DS 10

Devoir surveillé du Mardi 12 Mars

La calculatrice est interdite. Durée : 2h

Dans tout ce devoir, on suppose avoir importé sur `Python` les bibliothèques :

- `numpy` avec le raccourci `np` ;
- `numpy.random` avec le raccourci `rd` ;
- `numpy.linalg` avec le raccourci `al` ;
- `matplotlib.pyplot` avec le raccourci `plt`.

Exercice 1

1. Soit $\lambda > 0$. On admet que si $U \hookrightarrow \mathcal{U}([0,1])$, alors la variable $X = -\frac{1}{\lambda}\ln(1-U)$ suit une loi exponentielle de paramètre λ.

   Écrire une fonction `exponentielle(lambda)` en langage `Python` simulant une loi $\mathcal{E}(\lambda)$ à partir de la fonction `rd.random()` (sans utiliser `rd.exponential`).

2. Soit $\lambda > 0$. On admet que si $X \hookrightarrow \mathcal{E}(\lambda)$, alors $Y = \lfloor X \rfloor + 1 \hookrightarrow \mathcal{G}(1-e^{-\lambda})$.

   (a) Écrire une fonction `geom(p)` en langage `Python` simulant une loi $\mathcal{G}(p)$ à partir de la fonction `exponentielle` (sans utiliser `rd.geometric`).

   (b) On souhaite construire un vecteur `x` de taille 10000 contenant 10000 simulations de la loi $\mathcal{G}(0.2)$ à partir de la fonction `geom`. Compléter pour cela le programme `Python` suivant :

```
x = .....
for k in range(10000) :
    x[k] = .....
```

   (c) En exécutant l'instruction suivante :

```
m = np.mean(x)
v = np.mean((x-m)**2)
print(m, v)
```

   on obtient `5.0404` et `20.133581` . Ces résultats sont-ils conforme à vos attentes ? Expliquer.

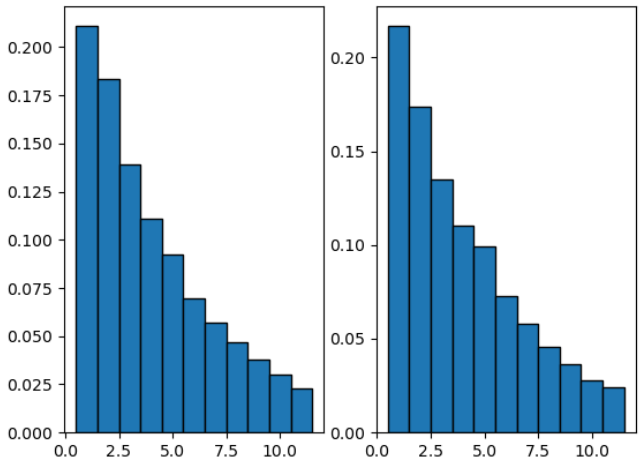
   (d) En exécutant le script :

```
c = np.arange(0.5, 12.5)
plt.subplot(1, 2, 1)
plt.hist(x, c, density='True', edgecolor='k')

plt.subplot(1, 2, 2)
plt.hist(rd.geometric(0.2, 10000), c, density='True',
edgecolor='k')

plt.show()
```

   on obtient les diagrammes en bâtons suivants :



A quoi correspondent chacun des diagrammes en bâtons ? Que peut-on conclure de ces diagrammes ?

Exercice 2

Soit $k \in \mathbb{N}^*$ et $\lambda > 0$. On considère une variable aléatoire à densité X donc la fonction de répartition est :

$$F(x) = \begin{cases} 1 - \dfrac{\lambda^k}{x^k} & \text{si } x > \lambda \\ 0 & \text{sinon.} \end{cases}$$

On dit que X suit la loi de Pareto de paramètre λ et k.

1. (a) Justifier que F réalise une bijection de $]\lambda, +\infty[$ dans $]0, 1[$ et déterminer sa bijection réciproque.
   (b) En utilisant la méthode d'inversion, en déduire une fonction `Python` qui simule une variable aléatoire suivant une loi de Pareto de paramètres λ et k.

2. Soient $X_1, \cdots, X_k$ k variables aléatoires indépendantes suivant toutes la loi uniforme sur $]0, 1]$. On pose alors $Y = \dfrac{\lambda}{\max(X_1, \ldots, X_k)}$.
   (a) Montrer que Y suit une loi de Pareto de paramètres λ et k.
   (b) En déduire une autre fonction `Python` pour simuler la loi de Pareto.

Exercice 3

On considère les trois tables suivantes :

`Voiture`

Plaque	Propriétaire	Marque	Modèle	Atelier
DG 103 HY	3	Peugeot	807	Carrosserie
EF 334 GA	1	Renault	Clio	Pneu
EI 189 KA	4	Fiat	Uno	Pneu
FA 934 TH	5	Lancia	Delta	Mécanique
BB 512 IJ	6	Renault	Twingo	Pneu
FD 246 MT	2	Fiat	500	Mécanique

Client

Id_client	Nom	Prénom	Téléphone
1	Perha	Marie	0668543452
2	Ridy	Anne	0675463309
3	Leclerc	Louis	0667240908
4	McGregor	Thomas	0679856423
5	Ricciardo	Constance	0619798669
6	Lauquie	Faustine	0638899821

Tarif

Atelier	Chef	Prix
Pneu	Paul	75
Mécanique	Roger	120
Carrosserie	Gérard	145

1. Au vu des relations entre les différentes entités représentées par ces trois tables :
   (a) Donner les clés primaires de chacune de ces trois tables.
   (b) Identifier des clés étrangères sur la table `Voiture` référençant les clés primaires des deux autres tables.
2. Que permettent d'effectuer les requêtes `SQL` suivantes :
   (a) `UPDATE Tarif SET Chef="Yves" WHERE Chef="Paul"`
   (b) `SELECT Marque,Modèle FROM Voiture WHERE Atelier="Pneu"`
   (c) `SELECT Propriétaire FROM Voiture WHERE Marque="Renault" AND Atelier="Pneu"`
   (d) `DELETE FROM Voiture WHERE Plaque="DG 103 HY"`
3. Écrire les requêtes `SQL` permettant de :
   (a) Sélectionner les colonnes `Plaque` et `Atelier` de la table `Voiture`.
   (b) Sélectionner les noms des propriétaires d'une Peugeot (on veut une seule requête).
   (c) Sélectionner les ateliers dont le taux horaire HT ne dépasse pas 100 euros.
   (d) Sélectionner les numéros de téléphone des propriétaires de Renault venus pour un problème de pneu.
   (e) Sélectionner les marques et les modèles des voitures réparées aux ateliers Carrosserie ou Mécanique.
   (f) Modifier le numéro de téléphone de Mme Ridy (son nouveau numéro est le 0651096754).
   (g) La voiture de M. McGregor est réparée. Supprimer la ligne correspondante dans la table `Voiture`.

Exercice 4

Un joueur joue à un jeu au casino. On suppose qu'il dispose d'une fortune de a euros et que celle du casino est de b euros. A chaque tour, il gagne un euro avec la probabilité $p \in]0,1[$ ou perd un euro (donné au casino) avec la probabilité $q = 1 - p$. Le jeu s'arrête lorsque le joueur ou le casino se retrouve ruiné.
On peut alors modéliser la situation par une chaîne de Markov homogène $(X_n)_{n\in\mathbb{N}}$ où X_n représente la fortune du joueur à l'issue du n-ième tour.

1. Déterminer la matrice de transition A associée à cette chaîne de Markov.

   Donner sans calcul deux états stables.

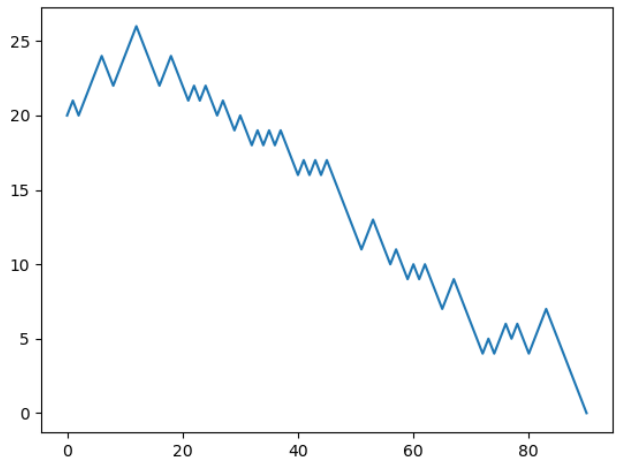
2. Compléter et exécuter la fonction suivante qui prend en entrée p, a et b, simule l'expérience décrite dans l'énoncé et renvoie en sortie une liste contenant l'état de la fortune du joueur à chaque tour :

```
def ruinejoueur(p,a,b):
    L = [a]
    while ..... and ..... :
        if ..... :
            L.append(L[-1]+1)
        else:
            L.append( ..... )
    return(L)
```

3. On ajoute les instructions suivantes à la suite du programme précédent :

```
p = 1/3
a = 20
b = 300
L = ruinejoueur(p,a,b)
N = list(range(len(L))
plt.plot(N, L)
plt.show()
```

   On obtient le résultat graphique suivant :



   Que fait ce programme ? Comment interpréter le résultat graphique obtenu ?

4. Compléter la fonction suivante qui simule 1000 chaînes de Markov indépendantes et calcule à partir de ces 1000 simulations l'effectif N puis la fréquence f de l'événement "le joueur est ruiné à la fin du jeu" :

```
def ruinejoueurfrequence(p,a,b)
      N = 0
      for k in range(1000):
          L = ruinejoueur(p,a,b)
          if ..... :
              N = N+1
      f = .....
      return(f)
```

Exercice 5

On admet que le nombre X de têtards issus des oeufs pondus en mars et avril d'une année suit une loi de Poisson de paramètre $\lambda = 20$.
Ces têtards sont soumis à des prédateurs nombreux et voraces, et on admet que chacun d'entre eux à une probabilité $p = 0,05$ de parvenir à son développement complet, et qu'ils se développent de façon indépendante.
On note Y le nombre de têtards qui parviennent à leur développement complet et se transforment donc en une grenouille.

1. Quelle est la loi de Y sachant $(X = k)$? Justifier.

2. Recopier et compléter la fonction `Python` suivante qui pour qu'elle simule une fois cette expérience et retourne la valeur y prise par la variable Y :

```
def simulY()
    x = .....
    y = .....
    return(y)
```

3. Écrire une fonction d'en-tête `def SimulY(N)` en langage `Python` donnant un échantillon de taille N de la loi de Y.

4. Recopier et compléter la fonction `Python` suivante pour qu'elle renvoie un vecteur $V = [v_0, \dots, v_{10}]$ tel que :

$$\forall k \in [\![0, 10]\!], \quad v_k = \frac{\lambda^k}{k!} e^{-\lambda}.$$

```
def loipoisson(lb):
    V = np.zeros(11)
    V[0] = .....
    for k in range(1,11):
        V[k] = .....
    return(V)
```

5. A la suite des instructions précédentes, on ajoute les commandes `Python` suivantes :

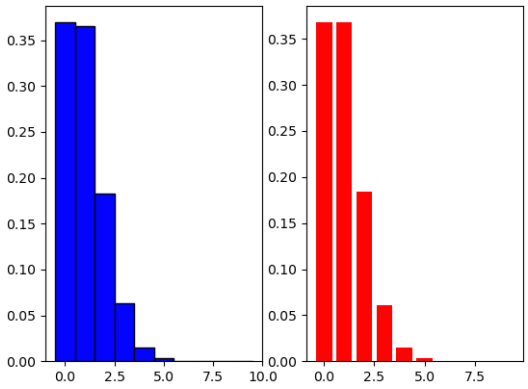
```
U = SimulY(100000)
c = np.arange(-0.5, 10)
plt.subplot(1, 2, 1)
plt.hist(U, c, density='True', edgecolor='k', color='blue')

n = np.arange(10)
V = loipoisson(1)
```

```
plt.subplot(1, 2, 2)
plt.bar(n, V, color='red')
plt.show()
```

Que contiennent les variables `U`, `c`, `n` et `V` ? Que fait ce programme ?

6. Après exécution, on obtient les graphiques suivants :



Quelle conjecture peut-on faire sur la variable aléatoire Y ? Justifier votre réponse.

Exercice 6

1. Écrire une fonction `indicemin` en langage `Python` qui, pour une liste `L`, renvoie l'indice du minimum de la liste.

2. Si `L` est une liste, que fait l'instruction `del L[i]` ?

3. Déduire des deux premières questions une fonction `min2` qui, pour une liste `L`, renvoie la valeur du deuxième minimum de la liste.

4. Compléter la fonction `tri` suivante pour que, étant donné une liste `L`, elle renvoie la liste triée dans l'ordre croissant des éléments de `L`. On détaillera ligne par ligne ce que fait cette fonction.

```
def tri(L):
    M = []
    while len(L) > 0 :
        i = .....
        M.append( ..... )
        del .....
    return M
```
