Master Probabilités et Finance:

Rapport de projet informatique

Omar El-Euch, Jonathan Visbecq

Stratification adaptative vs randomized QMC

28 avril 2015

Table des matières

1 Présentation du sujet

1.1 Sujet

Le sujet choisi est le suivant :

2.21 Stratification adaptative vs randomized QMC

Comparer les performances des méthodes de stratification exposées dans [EJ08] et de QMC randomisées exposées dans [Tuf04] sur les exemples présentés dans [EJ08].

1.2 Présentation mathématique des modèles

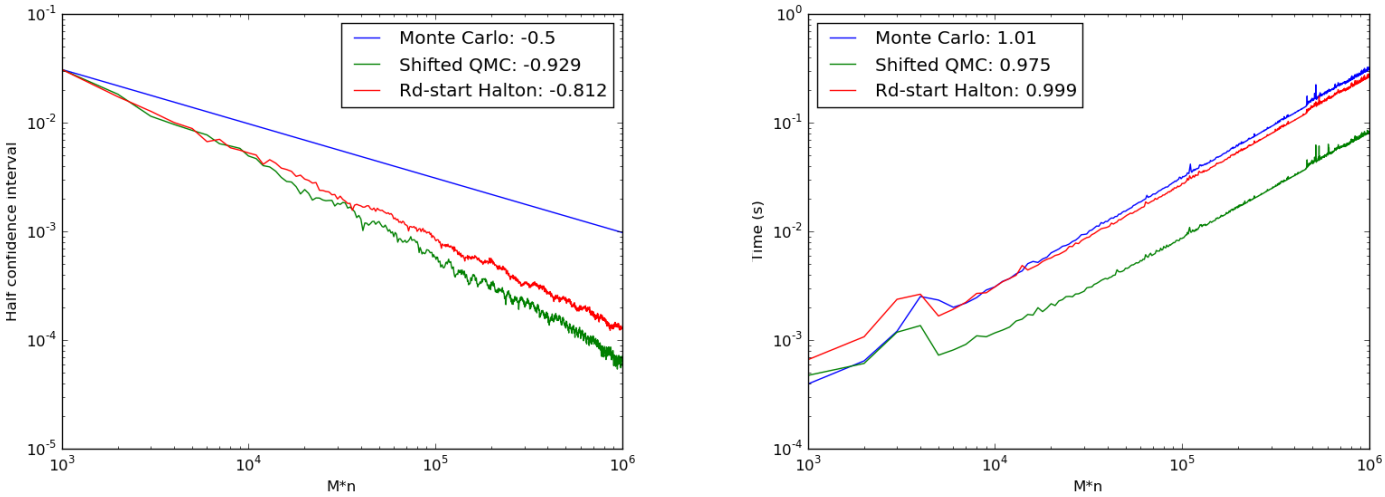
1.2.1 Quasi-Monte Carlo randomisé

1.2.2 Stratification adaptative

2 Tests numériques du code C++

2.1 Quasi-Monte Carlo randomisé

2.1.1 Comparaison des vitesses de convergence

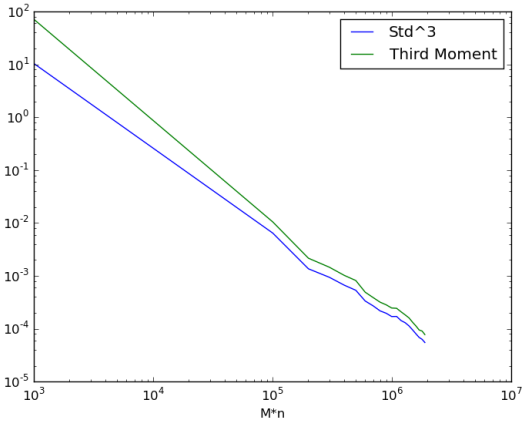


(a) Demi-interval de confiance en fonction du nombre de points générés (échelle 'log-log')

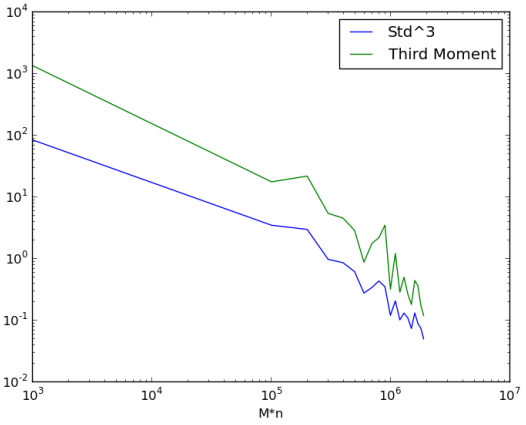
(b) Temps (s) de calcul nécessaire en fonction du nombre de points générés (échelle 'log-log')

FIGURE 2.1: Comparaison des temps et vitesses de convergence des méthodes de Monte Carlo randomisées et de Monte Carlo standard pour le calcul de l'intégrale de $f(x_1, x_2) = 1_{x_1<x_2}$. M (nombre de pseudo-nombres aléatoires) est fixé à 1000 tandis que N (nombres de quasi-nombres aléatoires) varie de 1 à 1000. Les valeurs indiquées sont les coefficients des pentes (vitesses de convergences).

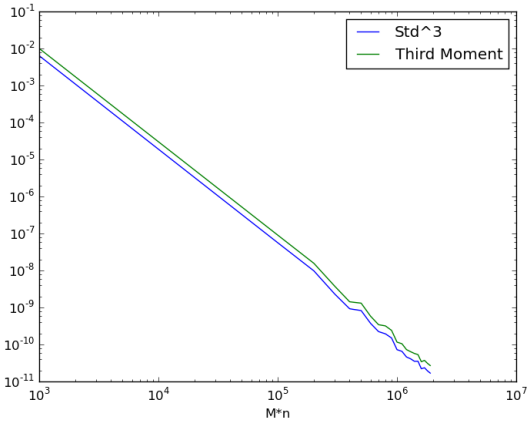
2.1.2 Étude du terme de Berry-Essen pour l'intervalle de confiance



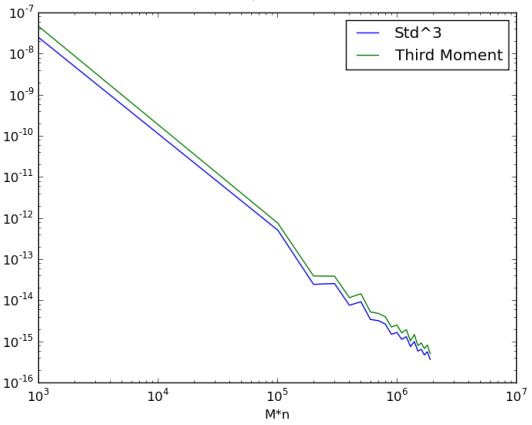
(a) $\prod_{i=1}^{10} \frac{\pi}{2} \sin(\pi x_i)$



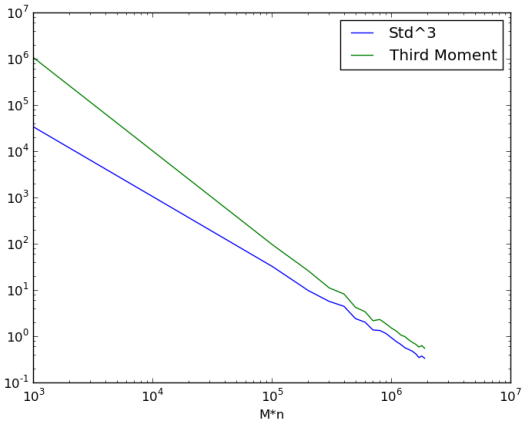
(b) $\prod_{i=1}^{10} 12(x_i - 0.5)$



(c) $\frac{1}{5} \sum_{i=1}^{10} x_i$



(d) $\prod_{i=1}^{10} \frac{e^{-|x_i - 0.5|}}{2 - 2e^{0.5}}$



(e) $\prod_{i=1}^{10} 21_{\{x_i > 0.5\}}$

FIGURE 2.2: Comparaison des vitesses de convergence de σ_N^3 et de β_N. Comme pour les simulations présentées dans [Tuf04], la vitesse de convergence est sensiblement la même pour toutes les fonctions étudiées.

2.2 Stratification adaptative

2.2.1 Convergence vers la proportion optimale

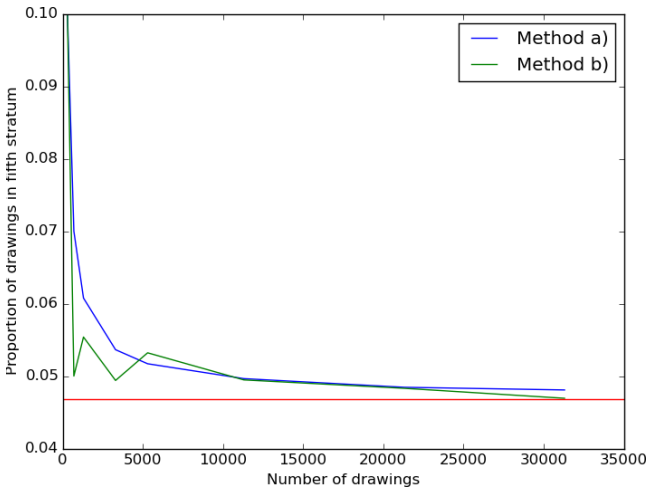
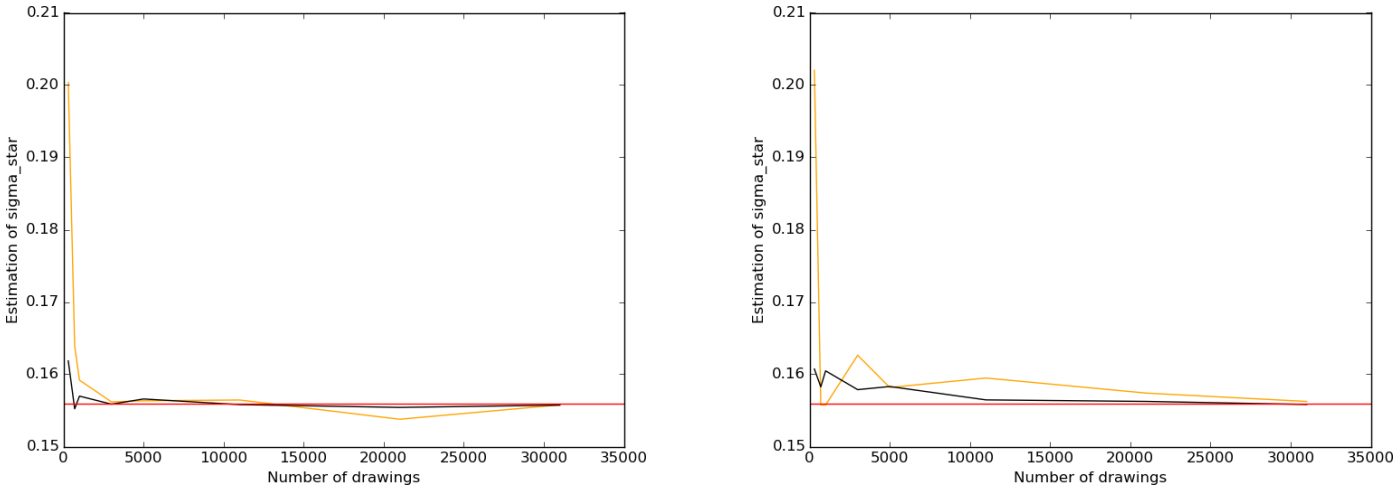


FIGURE 2.3: Convergence de $\frac{N_5^k}{N^k}$ vers la valeur optimale $q_5^* = 0.04685$ (droite rouge) en fonction de N^k, pour $k \in [|1, 8|]$ et N^k prenant les valeurs 300, 700, 1300, 3300, 5300, 11300, 21300, 31300. Les deux méthodes de calcul possible pour la stratification adaptative sont envisagées.

2.2.2 Comparaison de l'intervalle de confiance estimé à la simulation Monte Carlo



(a) Méthode a) (b) Méthode b)

FIGURE 2.4: Comparaison entre l'intervalle de confiance estimé à partir de l'estimation de la variance asymptotique σ^* (en orange) et celui obtenu par simulation Monte Carlo avec 10000 tirages (en noir) pour l'estimation de l'espérance d'une loi normale standard. Les courbes tracées sont estimations de σ^* (proportionnel à la largeur de intervalle de confiance et la vraie valeur (en rouge)). Avec notre implémentation, la méthode b) semble la moins fiable.

2.2.3 Comparaison de la stratification proportionnelle et de la méthode adaptative

	Stratification proportionnelle	Stratification adaptative
Variance	1.26e-06	7.82e-07
Temps (s)	0.0249	0.0256
Variance*temps	3.14e-08	2.00e-08

TABLE 2.1: Comparaison des résultats entre les méthodes d'allocation proportionnelles et adaptative pour l'estimation de l'espérance d'une loi normale standard. Nous utilisons 31300 tirages pour la stratification, répartie successivement en 300, 1300, 11300 puis 31300 tirages pour la méthode adaptative. L'expérience est répétée 10000 fois et les valeurs présentées sont les moyennes des résultats obtenus.

2.2.4 Comparaison avec les résultats de [EJ08] pour les options asiatiques

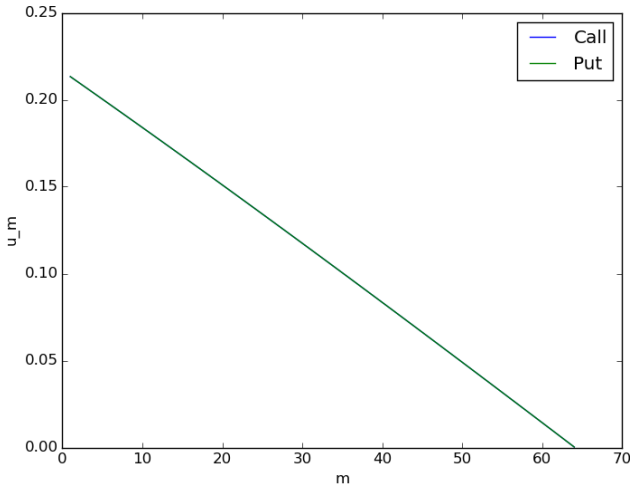
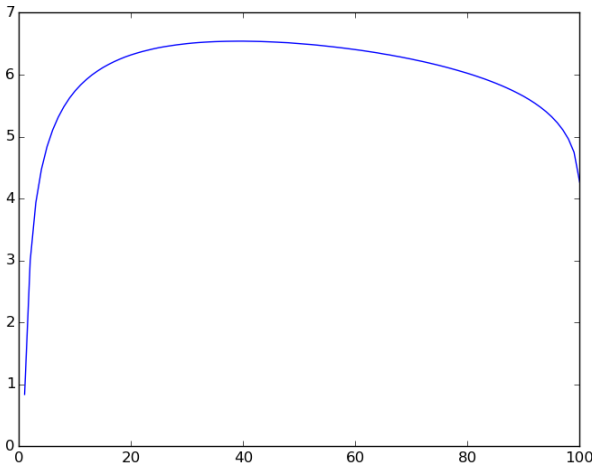
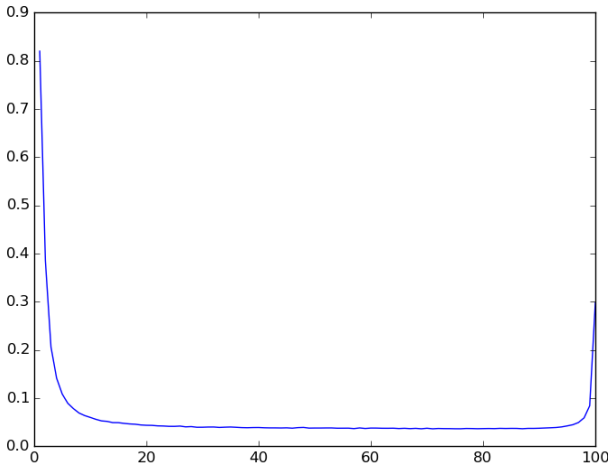


FIGURE 2.5: Composantes u_m du vecteur unitaire u donnant la direction de stratification en fonction de m pour le call et le put asiatique. Paramètres : d=64, K=45.

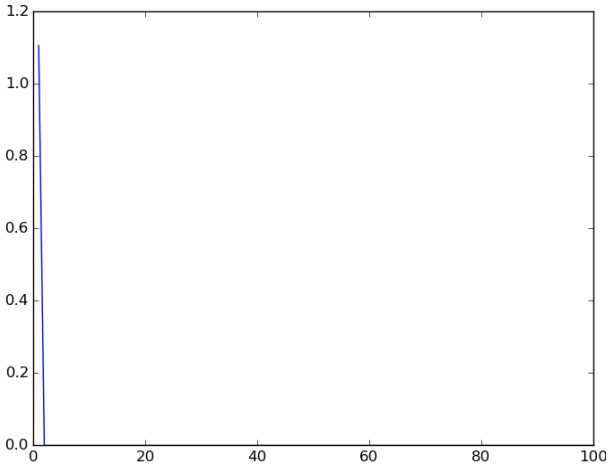


(a) Estimation de $\mathbb{E}\left[f\left(X_i\right)\right]$.

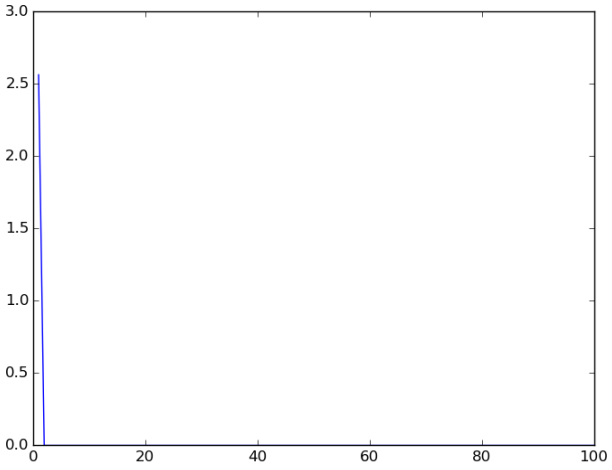


(b) Estimation de σ_i.

FIGURE 2.6: Estimation de $\mathbb{E}\left[f\left(X_i\right)\right]$ et de σ_i sur chaque strate en fonction de i pour le call asiatique. Paramètres : $d = 64$ et $K = 45$.



(a) Estimation de $\mathbb{E}\left[f\left(X_i\right)\right]$.



(b) Estimation de σ_i.

t

FIGURE 2.7: Estimation de $\mathbb{E}\left[f\left(X_i\right)\right]$ et de σ_i sur chaque strate en fonction de i pour le put asiatique. Paramètres : $d = 64$ et $K = 45$.

			Variance		
d	K	Price	Adap.	Prop.	Ratio Prop. / Adap
	45	6.05	2.63e-08	1.08e-07	4.11
16	50	1.91	1.20e-07	7.76e-06	64.69
	55	0.20	5.29e-09	4.33e-07	81.74
	45	6.00	3.53e-09	2.32e-08	6.58
64	50	1.84	1.00e-09	3.05e-09	3.04
	55	0.17	6.55e-09	7.36e-07	112.35

TABLE 2.2: Comparaison des allocations proportionnelles et adaptative pour l'exemple du call asiatique. Les ratios trouvés sont environ 2 fois supérieures à ceux de [EJ08].

			Variance		
d	K	Price	Adap.	Prop.	Ratio Prop. / Adap
	45				
16	50				
	55				
	45				
64	50				
	55				

TABLE 2.3: Comparaison des allocations proportionnelles et adaptative pour l'exemple du put asiatique avec $S0 = 50$, $r = 0.05$, $\sigma = 0.1$, $T = 1.$ et 100 strates. 1000000 de tirages sont utilisés, répartie progressivement en 100000, 400000, 500000 et 1000000 pour la méthode adaptative.

3 Comparaison des deux méthodes de réducti de variance sur les exemples de [EJ08]

3.0.5 Loi normale standard

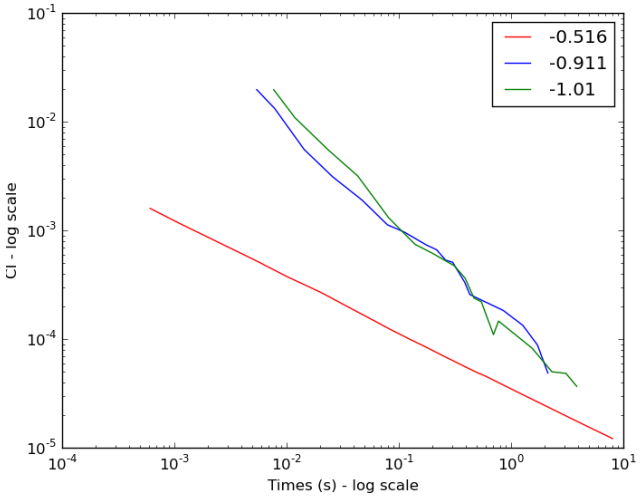
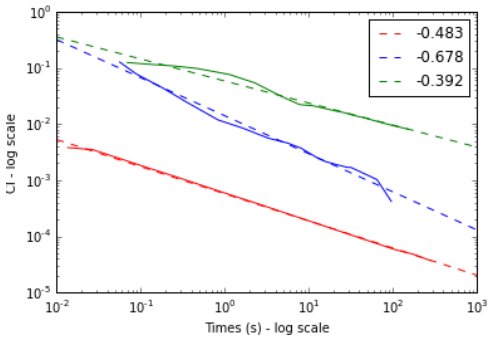
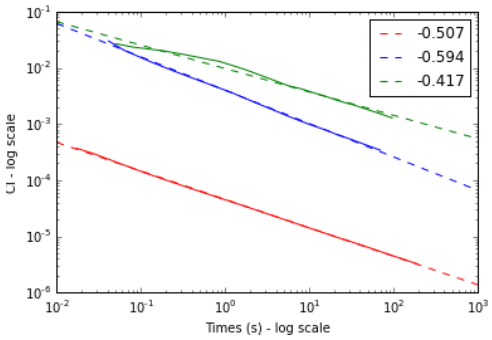


FIGURE 3.1: Etude des vitesses de convergence entre les trois méthodes de réduction de variance : stratification adaptative (en rouge), Shifted SQRT (en bleu) et Random-start Halton (en vert) pour l'example de la gaussienne standard. Les valeurs indiquées sont les pentes des droites (vitesses de convergence).

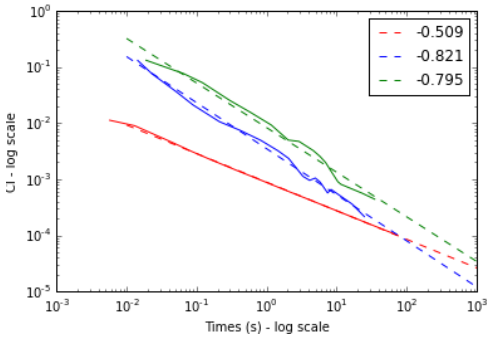
3.0.6 Options asiatiques



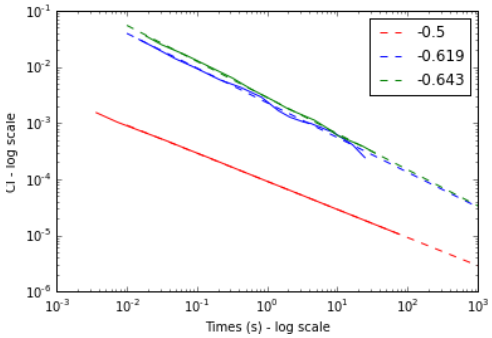
(a) $d = 64$ et $K = 45$



(b) $d = 64$ et $K = 45$



(c) $d = 64$ et $K = 45$



(d) $d = 64$ et $K = 45$

FIGURE 3.2: Etude des vitesses de convergence entre les trois méthodes de réduction de variance : stratification adaptative (en rouge), Shifted SQRT (en bleu) et Random-start Halton (en vert) pour l'example du call asiatique. Les valeurs indiquées sont les pentes des droites (vitesses de convergence).

4 Présentation du code C++

L'intégrité du code informatique est écrit en C++11. Le type de programmation employée est générique, afin de faciliter l'écriture des tests et scripts.

Le code se compose notamment de :

— Alias ARRAY<DIM> et fonctions template du fichier *my_array.hpp* pour simuler les opérations basiques d'algèbre linéaire sans avoir à faire appel à une librairie externe.

— Des classes représentant des générateurs (*generator.hpp*) de nombres pseudo-aléatoires ou quasi-aléatoires respectant les caractéristiques suivantes :
- argument template *unsigned dim* représentant la dimension des points à générer.
- surcharge de l'opérateur () sans arguments renvoyant un *Array<dim>*.

Les générateurs présents sont les suivants :
. UNIFORM_GEN (*uniform_generator.hpp*) qui appel une fonction externe dont un champ statique contient un générateur MT19937 de la librairie standard. Ainsi, toutes les instance de cette classes représenteront des variable aléatoires uniforme indépendantes.
. UNIFORM_GEN_FIXED (*uniform_generator.hpp*) qui utilise un générateur MT19937 différent pour chaque instance mais de seed identique.
. KAKUTANI (*kakutani.hpp*) représentant une suite de Kakutani.
. HALTON (*kakutani.hpp*) représentant une suite de Halton comme classe dérivée d'une suite de Kakutani.
. HALTON_FAST (*kakutani.hpp*) représentant une suite de Halton, mais pour laquelle la génération des points est plus rapide qu'avec l'addition de nombre p-adique vue en cours. La méthode est inspirée de ...
. FAURE (*faure.hpp*) représentant une suite de Faure.
. TORE (*tore.hpp*) représentant une suite générée par rotation ...
. SQRT (*tore.hpp*) représentant ...

— Des distributions (*distribution.hpp*), permettant de calculer des réalisations de variables aléatoires à partir des points générés par un générateur. Les distribution respectent les caractéristique suivantes :
- typedef *result_type* pour le type de retour de la distribution
- attribut *static constexpr unsigned dim_alea* pour la dimension des points nécessaires au calcul des réalisations
- surcharge de l'opérateur () prenant en argument un *const Array<dim>&* et renvoyant un objet de type *result_type*

Les distributions présentes sont les suivantes :
. UNIFORM (*uniform.hpp*) pour les variables aléatoires uniformes sur un intervalle de $\mathbb{R}$
. GAUSSIAN_IND<unsigned dim> (*gaussian_ind.hpp*) pour les variables aléatoires gaussiennes multidimensionnelles dont toutes les composantes sont mutuellement indépendantes. On utilise la méthode de Box-Müller pour être compatible avec les générateurs QMC.
. STDBROWNIAN<unsigned t_dim> (*process.hpp*) pour les mouvements Browniens (le type

de retour est un alias sur un tableau de pairs (temps,valeur)). Le template repsésente le nombre de temps du processus.
. BLACK_SCHOLES<unsigned t_dim> pour les processus de Black-Scholes (*process.hpp*).
. NORMAL_INTERVAL<unsigned dim> (*normal_interval.hpp*)pour les variable aléatoires gaussiennes multidimensionnelles standards conditionnellement à ce que leur projection sur un axe soit dans un intervalle donné (pour les méthodes de stratification). On utilise ici une approximation de l'inverse de la fonction de répartition par la méthode de Beasley-Springer-Moro [Glasserman 2004].
. COMPOSED_DIST<typename Dist> permettant de composer une distribution avec une fonction.

— Sont également implémentés comme des distributions les méthodes de Monte Carlo randomisées (pour pouvoir les passer à un objet MONTE_CARLO, voire ci-après) :
. SHIFTED_QMC<typename Dist, typename QMC_Generator> (*shifted_qmc.hpp*) pour les méthodes de quasi-Monte Carlo translatées.
. RANDSTART_HALTON<typename Dist> (*random_start_halton.hpp*) pour la méthode de Halton à point initial aléatoire.

— RAND_VAR (*rand_var.hpp*)<typename Dist, typename Generator> permet d'associer un générateur à une distribution.

— MONTE_CARLO<typename Dist> pour les simulations Monte Carlo. Le choix adopté est d'avoir la distribution en argument template et de passer à chaque appel le générateur à utiliser et le nombre de réalisations, afin de faciliter les comparaisons entre plusieurs générateurs.

— STRATIFICATION<template <unsigned> Dist_Strat, unsigned dim, nb_strats> (*stratification.hpp*) implémentant la méthode de stratification standard pour l'évaluation de fonctionnelles de la distribution Dist_Strat¡dim¿ en dimension dim, avec nb_strats strates. A l'instanciation, la fonctionnelle, les bornes des strates, la direction de stratification et les probabilités des strates sont précisées. L'appel à l'algorithme nécessite un générateur, le nombre de réalisation et les allocations sur chaque strate (dont l'omission conduit à l'algorithme proportionnel).

— ADAPTIVE_STRATIFICATION<template <unsigned> Dist_Strat, unsigned dim, nb_strats> (*adaptive_stratification.hpp*) pour l'algorithme de stratification adaptative de [EJ08]. Les paramètres d'instanciation sont les mêmes que précédemment, plus le choix de la méthode (a) ou b)). La première de l'appeler est de fournir le générateur, et un tableau contenant les nombres successifs de réalisations à chaque itération. Une seconde fonction d'appel permet de réaliser un itération à la fois affin de raffiner. Cela permet d'avoir accès à l'état de l'algorithme itération par itération.

— P_ADIC (*p_adic.hpp, p_adic.cpp*) permet de représenter et de manipuler les nombres p-adiques.

Le fichier *payoff.hpp* contient notamment les fonctions de payoff des options asiatiques nécessaires pour le deuxième exemple de [EJ08].

Enfin, le *scripts.hpp* et *scripts.cpp* contiennent les fonctions effectuant les simulations et tests et sont

lancés depuis le *main.cpp*.

Bibliographie

[EJ08] P. Etore and B. Jourdain. Adaptive optimal allocation in stratified sampling methods. *Methodology and Computing in Applied Probability*, pages 1–26, 2008.

[GHS99] P. Glasserman, P. Heidelberger, and P. Shahabuddin. Asymptotically optimal importance sampling and stratification for path-dependant options. *Mathematical Finance*, 9(2) :117–152, 1999.

[Tuf04] B. Tuffin. Randomization of quasi-monte carlo methods for error estimation : survey and normal approximation. *Monte Carlo Methods Appl.*, 10(3-4) :617–628, 2004.