TP 4: Loi géométrique.

5 juillet 2023

Partie 1 : Etude théorique.

Dans le dossier "infoECGMA2", copier le fichier `loi_geometrique_theorique_a_trou.py` le coller dans votre dossier. Lancer Pyzo puis ouvrir ce fichier.

1. On veut tracer l'histogramme théorique d'une variable aléatoire $X \hookrightarrow \mathcal{G}(p)$. Le réel p étant donné par l'utilisateur. On sait que dans ce cas $X(\Omega) = \mathbb{N}^*$. Pour simuler une telle loi discrète où $X(\Omega)$ est infini, on trace l'histogramme théorique sur $[\![1, n]\!]$ ou $n = \lfloor 3\mathbb{E}(X) \rfloor$. Compléter la ligne 8 pour que n soit égal à $\lfloor 3\mathbb{E}(X) \rfloor$.
2. Rappeler pour tout entier $k \in X(\Omega)$ l'expression de $\mathbb{P}(X = k)$ Compléter les lignes 9, 10 et 11 pour que le vecteur `Xt` contienne les valeurs théorique de la loi de X.

```
Xt = ...
for k in ... :
    Xt[k] = ...
```

   Le vecteur ligne `Xt` est de la forme `Xt = [Xt[0],Xt[1],..,Xt[n]]`. La valeur de `Xt[0]` reste donc 0.

3. Compléter la ligne 17 en utilisant `plt.bar()` afin que le programme affiche en rouge (`color = 'red'`) l'histogramme théorique de X.
4. Compléter la ligne 22 en utilisant `plt.step()` afin que le programme affiche en bleu la fonction de répartition théorique de X.

Partie 2 : Etude empirique

On cherche maintenant à simuler un grand nombre de fois, disons 10000 une même loi géométrique et à comparer l'histogramme et la répartition empirique avec les graphiques obtenus au point précédent. Dans le dossier "infoECGMA2", copier le fichier `loi_geometrique_atrou.py` le coller dans votre dossier. Lancer Pyzo puis ouvrez ce fichier. Dans la suite on va faire les simulations de trois façons différentes. On comparera les performances en observant les données empiriques mais également en mesurant le temps nécessaire aux simulations en utilisant la fonction `time()` de la bibliothèque `time`.

1. Compléter la ligne 12 afin que l'algorithme affiche l'histogramme théorique d'une variable aléatoire X suivant une loi géométrique de paramètre p. Le réel p étant donné par l'utilisateur.
2. **Simulation par utilisation de `rd.geometric()`**

(a) Compléter la ligne 29 du programme de manière à ce qu'il crée une série statistique X en utilisant `rd.geometric()`

(b) Compléter les 34 et 35 afin que le programme affiche l'espérance théorique et l'espérance empirique de la variable aléatoire X.

(c) Compléter la ligne 39 afin que le programme affiche l'histogramme empirique lié à la série statistique X.

(d) Exécuter votre programme en choisissant $p = 0.1$. Combien de temps faut-il pour créer l'échantillon ?

3. **Utilisation de `while`** On veut maintenant faire la simulation d'une loi géométrique de paramètre p en simulant le temps d'attente d'un événement de probabilité p lors de tentatives identiques et indépendantes. Compléter les lignes suivantes et insérer les dans le programme à la place de l'instruction ajoutée en ligne 35 dans le point précédent :

```
X = np.zeros(N)
for j in range(N):
    k = ...
    while ...
        k = ...
    X[j] = k
```

Combien de temps faut-il pour créer l'échantillon ?

4. **Utilisation d'une loi exponentielle** On suppose que Y est une variable aléatoire qui suit la loi exponentielle de paramètre a.

(a) Prouver que, pour tout entier naturel k non nul :

$$\mathcal{P}(k-1 \leq Y < k) = (e^{-a})^{k-1}(1-e^{-a}).$$

(b) En déduire une valeur de a telle que $\lfloor Y \rfloor + 1$ suive une loi géométrique de paramètre p.

(c) Compléter les lignes suivantes et insérer les dans le programme :

```
a = ...
Y = ...
X = ...
```

```python
# Importation de bibliotheques
import numpy as np
import numpy.random as rd
import matplotlib.pyplot as plt

# Loi geometrique theorique
p = float(input("valeur de p ? "))
n = ...
Xt = ...
for k in ...:
    Xt[k] = ...

plt.close()

# Histograme theorique
plt.subplot(2,1,1)
plt.bar(...)
plt.title("histogramme theorique")

# Repartition theorique
plt.subplot(2,1,2)
plt.step(...)
plt.title("Repartition theorique")
plt.show()
```

```python
# Importation de bibliotheques
import numpy as np
import numpy.random as rd
import matplotlib.pyplot as plt
from time import time

# Loi geometrique theorique
p = float(input("valeur de p ? "))
n = int(np.floor(3 / p))
Xt = np.zeros(n+1)
for k in range(1,n+1):
    Xt[k] = ...

# Histograme theorique
plt.close()
plt.subplot(2,2,1)
plt.bar(range(n+1),Xt,width=0.5,color='red')
plt.title("histogramme theorique")

# Repartition theorique
plt.subplot(2,2,2)
plt.step(range(n+1),np.cumsum(Xt),color="blue")
```

```python
plt.title("Repartition theorique")
plt.show()

# Simulation
N = 10000
t0 = time()
X = ...
duree = time()-t0
print("temps de simulation : ",duree)

# Esperance
print("Esperance theorique : ",...)
print("Esperance empirique : ",...)

# Histogramme empirique
plt.subplot(2,2,3)
plt.hist(... , ... , color='yellow' , edgecolor='red' , density=True)
plt.title("Histogramme empirique")

# Repartition empirique
def Fr_empirique(X):
    a = np.sort(X)
    N = len(X)
    F = np.zeros(n+1)
    for j in range(n+1):
        k = 0
        while k<N and a[k] <= j:
            F[j] += 1
            k = k+1
    return F/N

plt.subplot(2,2,4)
plt.step(range(n+1),Fr_empirique(X))
plt.title("Repartition empirique")
plt.show()
```