Correction – TD/TP 4

Réduction de variance II : stratification, importance sampling et événements rares

M2 Finance – Université Gustave Eiffel

Exercice 1 – Stratification : allocation proportionnelle ou allocation de Neyman ?

1) Pour l'estimateur naïf,

$$\widehat{I}_n = \frac{1}{n}\sum_{i=1}^{n} U_i^2.$$

On a

$$\mathbb{E}(U^2) = \frac{1}{3}, \qquad \mathbb{E}(U^4) = \frac{1}{5},$$

donc

$$\mathrm{Var}(U^2) = \frac{1}{5} - \frac{1}{9} = \frac{4}{45}.$$

Ainsi

$$\boxed{\mathrm{Var}(\widehat{I}_n) = \frac{4}{45n}}.$$

2) Les deux strates ont même probabilité :

$$p_1 = p_2 = \frac{1}{2}.$$

Puis

$$I_1 = 2\int_0^{1/2} x^2\,\mathrm{d}x = \frac{1}{12},$$
$$I_2 = 2\int_{1/2}^{1} x^2\,\mathrm{d}x = \frac{7}{12}.$$

On vérifie

$$\frac{1}{2}\frac{1}{12} + \frac{1}{2}\frac{7}{12} = \frac{1}{3}.$$

3) Dans la première strate,

$$\mathbb{E}(U^4 \mid U \in A_1) = 2\int_0^{1/2} x^4\,\mathrm{d}x = \frac{1}{80},$$

donc

$$\sigma_1^2 = \frac{1}{80} - \frac{1}{144} = \boxed{\frac{1}{180}}.$$

Dans la seconde,

$$\mathbb{E}(U^4 \mid U \in A_2) = 2\int_{1/2}^{1} x^4\,\mathrm{d}x = \frac{31}{80},$$

d'où

$$\sigma_2^2 = \frac{31}{80} - \left(\frac{7}{12}\right)^2 = \boxed{\frac{17}{360}}.$$

La seconde strate est nettement plus variable.

4) Avec $n_1 = n_2 = n/2$,

$$\begin{aligned}\mathrm{Var}(\widehat{I}_{\mathrm{strat}}) &= \frac{p_1^2\sigma_1^2}{n/2} + \frac{p_2^2\sigma_2^2}{n/2} \\ &= \frac{1}{2n}\left(\frac{1}{180} + \frac{17}{360}\right) = \boxed{\frac{19}{720n}}.\end{aligned}$$

Le rapport des variances naïve / stratifiée vaut

$$\frac{4/45}{19/720} = \frac{64}{19} \simeq 3.37.$$

À nombre total de simulations identique, la stratification proportionnelle divise donc déjà la variance par environ 3.37.

5) L'allocation de Neyman vérifie

$$n_k \propto p_k\sigma_k.$$

Ici $p_1 = p_2$, donc

$$\frac{n_2}{n_1} = \frac{\sigma_2}{\sigma_1} = \sqrt{\frac{17/360}{1/180}} = \boxed{\sqrt{\frac{17}{2}}} \simeq 2.92.$$

Pour $n = 120$,

$$n_1 \simeq 30.65, \qquad n_2 \simeq 89.35.$$

En pratique on pourrait prendre, par exemple, $n_1 = 31$ et $n_2 = 89$.

6) La variance minimale théorique est

$$\begin{aligned}\mathrm{Var}_{\min} &= \frac{1}{n}\left(p_1\sigma_1 + p_2\sigma_2\right)^2 \\ &= \frac{1}{n}\left(\frac{1}{2}\sqrt{\frac{1}{180}} + \frac{1}{2}\sqrt{\frac{17}{360}}\right)^2 \\ &= \boxed{\frac{19 + 2\sqrt{34}}{1440n}}.\end{aligned}$$

Numériquement, les coefficients devant $1/n$ sont

méthode	coefficient de variance
naïve	0.08889
stratification proportionnelle	0.02639
allocation de Neyman	0.02129

Le gain naïve / Neyman est environ 4.17 en variance.

7) Une procédure pratique est :

   a) effectuer un petit échantillon pilote dans chaque strate ;

   b) estimer σ_1, σ_2 par les écarts-types empiriques ;

   c) allouer le budget restant proportionnellement à $p_k\widehat{\sigma}_k$;

   d) imposer éventuellement un nombre minimal de tirages par strate et arrondir les n_k en conservant leur somme égale à n.

Exercice 2 – Importance sampling : choisir le paramètre d'une loi exponentielle

1) Le rapport de vraisemblance vaut

$$\frac{p(y)}{q_\lambda(y)} = \frac{1}{\lambda} \mathrm{e}^{-(1-\lambda)y}.$$

Une réplication sans biais est donc

$$\boxed{H_\lambda = \mathbf{1}_{\{Y>\ell\}} \frac{1}{\lambda} \mathrm{e}^{-(1-\lambda)Y}, \qquad Y \sim \mathcal{E}(\lambda).}$$

2) Son second moment est

$$\begin{aligned}\mathbb{E}_\lambda(H_\lambda^2) &= \int_\ell^\infty \frac{p(y)^2}{q_\lambda(y)} \,\mathrm{d}y \\ &= \frac{1}{\lambda} \int_\ell^\infty \mathrm{e}^{-(2-\lambda)y} \,\mathrm{d}y.\end{aligned}$$

Il est fini si et seulement si $\lambda < 2$, et alors

$$\boxed{\mathbb{E}_\lambda(H_\lambda^2) = \frac{\mathrm{e}^{-(2-\lambda)\ell}}{\lambda(2-\lambda)}.}$$

Pour $\lambda \geq 2$, le second moment diverge : l'estimateur reste formellement sans biais pour $\lambda > 0$, mais sa variance est infinie.

3) Comme $p_\ell^2 = \mathrm{e}^{-2\ell}$,

$$\begin{aligned}R_\ell(\lambda) &= \frac{\mathbb{E}(H_\lambda^2)}{p_\ell^2} - 1 \\ &= \boxed{\frac{\mathrm{e}^{\lambda\ell}}{\lambda(2-\lambda)} - 1.}\end{aligned}$$

Pour $\lambda = 1$,

$$R_\ell(1) = \mathrm{e}^\ell - 1 = \frac{1-p_\ell}{p_\ell},$$

qui est bien la variance relative d'une Bernoulli de paramètre p_ℓ.

4) Il suffit de minimiser

$$g(\lambda) = \lambda\ell - \log\lambda - \log(2-\lambda).$$

On a

$$g'(\lambda) = \ell - \frac{1}{\lambda} + \frac{1}{2-\lambda}.$$

L'équation $g'(\lambda) = 0$ équivaut à

$$\ell\lambda^2 - (2\ell+2)\lambda + 2 = 0.$$

La seule racine appartenant à $(0,2)$ est

$$\boxed{\lambda_\ell^\star = \frac{\ell + 1 - \sqrt{\ell^2+1}}{\ell}.}$$

De plus,

$$g''(\lambda) = \frac{1}{\lambda^2} + \frac{1}{(2-\lambda)^2} > 0,$$

donc il s'agit bien du minimum. Le choix $\lambda_\ell^\star < 1$ donne une loi exponentielle à queue plus lourde que la loi naturelle, ce qui rend $Y > \ell$ beaucoup moins rare.

5) Pour $\ell = 5$,

$$\lambda_5^\star = \frac{6 - \sqrt{26}}{5} \simeq 0.1802.$$

La variance relative naïve vaut

$$\mathrm{e}^5 - 1 \simeq 147.4,$$

tandis que

$$R_5(\lambda_5^\star) \simeq 6.51.$$

Le gain est donc d'environ 22.6 en variance relative.

6) Un développement de $\sqrt{\ell^2+1}$ donne

$$\sqrt{\ell^2+1} = \ell + \frac{1}{2\ell} + o(\ell^{-1}),$$

donc

$$\lambda_\ell^\star = \frac{1}{\ell} - \frac{1}{2\ell^2} + o(\ell^{-2}) \sim \frac{1}{\ell}.$$

Ainsi $\lambda_\ell^\star \ell \to 1$ et

$$\frac{\mathrm{e}^{\lambda_\ell^\star \ell}}{\lambda_\ell^\star(2-\lambda_\ell^\star)} \sim \frac{\mathrm{e}\,\ell}{2}.$$

Donc

$$R_\ell(\lambda_\ell^\star) \sim \frac{\mathrm{e}}{2}\ell.$$

La difficulté ne croît plus exponentiellement, mais elle croît encore : la méthode n'a donc pas une erreur relative bornée.

Exercice 3 – Un événement rare à deux régions : pourquoi une seule translation peut échouer

1) On utilise

$$\varphi(y-a) = \varphi(y)\exp\left(ay - \frac{a^2}{2}\right),$$

et de même

$$\varphi(y+a) = \varphi(y)\exp\left(-ay - \frac{a^2}{2}\right).$$

Ainsi

$$\begin{aligned} q_a(y) &= \frac{1}{2}\varphi(y)\mathrm{e}^{-a^2/2}\left(\mathrm{e}^{ay} + \mathrm{e}^{-ay}\right) \\ &= \boxed{\varphi(y)\mathrm{e}^{-a^2/2}\cosh(ay)}. \end{aligned}$$

Le poids est donc

$$\boxed{w_a(y) = \frac{\mathrm{e}^{a^2/2}}{\cosh(ay)}}.$$

2) Une réplication est

$$\boxed{H_a = \mathbf{1}_{\{|Y|>a\}} w_a(Y), \qquad Y \sim q_a.}$$

En effet,

$$\mathbb{E}_{q_a}(H_a) = \int_{|y|>a} \frac{\varphi(y)}{q_a(y)} q_a(y)\,\mathrm{d}y = p_a.$$

3) Pour tout $t \geq 0$,

$$\cosh(t) = \frac{\mathrm{e}^t + \mathrm{e}^{-t}}{2} \geq \frac{1}{2}\mathrm{e}^t.$$

Sur $|y| > a$,

$$\cosh(ay) = \cosh(a|y|) \geq \frac{1}{2}\mathrm{e}^{a|y|} \geq \frac{1}{2}\mathrm{e}^{a^2}.$$

Donc

$$\boxed{w_a(y) \leq 2\mathrm{e}^{-a^2/2}}.$$

Comme $H_a = w_a(Y)\mathbf{1}_{\{|Y|>a\}}$,

$$H_a^2 \leq 2\mathrm{e}^{-a^2/2} H_a.$$

En prenant l'espérance,

$$\boxed{\mathbb{E}_{q_a}(H_a^2) \leq 2\mathrm{e}^{-a^2/2} p_a.}$$

4) Pour la proposition unilatérale $q_+(y) = \varphi(y-a)$, le poids vaut

$$L_+(y) = \exp\left(-ay + \frac{a^2}{2}\right).$$

Le second moment est

$$\begin{aligned}
\mathbb{E}_{q_+}\left[\mathbf{1}_{\{|Y|>a\}} L_+(Y)^2\right] &= \int_{|y|>a} \frac{\varphi(y)^2}{\varphi(y-a)}\,\mathrm{d}y \\
&= \mathrm{e}^{a^2} \int_{|y|>a} \varphi(y+a)\,\mathrm{d}y \\
&= \mathrm{e}^{a^2}\left(\overline{\Phi}(2a) + \frac{1}{2}\right).
\end{aligned}$$

Le terme

$$\boxed{\frac{1}{2}\mathrm{e}^{a^2}}$$

provient exactement de la queue gauche $y < -a$. Il explose extrêmement vite avec a.

La raison est instructive : sous $\mathcal{N}(a,1)$, atteindre la queue gauche $(-\infty, -a)$ est extraordinairement rare ; mais lorsqu'un tel tirage survient, son poids est gigantesque. La variance est donc dominée par des événements que l'on ne verra pratiquement jamais dans une simulation de taille réaliste.

5) Le mélange symétrique place, par construction, environ la moitié des simulations près de $+a$ et l'autre moitié près de $-a$. Les deux régions qui contribuent à p_a sont donc explorées directement. Le poids est en outre borné sur l'événement rare par $2\mathrm{e}^{-a^2/2}$, ce qui empêche les contributions gigantesques responsables de l'instabilité de la translation unilatérale.

Exercice 4 – TP Python : queue gaussienne bilatérale et faux sentiment de convergence

1) **Compléter les trois fonctions.**
Une solution possible est la suivante.

```python
import numpy as np
from scipy.stats import norm


def summary(samples, exact):
    n = len(samples)
    mean = np.mean(samples)
    var = np.var(samples, ddof=1)
    half_width = 1.96 * np.sqrt(var / n)
    rel_var = var / exact**2
    return mean, var, rel_var, mean-half_width, mean+half_width


def naive_samples(rng, n, a):
    z = rng.normal(size=n)
    return (np.abs(z) > a).astype(float)


def one_sided_samples(rng, n, a):
    y = rng.normal(loc=a, scale=1.0, size=n)
    w = np.exp(-a*y + 0.5*a**2)
    return (np.abs(y) > a) * w


def mixture_samples(rng, n, a):
    s = rng.choice([-1.0, 1.0], size=n)
    y = rng.normal(loc=s*a, scale=1.0, size=n)
    w = np.exp(0.5*a**2) / np.cosh(a*y)
    return (np.abs(y) > a) * w


a = 4.0
n = 200_000
exact = 2 * norm.sf(a)
print("valeur exacte", exact)

for seed in [1, 2, 3, 4, 5]:
    rng = np.random.default_rng(seed)
    print("seed", seed)
    for name, fun in [
        ("naif", naive_samples),
        ("translation +a", one_sided_samples),
        ("melange symetrique", mixture_samples),
    ]:
        x = fun(rng, n, a)
        print(name, summary(x, exact))
    print()
```

2) **Comparer les trois méthodes pour les cinq graines.**
Pour $a = 4$, la probabilité exacte vaut environ

$$p_4 \simeq 6.33 \times 10^{-5}.$$

Avec $n = 200\,000$, l'estimateur naïf ne voit en moyenne qu'une douzaine d'événements. Il fluctue donc fortement d'une graine à l'autre.

Le mélange symétrique, au contraire, force l'exploration des deux queues. Il donne des estimations beaucoup plus stables autour de la valeur exacte et une variance empirique nettement plus faible.

3) **Pourquoi la translation unilatérale peut-elle donner un intervalle de confiance très court tout en étant loin de p_a ?**
La translation unilatérale $\mathcal{N}(a, 1)$ visite très bien la queue droite mais pratiquement jamais la

queue gauche. Pour $a = 4$,

$$\mathbb{P}_{\mathcal{N}(4,1)}(Y < -4) = \Phi(-8),$$

qui est de l'ordre de 10^{-15}. Dans une simulation de taille 200 000, on ne verra donc presque sûrement aucune contribution provenant de la queue gauche.

L'estimateur reste sans biais mathématiquement, car la loi normale translatée a un support sur tout $\mathbb{R}$. Mais les très rares tirages de la queue gauche portent des poids énormes et sont précisément ceux qui rétablissent l'espérance correcte. Une simulation ordinaire peut donc sembler stable autour d'une valeur proche de la seule probabilité de queue droite, soit environ la moitié de p_a.

L'intervalle de confiance calculé à partir de cette simulation ne « sait » pas que des contributions extrêmement rares n'ont pas encore été observées : la variance empirique peut alors être artificiellement petite. C'est exactement le type de faux sentiment de convergence que l'on veut mettre en évidence ici.

4) **Que se passe-t-il lorsque l'on remplace $a = 4$ par $a = 2$ puis $a = 5$?**

Pour $a = 2$, l'événement est beaucoup moins rare. L'estimateur naïf observe suffisamment d'événements pour être relativement stable, et la translation unilatérale a davantage de chances de rencontrer la queue gauche, même si elle reste mal adaptée à un événement bilatéral.

Pour $a = 5$, la situation se dégrade fortement pour la méthode naïve : les événements deviennent très rares et de nombreuses simulations peuvent ne contenir que très peu, voire aucun, dépassement. La translation unilatérale devient encore plus trompeuse, car la queue gauche est alors pratiquement inaccessible sous $\mathcal{N}(5, 1)$.

Le mélange symétrique, lui, continue à placer directement des simulations dans les deux régions $(-\infty, -a)$ et $(a, +\infty)$. Sa difficulté augmente beaucoup moins vite avec a.

5) **Calculer l'ESS du mélange symétrique et l'interpréter.**

Pour calculer l'ESS des poids du mélange, on peut utiliser :

```
def mixture_weights(rng, n, a):
    s = rng.choice([-1.0, 1.0], size=n)
    y = rng.normal(loc=s*a, scale=1.0, size=n)
    w = np.exp(0.5*a**2) / np.cosh(a*y)
    return y, w

rng = np.random.default_rng(123)
y, w = mixture_weights(rng, 200_000, 4.0)
ess = np.sum(w)**2 / np.sum(w**2)
print("ESS/n =", ess / len(w))
```

Une ESS modeste n'implique pas nécessairement que l'estimateur de p_a soit mauvais. L'ESS mesure la dispersion globale des poids, indépendamment de la fonction que l'on cherche à moyenner. Ici, ce qui compte pour p_a est surtout le comportement des poids dans les deux queues $|Y| > a$. Le mélange est précisément construit pour bien visiter ces deux régions.

Ainsi, une ESS élevée est un diagnostic favorable, mais elle n'est ni une condition nécessaire ni une mesure exacte de l'efficacité pour une quantité particulière.