TP2 : Simulation de Variables Aléatoires

Tout au long de l'année, on utilisera régulièrement la librairie `scipy.stats`, qui contient une riche collection de lois usuelles. Vous commencerez donc vos codes en important les librairies suivantes :

```
import numpy as np
import scipy.stats as stat
import matplotlib.pyplot as plt
```

Ainsi, la loi usuelle XXX est accessible via `stat.XXX`. En Python, une telle loi de probabilité contient plusieurs méthodes permettant par exemple de tirer des échantillons (`rvs`), de calculer la fonction de répartition (`cdf`), de calculer la densité pour les v.a. continues (`pdf`) ou les probabilités élémentaires pour les v.a. discrètes (`pmf`).

Néanmoins, vous devez connaître des techniques de simulation de lois usuelles en partant de la loi uniforme. Ainsi, dans les exercices suivants, la seule fonction de simulation Python que vous utiliserez est `stat.uniform.rvs`. Celle-ci fait appel à un générateur de nombres pseudo-aléatoires ; elle permet de générer des nombres dont la distribution est proche d'une loi uniforme. De plus, plusieurs appels à cette fonction génèrent des nombres aléatoires qu'on peut considérer indépendants.

Exercice 1. Simulation de lois discrètes

1. Simuler un nombre aléatoire de loi de Bernoulli $\mathcal{B}(p)$.

2. Écrire une fonction `bern(p,s)` permettant de générer un tableau de taille `s` de nombres aléatoires indépendants suivant la loi $\mathcal{B}(p)$.

3. Écrire une fonction `binomrvs(n,p,ns)` permettant de générer `ns` nombre aléatoires indépendants suivant la loi $\mathcal{B}(n,p)$.

4. Simuler un nombre aléatoire de loi géométrique $\mathcal{G}(p)$.

5. Écrire une fonction `unifdiscrete(n)` qui simule un nombre aléatoire de loi uniforme sur $\{1,2,\dots,n\}$.

6. Simuler la loi P sur $\{-2,3,8\}$ donnée par $P(\{-2\}) = \frac{2}{3}$, $P(\{3\}) = \frac{1}{6}$ et $P(\{8\}) = \frac{1}{6}$.

Exercice 2. Méthode d'inversion

Soit X une v.a.r. de fonction de répartition F. On définit la fonction quantile (ou "inverse généralisée" de F) par

$$F^-(u) = \inf\{\, x \in \mathbb{R} \mid F(x) \geqslant u \,\} \quad (u \in]0,1[) \,.$$

1. Montrer que F^- est bien définie, et que l'inf est en fait un min. Autrement dit,

$$\forall x \in \mathbb{R}, \forall u \in]0,1[, \quad u \leqslant F(x) \iff F^-(u) \leqslant x \,.$$

2. En déduire que si U est une loi uniforme sur $[0,1]$, alors $F^-(U)$ admet F pour fonction de répartition.

3. Que vaut F^- si F réalise une bijection d'un intervalle ouvert I sur $]0,1[$?

4. Écrire une fonction `exprvs(lambda,s)` qui génère un tableau de taille `s` de nombres aléatoires i.i.d. de loi $\mathcal{E}(\lambda)$.

5. (Bonus) Écrire une fonction `cauchy` permettant de simuler la loi de Cauchy $\dfrac{dx}{\pi(1+x^2)}$.

6. (Bonus) Écrire une fonction `discrete(p)` qui simule un nombre aléatoire dont la loi discrète sur $\{1,\dots,n\}$ est donnée par les éléments `p(1), ..., p(n)` du paramètre `p`.

Indice : On pourra utiliser les fonctions `np.cumsum` et `np.nonzero`.

Exercice 3. Affichage de lois et lois empiriques

1. Afficher la densité gaussienne $x \mapsto \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(x-m)^2}{2\sigma^2}\right)$.

2. Calculer et afficher la fonction de répartition de la loi $\mathcal{E}(\lambda)$.

3. À des échantillons $(x_1, \ldots, x_n)$ de loi μ on associe la loi empirique et la fonction de répartition empirique

$$\widehat{\mu}_n = \frac{1}{n}\sum_{i=1}^{n} \delta_{x_i} \quad , \quad \widehat{F}_n(x) = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}_{x_i \leqslant x} \, .$$

a. Étant donné un échantillon stocké dans le vecteur `x` tiré avec la fonction `exprvs` de l'exercice précédent, tracer la fonction de répartition empirique associée.

Indice : On pourra utiliser les fonctions `np.sort` et `plt.step`.

b. Vérifier ainsi votre fonction `exprvs` en comparant la fonction de répartition empirique avec la fonction de répartition de $\mathcal{E}(\lambda)$. (Bonus : calculer l'erreur $\|F - \widehat{F}_n\|_\infty$ en norme uniforme.)

4. Une autre manière de visualiser la répartition de l'échantillon $(x_1, \ldots, x_n)$ est d'afficher un histogramme. Ceci est particulièrement utile lorsque la loi μ est absolument continue, auquel cas pour n grand, l'histogramme est comparable à la densité sous-jacente. Pour visualiser l'histogramme d'un échantillon `x`, on utilisera

```
plt.hist(x, nclasses, normed=True)
```

En précisant `normed=True`, on impose que l'aire sous l'histogramme soit égale à 1. De plus, à nombre de classes donné, `plt.hist` choisit automatiquement les classes utilisées. Le nombre de classes peut être fixé à la main. Un choix standard est `nclasses` $= \lceil 4n^{1/4} \rceil$ (mais `nclasses=30` donne souvent un joli histogramme à partir du moment où l'on a $n \geqslant 1000$...)

⟶ Vérifier de nouveau votre fonction `exprvs` en superposant histogramme et densité théorique.

NB : Le paramètre `rwidth` de `plt.hist` ne modifie pas la hauteur des barres (en précisant ce paramètre, l'aire sous la courbe devient `rwidth`).

5. Pour visualiser des lois discrètes sur un ensemble fini de nombres, on utilise un diagramme en bâtons affiché avec `plt.vlines`. Visualiser ainsi la loi binomiale $\mathcal{B}(n,p)$ pour différentes valeurs de n et p (on pourra utiliser la fonction `stat.binom.pmf`, `scipy.special.binom` ou `math.factorial`).

6. Si $(x_1, \ldots, x_n)$ est un échantillon d'une loi discrète à valeurs dans $\{b_1, \ldots, b_p\} \subset \mathbb{R}$ avec $b_1 < \ldots < b_p$, on peut compter le nombre d'éléments tombant sur chaque b_i en utilisant

```
(counts,_) = np.histogram(x,bins=bt)
```

où `bt` contient les valeurs $b_1, \ldots, b_p, b_p + 1$ (ici le paramètre de normalisation `normed` est par défaut à `False`). Notez bien qu'il faut diviser par n pour en déduire la loi empirique (Attention ! ceci n'est pas équivalent à préciser `normed=True` lorsque les intervalles $[b_i, b_{i+1}[$ ne sont pas de longueur 1).

⟶ Vérifier ainsi la fonction `binomrvs` que vous avez écrite dans l'Exercice 1.

Exercice 4. Méthode de Box-Muller pour simuler $\mathcal{N}(0,1)$

1. Soit S et Θ deux v.a. indépendantes de lois respectives $\mathcal{E}(\frac{1}{2})$ et $\mathcal{U}([0, 2\pi])$.

Montrer que $X = \sqrt{S}\cos(\Theta)$ et $Y = \sqrt{S}\sin(\Theta)$ sont indépendantes de loi $\mathcal{N}(0,1)$.

2. En déduire une méthode de simulation de la loi $\mathcal{N}(0,1)$ et l'implémenter.

3. Vérifier votre fonction en superposant histogramme et densité théorique.