Pricer Monte–Carlo générique

Jérôme Lelong

Ensimag

Année 2017-2018

Organisation du projet

- Travail par groupe de 4.
- Rendu sur TEIDE pour le jeudi 28 septembre à 14h
  - Code + documentation
  - Un diagramme de GANTT de l'équipe
- Suivre impérativement les instructions de rendu données sur la page du projet.
- Les soutenances auront lieu le vendredi 29 septembre matin : 25 minutes par groupe. Tous les membres du groupe doivent être capables d'expliquer l'intégralité de ce qui a été implémenté.
- Au besoin, les notes pourront être différenciées.

Objectifs

But : disposer d'un outil permettant de calculer les prix et la couverture de produits financiers dans le modèle de Black–Scholes multidimensionnel.
Cet outil devra implémenter de la généricité sur les produits considérés.

- Projet **clé** dans la filière IF
- Sert de base au projet "Evaluation de produits structurés"
- Réutilisé dans le cours "Calcul Parallèle et Applications en finance"
- Les choix d'architecture et de librairies sont **centraux**.

Formalisme mathématique (I)

Soit $S = (S_t,\ t \geq 0)$ la dynamique du sous-jacent. Nous allons supposer que le payoff des options considérées s'écrit sous la forme

$$\varphi(S_{t_0}, S_{t_1}, \ldots, S_{t_N})$$

où $0 = t_0 < t_1 < \cdots < t_N = T$.
Soit $r > 0$ le taux d'intérêt instantané supposé constant. Le prix à l'instant $0 \leq t \leq T$, avec $t_i \leq t < t_{i+1}$ d'une telle option est donné par

$$v(t, S_{t_0}, \ldots, S_{t_i}, S_t) = \mathrm{e}^{-r(T-t)}\, \mathbb{E}(\varphi(S_{t_0}, S_{t_1}, \ldots, S_{t_N})|\mathcal{F}_t)$$

Cette quantité sera calculée par une méthode de Monte–Carlo.
Un estimateur Monte–Carlo est toujours accompagné de la donnée d'un intervalle de confiance.

Le modèle de Black–Scholes

En dimension 1, le sous–jacent est modélisé par

$$S_t = S_0 \, \mathrm{e}^{(r-\sigma^2/2)t+\sigma W_t}$$

où $\sigma > 0$ est la volatilité, r le taux d'intérêt instantané et W un mouvement Brownien standard.
Pour tout u, t, on a l'égalité

$$S_{t+u} = S_t \, \mathrm{e}^{(r-\sigma^2/2)u+\sigma(W_{t+u}-W_t)} \overset{Loi}{=} S_t \tilde{S}_u$$

où

$$\tilde{S}_u = \mathrm{e}^{(r-\sigma^2/2)u+\sigma\tilde{W}_u}$$

avec $\tilde{W}$ un mouvement Brownien standard indépendant de $\mathcal{F}_t$

Simulation du modèle de Black–Scholes

Il faut être capable de simuler un échantillon selon la loi de $(S_{t_1}, \ldots, S_{t_N})$.

$$S_{t_i} = S_{t_{i-1}} \, \mathrm{e}^{(r-\sigma^2/2)(t_i - t_{i-1}) + \sigma(W_{t_i} - W_{t_{i-1}})}$$

Les accroissements $W_{t_i} - W_{t_{i-1}}$ sont indépendants et de loi $\mathcal{N}(0, t_i - t_{i-1})$. Soit $G_1, \ldots, G_N$ N v.a. indépendantes de loi normale centrée et réduite et posons

$$X_i = X_{i-1} \, \mathrm{e}^{(r-\sigma^2/2)(t_i - t_{i-1}) + \sigma\sqrt{t_i - t_{i-1}} G_i}$$

Alors

$$(S_{t_1}, \ldots, S_{t_N}) \overset{Loi}{=} (X_1, \ldots, X_N).$$

Prix à l'instant 0

On approche le prix en zéro

$$v(0, S_0) = \mathrm{e}^{-rT}\, \mathbb{E}(\varphi(S_{t_0}, S_{t_1}, \ldots, S_{t_N}))$$

par une méthode de Monte Carlo

$$\mathrm{e}^{-rT}\, \frac{1}{M} \sum_{j=1}^{M} \varphi(S_{t_0}^{(j)}, S_{t_1}^{(j)}, \ldots, S_{t_N}^{(j)})$$

où les $N-$uplets $(S_{t_0}^{(j)}, S_{t_1}^{(j)}, \ldots, S_{t_N}^{(j)})$ pour $j = 1, \cdots, M$ sont i.i.d selon la loi du vecteur $(S_{t_0}, S_{t_1}, \ldots, S_{t_N})$.

Prix à l'instant $t > 0$ (I)

On a vu que pour tout u, t

$$S_{t+u} = S_t \tilde{S}_u \tag{1}$$

où $\tilde{S}$ est **indépendant** de $\mathcal{F}_t$. Ainsi, le prix à l'instant t se réécrit

$$v(t, S_{t_0}, \ldots, S_{t_i}, S_t) =$$
$$\mathrm{e}^{-r(T-t)}\, \mathbb{E}(\varphi(s_{t_0}, \ldots, s_{t_i}, s_t \tilde{S}_{t_{i+1}-t}, \ldots, s_t \tilde{S}_{t_N-t})) \Big| \begin{array}{l} s_{t_k} = S_{t_k},\ k = 0, \cdots, i \\ s_t = S_t \end{array}$$

Les lettres s minuscules désignent des variables déterministes.

Prix à l'instant $t > 0$ (II)

On peut alors approcher le prix à l'instant t par la moyenne Monte Carlo suivante

$$\mathrm{e}^{-r(T-t)} \frac{1}{M} \sum_{j=1}^{M} \varphi(s_{t_0}, s_{t_1}, \ldots, s_{t_i}, s_t \tilde{S}^{(j)}_{t_{i+1}-t}, \ldots, s_t \tilde{S}^{(j)}_{t_N-t}) \tag{2}$$

où les $N-$uplets $(\tilde{S}^{(j)}_{t_{i+1}-t}, \ldots, \tilde{S}^{(j)}_{t_N-t})$ sont i.i.d selon la loi de $(\tilde{S}_{t_{i+1}-t}, \ldots, \tilde{S}_{t_N-t})$.
$(s_{t_0}, s_{t_1}, \ldots, s_{t_i}, s_t)$ représentent les valeurs réellement observées jusqu'à l'instant t, ce sont les cotations du sous-jacent observées sur le marché jusqu'à la date t.

Les fonctionnalités à implémenter

- Simuler une trajectoire du modèle de Black–Scholes sur la grille $0 = t_0 < t_1 < \cdots < t_N$.
- Simuler une trajectoire du modèle de Black–Scholes sur la grille $0 = t_0 < t_1 < \cdots < t_N$ conditionnellement à la donnée du passé jusqu'à la date t $S_{t_0}, S_{t_1}, \cdots, S_{t_i}, S_t$.
- Calculer le payoff de l'option étant donnée une trajectoire du sous–jacent.
- Calculer le prix à un instant t quelconque d'une option connaissant son payoff.
- Calculer le portefeuille de couverture de l'option.

Le fichier d'entrée

Le pricer sera alimenté par un fichier du type

```
option size <int> 40
strike <float> 100
spot <vector> 100
maturity <float>  3
volatility <vector> 0.2
interest rate <float>  0.04879
correlation <float> 0.0

option type <string> basket
payoff coefficients <vector> 0.025

TimeStep Number <int> 1
Sample Number <long> 50000
```

Le parser

- La liste des paires (clés, type) est fournie dans le sujet.
- Les clés sont insensibles à la casse.
- Un parser écrit en C++ vous est fourni. Cf. l'exemple d'utilisation fourni dans le squelette du projet disponible sur la page du projet.

Les grandes étapes du projet

1. Du formalisme mathématique à la conception informatique
   - Comment représenter le modèle d'un point de vue informatique ?
   - Qu'est-ce qui caractérise une option ?
   - De quelles fonctionnalités a-t-on besoin pour valoriser et couvrir une option dans un modèle donné ?
2. Proposer une architecture pour votre pricer.
3. Proposer des tests pour les différentes fonctionnalités.
4. Implémenter les différents composants : simulation du sous-jacent, payoffs, moteur Monte Carlo
   - Penser à tester séparément les différentes fonctionnalités au fur et à mesure.
   - Comment puis–je optimiser mon code ?
   - Ne pas hésiter à refactoriser le code.

Le choix de la librairie numérique

- Manipulation matricielle efficace.
- Génération de vecteurs aléatoires.

PNL : `https://github.com/pnlnum/pnl`. Librairie numérique écrite en C sous licence LGPL.
Elle est installée sur les machines de l'Ensimag dans

- `/matieres/5MMPMP6/pnl` (version optimisée)
- `/matieres/5MMPMP6/pnl-dbg` (version avec symboles de debug)

Lisez la documentation !

Compilation du projet

La compilation du projet se fera à l'aide de l'outil *CMake* dédié à la compilation multi–plateforme. Exemple type d'utilisation

```
mkdir build; cd build
cmake [-DVAR1=] [-DVAR2=] ../
```

Quelques variables utiles

- `CMAKE_PREFIX_PATH` : répertoire supplémentaire dans lequel chercher des modules.

  `cmake -DCMAKE_PREFIX_PATH=/matieres/5MMPMP6/pnl`.
- `CMAKE_CXX_COMPILER` : compilateur C++
- `CMAKE_C_COMPILER` : compilateur C
- `CMAKE_BUILD_TYPE` : Release ou Debug