Méthodes de Monte-Carlo et algorithmes stochastiques – TD/TP 1

Fondements de Monte-Carlo : représentations, variance et précision

M2 Finance – Université Gustave Eiffel

Objectifs

Cette séance reprend le début du chapitre 1. L'objectif est de savoir choisir une représentation probabiliste, construire un estimateur, quantifier sa variance et dimensionner le nombre de simulations.

Exercice 1 – Une même intégrale, plusieurs estimateurs

On considère l'intégrale

$$I = \int_0^1 \cos\left(x^3\right) \mathrm{e}^{-x}\, dx.$$

Le but est de construire plusieurs estimateurs de Monte-Carlo de la même quantité. On rappelle qu'un estimateur est dit sans biais si son espérance vaut exactement la quantité cherchée.

1) Soit $U \sim \mathcal{U}([0,1])$. Montrer que

$$I = \mathbb{E}\left[\cos\left(U^3\right) \mathrm{e}^{-U}\right].$$

En déduire un premier estimateur $\widehat{I}_n^{(1)}$.

2) Soit $X \sim \mathcal{E}(1)$, c'est-à-dire de densité $x \mapsto \mathrm{e}^{-x}\mathbf{1}_{x\geq 0}$. Montrer que

$$I = \mathbb{E}\left[\cos\left(X^3\right) \mathbf{1}_{\{X\leq 1\}}\right].$$

En déduire un second estimateur $\widehat{I}_n^{(2)}$.

3) Soit maintenant Y une variable de densité

$$p(y) = \frac{\mathrm{e}^{-y}}{1-\mathrm{e}^{-1}} \mathbf{1}_{0\leq y\leq 1}.$$

Montrer que

$$I = \left(1-\mathrm{e}^{-1}\right) \mathbb{E}\left[\cos\left(Y^3\right)\right].$$

En déduire un troisième estimateur $\widehat{I}_n^{(3)}$.

4) Les trois estimateurs sont-ils sans biais ? Pour comparer leur précision, on note

$$\begin{aligned} Z_1 &= \cos(U^3)\mathrm{e}^{-U}, \\ Z_2 &= \cos(X^3)\mathbf{1}_{\{X\leq 1\}}, \\ Z_3 &= (1-\mathrm{e}^{-1})\cos(Y^3), \end{aligned}$$

de sorte que chaque estimateur est la moyenne de n copies indépendantes de la variable correspondante Z_j.

Montrer que

$$\mathrm{Var}\left(\widehat{I}_n^{(j)}\right) = \frac{\mathrm{Var}(Z_j)}{n}.$$

Puis montrer que, bien que

$$\mathrm{Var}(Z_j) = \mathbb{E}(Z_j^2) - I^2,$$

il n'est pas nécessaire de connaître I pour comparer les trois variances : il suffit de comparer les seconds moments $\mathbb{E}(Z_j^2)$.

Écrire explicitement ces trois seconds moments et montrer directement que

$$\mathrm{Var}\left(\widehat{I}_n^{(3)}\right) < \mathrm{Var}\left(\widehat{I}_n^{(2)}\right).$$

La comparaison numérique complète des trois estimateurs sera effectuée dans le TP Python de l'exercice 6. Pourquoi la variance est-elle un critère essentiel pour choisir entre plusieurs estimateurs sans biais ?

Idée à retenir

Lorsque plusieurs estimateurs sans biais visent la même quantité I, le terme $-I^2$ est commun à toutes leurs variances. Pour les comparer, il n'est donc pas nécessaire de connaître I : comparer les seconds moments des variables que l'on moyenne suffit. C'est particulièrement utile en Monte-Carlo, puisque I est précisément la quantité que l'on cherche à estimer.

Exercice 2 – Estimation de π, variance et taille d'échantillon

Soit $(U_i, V_i)_{i\geq 1}$ une suite de variables indépendantes uniformes sur le carré $[-1,1]^2$. On pose

$$B_i = \mathbf{1}_{\{U_i^2+V_i^2\leq 1\}}.$$

1) Calculer $p = \mathbb{E}(B_1)$. En déduire un estimateur naturel de π.
2) Calculer la variance de cet estimateur en fonction de p et n.
3) Construire un estimateur empirique de cette variance à partir des tirages $B_1, \ldots, B_n$.
4) En utilisant l'inégalité de Chebyshev et la borne $p(1-p) \leq 1/4$, trouver une condition suffisante sur n pour que
$$\mathbb{P}\left(|\widehat{\pi}_n - \pi| \leq 0.01\right) \geq 0.95.$$
5) En utilisant l'approximation normale, donner une taille d'échantillon approximative beaucoup moins pessimiste. On pourra utiliser $p \simeq \pi/4$.

Idée à retenir

Chebyshev est universel mais très conservateur. L'approximation normale est asymptotique, donc moins sûre pour de très petits échantillons, mais beaucoup plus informative en pratique lorsque n est grand.

Exercice 3 – Dimensionnement d'une simulation par le TCL

On s'intéresse à

$$I = \int_0^1 \sin(\sqrt{x})\,\mathrm{d}x.$$

1) Si $U \sim \mathcal{U}([0,1])$, écrire I comme une espérance et proposer l'estimateur de Monte-Carlo naturel $\widehat{I}_n$.
2) Calculer explicitement I en effectuant le changement de variable $x = t^2$.
3) On note $\sigma^2 = \mathrm{Var}(\sin(\sqrt{U}))$. Donner la variance de $\widehat{I}_n$ et un intervalle de confiance asymptotique de niveau 99%.

4) Donner une condition sur n, en fonction de σ, pour que la demi-largeur de cet intervalle soit au plus 0.01. On pourra utiliser $z_{0.995} \simeq 2.576$.

5) Calculer σ^2 explicitement puis donner une valeur numérique du nombre de simulations obtenu.

Idée à retenir

Le TCL transforme une exigence de précision en une règle de dimensionnement très concrète. La dépendance en $1/\sqrt{n}$ implique qu'une précision dix fois meilleure coûte environ cent fois plus de simulations si l'on ne réduit pas la variance.

Exercice 4 – Aiguille de Buffon : une autre estimation de π

On jette une aiguille de longueur 2 sur un parquet dont les rainures parallèles sont espacées de 2. Par symétrie, on peut modéliser la distance X du centre de l'aiguille à la rainure la plus proche par $X \sim \mathcal{U}([0,1])$, et l'angle Θ avec les rainures par $\Theta \sim \mathcal{U}([0,\pi/2])$, indépendamment de X.

1) Montrer que l'aiguille coupe une rainure si et seulement si $X \leq \sin\Theta$.

2) En déduire que la probabilité de croisement vaut $p = 2/\pi$.

3) À partir de n couples (X_i, Θ_i), proposer une estimation Monte-Carlo de p, puis de π.

4) L'estimateur de π ainsi obtenu est-il exactement sans biais ? Expliquer brièvement.

5) Proposer quelques lignes de Python permettant de vérifier numériquement la convergence.

Exercice 5 – Volume d'une boule par Monte-Carlo

Soient $(U_i^{(1)}, U_i^{(2)}, U_i^{(3)})$, $i \geq 1$, des vecteurs i.i.d. uniformes dans le cube $[-1,1]^3$. On pose

$$B_i = \mathbf{1}_{\{(U_i^{(1)})^2+(U_i^{(2)})^2+(U_i^{(3)})^2 \leq 1\}}.$$

1) Calculer $\mathbb{E}(B_i)$ en fonction du volume V_3 de la boule unité de $\mathbb{R}^3$.

2) En déduire un estimateur Monte-Carlo de V_3, puis retrouver la valeur théorique $4\pi/3$.

3) Donner la variance de cet estimateur.

4) Généraliser l'idée à la boule unité de $\mathbb{R}^d$. Pourquoi cette méthode devient-elle peu efficace lorsque d est grand ?

5) Écrire quelques lignes de Python qui estiment V_3 pour $n = 10^3, 10^4, 10^5$.

Exercice 6 – TP Python : comparer plusieurs estimateurs d'une même intégrale

On reprend l'intégrale de l'exercice 1 :

$$I = \int_0^1 \cos\left(x^3\right) \mathrm{e}^{-x}\, dx.$$

On compare trois représentations probabilistes : uniforme, exponentielle non tronquée, exponentielle tronquée. L'objectif n'est pas seulement d'obtenir un nombre : il faut comparer les variances et comprendre pourquoi certains estimateurs sont plus efficaces que d'autres.

À programmer vous-même

Dans ce TP, une partie du code utilitaire est fournie, mais les fonctions correspondant aux estimateurs Monte-Carlo sont à compléter. Le point important est d'écrire vous-même les variables aléatoires dont on prend la moyenne empirique.

Code de départ

```python
import numpy as np
from scipy.integrate import quad

rng = np.random.default_rng(123)


def mc_summary(samples):
    """Renvoie moyenne, variance empirique des valeurs et IC asymptotique 95%."""
    n = len(samples)
    mean = np.mean(samples)
    sd = np.std(samples, ddof=1)
    half_width = 1.96 * sd / np.sqrt(n)
    return mean, sd**2, mean - half_width, mean + half_width


def estimator_uniform(n):
    # TODO : simuler U uniforme sur [0,1]
    # TODO : renvoyer le tableau cos(U**3) * exp(-U)
    pass


def estimator_exponential(n):
    # TODO : simuler X exponentielle de parametre 1
    # TODO : renvoyer le tableau cos(X**3) * 1_{X <= 1}
    pass


def estimator_truncated_exponential(n):
    # TODO : simuler Y de densite proportionnelle a exp(-y) sur [0,1]
    # Indication : si U est uniforme sur [0,1], poser
    # Y = -log(1 - U * (1 - exp(-1)))
    # TODO : renvoyer le tableau (1-exp(-1))*cos(Y**3)
    pass

true_value, _ = quad(lambda x: np.cos(x**3) * np.exp(-x), 0.0, 1.0)
print("valeur de reference :", true_value)

for n in [1_000, 10_000, 100_000]:
    print("\nn =", n)
    # TODO : appeler les trois fonctions ci-dessus et afficher mc_summary(...)
```

Décodage du code Python

- `mc_summary(samples)` est fourni pour éviter de réécrire à chaque fois les mêmes calculs. Si `samples` contient les valeurs individuelles $Z_1, \dots, Z_n$ que l'on moyenne, la fonction renvoie leur moyenne $\widehat{I}_n$, leur variance empirique S_n^2, puis l'intervalle de confiance construit à partir de l'erreur-type $S_n/\sqrt{n}$.
- Les fonctions `estimator_uniform`, `estimator_exponential` et `estimator_truncated_exponential` doivent donc renvoyer le *tableau des n valeurs individuelles à moyenner*, et non directement leur moyenne. Par exemple, la première fonction doit renvoyer les valeurs $\cos(U_k^3)\mathrm{e}^{-U_k}$, $1 \le k \le n$.
- Dans la loi exponentielle tronquée, la formule donnée vient de la méthode d'inversion appliquée à la fonction de répartition conditionnelle sur $[0,1]$.
- `quad(...)` sert uniquement de valeur de référence pour comparer les résultats numériques. Ce n'est pas la méthode Monte-Carlo.

1) Compléter les trois fonctions d'estimation définies dans le code de départ.
2) Pour chaque valeur de $n \in \{10^3, 10^4, 10^5\}$, afficher l'estimation $\widehat{I}_n$, la variance empirique S_n^2 des n valeurs individuelles qui sont moyennées, ainsi que l'intervalle de confiance. En déduire également, si besoin, une estimation de la variance de $\widehat{I}_n$ en divisant S_n^2 par n.
3) Les intervalles de confiance contiennent-ils la valeur de référence ? Est-ce systématique ?

4) Comparer les variances empiriques des valeurs individuelles qui sont moyennées. Pour un même n, cela revient aussi à comparer les variances estimées des trois estimateurs Monte-Carlo, puisque celles-ci valent S_n^2/n. Quel estimateur choisiriez-vous ?

5) Refaire l'expérience avec une graine aléatoire différente. Les conclusions changent-elles ?