Ecole Nationale de la Statistique
et de l'Administration Économique
Paris

Méthode de Black-Scholes + Méthode de Monte-Carlo

Projet de Programmation C++

Samy Mekkaoui Eros Giovanetto Augustin Mangion

Sommaire

1 Introduction

Le modèle de Black-Scholes constitue l'un des paradigmes de la finance moderne. Il permet de calculer la valeur théorique de certains types d'options à l'aide de 5 paramètres :

— le prix initial du sous-jacent S_0
— le temps avant Maturité T
— la valeur du strike K
— le taux d'intérêt sans risque r
— la volatilité σ

Sous ces conditions, on peut retrouver la formule de Black-Scholes qui nous donne directement la valeur théorique d'un Call vanille européen :

$$C(\mathrm{S}_0,K,r,T,\sigma)=S_0\Phi(d_1) - K\,\mathrm{e}^{-rT}\Phi(d_2)$$

avec :

1. la fonction de répartition de la loi normale centrée réduite Φ
2. $d_1 = \frac{1}{\sigma\sqrt{T}}(\ln(\frac{S_0}{K}) + (r + \frac{\sigma^2}{2})T)$
3. $d_2 = d_1 - \sigma\ \sqrt{T}$

Par ailleurs, on peut retrouver la valeur d'un Put vanille européen possédant les mêmes caractéristiques via la Parité Call-Put et on obtient alors :

$$P(\mathrm{S}_0,K,r,T,\sigma)=\text{-}\mathrm{S}_0\Phi(-d_1) + K\,\mathrm{e}^{-rT}\,\Phi(-d_2)$$

Il existe également de nombreuses autres méthodes qui permettent de calculer les prix des options mais celle-ci constitue l'une des plus simples même si en réalité la volatilité n'est pas un paramètre simple à évaluer car selon le type d'option sur actions, elle a tendance à varier c'est ce qu'on appelle : le smile de volatilité.

Une autre méthode très classique se basant sur le modèle de Black-Scholes est l'utilisation des procédés Monte-Carlo. En effet, lorsque l'option n'est pas une option vanille (c'est à dire lorsqu'on a besoin de la trajectoire du cours du sous-jacent dans son ensemble et non seulement de sa valeur à maturité), on ne peut pas utiliser les résultats présentés ci-dessus et il est alors impossible de pouvoir calculer la vraie valeur de l'option en se basant sur les formules ci-dessus. Cependant, le procédé Monte Carlo va lui chercher à simuler différentes valeurs de trajectoire du sous-jacent et à faire une moyenne actualisée de ces valeurs, ce qui donnera une valeur approchée de la valeur de l'option. Ce résultat est issu du fait que dans le modèle de Black-Scholes , le sous-jacent est censé suivre une loi log-normale.

Notons S_t la valeur du sous-jacent à l'instant t. Alors la théorie Black-Scholes nous dit :

$$S_t = S_0\,\mathrm{e}^{(\mu - \frac{\sigma^2}{2})t + \sigma * \sqrt{dt}\Phi}$$

avec :

1. la tendance μ
2. la loi normale centrée réduite Φ

On obtient alors la valeur actualisée en multipliant par le *"discount factor"* e^{-rT}

Nous allons également nous intéresser aux grecques qui constituent les instruments de base de la gestion financières des options. Il en existe 5 que nous allons introduire ci-dessous.

Notons C le prix de l'option. Alors, on a :

- le Δ qui s'écrit : $\frac{\partial C}{\partial S}$ et qui correspond à la variation de valeur de l'option pour un faible mouvement du sous-jacent
- le Γ qui s'écrit : $\frac{\partial^2 C}{\partial S^2}$ et qui correspond à la variation de la valeur du Delta, c'est à dire si la variation est plus ou moins importante suite à faible mouvement du sous-jacent.
- le Θ qui s'écrit : $\frac{\partial C}{\partial T}$ et qui correspond à la variation de la valeur de l'option lorsque sa maturité résiduelle diminue
- le ν qui s'écrit : $\frac{\partial C}{\partial \sigma}$ et qui correspond à une mesure de la sensibilité de l'action
- le ρ qui s'écrit :$\frac{\partial C}{\partial r}$ et qui correspond à la variation de l'option suite à un mouvement du taux d'intérêt sans risque

Dans le cadre du modèle de Black-Scholes et pour des options vanilles européennes, il existe comme pour les valeurs des Call et Put des formules explicites de ces lettres grecques.

En effet, on peut montrer que dans le cas d'options vanilles européennes :

- le Δ_{Call}=$\Phi(d_1)$ et Δ_{Put}=$\Phi(d_1) - 1$
- le Γ_{Call}=Γ_{Put}=$\frac{\Phi'(d_1)}{S_0\sigma\sqrt{T}}$
- le Θ_{Call}=$\frac{-\phi'(d_1)\sigma S_0}{2\sqrt{T}}$-$rK\,\mathrm{e}^{-rT}\,\Phi(d_2)$ et Θ_{Put}=$\frac{-\phi'(d_1)\sigma S_0}{2\sqrt{T}}$+$rK\,\mathrm{e}^{-rT}\,\Phi(-d_2)$
- le ν_{Call}=ν_{Put}=$\Phi'(d_1)S_0\sqrt{T}$
- le ρ_{Call}=$K\,\mathrm{e}^{-rT}\,T\Phi(d_2)$ et ρ_{Put}= $-K\,\mathrm{e}^{-rT}\,T\Phi(-d_2)$

Ainsi, pour des options vanilles européennes, on a accès à ces valeurs qui permettent de décrire le comportement du prix de l'option si certains paramètres sont modifiés.
Lorsque les options ne seront pas vanilles européennes, on ne pourra pas avoir réutiliser ces formules explicites mais on pourra estimer ces valeurs par méthode de Monte-Carlo

2 Présentation de notre projet ainsi que des méthodes utilisées

2.1 Objectif de notre projet

L'objectif de ce projet est de développer un "pricer" d'options. Il existe de multiples manières d'évaluer des options : l'approche par arbres, par équations aux dérivées partielles. Ici, nous nous concentrons sur l'approche Monte-Carlo. Elle consiste à simuler un grand nombre de réalisations et d'en déduire par la loi des grands nombres une approximation du prix. Nous nous plaçons dans le modèle de Black-Scholes, qui en fonction d'un paramètre de taux d'intérêt et d'un paramètre de volatilité propose une génération de la trajectoire de prix. Nous nous appuyons donc sur ce modèle pour simuler l'évolution du prix d'un sous-jacent fictif et déterminer le prix d'une option dérivée de celui-ci.

2.2 Description de l'architecture générale du programme

Nous avons utilisé une architecture organisée selon des classes dans notre programme : cela va nous permettre de faire de la programmation orientée objet (POO) et d'utiliser le polymorphisme grâce à l'héritage.

Nous allons désormais expliquer les classes que nous avons utilisé dans notre projet :

— La classe *BlackSholesModel*, qui est une classe contenant l'objet *BlackScholesModel* permettant de générer des prix, des trajectoires de prix selon le modèle de Black Scholes. Elle contient tous les paramètres relatifs seulement au modèle de Black Scholes (risk free rate, date, volatility...)

— La classe *ContinuousTimeOption*, qui est une classe virtuelle contenant toutes les options qui sont fonctions continues du temps, en opposition aux modèles de pricing d'options en temps discret (ex : arbre binomial). Cette classe contient les termes du contrat de l'option : la maturité (maturity) et le strike (strike) et la barrière (barrier) (spécifique aux options up and out knock-out)

Les fichiers suivants sont relatifs aux options :

— La classe *PathIndependent*, qui est une classe virtuelle contenant la structure basique des options dont le payoff ne dépend pas de la trajectoire du prix.

— La classe *PathDependent*, qui classe virtuelle contenant la structure basique des options dont le payoff dépend de la trajectoire du prix.

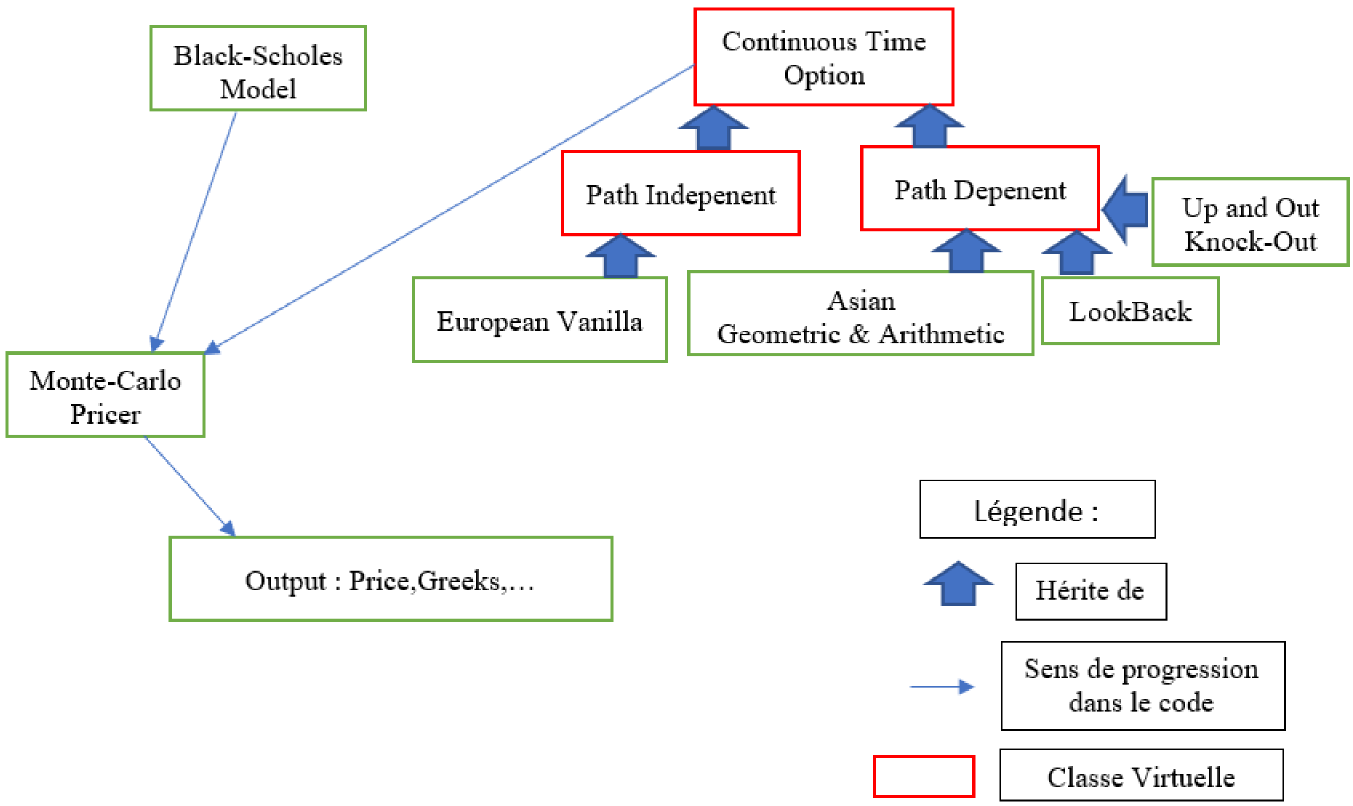
— La classe *MonteCarloPrice*, qui est une classe contenant l'objet MonteCarloPricer permettant d'appliquer la méthode Monte-Carlo afin d'estimer le prix de l'option, de calculer les grecques (avec Finite Difference Method et Monte-Carlo) et l'écart-type utilisé pour construire l'intervalle de confiance à 95% du prix de l'option déterminé par la méthode de Monte-Carlo. On impose le nombre de simulations à 100000 et le nombre d'étapes de calcul à 250.

— Les Classes *EuropeanVanillaCallOption* et *EuropeanVanillaPutOption*, qui contiennent les options vanilles Européennes.

- Les Classes *AsianArithmeticCallOption* et *AsianArithmeticPutOption*, qui contiennent les options Asiatiques Arithmétiques.
- Les Classes *AsianGeometricCallOption* et *AsianGeometricPutOption*, qui contiennent les options Asiatiques Géométriques.
- Les Classes *upandoutknockoutcalloption* et *upandoutknockoutputoption* qui contiennent les options Up and Out Knock-Out.
- Les Classes LookBackCallOption et *LookBackPutOption* qui contiennent les options LookBack

Enfin, nous avons la fonction Main qui nous a permis de coder l'interface facilitant l'utilisation de notre programme.

Nous modélisons l'architecture de notre programme au travers du schéma ci-dessous :



2.3 Méthode explicite utilisant les formules analytiques de Black-Scholes pour pricer des options vanilles européennes

Nous avons donc dans une première partie crée un programme permettant d'utiliser les formules analytiques que nous avons présentées dans l'introduction dans le cas d'options vanilles européennes. Nous avons donc pu obtenir la valeur exacte de cette option (Call ou Put)

2.4 Méthode de Monte Carlo se basant sur les résultats de Black-Scholes pour pricer tout type d'option

Nous avons dans cette partie pu mettre en pratique la méthode de Monte Carlo utilisant la formule log-normale de la valeur du sous-jacent et nous avons pu estimer la valeur de l'option à partir de simulations donnant la valeur du sous-jacent. Cela nous permet d'estimer la valeur de l'option alors qu'on ne peut la calculer explicitement à l'aide de la formule de Black-Scholes.

Le principe de Monte-carlo s'appuie sur la loi forte des grands nombres. En prenant f une fonction de payoff et $(S^i)_{i\in[|1,N|]}$ des prix de sous-jacents identiques, on a, quand N tend vers l'infini :

$$\overline{X_N} = e^{-rT}\frac{1}{N}\sum_{i=1}^{N} f(S_T^i) \longrightarrow p = \mathbb{E}[e^{-rT}f(S_T)]$$

avec p le prix de l'option sur le sous-jacent S_T.

De plus le théorème centrale limite nous donne une estimation de l'erreur :

$$\sqrt{N}(\overline{X_N} - p) \longrightarrow \mathcal{N}(0, \sigma^2)$$

En temps discret, l'équation de Black-Scholes devient :

$$S_{t_{j+1}} = S_{t_j}exp((r - \frac{\sigma^2}{2})(t_{j+1} - t_j) + \sigma(W_{t_{j+1}} - W_{t_j}))$$

En considérant (ε_i) des variables aléatoires indépendantes de loi normale centrée réduite, le terme $(W_{t_{j+1}} - W_{t_j})$ a la même loi que $\sqrt{t_{j+1} - t_j}\varepsilon_i$.

La première étape est donc de générer ces variables aléatoires centrées réduites. C'est ce que fait la fonction *gaussian_box_muller*, qui utilise l'algorithme de Box-Muller.

Ensuite, il nous faut écrire la fonction *generateRiskNeutralPricePath* qui, pour un entier *nSteps* donné, qui correspond au nombre de t_j simulés, va calculer le prix de l'option $\forall i < nSteps$

Ensuite, il nous reste à écrire la fonction qui donne le prix estimé par la méthode de Monte Carlo. Pour cela, on va utiliser la fonction précédente sur un grand nombre *nScenarios* d'options, pour ensuite pouvoir prendre la moyenne et multiplier par e^{-rT} pour ainsi obtenir le prix estimé, via la loi des grands nombres et la première formule de ce paragraphe.

3 Résultats Obtenus

3.1 Bonne Pratique d'utilisation du programme

Nous avons construit notre programme de telle sorte que l'interface permette à l'utilisateur d'entrer ses choix au fur et à mesure. Ainsi, une fois le code compilé, il suffit de suivre les directives sur l'interface. Nous allons désormais expliquer l'utilisation du programme à proprement parler :

1. Choisir si l'on veut juste le prix d'une option (=1) ou le prix d'une stratégie (=2)
2. Si l'on a choisi juste le prix de l'option : choisir le type d'option (1 à 10). Si l'on a choisi le prix d'une stratégie : choisir le type de Stratégie (1 à 6)
3. Suivre les instructions demandées par l'interface. Ici mentionner les paramètres du modèle de Black-Scholes
4. Choisir si l'on veut obtenir le calcul des Grecques
5. Dépend du type d'option : 2 méthodes possibles de calcul pour les options vanilles européennes : Méthode Explicite (avec la Formule explicite) et Méthode Monte-Carlo

En effet, deux méthodes sont disponibles pour calculer le prix d'une option : *priceMonteCarlo* et *priceclassic*. La première accepte toutes les options, et génère une trajectoire complète du prix à chaque itération de l'algorithme de Monte-Carlo. Cela permet d'évaluer les options selon la trajectoire du prix. Pour les options qui ne dépendent pas de la trajectoire du prix, on peut gagner du temps en ne calculant que le prix à maturité sans générer la trajectoire du prix grâce à la méthode priceclassic. On ne peut donc pas utiliser cette méthode pour estimer le prix d'une option Asiatique ou Up and Out Knock-Out qui dépendent de la trajectoire du prix.

Ensuite, valider et laisser au programme le temps de faire les calculs (**NE PAS TOUCHER AU PROGRAMME** avant d'avoir ce message « Program ended with exit code : 0 »).

Dans un premier temps, les paramètres du modèle vont apparaitre puis au fur et à mesure les différents résultats des calculs vont apparaître.

3.2 Résultats du programme utilisant les formules explicites de Black-Scholes

Nous avons donc programmé un programme C++ permettant de calculer la valorisation d'une option vanille européeenne avec les formules explicites du modèle de Black-Scholes.

Voici ci-dessous le résultat :

```
The price of the Stock is 100
The Volatility of the Stock is 20 %
The Risk-free rate is 5 %
The Date is 0
The Drift is 5 %
The Maturity is of the European Vanilla Call is 1
The Strike is of the European Vanilla Call is 100
The Delta of the Call:0.636831
The Gamma of the Call is 0.018762
The Theta of the Call is -6.41403
The Vega of the Call is 37.524
The Rho of the Call is 53.2325
The price of the European Vanilla Call is 10.4506
Thank you for using this program
```

Figure 1 : Valeur d'un Call vanille européen obtenu par la formule analytique de Black-Scholes

3.3 Résultats du progamme utilisant la méthode de Monte Carlo

Nous avons également programmé la méthode de Monte-Carlo afin d'estimateur la valeur d'une option et afin de vérifier la pertinence de notre méthode , nous avons comparé la vraie valeur de l'option issue du modèle Black-Scholes utilisant les formules explicites pour des options vanilles européennes à la valeur issue de la méthode de Monte-Carlo.

Voici-ci dessous le résultat :

```
The price of the Stock is 100
The Volatility of the Stock is 20 %
The Risk-free rate is 5 %
The Date is 0
The Drift is 5 %
The Maturity is of the European Vanilla Call is 1
The Strike is of the European Vanilla Call is 100
The Delta of the Call:0.638568
The Gamma of the Call:0.0107471
The Theta of the Call is -6.45007
The Vega of the Call is 37.2193
The Rho of the Call is 53.2325
The price of the European Vanilla Call is 10.5249
Confidence Interval at level 95% : [9.19 , 11.86]

Thank you for using this program
```

Figure 2 :Valeur d'un Call vanille européen obtenu avec la méthode de Monte-Carlo

La première chose que l'on peut remarquer est que la méthode de Monte-Carlo donne une assez bonne approximation de la vraie valeur du Call Vanille Européen ce qui montre la convergence du modèle.
Par ailleurs, cette méthode de Monte-Carlo nous permettra d'estimer des valeurs d'options pour des stratégies de réplication pour lesquelles le modèle Black-Scholes théorique n'est plus utilisable (les options non vanilles européennes avec une expression de pay-off qui dépend de la trajectoire complète du sous-jacent et non seulement de sa valeur à maturité)

3.4 Résultats de la méthode de Monte Carlo pour d' autres options classiques

Dans cette partie, on va s'intéresser à l'implémentation de la méthde de Monte-Carlo pour d'autres options que celles classiques européennes

On va d'abord s'intéresser aux options asiatiques. On distingue deux types d'options : les options dites arithmétiques de payoff pour un call : $\max(\frac{1}{N}\sum_{l=1}^{N} S_l - K, 0)$ avec N le nombre d'observations de la valeur du sous-jacent.
On a également les options dites géométriques de payoff pour un call : $\max((\prod_{l=1}^{N} S_l)^{\frac{1}{N}} - K, 0)$
Voici les résultats obtenus pour des calls respectivement asiatique arithmétique et asiatique géométrique :

```
The price of the Stock is 100
The Volatility of the Stock is 20 %
The Risk-free rate is 5 %
The Date is 0
The Drift is 5 %
The Maturity is of the Asian Arithmetic Call is 1
The Strike is of the Asian Arithmetic Call is 100
The Delta of the Call:0.590081
The Gamma of the Call:0.0252865
The Theta of the Call is -3.41256
The Vega of the Call is 21.5095
The Rho of the Call is 24.3957
The price of the Asian Arithmetic Call is 5.80192
Confidence Interval at level 95% : [5.41 , 6.19]

Thank you for using this program
```

Figure 3 :Valeur d'un Call asiatique arithmétique obtenu avec la méthode de Monte-Carlo

```
The price of the Stock is 100
The Volatility of the Stock is 20 %
The Risk-free rate is 5 %
The Date is 0
The Drift is 5 %
The Maturity is of the Asian Geometric Call is 1
The Strike is of the Asian Geometric Call is 100
The Delta of the Call:0.581484
The Gamma of the Call:0.0726808
The Theta of the Call is -3.16941
The Vega of the Call is 19.6152
The Rho of the Call is 23.3613
The price of the Asian Geometric Call is 5.58305
Confidence Interval at level 95% : [5.22 , 5.95]

Thank you for using this program
```

Figure 4 :Valeur d'un Call asiatique géométrique obtenu avec la méthode de Monte-Carlo

Enfin, on s'est également intéressés à 2 types d'options classiques : Up-And-Out et LookBack pour lesquelles on a calculé pour chacune le prix d'un call et d'un put.

3.5 Résultats obtenus pour des stratégies d'option

Nous avons également trouver judicieux de programmer différentes statégies d'options classiques qui peuvent exister dans le monde financier telles que celles ci-dessous :

1. La stratégie Stellage Bought qui consiste à acheter 1 Call européen vanille et 1 Put européen vanille de même strike
2. La stratégie Strangle Bought qui consiste à acheter 1 Call vanille européen et 1 Put vanille européen de strikes différents avec celui du Put plus faible que celui du Call
3. La stratégie Bull Spread qui consiste à acheter 1 Call vanille européen et à vendre 1 call vanille européen de strikes différents avec le strike du Call acheté plus faible que celui vendu
4. La stratégie Bear Spread qui consiste à acheter 1 Call vanille européen et vendre 1 call vanille européen de strikes différents avec celui du Call que l'on vend plus faible
5. La stratégie Butterfly Bought qui consiste à acheter 1 Call vanille européen, vendre 2 Calls vanilles européens et acheter 1 Call vanille européen, avec 3 strikes différents pour chaque "move"
6. La stratégie Condor Sold qui consiste à vendre 1 call vanille européen, acheter 1 call vanille européen puis réacheter 1 call vanille européen et enfin vendre 1 call vanille européen avec 4 strikes différents pour chaque "move"

Nous allons utiliser notre programme pour déterminer le prix d'une stratégie. Prenons par exemple la stratégie Stellage Bought et regardons l'output obtenu :

```
The price of the Stock is 100
The Volatility of the Stock is 20 %
The Risk-free rate is 5 %
The Date is 0
The Drift is 5 %
The Maturity of this Strategy is 1
The shared Strike of this Strategy is 100
The price of the Call you'll buy is 10.5249
The price of the Put you'll buy is 5.54706
The price of this Strategy is 16.072
You have to spend this money to use this strategy
Thank you for using this program


--------------------------------
Process exited after 26.28 seconds with return value 0
```

Figure 5 : Output obtenu pour caractériser la stratégie Stellage Bought à partir de notre programme

On voit ainsi qu'on a pu caractériser le prix des options qui constituent la stratégie et ainsi caractériser le prix de la stratégie qui est ici de 16.072.

4 Points importants sur la réalisation de notre projet

4.1 Problèmes rencontrés lors de la réalisation du projet

Le premier problème que nous avons rencontré était de créer l'architecture du programme. En effet, il fallait choisir judicieusement le type de classes et ce qu'elles allaient contenir pour utiliser la POO et le Polymorphisme de la manière la plus efficace possible. Ainsi c'est seulement après avoir réalisé le schéma d'architecture du programme situé ci-dessus que nous avons pu commencer à implémenter à proprement parler notre programme.

Le second accroc s'est situé au niveau de la génération des chiffres aléatoires nécessaires à la génération des prix selon le modèle de Black-Scholes. En particulier, nous devons simuler une loi normale pour calculer les payoffs et en déduire une estimation du prix des options. Le choix de notre générateur pseudo-aléatoire est donc crucial. Nous avons utilisé le générateur pseudo-aléatoire rand() qui nous sert ensuite à obtenir le chiffre voulu grâce à l'algorithme de Box-Müller.

Enfin, nous avons décidé de calculer les Grecques en combinant les méthodes de Finite Difference et Monte-Carlo. Ainsi, l'utilisation de la méthode Finite Difference nous permet de calculer les Grecques en se basant sur ce type de formules (celle du haut pour le Delta et celle du Bas pour le Gamma)

$$\frac{\partial C}{\partial S} \approx \frac{C(S+\Delta S, T, \sigma, r, K) - C(S, T, \sigma, r, K)}{\Delta S}$$

$$\frac{\partial^2 C}{\partial S^2} \approx \frac{C(S+\Delta S, T, \sigma, r, K) - 2C(S, T, \sigma, r, K) + C(S-\Delta S, T, \sigma, r, K)}{(\Delta S)^2}$$

Cependant, étant donné la concavité ou convexité de ces courbes, ce type de calcul approxime bien la vraie valeur des Grecques lorsque le prix du sous-jacent est assez proche du Strike mais perd en précision lorsque ceux deux dernières variables s'éloignent l'une de l'autre.

4.2 Pistes d'Amélioration de notre projet

Ajout d'une interface graphique :

Il est compliqué d'implémenter une interface graphique en C++ et nous ne possédons pas encore les compétences nécessaires pour faire cela. Malgré tout, il pourrait être intéressant d'ajouter cette fonctionnalité dans un programme futur.

Laisser une plus grande liberté à l'utilisateur dans la construction de ses stratégies d'options :

Dans l'implémentation de notre programme, nous permettons à l'utilisateur de créer 6 stratégies différentes avec des options Européennes : le Stellage Bought, le Strangle Bought, le Bull Spread, le Bear Spread, le Butterfly Bought et enfin le Condor Sold. Un moyen d'améliorer ce programme serait de permettre à l'utilisateur de créer des stratégies encore plus "originales", c'est-à-dire moins standardisées, et avec un autre type d'option que les options vanilles Européennes, par exemple les options Asiatiques ou Up and Out Knock-Out.

Réduction de la variance :

La simulation par la méthode de Monte-Carlo nécessite un nombre conséquent de simulations pour être utile afin d'obtenir une estimation pertinente de la vraie valeur du paramètre. Ainsi, pour avoir des estimations avec une plus grande précision nous pouvons augmenter le nombre de simulations. Cependant, cela se fait au détriment temps de calcul, ce qui le fait donc logiquement augmenter. Il faut donc trouver un juste milieu, c'est le trade-off classique entre précision et temps de calcul.

Pour améliorer la rapidité de la convergence de notre estimateur nous aurions pu utiliser la méthode de réduction de variance. En effet, réduire la variance de notre estimateur améliore la vitesse de convergence. Cependant, il faut toutefois veiller à ce que les calculs supplémentaires engendrés par cette méthode n'augmentent pas trop le temps de calcul.

5 Annexes

5.1 Compréhension des modèles de Black-Scholes et de Monte-Carlo

Dans une première partie, nous avons cherché à comprendre les différents modèles que nous allions utiliser et nous nous avons utilisé les liens suivants :

1. Cours d'Instruments Financiers pour comprendre le principe de valorisation des options sur actions ainsi que la théorie sur le modèle de Black-Scholes et la Théorie Call-Put.
2. Explication du modèle de Black-Scholes pour comprendre la formule explicite de la valeur d'un Call/Put européen.
3. Explication de la Méthode de Monte-Carlo pour évaluer la valeur de différentes options pour des stratégies non européennes via le modèle de Black-Scholes.
4. Simulateur en ligne permettant de calculer le prix d'une action européenne en utilisant la méthode de Black-Scholes
5. Explication du concept des lettres grecques dans le modèle Black-Scholes ainsi que leurs formule explicites dans le cas d'options européennes

5.2 Compréhension de la structure des codes

Ensuite, nous avons cherché à comment structurer notre code pour pouvoir modéliser ce que l'on voulait et nous avons utilisé les liens suivants

1. Cours n°1 de Roxana Dumitrescu pour structurer son code.
2. Cours n°9 de Roxana Dumitrescu pour comprendre la construction des classes pertinentes.
3. Explication de l'implémentation d'un programme en C++ pour appliquer la méthode de Black-Scholes pour des options européennes.
4. Explication de l'implémentation du calcul des grecques par la méthode des différences finies et de Monte-Carlo