TD Monte Carlo et simulation

Il est conseillé d'effectuer les TD en R, mais les étudiants qui souhaitent utiliser un autre langage de programmation (Matlab, Python par ex.) peuvent en demander l'autorisation à leur chargé de TD.

1 RANDU

Le générateur congruentiel RANDU, défini par la récurence (sur l'ensemble des entiers $\{1, \dots, 2^{31} - 1\}$)

$$x_i = 65539 x_{i-1} \mod 2^{31}$$

et $u_i = x_i / 2^{31} \in (0, 1)$ a acquis la réputation du plus mauvais générateur jamais utilisé en pratique. Le but de l'exercice est de vérifier que cette réputation n'est pas usurpée.

1. Montrer que les triplets (u_i, u_{i-1}, u_{i-2}) se retrouvent forcément sur 15 hyperplans dans $[0, 1]^3$. (Note: $65539 = 2^{16} + 3$)
2. Programmer ce générateur, générer 20000 points, et tracer u_i en fonction de u_{i-2} pour tous les i tels que $0.5 \leq u_{i-1} \leq 0.51$. Commenter.
3. Tracer ensuite en 3 dimensions les triplets (u_i, u_{i-1}, u_{i-2}) pour illustrer le point 1.
4. Un autre défaut de RANDU est la faible période des bits de poids faible des x_i. Mettre en évidence ce phénomène.
5. Promettre à son chargé de TD que même sous la torture l'on n'utilisera jamais RANDU (ou tout autre générateur obsolète).
6. Si vous avez le temps, vous pouvez vérifier numériquement que le générateur par défaut sous votre logiciel a de meilleures propriétés que RANDU. (Réfléchir à différentes façons de tester un générateur uniforme.)

2 Rejet et loi de Laplace

1. Proposer et programmer un générateur de la loi de Laplace:

$$p(x) = \frac{1}{2} \exp(-|x|)$$

2. Proposer un générateur de la loi $N(0, 1)$ basé sur l'algorithme d'acceptation-rejet et la loi de Laplace. Le programmer et le tester (justifier).
3. Donner le taux d'acceptation de l'algorithme. Simplifier la condition d'acceptation.
4. Peut-on faire l'inverse? (i.e. générer la loi de Laplace par rejet, en utilisant une loi Normale.)

3 Box-Muller amélioré

En cours, on a évoqué l'algorithme de Box-Muller amélioré pour simuler selon une loi $N(0, 1)$:

- Générer U_1, $U_2 \sim \mathcal{U}[-1, 1]$ jusqu'à ce que $U_1^2 + U_2^2 \leq 1$.
- Renvoyer $X = U_1 \sqrt{-2(\log S)/S}$ et $Y = U_2 \sqrt{-2(\log S)/S}$, où $S = U_1^2 + U_2^2$.

1. Démontrer que cet algorithme est bien tel que X, $Y \sim N(0,1)$, indépendamment.
2. Comment peut-on comparer la performance de cet algorithme avec celle de l'algorithme de Box-Muller standard? Mettre en oeuvre cette comparaison.

Rappel: Box-Muller simule U_1, $U_2 \sim \mathcal{U}[0,1]$ et renvoie $X = \sqrt{-2\log U_1}\sin(2\pi U_2)$, $Y = \sqrt{-2\log U_1}\cos(2\pi U_2)$.

4 Loi géométrique

Proposer un algorithme pour simuler selon une loi géométrique. Le mettre en oeuvre, et proposer une méthode pour vérifier les résultats.

5 Control variates, variables antithétiques, QMC

Soit X un vecteur Gaussien de dimension $d \geq 2$, de moyenne nulle et de variance identité. On veut calculer la probabilité que $X \in A$ pour une certaine région A. On prend par exemple:

$$A = \{(x_1, \ldots, x_d), \quad \left|\prod_{i=1}^{d} x_i\right| \leq c\}.$$

1. Proposer un algorithme de MC pour approcher cette probabilité. Quel problème se pose lorsque c devient petit?
2. Utiliser le principe des variables de contrôle pour proposer un algorithme amélioré. (Quelles variables de contrôle peut-on prendre ici)?
3. Peut-on utiliser le principe des variables antithétiques pour améliorer l'estimateur de MC standard?
4. Facultatif: commment utiliser le Quasi-Monte Carlo pour améliorer les méthodes précédentes? Faire la comparaison numérique avec les algorithmes précédents.

6 Importance sampling

On veut cette fois calculer des espérances selon la loi de X conditionnellement à l'événement $X \in A$; expliquer pourquoi cette loi conditionnelle à pour densité:

$$f(x) = \frac{(2\pi)^{-d/2}}{P(X \in A)} \exp\{-\frac{1}{2}\sum_{i=1}^{d} x_i^2\}.$$

1. Un de vos camarades vous propose la méthode suivante: (a) simuler des X_i selon la loi Normale centrée réduite de dimension d; (b) calculer la moyenne $\hat{\mu}$ et la matrice empirique $\hat{\Sigma}$ du sous-échantillon des X_i qui sont tombés dans l'ensemble A; (c) faire de l'importance sampling avec loi de proposition la loi $N(\hat{\mu}, \hat{\Sigma})$. Est-ce à votre avis une bonne idée? (Vous pouvez répondre soit formellement, soit en essayant de mettre en oeuvre cette méthode, par exemple pour estimer l'espérance de la fonction $\log(1 + |x_1 - x_2|)$ pour $d = 2$).
2. Modifer la méthode proposée pour obtenir une méthode valide.

7 MCMC

Soit la loi de densité

$$p(x_1, x_2) = \frac{1}{2\pi\sqrt{1-\rho^2}} \exp\left\{-\frac{1}{2(1-\rho^2)}(x_1^2 + x_2^2 - 2\rho x_1 x_2)\right\}$$

avec $\rho \in [-1, 1]$.

1. Quelle loi reconnaissez-vous?
2. Calculer les lois conditionnelles de chaque composante, et mettre en oeuvre le Gibbs sampler correspondant. Représenter dans le plan des contours de la densité cible, et l'évolution de la chaîne de Markov simulée. Que se passe-t-il quand $\rho \to 1$?
3. Programmer un algorithme RWHM (Hastings-Metropolis avec marche aléatoire), où la variance de la loi de proposition est Σ. Commencer par $\Sigma = \tau I_2$, et essayer de voir ce qui se passe pour différentes valeurs de τ, en particulier quand ρ est proche de 1. Essayer ensuite avec Σ proportionnel à la la variance de la loi cible. Commenter.

8 Erreur de discrétisation dans le pricing d'option

1. Soit (W_t) un processus de Wiener. Retrouver la loi de W_t conditionellement en W_a et W_b pour $a \leq t \leq b$.
2. Soit le processus de Black-Scholes défini par

   $$S(t) = \exp\{(\mu - \sigma^2/2)t + \sigma W_t\}$$

   où de manière équivalente par l'équation stochastique:

   $$dS(t) = \mu S(t)\, dt + \sigma S(t) dW_t$$

Calculer par Monte Carlo le prix d'une option européenne définie par:

$$V = \mathbb{E}\left[e^{-rT}\left(K - S(T)\right)^+\right]$$

pour $T = 1$, $r = 0.02$, $\mu = 0.5$, $\sigma = 0.5$, $K = 2$.

3. On traite désormais le processus de BS comme un processus que l'on ne sait pas simuler exactement (à certains temps donnés), et qu'il faut donc discrétiser. Utiliser la question 1. pour construire un algorithme itératif qui réduit graduellement le pas de discrétisation, jusqu'à ce que l'erreur de discrétisation semble petite. **Indication**: à l'étape k, le pas de discrétisation sera $T2^{-k-k_0}$, et le but est de recycler les simulations de l'étape $k-1$ en insérant 2^{-k-k_0+1} nouveaux points. On pourra comparer les intervalles de confiance sur V obtenus à l'itération k et $k-1$, et décider en fonction s'il faut continuer, ou s'arrêter, ou éventuellement augmenter le nombre de simulations pour permettre une comparaison plus fine.

9 Cross-Entropy Method

La fonction de Rosenbrock en dimension d est définie de la façon suivante:

$$S(x) = \sum_{i=1}^{d} 100(x_{i+1} - x_i)^2 + (x_i - 1)^2$$

et admet comme minmimum global le point $x^\star = (1, \ldots, 1)$. C'est un benchmark populaire en optimisation.

Proposer un algorithme de type CE (Cross-Entropy) pour obtenir le minimum global d'une fonction quelconque $S : \mathbb{R}^d \to \mathbb{R}$, le programmer et l'appliquer à la fonction de Rosenbrock, pour différentes valeurs de d.