M2 IF Apprentissage projet Programmation par composants

Spécifications du composant 6
Constructeur de chemin

Groupe 4
Magali BIT
Laurent MARY
Rémy MILIA

Version doc	Date	Auteur(s)	Modifications
1.0	21/02/2015	Magali BIT Laurent MARY Rémy MILIA	Version initiale
1.1	14/03/2015	Laurent MARY	Précisions dans l'enchaînement des appels
1.2	04/04/2015	Magali BIT Laurent MARY Rémy MILIA	Signatures des fonctions Valeurs de retours attendues (fonctionnement nominal et cas d'erreurs)
1.3	11/04/2015	Laurent MARY	Petite precision dans l'appel vers le GNA Gaussien Ajout de headers de fonctions à des fins de test
1.4	14/04/2015	Laurent MARY	Ajustement pour l'appel au composant 3
1.5	15/04/2015	Laurent MARY	Précisions dans l'enchaînement de appels
1.6	11/05/2015	Laurent MARY	La function getChemin prend 2 paramètres : le nombre de jours et le spot de départ
1.7	20/05/2015	Laurent MARY	Harmonisation du type des vecteurs (double) et precisions dans les appels

1. Contexte

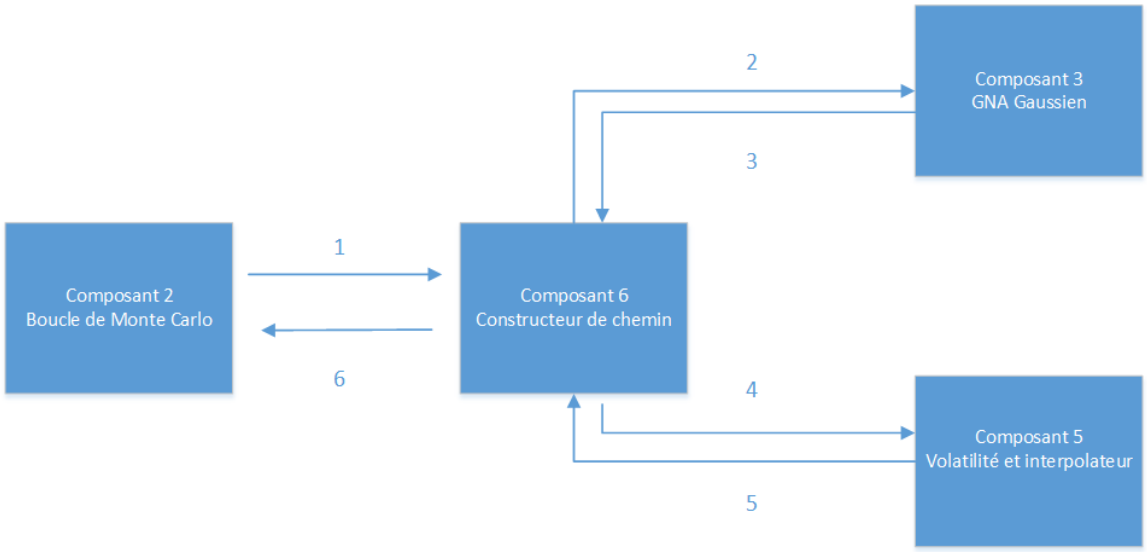
L'objectif de cette spécification est de décrire les fonctionnalités et le comportement attendu d'un composant n°6 « Constructeur de chemin » qui sera compilé sous forme d'une DLL et qui sera utilisé dans le cadre d'une application permettant le pricing d'options (européennes, américaines ou bermudéennes).

2. Objectif du composant

Le composant « Constructeur de chemin » devra être capable à partir du spot de départ S_0 et de la maturité T de fournir un vecteur de taille *size* (vecteur allant de 0 à T **inclus**) correspondant à la valeur estimée de l'actif sous-jacent pour chaque jour de cotation.

3. Fonctionnement

a. Enchaînement des appels



1. Le composant « Constructeur de chemin » est appelé par le composant « Boucle de Monte Carlo », ce dernier lui fournit en paramètre la valeur du spot de départ S_0 et la maturité T attendue

2. Le « Constructeur de chemin » appelle le « GNA Gaussien » (préalablement initialisé par l'interface XLL) avec la fonction double normalRandom(T). T désignant la maturité en jours

3. Le « GNA Gaussien » suite à l'appel précédent lui renvoie un vecteur de variables aléatoires suivant une loi normale gaussienne de moyenne 0 et de variance 1 qu'on nommera N_t $\forall t \in N^* \cap$

$[0; T-1]$ T étant la date de maturité exprimée en jours (les différentes valeurs sont donc stockées dans un vector<double> N de taille T).

4. Le « Constructeur de chemin » interroge le composant « Volatilité et interpolateur » par le biais de la fonction double getLocalVol(double strike, double maturity), il lui fournit en paramètre le strike S_t (en unité de 1) et la maturité t (exprimée en jours). **Attention : Le paramètre *strike* de *getLocalVol* est un pourcentage exprimé en unité de 1, ce qui signifie que pour un *strike* de 100, ce qui faudra passer en paramètre est 1 car le strike est exprimé en pourcentage dans ce module, en généralisant, pour un strike de K, la valeur à passer en paramètre est K/100.**

Cet appel sera répété autant de fois qu'il y a de jours de cotations soit T fois, les étapes 4 et 5 seront donc répétées T fois.

5. Le « Constructeur de chemin » récupère la volatilité locale σ_t en t (t désignant le temps courant en jours) et la stocke dans un vector<double> sigma.

6. Le module « Constructeur de chemin » renvoie au module « Boucle de Monte Carlo » un vecteur contenant les valeurs estimées du sous-jacent pour chaque jour de cotation.

a. Calcul de la valeur du sous-jacent en t

La valeur du sous-jacent S en t+1 dépend de la valeur calculée sous-jacent en t, de la volatilité locale σ_t et de la variable aléatoire N_t.

La formule permettant de calculer le prix du sous-jacent est la suivante :

$$S_{t+1} = S_t\left(1 + \frac{\sigma_t}{\sqrt{252}} N_t\right)$$

La valeur t correspond à fois au temps et à la position dans le vecteur généré. On divise par la racine de 252 car σ_t constitue une volatilité annualisée qu'on doit rapporter à une journée.

4. Header

Ce composant met à disposition une fonction polymorphe permettant d'obtenir un chemin (vecteur de « double » de taille T + 1, T désignant la maturité) à partir d'un spot de départ et d'une maturité :

vector<Double> getChemin(int jours, double spot)

Les fonctions suivantes sont également à implémenter à des fins de test :

Vector<double> getN(int jours) ; // renvoie le tableau N de taille jours (vecteur de variables aléatoires suivant une loi N(0,1)

Vector<double> getSigma(int jours) ; // renvoie le tableau sigma de taille jours (volatilité locale en t)

5. Cas d'erreurs

Dans tous les cas, lorsque le composant rencontre une erreur, il renvoie (« throw ») une chaine de caractères :

Tableau 1 Correspondance entre type d'erreur et valeur renvoyée

Libellé de l'erreur	Code d'erreur
Spot de départ (strike) négatif ou nul (non respect de l'hypothèse log-normale)	« valeur_neg : 0 »
Spot de départ (strike) trop grand	« valeur_big : 0 »
Maturité négative ou nulle	« valeur_neg : [indice_maturite] » (remplacer [indice_maturite] par le dernier indice du vecteur : la maturité T)
Taille du vecteur renvoyée par le GNA gaussien incorrecte	« unvalid_size : [size] » (remplacer size par la taille du vecteur renvoyé)
Le prix S_t du sous-jacent est négatif à une valeur t donnée	« valeur_neg : [indice] » (remplacer indice par l'indice dans le vecteur S où la condition n'est pas respectée)

On remarquera que dès le composant rencontre une erreur, il se termine immédiatement et renvoie le code erreur correspondant.