TP n°2 : Variables et Vecteurs Aléatoires (Sous Python)

Rappels

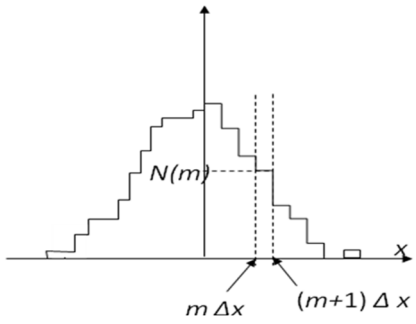
Un générateur de nombres aléatoires dans l'intervalle [0, 1] est une fonction rand qui vérifie les deux propriétés suivantes :

- un appel à la fonction **numpy.random.uniform** (a,b, N) donne une réalisation d'une variable aléatoire de loi uniforme sur [a,b].

- les appels successifs à la fonction **numpy.random.uniform** fournissent une réalisation d'une suite de variables aléatoires indépendantes.

Pour générer une loi Gaussienne de moyenne M et variance V : V* **numpy.random.randn**(N) + M
Pour générer une loi exponentielle de paramètre Landa : **numpy.random.exponential(Landa,N)**

- **sum(X**) Calcule la somme des composantes de X .
- **numpy.mean(X)** Calcule la moyenne empirique de X
- **numpy.var(X)** Calcule la variance empirique.
- **numpy.std(X)** Calcule l'écart-type empirique de X
- **numpy.histogram (X, bins): T**race un histogramme non normalisé de X sur 'bins' intervalles. Soit une suite finie de points issus de la réalisation d'une VA X avec Xmin=valeur minimale de la suite de valeurs et Xmax=valeur maximale de la suite de valeurs. On partage le segment [Xmin, Xmax] en 'bins' classes équidistantes et on compte le nombre de valeurs dans chaque classe, on a alors une estimation de la densité de probabilité:



Remarques :

- ✓ Sous Python au début d'un programme, on peut initialiser le générateur en lui fournissant la première valeur appelée la graine du générateur numpy.random.seed(nbre). Sinon, à chaque fois que l'on démarre Python, c'est la même suite de valeurs qui sera donnée.

- ✓ Python dispose d'une classe **stats** de la bibliothèque **scipy** (**scipy.stats**) qui contient des fonctions prédéfinies permettant de simuler des réalisations de variables de la plupart des lois usuelles ainsi que de nombreuses fonctions statistiques.

Exercice 1

On veut simuler le lancer d'un dé avec 60 jets. Donner la loi de probabilité puis la fonction de répartition et calculer la moyenne et la variance théoriques.

```
import numpy as np; import matplotlib.pyplot as plt
N = 60; a = 1 ; b = 6  ;
X = np.random.randint(a,b+1,N) ;
plt.figure(1); plt.stem(X,use_line_collection=True); plt.grid(True);
plt.title("lancer du dé")
bins=(b-a)+1; I = np.linspace(a,b,bins)
(P,J) = np.histogram(X,bins);
plt.figure(2);        plt.subplot(211);        plt.stem(I,P,use_line_collection=True);
plt.grid(True);
plt.subplot(212); plt.hist(X, bins, density = True)
mu=np.mean(X);vv=np.var(X) ; vvv=np.std(X)
```

1. Prendre N=6 puis 12 puis 6000 puis 60000 et commenter l'histogramme
2. Quelle est l'effet de l'augmentation de N sur la moyenne et la variance estimées.
3. Peut-on considérer que l'histogramme représente la densité de probabilité?
4. Utiliser `np.cumsum` pour tracer la fonction de répartition

```
F=np.cumsum(P/N);
plt.figure(3); plt.stem(I,F,use_line_collection=True); plt.grid(True);
```

Exercice 2

En vous servant de `np.random.randn`, écrire un programme permettant de simuler une loi gaussienne représentant les notes de 1000 étudiants. Utiliser différentes valeurs de bins (10, 50 et 100) et commenter.

Exercice 3

On veut vérifier le théorème central limite. Pour cela, on crée une matrice A dont chaque ligne représente un échantillon d'une loi uniforme.

1. Créer la matrice `A = np.random.uniform(a,b,(12,1000))`, et un vecteur X de taille 1000 dont les composantes représentent la moyenne de chaque colonne de A : `np.mean(A,0)`
2. Vérifier que chaque ligne de A suit une loi uniforme en traçant leurs histogrammes sur 50 valeurs (bins=50) en employant une boucle.
3. Utiliser `np.histogram(X,bins)` pour visualiser la loi de X. A quoi la loi de X ressemble-t-elle?

Exercice 4

Ecrire un programme de générer une loi X une v.a ayant une densité de probabilité uniforme pour $0 \leq x \leq 1$ (N=10000) et nulle ailleurs ;et soit Y une v.a indépendante de X et de densité de probabilité uniforme pour $0 \leq y \leq 1$ et nulle ailleurs. On considère la variable aléatoire Z = X + Y,

1. Sachant que $p_Z(z) = p_X(x)*p_Y(y)$, montrer théoriquement que $p_Z(z)$ aura l'allure d'un triangle
2. Vérifier par `np.histogram` que c'est le cas.

Exercice 5

On modélise fréquemment la durée de vie d'un composant électronique par une loi exponentielle. On suppose que la durée de vie, en jours, d'une ampoule, est une variable aléatoire de loi exponentielle de paramètre $\lambda=1/100$.

1. Donner sa moyenne et sa variance théorique
2. Utiliser X = `numpy.random.exponential(1/Landa,N)` pour générer cette loi. Tracer sa loi.
3. Utiliser `np.mean` pour déterminer la durée de vie moyenne d'une ampoule.

Exercice 6

Soi φ une v.a. uniforme entre 0 et 2π et soit $X=\cos(\varphi)$ et $Y=\sin(\varphi)$

1. Déterminer théoriquement p(x) et p(y) puis tracer les lois de X et Y et commenter.
2. Utiliser `plt.scatter(X, Y)`
2. Calculer leurs moyennes et variances théoriques puis empiriques.
3. Utiliser `np.corrcoef(X,Y)` pour déterminer le coefficient de corrélation.
4. X et Y sont-elles décorrélées? Sont-elles indépendantes?.
4. Reprendre ces questions en considérant Y=aX+b avec a= 2 et b=3 puis a= - 4. Conclure.