INSTITUT NATIONAL DE STATISTIQUE ET D'ÉCONOMIE APPLIQUÉE

Projet : Le calcul des options asiatiques par les méthodes Kemnna-Vorst et EDP de Rogers-Shi

Simulation des modèles financiers

Péparé par :

EL YOUSEFI AHMED

Encadré par :

M.EL QALLI Yassine

Filière : Actuariat-Finance

Année 2022/2023

Table des matières

1 La Méthode de Kemna-Vorst pour les options Asiatiques **6**

1.1 Le cadre général de la méthode Kemna-Vorst 6

1.2 Application de la formule close : à l'aide d'un programme C++ 7

1.3 La méthode des variables antithétiques : en utilisant la simulation Monte-Carlo 8

2 Le prix d'une option asiatique via l'EDP de Rogers-Shi **9**

2.1 Rogers-Shi par les Méthodes des différences finies : le schema implicite . . . 9

2.2 Application 11

2.3 Comparaison des 2 méthodes 12

A Démonstration **16**

B Code C++ pour la méthode Kemna-Vorst **17**

B.1 La méthode Kemna-Vorst en utilisant les variables Variables de contrôles . 17

B.2 Autre méthode : Algorithme Monte-Carlo et les variables antithétique . . . 18

C Code C++ pour l'équation Rogers-Shi : méthode implicite **21**

Table des figures

1.1 Le prix d'un Put et d'un Call asiatiques par la formule close 8
1.2 Le prix d'un Put etd'un Call asiatiques par les variables antithétiques . . . 8

2.1 Le prix Call et Put d'une option asiatique par le schéma implicite 12
2.2 Evolution de $f(0, x)$ 13
2.3 Comparaison des 2 méthodes 14

Liste des tableaux

2.1 Comparaison des méthodes Rogers-Shi and Kemnna-Vorst 13

Introduction

Les options asiatiques sont couramment négociées ; elles ont été introduites en partie pour éviter un problème commun aux options européennes, à savoir qu'en manipulant le prix d'un actif à l'approche de la date d'échéance, les spéculateurs pouvaient augmenter les gains de l'option.
Malgré cela, il n'y a pas encore d'expression analytique simple pour la valoriser, contrairement à la situation pour un call européen, où la célèbre formule de Black-Scholes est disponible.

Autrefois La valorisation des options asiatiques peut être effectuée à l'aide de différentes méthodes numériques.Dans ce projet nous entament la valorisation des options asiatiques par la méthode de Kemna-Vorst puis par l'EDP de Rogers-Shi.

Dans un premier temps nous abordons la valorisaton par la méthode de Kemna-Vorst, qui utilise des techniques d'évaluation de Monte Carlo pour estimer la valeur de l'option asiatique. La méthode de Kemna-Vorst est basée sur l'hypothèse que le prix de l'actif sous-jacent suit une dynamique de diffusion générale. Cette méthode est considérée comme une méthode fiable pour valoriser les options asiatiques, mais elle peut être coûteuse en termes de temps et de calculs.

Dans un deuxième temps nous utilisons l'EDP (Equation Différentielle Partielle) de Rogers-Shi pour valoriser les options asiatiques.
Cette méthode utilise une approche mathématique pour résoudre l'équation de Black-Scholes basée sur les différences finies.
L'EDP de Rogers-Shi est basée sur l'hypothèse que le prix de l'actif sous-jacent suit une dynamique de diffusion générale. Cette méthode est plus rapide et moins coûteuse en termes de calculs que la méthode de Kemna-Vorst, mais elle peut présenter des erreurs dans les cas où les hypothèses sont violées.

Finalement nous clôturons ce travail par une comparaison entre les deux méthodes en terme de résultats de valorisation.

Chapitre 1

La Méthode de Kemna-Vorst pour les options Asiatiques

1.1 Le cadre général de la méthode Kemna-Vorst

On suppose que le prix d'un actif à l'instant t est donné par un modèle de *Black-Scholes* :

$$S_t = S_0 \times exp\left(\sigma B_t + (r - \frac{\sigma^2}{2})t\right)$$

Où : B_t est le mouvement brownien standard unidimensionnel sous la probabilité risque-neutre $\mathbb{Q}$, σ, r et S_0 [1]
Une option de vente asiatique est défini par le Payoff :

$$(K - \bar{S_T})^+$$

Où $\bar{S_T} = \frac{1}{T}\int_0^T S_t dt$.
Le prix de cette option,à l'instant $t = 0$, est donné par la formule de valorisation risque-neutre avec un taux d'intérêt constant r :

$$P(S_0, \sigma, r, K, T) = \mathbb{E}_{\mathbb{Q}}\left(e^{-rT}\left(K - \bar{S_T}\right)^+\right)$$

Sous la condition que σ et r sont assez petits le prix d'une option de vente de strike K peut s'écrire comme suit :

$$P(S_0, \sigma, r, K, T) = \mathbb{E}_{\mathbb{Q}}\left(e^{-rT}\left(K - exp(Z)\right)^+\right)$$

Où $Z = \frac{1}{T}\int_0^T log(S_t)dt$.
Cette formule est obtenue à l'aide de l'approximation $exp(Z) \approx \bar{S_T}$ (sous la condition précédente).

Dans le cadre du modèle de $Black - Scholes$ la variable aléatoire Z suit, sous $\mathbb{Q}$, une loi Gaussienne :

$$E_{\mathbb{Q}}(Z) = log(S_0) + \frac{(r - \frac{\sigma^2}{2})T}{2} \qquad et \qquad Var_{\mathbb{Q}}(Z) = \frac{\sigma^2 T}{3}$$

1. On suppose que nous nous plaçons à l'instant $t = 0$ où le prix S_0 est connu donc on peut le considérer comme constant et non aléatoire.

La démonstration est dans l'annexe A.

On calcule donc le prix du put à l'instant $t = 0$:

$$P(S_0, \sigma, r, K, T) = exp(-rT) \times \left(K\mathcal{N}(-d) - \mathcal{N}\left(-d - \sqrt{Var_{\mathbb{Q}}(Z)}\right) \times exp\left(E_{\mathbb{Q}}(Z) + \frac{Var_{\mathbb{Q}}(Z)}{2} \right) \right) \tag{1.1}$$

Où $d = \frac{E_{\mathbb{Q}}(Z) - log(K)}{\sqrt{Var_{\mathbb{Q}}(Z)}}$.

Remarque

Dans cette partie on suppose que $\sigma \approx 0.3$ par an et $r \approx 0.1$ par an et $T \approx 1$ année. Car la méthode de Kemna-Vorst est effciente tant que la supposition précédente est vérifiée.

1.2 Application de la formule close : à l'aide d'un programme C++

Prix d'un put asiatique

En utilisant la formule du princing 1.1 on peut construire une fonction `KemnaVorst_put1()` qui prend comme entrées les éléments suivants :

— $\mathbf{S_0}$: le prix du sous-jacent

— **sigma** (σ) : la volalité qu'on fixée à la valeur 30%.

— **r** : le taux d'intérêt qui est aussi fixé à 10%.

— **K** : le strike de l'option.

— **T** : Le temps à la maturité de l'option fixé à 1 année.

La fonction `KemnaVorst_put1()` retourne comme output le prix de l'option pour des valeurs des inputs.
Dans l'annexe B on donne la fonction `KemnaVorst_put1()` ainsi qu'on fait appel de cette fonction pour les valeurs suivantes ($S_0 = 120, K = 130, sigma = 0.3, r = 0.1, T = 1$).

Prix d'un call asiatique

A l'aide de la formule de parité call-put, on onbtient le prix du call :

$$C(S_0, \sigma, r, K, T) = \mathbb{E}_{\mathbb{Q}}\left(e^{-rT}\left(\bar{S_T} - K\right)^+\right) = P(S_0, \sigma, r, K, T) + (\mathbb{E}(\bar{S_T}) - K)e^{-rT} \tag{1.2}$$

Avec : $\mathbb{E}(\bar{S_T}) = S_0 \frac{e^{rT} - 1}{rT}$.
A l'aide de l'equation 1.2 on peut écrire facilement un programme `C++` en utilisant la fonction déjà définit `KemnaVorst_put1()`. Dans l'annexe B on définit la fonction `KemnaVorst_call1()` qui donne le prix du Call, et un programme qui donne le prix du Put et du Call ci-dessous le résultat de ce programme pour les mêmes Inputs précédents :

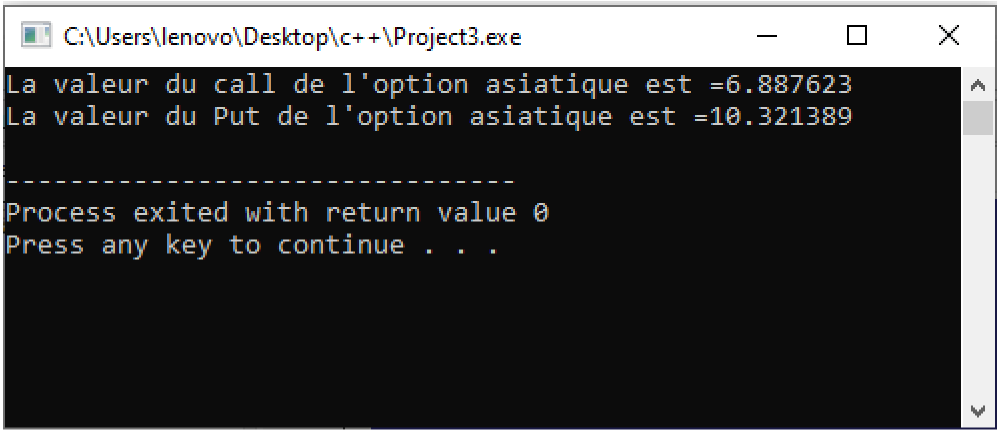

FIGURE 1.1 – Le prix d'un Put et d'un Call asiatiques par la formule close

1.3 La méthode des variables antithétiques : en utilisant la simulation Monte-Carlo

Puisque on cherche la quantité $\mathbb{E}_{\mathbb{Q}}\left(e^{-rT}\left(K - exp(Z)\right)\right)^{+}\big)$ on peut utiliser la simulation $Monte - Carlo$ pour la calculer, en simulant la variable aléatoire $e^{-rT}(K - exp(Z))^{+}$. Cela nous mene à simuler Z qui est une v.a normale.

Les étapes

1. On simule $X_1, ..., X_n$ v.a iid de loi normale centré réduite.
2. Pour chaque i on calcule $g(X_i)$ et $g(-X_i)$ avec $g(x) = max(K - e^{xVar_{\mathbb{Q}}(Z)+E_{\mathbb{Q}}(Z)}, 0)$
3. Le prix du Put est la quantité : $\frac{e^{-rT}}{2n} \sum_{i=1}^{n} g(X_i) + g(-X_i)$

Dans l'Annexe B Un programme contenant 2 fonctions qui donne le prix du call et du put pour les mêmes inputs déjas définie dans la partie précedante et un $n = 100$:

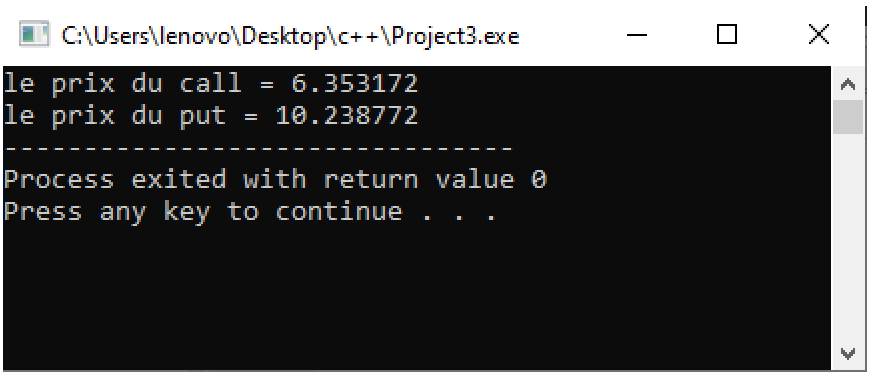

FIGURE 1.2 – Le prix d'un Put etd'un Call asiatiques par les variables antithétiques

On remarque que les 2 méthodes donnent des résultats proches.

Chapitre 2

Le prix d'une option asiatique via l'EDP de Rogers-Shi

2.1 Rogers-Shi par les Méthodes des différences finies : le schema implicite

On suppose que le prix de l'option asiatique de Strike K est donné par :$P = S_0 f(0, \frac{K}{S_0})$
Et vérifie l'EDP suivante :

$$\begin{cases} \frac{\partial f}{\partial t} + \frac{\sigma^2 x^2}{2}\frac{\partial^2 f}{\partial x^2} - (\frac{1}{T} + rx)\frac{\partial f}{\partial x} = 0 \\ f(T, x) = x^- \end{cases} \tag{2.1}$$

Afin de résoudre cette EDP, nous nous basons sur la méthodes des différences finies, donc nous allons d'abord discrétiser l'EDP et ceci en remplaçant les dérivées partielles par des approximations discrètes.

On se place sur $(t, x) \in [0, T] \times [0, b]$ avec $0 < b$.
Pour $M \in \mathbb{N}^*$ et $N \in \mathbb{N}^*$, on pose

$k = \Delta t = \frac{T}{M}$ $\qquad h = \Delta x = \frac{b}{N+1}$
$t_j = jk \qquad 0 \leq j \leq M$
$x_i = ih \qquad 0 \leq i \leq N$

En raison de l'indisponibilité de la condition initile $f(0, x)$ on définit

$$U(x, \tau) = f(T - \tau, x) \tag{2.2}$$

tel que $U_{i,j} = u(x_i, t_j)$, $k = \Delta t$ est le pas de discrétisation en temps et $h = \Delta x$ est le pas de discrétisation en espace.

La solution de cette EDP pour chaque x négatif est donnée par :

Pour chaque $x \leq 0$,

$$f(t, x) = \frac{1}{rT}(1 - e^{-r(T-t)} - xe^{-r(T-t)}) \tag{2.3}$$

Ainsi en considérant les conditions initiales et limites le système peut être reécrit comme suit :

$$\begin{cases} \frac{\partial U}{\partial t} - \frac{\sigma^2 x^2}{2}\frac{\partial^2 U}{\partial x^2} + (\frac{1}{T} + rx)\frac{\partial U}{\partial x} = 0 & \text{pour (x,t)} \in [0,b] \times [0,T] \\ U(t=0,x) = U_0(x) = f(T,x) = 0 & \\ U(t,x=0) = f(T-\tau,0) = \alpha_t = \frac{1-e^{-r\tau}}{rT} & \\ U(t,x=b) = \beta_t = 0 & \text{pour b très grand} \end{cases} \tag{2.4}$$

A l'aide des développements de Taylor et en remplaçant les dérivées partielles par leurs approximations discrètes de type différence progressive et centrée en utilisant le schéma implicite, nous obtiendrons donc pour tout $0 \leq j \leq M$ et $1 \leq i \leq N$:

$$\frac{U_{j+1,i} - U_{j,i}}{k} - \frac{\sigma^2 x^2}{2}\frac{U_{j+1,i+1} - 2U_{j+1,i} + U_{j+1,i-1}}{h^2} + \left(\frac{1}{T} + rx_i\right)\frac{U_{j+1,i+1} - U_{j+1,i-1}}{2h} = 0 \tag{2.5}$$

On peut réecrire ce schéma numérique : pour $0 \leq j \leq M$

$$U_{j+1,i} - U_{j,i} + \frac{k}{2h^2}\left(\alpha_i U_{j+1,i+1} + \gamma_i U_{j+1,i} + \beta_i\right) = 0 \tag{2.6}$$

Avec :

$\alpha_i = (\frac{1}{T} + rx_i)h - \sigma^2 {x_i}^2$

$\beta_i = -(h(\frac{1}{T} + rx_i) + \sigma^2 {x_i}^2)$

$\gamma_i = 2\sigma^2 {x_i}^2$

L'équation precédente peut s'écrire alors :

$$U_{j+1,i} - U_{j,i} + \frac{k}{2h^2}(\alpha_i U_{j+1,i+1} - 2U_{j+1,i} + \beta_i U_{j+1,i}) = 0 \tag{2.7}$$

Par suite :

$$U_{j,i} = U_{j+1,i} + \frac{k}{2h^2}(\alpha_i U_{j+1,i+1} - 2U_{j+1,i} + \beta_i U_{j+1,i}) \tag{2.8}$$

On constate par la suite que la connaissnace de $U_{j,i}$ pour tout $1 \leq i \leq N$ entraînz czllz de $U_{j+1,i}$ pour tout $1 \leq i \leq N$.
On retrouve ainsi le côté *évolutif* en temps de l'équation. En utilisant l'**Ecriture matricielle** on a alors :

$$F^{(j)} = \left(I + \frac{k}{2h^2}A\right)U^{(j+1)} \tag{2.9}$$

Avec :

$$F^{(j)} = U^{(j)} + \begin{pmatrix} \frac{-k}{2h^2}\beta_1 U_{j+1,0} \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix} ; U^{(j)} = \begin{pmatrix} U_{j,1} \\ U_{j,2} \\ \vdots \\ U_{j,N-1} \\ U_{j,N} \end{pmatrix} et A = \begin{pmatrix} \gamma_1 & \alpha_1 & & & \\ \beta_1 & \gamma_2 & \alpha_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & \alpha_N \\ & & & \beta_N & \gamma_N \end{pmatrix} \quad (2.10)$$

Donc

$$U^{(j+1)} = F^{(j)} \left(I + \frac{k}{2h^2} A \right)^{-1}$$

Avec $(I + \frac{k}{2h^2}A)$ est une matrice inversible.

Remarque

On peut appliquer également le schéma de Crank-Nicolson qui donne presque les mêmes résultat du Schéma implicite.

2.2 Application

A l'aide du schéma implicite, nous avons pu déscritiser l'EDP de Rogers-shi et ceci par la méthode des différences finies. A l'aide de l'écriture matricielle, on avons implementér sous un programme C++, la fonction **implicit**, la fonction prend en arguments les paramètres suivant :

— l'intervalle [a,b] avec a=0 et b=20 tel que b>> a
— T ltemps de maturité de l'option fixé à 1 année
— r taux d'itérêt fixé à 10%
— σ la volatilité fixée à 30%
— N est la partition sur [0,T] fixée à 1000
— M est la partition sur [a,b] fixée à 1000

Cette fonction nous retourne la matrice

$$U = (U(x_i, t_j))_{(i,j) \in [1,N] \times [1,M+1]}$$

Par la suite, le prix du Put de l'option asiatique est donné par,

$$P = S_0 \times f(0, \frac{K}{S_0}) \times \exp(-rT)$$

Donc

$$P = S_0 \times U(\frac{K}{S_0}, T) \times \exp(-rT)$$

Ceci est équinvalent à :

$$P = S_0 \times U(\frac{K}{S_0}, t_{M+1}) \times \exp(-rT)$$

Avec $U(\frac{K}{S_0}, T)$ est la valeur à la maturité T au point $x = \frac{K}{S_0}$, qui est égale à $U(\frac{K}{S_0}, t_{M+1})$ corresponadnt à l'élement du $\frac{K}{S_0}$ ligne et la dérnière colonne t_{M+1}.
On sait que $U(x,T) = f(0,x)$, nous répresentons alors la fonction $f(0,x)$ par la figure 2.2.Nous remarquons que $f(0,x)$ est fonction décroissante (décroit exponentiellement), ceci s'explique par le fait que si la valeur du Strike K augmente alors le prix du call diminue.

On prend maitenant, le cas oû $S_0 = 120$, $K = 130$
Donc $\frac{K}{S_0} = 1.083$ cette valeur correspond à $U[53]$
Ainsi la valeur du Call est égale à :

$$120 \times U[53, 1000] \times \exp -0.1 = 6.858$$

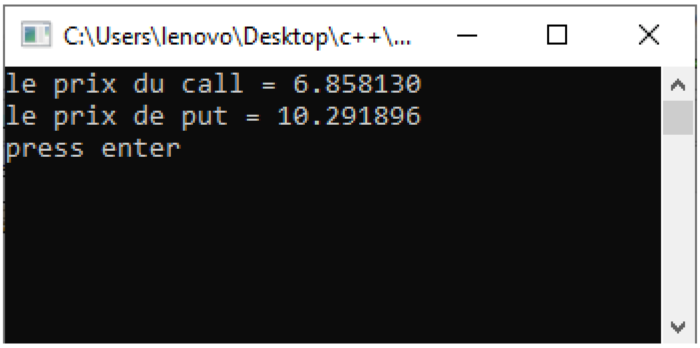

FIGURE 2.1 – Le prix Call et Put d'une option asiatique par le schéma implicite

2.3 Comparaison des 2 méthodes

Une comparaison entre les prix d'une option d'achat par la méthode de Kemnma et les prix de l'option d'achat par la méthode de Rogers-Shi, pour une même valeur du sous jacent $S_0 = 120$ et des valeurs croissante du Strike K pour les deux méthodes, est donné par le tableau 2.1.

Nous remarquons que les résultats des deux colonnes **prix call kemnna** et **prix call Rogers** pour la même valeurs du Strike sont **très proches**.

Pour mieux illustrer ce résultat, nous procédons par une illustration graphique 2.3, Nous constatons que les courbes des deux méthode sont presques identique avec une petite diffèrence positive de la courbe de la méthode de kemna-Vorst par rapport à la méthide de Rogers-Shi de l'ordre de 10^{-1}.

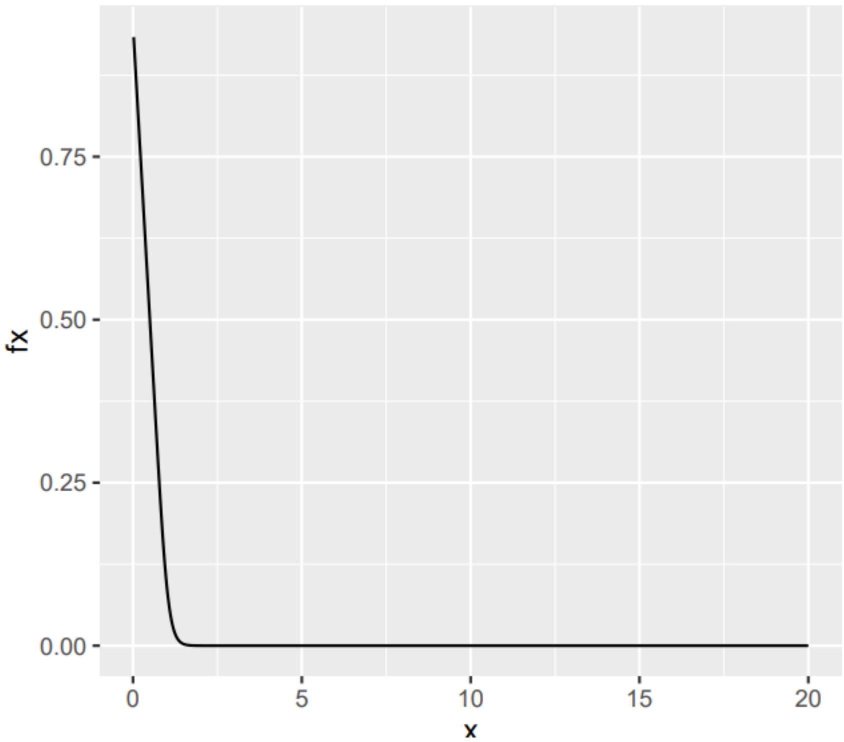

FIGURE 2.2 – Evolution de $f(0, x)$

Strike	prix_call kemnna	prix_call Rogers
96	27,80	27,775152
98,4	25,81	25,772184
100,8	23,87	23,822004
103,2	22,00	21,933396
105,6	20,19	20,114772
108	18,46	18,373764
110,4	16,81	16,717044
112,8	15,25	15,15
115,2	13,78	13,676724
117,6	12,41	12,299808
120	11,14	11,0204796
122,4	9,97	9,8385708
124,8	8,89	8,7526908
127,2	7,91	7,7603496
129,6	7,03	6,8581296
132	6,23	6,0418716
134,4	5,51	5,3068428
136,8	4,88	4,6479144
139,2	4,32	4,0597116
141,6	3,82	3,5367576

TABLE 2.1 – Comparaison des méthodes Rogers-Shi and Kemnna-Vorst

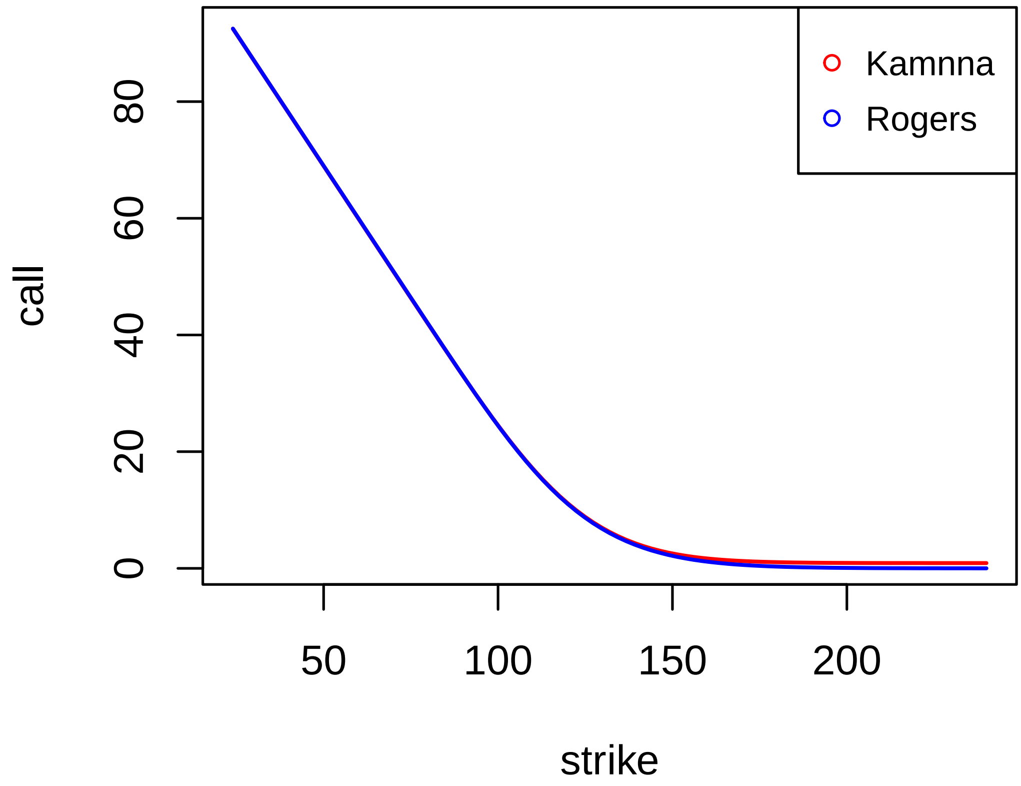

Figure 2.3 – Comparaison des 2 méthodes

Conclusion

En conclusion, la valorisation des options asiatiques peut être effectuée à l'aide de différentes méthodes, comme la méthode de Kemna-Vorst et l'EDP de Rogers-Shi.

La méthode de Kemna-Vorst utilise des techniques d'évaluation de Monte Carlo pour estimer la valeur de l'option asiatique, elle est considérée comme une méthode fiable mais elle est coûteuse en termes de temps et de calculs.

L'EDP de Rogers-Shi utilise une approche mathématique pour résoudre l'équation de Black-Scholes pour les options asiatiques, elle est plus rapide et moins coûteuse en termes de calculs, mais elle peut présenter des erreurs dans les cas où les hypothèses sont violées.

Concernant la valorisation du call, en terme de résultats nous pouvons constater que les deux méthodes génèrent des résultats très proche, avec une petite différence positive cocérnant la valorisation par la méthode de kemna-Vorst par rapport à la méthode de Rogers-Shi de l'ordre de 10^{-1}

Il est important de considérer les avantages et les inconvénients de chaque méthode avant de choisir celle qui convient le mieux pour un projet de valorisation et de considérer les risques et incertitudes associés à l'utilisation de ces méthodes pour valoriser les options asiatiques.

Annexe A

Démonstration

On a :

$$log(S_t) = log(S_0) + (r - \frac{\sigma^2}{2})t + \sigma B_t$$

Donc :

$$\int_0^T log(S_t)dt = Tlog(S_0) + \left(r - \frac{\sigma^2}{2}\right)\frac{T^2}{2} + \sigma \int_0^T B_t dt \tag{A.1}$$

B_t est un mouvement Brownien sous $\mathbb{Q}$, en applicant lemme d'Itô sur $f(B_t, t) = t \times B_t$ on trouve :

$$B_t dt = d(t \times B_t) - tdB_t$$

Par intégration on a :

$$\int_0^T B_t dt = B_T \times T - \int_0^T tdB_t = \int_0^T (T-t)dB_t$$

On a :

$$\int_0^T (T-t)dB_t \sim \mathcal{N}\left(0, \int_0^T (T-t)^2 dt\right)$$

Alors

$$\sigma \int_0^T B_t dt \sim \mathcal{N}\left(0, \sigma^2 \int_0^T (T-t)^2 dt\right)$$

Donc en remplaçant dans l'expression A.1 :

$$Z = \frac{1}{T}\int_0^T log(S_t)dt \sim \mathcal{N}\left(E_{\mathbb{Q}}(Z), Var_{\mathbb{Q}}(Z)\right)$$

$E_{\mathbb{Q}}(Z) = log(S_0) + \frac{(r-\frac{\sigma^2}{2})T}{2}$ et $Var_{\mathbb{Q}}(Z) = \frac{\sigma^2}{T^2}\int_0^T (T-t)^2 dt = \frac{\sigma^2 T}{3}$

Annexe B

Code C++ pour la méthode Kemna-Vorst

B.1 La méthode Kemna-Vorst en utilisant les variables Variables de contrôles

```
#include <iostream>
#include <stdlib.h>
#include <stdio.h>
#include<conio.h>
#include <math.h>
#include<gsl/gsl_sf_erf.h>
double KemnaVorst_put1(double S0, double K, double r, double sigma,
↪  double T)
{
if(S0>0)
{
//esperance et la variance de la variable Z
double Ez = log(S0)+(r-sigma*sigma*0.5)*0.5*T;
double Varz = (sigma*sigma*T)/3;

double d1 = (log(K)-Ez)/sqrt(Varz);
double d2 = d1 - sqrt(Varz);
return exp(-r*T)*(K*(1-gsl_sf_erf_Q(d1)) - exp(Ez+0.5*Varz) *(1-
↪  gsl_sf_erf_Q(d2)));
}
else
return 0;
}

double KemnaVorst_call1(double S0, double K, double r, double sigma,
↪  double T)
{
        double ES = S0*(exp(r*T)-1)/(r*T) ;
```

```
        return KemnaVorst_put1 (S0, K, r, sigma,T) + (ES-K)*exp(-r*T);
}

int main(){ printf("La valeur du call de l'option asiatique est
 ↪  =%f\n",KemnaVorst_call1(120, 130, 0.1, 0.3,1));
printf("La valeur du Put de l'option asiatique est
 ↪  =%f\n",KemnaVorst_put1(120, 130, 0.1, 0.3,1));
return 0;
getch();
}
```

B.2 Autre méthode : Algorithme Monte-Carlo et les variables antithétique

```
#include <iostream>
#include <stdlib.h>
#include <stdio.h>
#include<conio.h>
#include <math.h>
#include <time.h>
#include<gsl/gsl_sf_erf.h>
#include<gsl/gsl_rng.h>
#include<gsl/gsl_randist.h>
#include<gsl/gsl_statistics_int.h>
#include<gsl/gsl_statistics_double.h>

double max(double x,double y)
{
        if (x<y) return y;
        else return x;
}
double KemnaVorst_call2(double S, double K, double r, double sigma,
 ↪  double T, int n)
{
    // variables initial
    double sum = 0;
    double Ez = log(S)+(r-sigma*sigma*0.5)*0.5*T;
    double Varz = (sigma*sigma*T)/3;
    //gsl_rng *r2;
    //r2=gsl_rng_alloc(gsl_rng_taus);
    // gsl_rng_env_setup();
    //gsl_rng_set(r2,time(NULL));
    float u1,u2,X;
    // Boucle pour la simulation Monte Carlo
    for (int i = 0; i < n; i++)
```

```
    {
    // Generer des v.a normales centrées réduites
      u1=(float)rand()/RAND_MAX;
      u2=(float)rand()/RAND_MAX;
      if(u1>0)
      X = sqrt(-2*log(u1)) * cos(M_PI*2*u2);
      else continue;
      //float X = gsl_ran_gaussian(r2,1);// génère une gaussienne en
 ↪  utilisant le generateur r2 ~ taus

        // calculer la variable aléatoire exp(Z)
        double expZ = exp(X*sqrt(Varz)+Ez);
        double expZn = exp(-X*sqrt(Varz)+Ez);
        // Calculer le payoff de l'option
        double payoff = max(expZ-K, 0.0) + max(expZn-K, 0.0);

        // Cumuler le Payoff
        sum += payoff;
    }

    // Calculate and return the option price
    return (sum * exp(-r * T))/(2*n);
}

double KemnaVorst_put2(double S, double K, double r, double sigma,
 ↪  double T, int n)
{
    // variables initial
    double sum = 0;
    double Ez = log(S)+(r-sigma*sigma*0.5)*0.5*T;
    double Varz = (sigma*sigma*T)/3;

    float u1,u2,X;
    // Boucle pour la simulation Monte Carlo
    for (int i = 0; i < n; i++)
    {
    // Generer des v.a normales centrées réduites
      u1=(float)rand()/RAND_MAX;
      u2=(float)rand()/RAND_MAX;
      if(u1>0)
      X = sqrt(-2*log(u1)) * cos(M_PI*2*u2);
      else continue;
        // calculer la variable aléatoire exp(Z)
        double expZ = exp(X*sqrt(Varz)+Ez);
        double expZn = exp(-X*sqrt(Varz)+Ez);
        // Calculer le payoff de l'option
        double payoff = max(K-expZ, 0.0) + max(K-expZn, 0.0);
```

```
        // Cumuler le Payoff
        sum += payoff;
    }

    // Calculate and return the option price
    return (sum * exp(-r * T))/(2*n);
}

int main(){
printf("le prix du call =
↪  %f\n",KemnaVorst_call2(120,130,0.1,0.3,1,100));
printf("le prix du put = %f",KemnaVorst_put2(120,130,0.1,0.3,1,100));
return 0;
getch();
}
```

Annexe C

Code C++ pour l'équation Rogers-Shi : méthode implicite

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <math.h>
#include <iostream>
#include<time.h>
#include <gsl/gsl_vector.h>
#include <gsl/gsl_matrix.h>
#include <gsl/gsl_linalg.h>
#include <gsl/gsl_blas.h>
#include <gsl/gsl_sf_erf.h>
#include<gsl/gsl_rng.h>
#include<gsl/gsl_randist.h>
#include<gsl/gsl_statistics_int.h>
#include<gsl/gsl_statistics_double.h>

// condition au limite
double v_T(double x) {
if (x > 0) {
return 0;
} else {
return -x;
}
}

double v_0(double t,double r,double T){
        return (1-exp(-r*t))/(T*r);
}

gsl_matrix *implicit(double a, double b, double T,double r,double
↪  sigma,int N, int M)
{
```

```
double n,Delta_t,Delta_x,u,xi,t,beta_i,alpha_i,gamma_i,F_i;
float beta_1;
        //Initialisation du vecteur U1 pour les valeurs de U(j)
          gsl_vector *U1;
          U1 = gsl_vector_alloc(N);
        //Vecteur U2 pour stocker la valeur de U(j+1)
    gsl_vector *U2;
        U2 = gsl_vector_alloc(N);
        //Initialisation de la matrice de solutions U
    gsl_matrix *U;
        U=gsl_matrix_alloc(N,M+1);
    //La matrice A
        gsl_matrix *A;
        A=gsl_matrix_alloc(N,N);
    //La matice d'identite
        gsl_matrix *I;
         I=gsl_matrix_alloc(N,N);
    gsl_matrix_set_identity(I);
    //Allouer la memoire pour la matrice inverse
        gsl_matrix *invA;
        invA= gsl_matrix_alloc(N,N);
    //les pas
        Delta_x = (b-a)/(double)(N+1);
    Delta_t = T/(double)(M);
    beta_1 = -((1/T +r*Delta_x)*Delta_x+pow(sigma*Delta_x,2));
        //Remplissage du vecteur U(0)
    for(int i=0;i<=N-1;i++)
    {
            xi= (i+1)*Delta_x+a;
        u=v_T(xi);
        gsl_vector_set(U1,i,u);
    }
    //Inserer le vecteur U1 dans la matrice U
    gsl_matrix_set_col(U,0,U1);
        //Initialisation de A
    gsl_matrix_set_zero(A);
        //Remplissage de la matrice A
    for(int i=0;i<=N-1;i++)
    {

            xi = (i+1)*Delta_x+a;
            alpha_i = (1/T + r*xi)*Delta_x - pow(sigma*xi,2);
            beta_i = -((1/T +r*xi)*Delta_x + pow(sigma*xi,2));
            gamma_i =2 * pow(sigma*xi,2);
        for(int j=0;j<=N-1;j++)
        {
                //remplir la matrice A par les valeurs
```

```
            if(i==j){gsl_matrix_set(A,i,j,gamma_i);}
            if(i==j+1){gsl_matrix_set(A,i,j,beta_i);}
            if(i==j-1){gsl_matrix_set(A,i,j,alpha_i);}

        }
    }
    gsl_matrix_scale(A,Delta_t/(2*pow(Delta_x,2)));
        //la somme de la matrice identité et A
    gsl_matrix_add(A,I);
    int s;
    gsl_permutation *perm;
    perm = gsl_permutation_alloc(N);
    gsl_linalg_LU_decomp(A,perm,&s);
        //Inversion de (I+(k/h^2)*A)
    gsl_linalg_LU_invert(A,perm,invA);
        //Inserer les valeurs
    for(int i=1;i<=M;i++)
    {

        u = gsl_vector_get (U1,0);
        F_i = -v_0(i*Delta_t,r,T)*(Delta_t)*beta_1/(2*pow(Delta_x,2)) +
↪   u;
        gsl_vector_set(U1,0,F_i);
            gsl_blas_dgemv(CblasNoTrans,1,invA,U1,0,U2);//U2 contient le
↪   produit Inv*U1
            //Insérer le vecteur U2 dans la i-ième colonne de la
↪   matrice U
        gsl_matrix_set_col(U,i,U2);
        gsl_vector_memcpy(U1,U2);//Sauvegarder la valeur de U2 dans U1

    }
    return U;
}
```

Bibliographie

[1] Cours "Simulation des modèles financiers". M. EL QALLI Yassine.

[2] L.C.G. Rogers Z. Shi (1995), The value of an Asian option, J. Appl. Proba, 32(1077-1088).

[3] Germán I. Ramírez-Espinoza "Conservative and Finite Volume Methods for the Pricing Problem".

[4]D. Duffy, Finite difference methods in financial engineering : A partial differential equation approach, Wiley, 2006