -\lambda & 0 & \cdots & 0 \\ -\lambda & 1 + 2\lambda & -\lambda & \ddots & \vdots \\ 0 & -\lambda & 1 + 2\lambda & -\lambda & 0 \\ \vdots & \ddots & \ddots & \ddots & -\lambda \\ 0 & \cdots & 0 & -\lambda & 1 + 2\lambda \end{pmatrix} \begin{pmatrix} \tilde{C}_0^{i+1} \\ \tilde{C}_1^{i+1} \\ \tilde{C}_2^{i+1} \\ \vdots \\ \tilde{C}_N^{i+1} \end{pmatrix} = \begin{pmatrix} \tilde{C}_0^i \\ \tilde{C}_1^i \\ \tilde{C}_2^i \\ \vdots \\ \tilde{C}_N^i \end{pmatrix}$$

Avec

$$\lambda = \frac{\sigma^2 \Delta \tilde{t}}{2(\Delta \tilde{S})^2}$$

D'où l'expression de la solution à l'instant futur :

$$A \tilde{\mathbf{C}}^{i+1} = \tilde{\mathbf{C}}^i$$

où A est une **matrice tridiagonale** de taille $N + 1 \times N + 1$.

3.3 Schéma Crank-Nicholson pour l'EDP complète

Le schéma de Crank-Nicolson combine les deux approches en faisant la moyenne entre le schéma explicite (évalué à l'instant i) et le schéma implicite (évalué à l'instant $i + 1$). Cela permet d'obtenir un schéma inconditionnellement stable (comme l'implicite) mais beaucoup plus précis car contrairement aux schémas purement explicites ou implicites qui sont d'ordre 1 en temps ($O(\Delta t)$), Crank-Nicolson est d'ordre 2 ($O(\Delta t^2)$). Mathématiquement, en se plaçant au centre du pas de temps ($t_{i+1/2}$), les erreurs de développement de Taylor du premier ordre s'annulent par symétrie. Cela permet d'obtenir une précision bien supérieure pour un même nombre de pas de temps.

En appliquant cette moyenne directement à l'équation (2) de Black-Scholes, on obtient :

$$\frac{C_j^{i+1} - C_j^i}{\Delta t} + \frac{1}{2} [\text{Diff}_j^i + \text{Diff}_j^{i+1}] = r \frac{C_j^i + C_j^{i+1}}{2}$$

où l'approximation spatiale Diff_j à un instant donné est :

$$\text{Diff}_j = rj \frac{C_{j+1} - C_{j-1}}{2\Delta S} + \frac{1}{2} \sigma^2 j^2 \frac{C_{j+1} - 2C_j + C_{j-1}}{(\Delta S)^2}$$

En regroupant les termes inconnus (instant i) à gauche et les termes connus (instant $i + 1$) à droite, le schéma conduit à un système linéaire tridiagonal :

$$A\mathbf{C}^i = B\mathbf{C}^{i+1}$$

Où :

$$A = \begin{bmatrix} \beta'_0 & \gamma'_0 & 0 & \dots & 0 \\ \alpha'_1 & \beta'_1 & \gamma'_1 & \dots & 0 \\ 0 & \alpha'_2 & \beta'_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \gamma'_{N-1} \\ 0 & \dots & 0 & \alpha'_N & \beta'_N \end{bmatrix}$$

$$\begin{aligned} \alpha'_j &= \frac{j\Delta t}{4\Delta S} (r - \sigma^2 j / (\Delta S)) \\ \beta'_j &= 1 + \frac{\Delta t}{2} (\sigma^2 j^2 / (\Delta S)^2 + r) \\ \gamma'_j &= -\frac{j\Delta t}{4\Delta S} (r + \sigma^2 j / (\Delta S)) \end{aligned}$$

$$B = \begin{bmatrix} \beta_0 & \gamma_0 & 0 & \dots & 0 \\ \alpha_1 & \beta_1 & \gamma_1 & \dots & 0 \\ 0 & \alpha_2 & \beta_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \gamma_{N-1} \\ 0 & \dots & 0 & \alpha_N & \beta_N \end{bmatrix}$$

$$\begin{aligned} \alpha_j &= \frac{j\Delta t}{4\Delta S} (\sigma^2 j / (\Delta S) - r) \\ \beta_j &= 1 - \frac{\Delta t}{2} (\sigma^2 j^2 / (\Delta S)^2 + r) \\ \gamma_j &= \frac{j\Delta t}{4\Delta S} (\sigma^2 j / (\Delta S) + r) \end{aligned}$$

3.4 Résolution de $Ax = d$ avec A tridiagonale : l'algorithme de Thomas

La discréétisation des équations (complète et réduite) de Black-Scholes conduit à la résolution de systèmes linéaires de la forme

$$Ax = d,$$

où la matrice $A \in \mathbb{R}^{N \times N}$ est tridiagonale :

$$A = \begin{bmatrix} b_1 & c_1 & 0 & \dots & 0 \\ a_2 & b_2 & c_2 & \dots & 0 \\ 0 & a_3 & b_3 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & c_{N-1} \\ 0 & \dots & 0 & a_N & b_N \end{bmatrix}.$$

L'algorithme de Thomas est une adaptation de l'élimination de Gauss à ce type de matrices. Il consiste en une élimination avant visant à annuler la sous-diagonale :

$$\begin{aligned} \tilde{b}_1 &= b_1, & \tilde{c}_1 &= \frac{c_1}{b_1}, \\ \tilde{b}_i &= b_i - a_i \tilde{c}_{i-1}, & \tilde{c}_i &= \frac{c_i}{\tilde{b}_i}, & i &= 2, \dots, N-1, \end{aligned}$$

et à modifier le second membre :

$$\tilde{d}_1 = d_1, \quad \tilde{d}_i = d_i - a_i \frac{\tilde{d}_{i-1}}{\tilde{b}_{i-1}}, \quad i = 2, \dots, N.$$

Le système est alors triangulaire supérieur et la solution est obtenue par substitution arrière :

$$x_N = \frac{\tilde{d}_N}{\tilde{b}_N}, \quad x_i = \frac{\tilde{d}_i - c_i x_{i+1}}{\tilde{b}_i}, \quad i = N-1, \dots, 1.$$

Cet algorithme a une complexité linéaire $O(N)$ et un coût mémoire minimal, ce qui en fait une méthode particulièrement adaptée à la résolution répétée des systèmes issus des schémas implicites.

4 Implémentation et Architecture

Le projet est réalisé en C++ et respecte une architecture orientée objet, favorisant le polymorphisme et la séparation claire des responsabilités. Cette modularité permet de distinguer les modèles financiers, les solveurs numériques et l'interface graphique.

4.1 Structure des Classes et Modélisation UML

- **Classe abstraite Option** : Elle constitue le socle des produits financiers et encapsule les paramètres du contrat (K, L, r, T). Elle définit l'interface pour les conditions spécifiques :
 - `payoff(double S)` : calcule la valeur intrinsèque à l'échéance.
 - `boundary_condition_low/high(double L, double t)` : définissent les conditions aux bords. L'argument L est nécessaire pour avoir la bonne condition en $s = S_{max}$ pour l'option Call.
 - *Polymorphisme* : les classes dérivées `Call` et `Put` surchargent ces méthodes, permettant au solveur de traiter différentes options de manière transparente.
- **Classe abstraite EDP** : Représente le lien entre l'option et les paramètres de marché, notamment la volatilité σ . Elle sert de base de données pour les solveurs, fournissant un accès aux caractéristiques du modèle financier via des méthodes telles que `getOption()` et `getSigma()`.

- **Hiérarchie Solver (moteur de calcul)** : Cette famille de classes gère la résolution numérique par différences finies sur un maillage (N, M).
 - **Solver** (classe de base) : contient la matrice des résultats v_- , le maillage spatial S_- et le maillage temporel t_- . Elle implémente l'algorithme de Thomas (`thomas_algorithm`) pour résoudre les systèmes tridiagonaux.
 - **Crank_Nicolson** : implémente le schéma de Crank-Nicolson pour l'EDP complète de Black-Scholes.
 - **Implicite_solver** : dédié à l'EDP réduite (équation de la chaleur). Cette classe intègre des mécanismes spécifiques :
 - `change_variable()` et `reverse_variable()` : gèrent la transformation vers le domaine logarithmique et le retour au domaine réel, ajustant également la position exacte de S pour garantir une superposition correcte des courbes à $t = 0$.
 - `get_value_at_S(double s_target, int time_step)` : méthode d'interpolation linéaire cruciale, qui permet d'évaluer le prix de l'option pour une valeur cible de S , indépendamment du maillage logarithmique utilisé pour le calcul.
- **Classe Sdl** : Responsable de l'interface graphique. Elle est entièrement séparée de la logique métier et utilise `draw_curve()` pour projeter les vecteurs de résultats (`std::vector<double>`) sur une fenêtre graphique, facilitant la visualisation et la comparaison des courbes de prix et d'erreur.

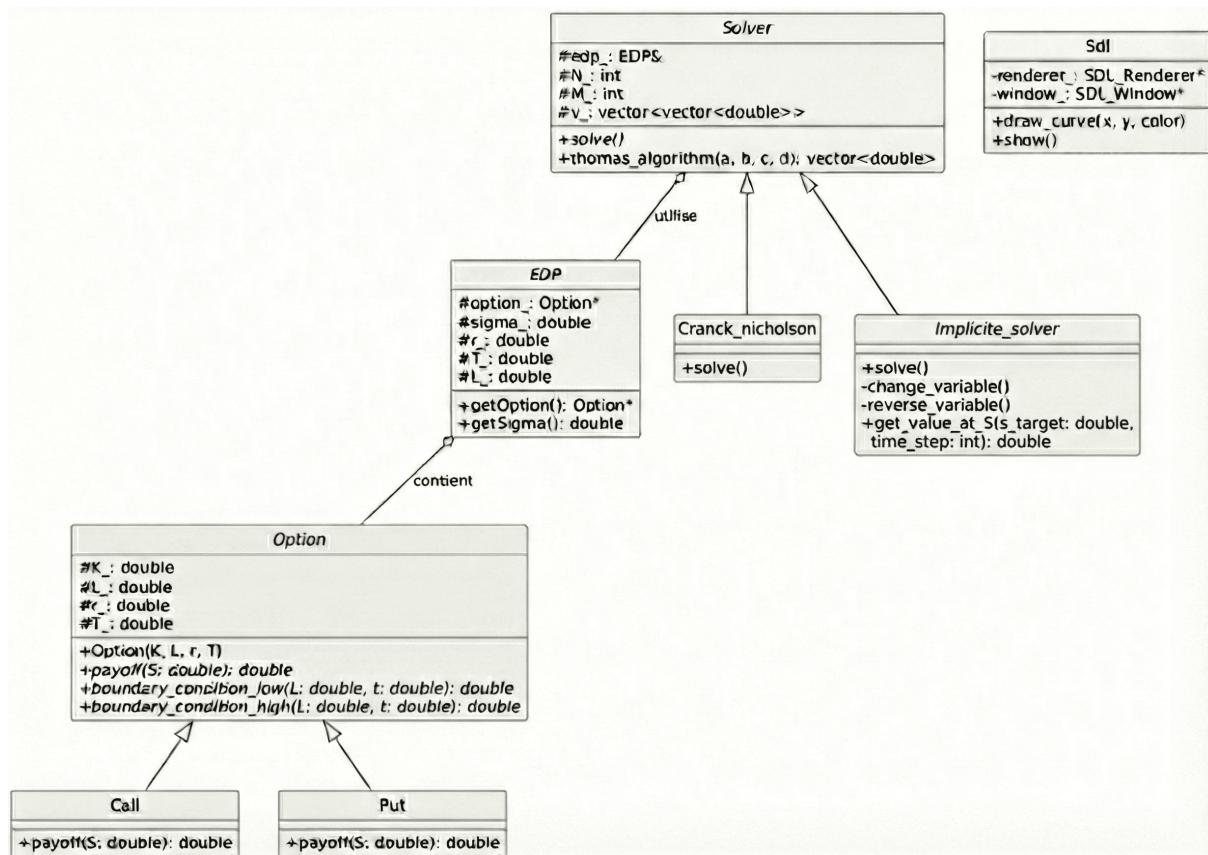


FIGURE 1 – Diagramme UML

4.2 Difficultés rencontrées et solutions

4.2.1 Le drift dans le changement de variable \tilde{S}

Lors de la résolution de l'EDP réduite, les prix des options sont calculés sur une grille transformée

$$\tilde{S} = \ln(S) + \text{drift} \times (T - t), \quad \text{avec drift} = r - \frac{\sigma^2}{2}.$$

Pour comparer les résultats avec l'EDP complète ou visualiser le prix de l'option en fonction de S , il est nécessaire de revenir à la variable originale S et de corriger la valeur de l'option.

La fonction `reverse_variable` effectue deux opérations clés :

1. Conversion des valeurs de \tilde{C} en prix d'option C :

$$C(t, S) = \tilde{C}(\hat{t}, \tilde{S}) e^{-r\hat{t}}.$$

2. Recalcul de la position exacte de S à chaque point de la grille pour le pas final ($t = 0$) :

$$S = \exp(\tilde{S} - \text{drift} \cdot T),$$

ce qui corrige l'effet de glissement du prix de l'actif le long de la grille transformée.

De plus, le vecteur de temps est inversé afin que l'indice $j = 0$ corresponde toujours au temps présent, garantissant ainsi que les courbes de prix puissent être superposées correctement pour l'analyse et la comparaison entre méthodes.

4.2.2 Interpolation linéaire

La principale difficulté provenait de l'incohérence entre les maillages spatiaux des deux solveurs : l'EDP complète est résolue sur une grille linéaire en S , tandis que l'EDP réduite utilise une grille logarithmique issue du changement de variable. Une comparaison directe par indice menait donc à des erreurs numériques artificielles.

Pour corriger ce problème, les comparaisons sont effectuées à prix S fixé au moyen d'une interpolation linéaire sur la grille de l'EDP réduite. Cette approche permet d'évaluer le prix de l'option pour une valeur cible de S , indépendamment du maillage utilisé, et garantit une superposition cohérente des courbes à $t = 0$.

L'interpolation linéaire constitue ainsi l'élément clé assurant une comparaison fiable entre les deux méthodes numériques.

4.2.3 Remarque : warning dû au mot-clé `override`

Le mot-clé `override` a été retiré. Il provoquait des warnings car la compilation demandait d'activer le standard C++11.

5 Affichage des courbes de $C(0, s)$ et conclusion

Pour les applications numériques, on utilisera les valeurs suivantes :

- $T = 1$
- $r = 0.1$
- $\sigma = 0.1$
- $K = 100$
- $L = 300$
- $N = 1000$
- $M = 1000$

On fixe également $s_{\min} = 10^{-5}$ pour la transformation logarithmique.

5.1 Analyse des courbes

5.1.1 Analyse du Call

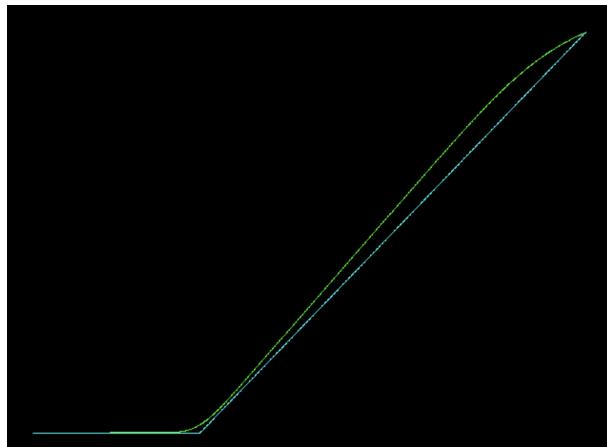


FIGURE 2 – Courbes qui approche $C(0, s)$ pour le call : en vert méthode de crank-nicholson sur l'edp complète et en cyan méthode implicite sur l'edp réduite.

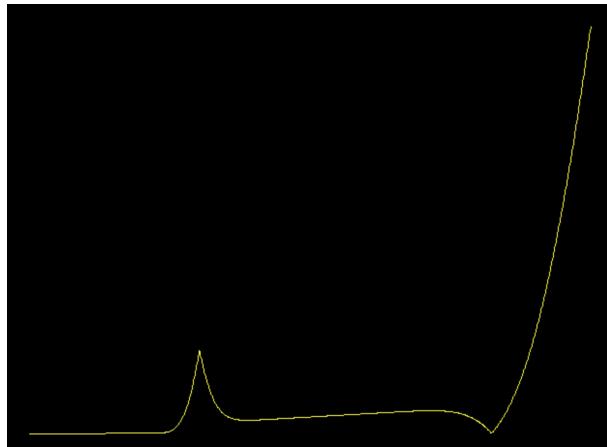


FIGURE 3 – Erreur entre les méthodes pour le call

La figure 2 montre une excellente adéquation entre les deux méthodes de valorisation. L’arrondi observé de la courbe à $t = 0$ par rapport au payoff terminal confirme la bonne résolution du terme de diffusion.

Un pic local de l’erreur apparaît au niveau du strike $s = K$. Il est dû à la discontinuité de la dérivée première du payoff, qui génère une erreur de discréétisation maximale.

Enfin, l’erreur augmente à l’approche de la borne supérieure $s = L$. Ce phénomène provient du maillage logarithmique de l’EDP réduite : le point x_{\max} ne correspond pas exactement à L après transformation, et ce décalage est amplifié pour les grandes valeurs de s .

5.1.2 Analyse du Put

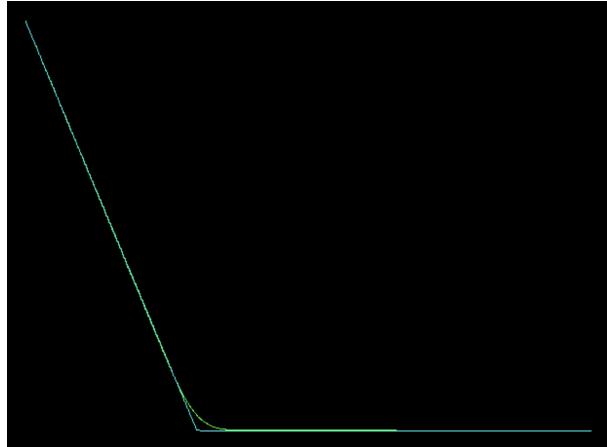


FIGURE 4 – Courbes qui approche $C(0, s)$ pour le put : en vert méthode de crank-nicholson sur l'edp complète et en cyan méthode implicite sur l'edp réduite.

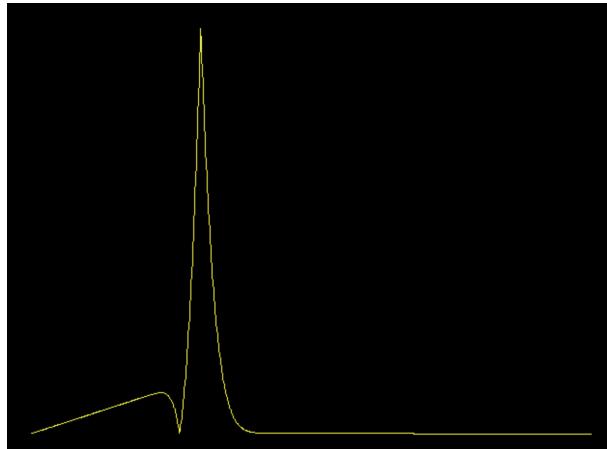


FIGURE 5 – Erreur entre les méthodes pour le put

Les résultats obtenus pour le Put sont en accord avec la théorie de Black–Scholes, les deux solveurs convergeant vers une même solution lissée.

La seule source d'erreur significative est localisée au niveau du strike $K = 100$. L'étroitesse de ce pic confirme la bonne performance de l'algorithme de Thomas pour l'inversion des matrices tridiagonales.

5.2 Conclusion

Les résultats montrent une très bonne concordance entre la méthode de Crank-Nicholson appliquée à l'EDP complète et le schéma implicite utilisé sur l'EDP réduite. Les deux méthodes convergent vers la solution de Black Scholes et reproduisent correctement le lissage du payoff.

La méthode de Crank-Nicholson est toutefois légèrement plus précise. Cela s'explique par son ordre deux en temps, qui réduit l'erreur de discréétisation, ainsi que par son caractère centré, limitant la diffusion numérique au voisinage du strike.