YNOV - Master 2 DATAEN

Maths DataScience

Probabilités et statistiques, Algèbre linéaire avec Python

Simulation d'échantillons statistiques et applications.

Ce premier atelier porte sur les échantillons statistiques. Un échantillon statistique est un ensemble $X_1, \cdots, X_n$ de variables aléatoires indépendantes et identiquement distribuées (i.i.d.) selon une même loi de distribution P. L'objectif est d'apprendre à simuler un échantillon, c'est à dire à faire un tirage aléatoire de la réalisation $x_1, \cdots, x_n$ d'un échantillon. Nous étudierons quelques méthodes de simulations pour des lois discrètes et continues. Nous verrons par ailleurs quelques applications de la simulation.

1. Loi uniforme.

1.1. Simulation.

Les langages de programmation proposent généralement des fonctions pour générer des suites pseudo-aléatoires de réalisations de variables aléatoires indépendantes de loi uniforme sur $[0,1)$. Python dispose notamment de la méthode **rand** du module **numpy.random**. Cette méthode peut s'importer dans un script sous le nom Uniforme avec l'instruction:

```
from numpy.random import rand as Uniforme
# Pour savoir comment utiliser cette méthode, décommenter la ligne suivante:
# help(Uniforme)
```

Exercice 1.

Ecrire un programme qui simule un échantillon de taille N fixée de loi uniforme sur $[0,1)$ et le stocke dans un vecteur y.

```
N = 1000
y = Uniforme(N)
```

1.2. Histogramme.

On peut analyser la distribution empirique d'un échantillon y à l'aide d'un histogramme. Un histogramme est une fonction qui à un intervalle associe le nombre (ou le pourcentage) d'éléments de l'échantillon dont la valeur est dans l'intervalle. En Python, un histogramme peut être calculé et visualisé au moyen de la méthode **hist** du module **pyplot** de **matplotlib**. On peut importer ce module avec l'instruction suivante.

```
import matplotlib.pyplot as plt
# help(plt.hist)
```

L'option **bins** de la fonction **hist** permet de prédéfinir les intervalles où l'histogramme est calculé. En mettant *bins=k*, on subdivise l'étendu des valeurs de y en k intervalles de même taille. En mettant *bins=lk* où *lk* est un vecteur de valeurs ordonnées, on précise les bornes des intervalles successifs. On peut en outre utiliser l'option *normed=True* pour préciser que l'on souhaite les pourcentages de valeurs dans chaque intervalle.

Exercice 2.

1. Afficher l'histogramme de l'échantillon simulé.
2. La distribution correspond-elle à celle d'une loi uniforme ?
3. Que se passe-t-il lorsque l'on augmente la taille N de l'échantillon ?
4. Que se passe-t-il lorsque l'on augmente la nombre K d'intervalles ?
5. En appliquant la loi des grands nombres, justifier mathématiquement que, sur chaque intervalle I, l'histogramme approche la probabilité $P(I)$ de la loi de l'échantillon.

Rappel de la ***loi des grands nombres***:

Soient $(X_n)_n$ une suite de variables aléatoires i.i.d. ayant la même loi qu'une variable aléatoire X. Si X est d'espérance et de variance finies, alors la moyenne empirique de l'échantillon,

$$\overline{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i,$$

converge presque sûrement vers l'espérance $E(X)$ de X.

```
# Vos réponses.
```

2. Loi de Bernoulli.

2.1. Simulation.

Une variable aléatoire suit une loi de Bernoulli de paramètre p dans $[0,1)$ si elle prend la valeur 1 avec probabilité p et 0 avec la probabilité $1-p$.

Soit U une variable uniforme sur $[0,1)$. On définit la variable aléatoire

$$X = \begin{pmatrix} 1 & ¿ si U < p, \\ 0 & ¿ sino n. \end{pmatrix}$$

Exercice 3.

1. Vérifier que X suit une loi de Bernoulli de paramètre p.
2. Ecrire une fonction nommée *Bernoulli* qui simule la réalisation d'un échantillon de loi de Bernoulli en suivant le modèle ci-dessous.
3. Simuler un échantillon de loi de Bernoulli de paramètre $p=0.2$ et de taille $N=1000$.

```
import numpy as np

def Bernoulli(p, n=1):
    """Crée un échantillon de taille n de la loi de Bernoulli de
paramètre p.

    Remarque: On utilise le fait que lorsque U est une loi uniforme
sur (0, 1),
    la variable aléatoire qui vaut 1 si U<p et 0 sinon est de loi de
Bernoulli
    de paramètre p. En effet, P(B=1) = P(U<p) = p, P(B=0) = P(U>p) =
1-p.

    Parameters
    ----------
    p : scalar (in (0, 1))
        Paramètre de la loi de Bernoulli.
    n : int, optional
        Taille de l'échantillon (Valeur par défaut = 1).

    Returns
    -------
    y : ndarray
        Echantillon.

    """
    y = np.zeros(n)

    # A compléter.
    # y[i] contient la réalisation de la ième variable de
l'échantillon.

    return(y)
```

Pour analyser la distribution empirique d'un échantillon de loi discrète, on peut utiliser un diagramme en bâton. Ce diagramme donne, pour chaque valeur possible a, le nombre (ou le pourcentage) d'éléments de l'échantillon dont la valeur est a. Ce diagramme peut se calculer en Python au moyen de la méthode **bar** du module **plt**.

Exercice 4.

A l'aide d'un diagramme en bâton, vérifier que l'échantillon de Bernoulli est correctement simulé.

Pour appliquer **bar**, il faut calculer au préalable le nombre d'occurrences de chaque valeur dans l'échantillon. Pour cela, on peut utiliser la fonction suivante.

```
import numpy as np

def occurrences(val, y):
```

```
    """Compte le nombre d'occurrences de chaque valeur de val dans
l'échantillon y.

    Parameters
    ----------
    val : ndarray
        Valeurs.
    y : ndarray
        Echantillon.

    Returns
    -------
    occ : ndarray
        Nombre d'occurrences de chaque valeur dans l'échantillon.
        occ[k] : nombre d'occurences de la valeur val[k] dans
l'échantillon y.
    """
    occ = np.zeros(val.size)
    for j in range(y.size):
        for k in range(val.size):
            if y[j] == val[k]:
                occ[k] += 1
                break
    return(occ)

# Votre réponse.
```

2.3. Quelques lois discrètes associées à la loi de Bernoulli.

Loi binomiale.

Une variable aléatoire X suit une loi binomiale $B(K, p)$ de paramètres (K, p) si, pour tout k entier compris entre 0 et K,

$$P(X=k)=\binom{K}{k}p^k(1-p)^{K-k}.$$

Exercice 6.

1. Montrer que la somme de K variables aléatoires indépendantes de loi de Bernoulli $B(p)$ suit une loi binomiale $B(K, p)$.
2. En vous aidant de cette propriété, écrire une fonction *Binomiale* qui simule la réalisation d'un échantillon de loi binomiale $B(K, p)$.
3. Simuler un N-échantillon de loi de Binomiale $B(K, p)$ et apprécier la qualité de la simulation à l'aide d'un diagramme en bâton.
4. Faire varier les valeurs de p et observer les changements d'allure de la distribution empirique.

```
def Binomiale(p, k=1, n=1):
    """
    Crée un échantillon de taille n de la loi de Bernoulli de
paramètre (k, p).

    Remarque: On s'appuie sur le fait que la somme de k variables
aléatoires
    indépendantes de loi de Bernoulli de paramètre p forme une
variable
    aléatoire de loi Binomiale de paramètres p et k.

    Parameters
    ----------
    p, k :
        Paramètres de la loi binomiale.
        Par défaut k=1, ce qui correspond à une loi de Bernoulli.
    n : int, optional
        Taille de l'échantillon (valeur par défaut=1).
    Returns
    -------
    y : ndarray
        Echantillon.

    """
    y = np.zeros(n)

    # A compléter.
    # y[i] contient la réalisation de la ième variable de
l'échantillon.

    return(y)
```

Loi géométrique.

Une variable aléatoire X à valeurs enti`eres suit une loi géométrique $G(p)$ de paramètre p dans ¿$0,1$¿ si, pour tout entier k>0,

$$P(X=k)=p(1-p)^{k-1}.$$

Exercice 7.

1. Vérifier que le rang de la première variable valant 1 dans une suite de variables aléeatoires indépendantes de loi de Bernoulli $B(p)$ suit une loi géométrique $G(p)$.
2. En vous aidant de cette propriété, écrire une fonction *Geometrique* qui permet de simuler un échantillon de loi géométrique $G(p)$.
3. Simuler des N-échantillons de loi géométrique pour différentes valeurs de p.
4. Comparer les diagrammes en bâton des échantillons obtenus et discuter de leur forme selon les valeurs du paramètre p.

```
def Geometrique(p, n=1):
    """
    Crée un échantillon de taille n de la loi géométrique de paramètre
p.

    Parameters
    ----------
    p :
        Paramètre de la loi géométrique.
    n : int, optional
        Taille de l'échantillon. La valeur par défaut est 1.
    Returns
    -------
    y : ndarray
        Echantillon.

    """
    y = np.zeros(n)

    # A compléter.
    # y[i] contient la réalisation de la ième variable de
l'échantillon.

    return(y)
```

3. Loi exponentielle.

La loi exponentielle $E(\lambda)$ de paramètre $\lambda>0$ est de densité

$$f(t)=\begin{pmatrix}\lambda e^{-\lambda t} & ¿sit>0,\\ 0 & ¿sinon.\end{pmatrix}$$

3.1. Méthode d'inversion de la fonction de répartition.

Exercice 8.

1. Soient F une fonction de répartition F strictement croissante et U une variable aléatoire de loi uniforme sur $[0,1)$. Vérifier que la variable aléatoire $X=F^{-1}(U)$ admet F comme fonction de répartition.
2. En déduire que, lorsque $U\sim U([0,1))$, $Y=-\frac{1}{\lambda}\log(U)\sim E(\lambda)$.
3. Construire une méthode pour simuler une variable de loi $E(\lambda)$ et la mettre en oeuvre dans une fonction *Exponentielle*.

```
def Exponentielle(lam, n=1):
    """
    Crée un échantillon de taille n de la loi exponentielle de
paramètre lambda.
```

```
    Parameters
    ----------
    lam :
        Paramètre de la loi exponentielle.
    n : int
        Taille de l'échantillon.
    Returns
    -------
    expo : ndarray
        Echantillon.

    """
    y = np.zeros(n)

    # A compléter.
    # y[i] contient la réalisation de la ième variable de
l'échantillon.

    return(y)
```

3.2. Approximation de la fonction Gamma par une méthode de Monte Carlo.

La fonction Γ est définie pour tout $a>0$ par l'intégrale

$$\Gamma(a)=\int_0^{+\infty} x^{a-1}e^{-x}dx.$$

Pour $n \in N^*$, cette fonction vaut $\Gamma(n)=(n-1)!$. Toutefois, les valeurs de Γ ne sont pas connues en général. On se propose de les approcher par simulation.

Exercice 9.

1. Soit $X_1,\cdots,X_n$ un échantillon de loi exponentielle $E(1)$. Pour $a>0$, on définit la variable aléatoire
$$G_n=\frac{1}{n}\sum_{i=1}^{n} X_i^{a-1}.$$
Justifier la convergence de G_n vers $\Gamma(a)$ en précisant le sens de cette convergence.
2. En déduire une méthode d'approximation de $\Gamma(a)$.
3. Ecrire un programme qui permet d'approcher $\Gamma(a)$ et évaluer sa précision pour $a \in N^*$.

```
# Vos réponses.
```

3.2. Approximation de la fonction Beta par une méthode de Monte Carlo.

La fonction β est définie pour tous $a,b>0$ par l'intégrale

$$\beta(a,b)=\int_0^1 x^{a-1}(1-x)^{b-1}\,dx.$$

Cette fonction est liée à la fonction Γ par la relation

$$\beta(a,b)=\frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}.$$

Exercice 10. En vous inspirant de l'exercice 6, construire une méthode d'approximation de cette fonction qui s'appuie sur un échantillon de loi uniforme.

```
# Vos réponses.
```

4. Loi normale.

4.1. Simulation approchée.

Soit une suite de variables aléatoires

$$(X_1,\cdots,X_M,\cdots)$$

indépendantes et identiquement distribuées selon une même loi d'espérance m et de variance v finies. D'après le théorème centrale limite, la loi de la variable

$$T_M=\sqrt{M}\,\frac{\frac{1}{M}\sum_{k=1}^{M}X_M-m}{\sqrt{v}}$$

tend vers une loi normale centrée réduite.

Exercice 13.

1. En vous appuyant sur ce théorème, construire une méthode pour simuler de manière approchée une variable gaussienne centrée réduite à partir d'un M-échantillon de loi uniforme sur $[0,1)$.
2. Mettre en oeuvre cette méthode dans une fonction *Normale*.
3. Générer des échantillons de loi normale centrée réduite avec différentes valeurs de M et comparer leurs distributions empiriques au moyen de l'histogramme.

```
def Normale(n , m=100000):
    """
    Crée un échantillon de taille n de la loi normale centrée réduite.
    Méthode approchée qui se base sur le TCL.

    Parameters
    ----------
    n : int
        Taille de l'échantillon.
    m : int, optionnel.
```

```
        Taille de l'échantillon utilisé pour approcher la loi normale.

        La valeur par défaut est 10000.
    Returns
    -------
    y: ndarray
        Echantillon.
    """
    y = np.zeros(n)

    # A compléter.
    # y[i] contient la réalisation de la ième variable de
l'échantillon.

    return(y)
```

4.2. Approximation de l'intégrale de Gauss tronquée.

Soit Z une loi normale centrée réduite $N(0,1)$. Pour tout $a<b$,

$$P(Z \in [a,b)) = \frac{1}{\sqrt{2\pi}} I(a,b)$$

où

$$I(a,b) = \int_a^b e^{-\frac{x^2}{2}} dx$$

est une intégrale de Gauss tronquée.

Exercice 14.

1. Construire une méthode pour approcher la valeur de cette intégrale à l'aide d'un échantillon de loi normale centrée réduite.
2. La mettre en oeuvre et évaluer sa précision pour $b=-a=1.96$ et différentes tailles d'échantillon.

```
# Votre réponse.
```