Devoir surveillé 8

Mercredi 19 février 2025 – 4h00

Consignes à lire !

La qualité de la rédaction, le soin porté à la copie, la lisibilité, l'orthographe, la rigueur du vocabulaire ainsi que la clarté des raisonnements sont des critères importants d'évaluation.

Quelques précisions :

- **la copie doit être prise de sorte que la marge se situe à droite de chaque page,**
- **la première page de la copie doit rester vierge et sera réservée aux appréciations,**
- **toutes les pages de la copie devront être numérotées et rangées dans l'ordre de lecture,**
- **les résultats finaux doivent être clairement mis en évidence (soulignés ou encadrés),**
- **les questions d'un même exercice doivent être présentées dans l'ordre du sujet.**
- un aide-mémoire `Python` est donné en fin de sujet.
  Pour toutes les questions `Python` du sujet, on supposera avoir importé les différents modules nécessaires de la sorte :

```
import numpy as np
import numpy.random as rd
import numpy.linalg as al
import matplotlib.pyplot as plt
import pandas as pd
```

L'usage de tout matériel électronique est interdit. Aucun document n'est autorisé.

Si, au cours de l'épreuve, un candidat repère ce qui lui semble être une erreur d'énoncé, il la signalera sur sa copie et poursuivra sa composition en expliquant les raisons des initiatives qu'il sera amené à prendre.

"*Tous les enfants ont du génie, le tout c'est de le faire apparaître.*"
Charlie Chaplin

Exercice 1

On considère un entier naturel n supérieur ou égal à 2. On note $\mathcal{S}_n$ l'ensemble des matrices $A = (a_{i,j})_{1\leqslant i,j\leqslant n}$ de $\mathcal{M}_n(\mathbb{R})$ qui vérifient les propriétés suivantes :

(1) $\forall (i,j) \in [\![1;n]\!]^2,\ a_{i,j} \geqslant 0$;

(2) $\forall i \in [\![1;n]\!],\ \sum_{j=1}^{n} a_{i,j} = 1$.

1. L'ensemble $\mathcal{S}_n$ est-il un sous-espace vectoriel de $\mathcal{M}_n(\mathbb{R})$?
2. Soit $A \in \mathcal{S}_n$. Justifier que pour tout $(i,j) \in [\![1;n]\!]^2$, on a $a_{i,j} \in [0;1]$.
3. Soit $A \in \mathcal{M}_n(\mathbb{R})$. Démontrer que A vérifie la propriété (2) si, et seulement si, le vecteur $\begin{pmatrix}1\\ \vdots \\ 1\end{pmatrix}$ est vecteur propre de A pour la valeur propre 1.
4. Démontrer que le produit de deux matrices de $\mathcal{S}_n$ est une matrice de $\mathcal{S}_n$.
5. Soit $A \in \mathcal{S}_n$ et $\lambda \in \mathrm{Sp}(A)$. Soit $X = \begin{pmatrix}x_1\\ \vdots \\ x_n\end{pmatrix}$ un vecteur propre de A associé à la valeur propre λ.

   On note k l'entier de $[\![1;n]\!]$ tel que $|x_k| = \max\big(|x_1|, \ldots, |x_n|\big)$.

   5.a. Justifier que $|x_k| > 0$.

   5.b. Établir :
   $$|\lambda x_k| \leqslant \sum_{j=1}^{n} a_{k,j}|x_j|$$
   En déduire :
   $$|\lambda| \leqslant 1$$

   5.c. Démontrer :
   $$|\lambda - a_{k,k}| \leqslant 1 - a_{k,k}$$
   En déduire que si les éléments diagonaux de A sont tous strictement supérieurs à $\frac{1}{2}$, alors la matrice A est inversible.

Exercice 2

Toutes les variables aléatoires du problème sont supposées définies sur le même espace probabilisé $(\Omega, \mathcal{A}, \mathbb{P})$. On considère une variable aléatoire X à valeurs strictement positives suivant la loi exponentielle de paramètre λ (avec $\lambda > 0$).
Un estimateur T_n d'un réel a est dit :

- **sans biais** lorsque : $\mathbb{E}(T_n) = a$,
- **convergent** lorsque : $\forall \varepsilon > 0,\ \lim\limits_{n\to+\infty} \mathbb{P}\big([|T_n - a| \geqslant \varepsilon]\big) = 0$.

Partie I. Comparaison de deux estimateurs de $\frac{1}{\lambda}$

L'objectif de cette partie est de comparer deux estimateurs sans biais et convergents du paramètre inconnu $\frac{1}{\lambda}$.
Pour n entier de $\mathbb{N}^*$, soit $(X_1, X_2, \ldots, X_n)$ un n-échantillon de variables aléatoires à valeurs strictement positives, indépendantes et de même loi que X.
On pose pour tout $n \in \mathbb{N}^*$:
$$Y_n = \sum_{i=1}^{n} X_i \quad ; \quad \overline{X}_n = \frac{1}{n} Y_n \quad ; \quad M_n = \max(X_1, X_2, \ldots, X_n)$$

1. Rappeler sans démonstration les valeurs de l'espérance $\mathbb{E}(X)$, de la variance $\mathbb{V}(X)$ ainsi que l'expression de la fonction de répartition F_X de la variable aléatoire X.
2. Démontrer que $\overline{X}_n$ est un estimateur sans biais et convergent du paramètre $\frac{1}{\lambda}$.
3. 3.a. Montrer qu'une densité f_{M_n} de M_n est donnée par :
   $$f_{M_n} : x \longmapsto \begin{cases} n\lambda \mathrm{e}^{-\lambda x}(1-\mathrm{e}^{-\lambda x})^{n-1} & \text{si } x > 0 \\ 0 & \text{si } x \leqslant 0 \end{cases}$$

   3.b. Établir pour tout $n \in \mathbb{N}^*$, l'existence de l'espérance $\mathbb{E}(M_n)$ de la variable aléatoire M_n.

   3.c. Soit $a > 0$. En posant $t = 1 - \mathrm{e}^{-\lambda x}$, justifier l'égalité :
   $$\int_0^a x\mathrm{e}^{-\lambda x}(1-\mathrm{e}^{-\lambda x})^{n-1}dx = -\frac{1}{\lambda^2}\int_0^{1-\mathrm{e}^{-\lambda a}} t^{n-1}\ln(1-t)dt$$

3.d. En déduire que l'on a : $\mathbb{E}(M_n) = -\dfrac{n}{\lambda}\displaystyle\int_0^1 t^{n-1}\ln(1-t)dt$.

3.e. Dériver la fonction $t \longmapsto (1-t)\big(1-\ln(1-t)\big)$ définie sur l'intervalle $[0,1[$.
À l'aide d'une intégration par parties, en déduire pour tout $n \in \mathbb{N}^*$ une relation entre $\mathbb{E}(M_{n+1})$ et $\mathbb{E}(M_n)$.

3.f. On pose pour tout $n \in \mathbb{N}^*$: $u_n = \displaystyle\sum_{j=1}^{n} \frac{1}{j}$. Déduire de la question précédente : $\mathbb{E}(M_n) = \dfrac{1}{\lambda}u_n$.

4. On pose pour tout $n \in \mathbb{N}^*$: $M'_n = \dfrac{M_n}{u_n}$ et $v_n = \displaystyle\sum_{j=1}^{n} \frac{1}{j^2}$. *On admet* : $\mathbb{V}(M_n) = \dfrac{1}{\lambda^2}v_n$.

4.a. Calculer $\mathbb{E}(M'_n)$ et $\mathbb{V}(M'_n)$.

4.b. Justifier la convergence de la suite de terme général v_n. Déterminer $\lim\limits_{n\to+\infty} u_n$.

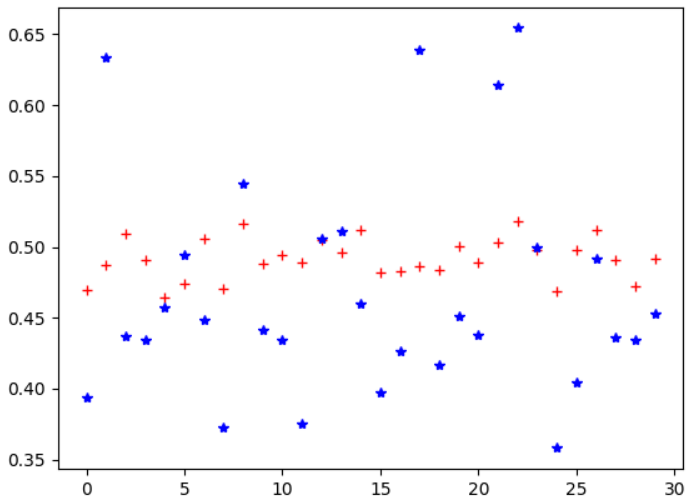
4.c. Déduire des questions 4.a et 4.b que M'_n est un estimateur sans biais et convergent du paramètre $\dfrac{1}{\lambda}$.

4.d. 4.d.i. Recopier et compléter les lignes manquantes du programme **Python** ci-dessous de sorte que son exécution affiche sur le même graphique un nuage de points de 30 réalisations de chacun des estimateurs $\overline{X_n}$ et M'_n dans le cas où $n = 1000$ et $\lambda = 2$.

```
import numpy as np
import numpy.random as rd
import matplotlib.pyplot as plt

n=1000
u=...
lam=2
for k in range(30):
    L=...
    X=...
    M=...
    plt.plot(k,X,'r+')
    plt.plot(k,M,'b*')
plt.show()
```

4.d.ii. L'exécution du programme ci-dessus, une fois complété, affiche le graphique suivant. Lequel des deux estimateurs $\overline{X_n}$ et M'_n semble à privilégier ?



4.e. 4.e.i. Soit Q la fonction polynomiale définie sur $\mathbb{R}$ par : $\forall x \in \mathbb{R},\ Q(x) = \displaystyle\sum_{j=1}^{n}\left(x - \frac{1}{j}\right)^2$.
À l'aide de l'étude de Q, établir l'inégalité : $u_n^2 \leqslant nv_n$.

4.e.ii. Comparer alors $\mathbb{V}(M'_n)$ et $\mathbb{V}(\overline{X_n})$. Retrouver le résultat proposé en question 4.d.ii.

Partie II. Un exemple.

Les notations et le contexte sont ceux de la partie I.
Dans cette partie, on suppose que la durée de vie (en heures) d'un composant électronique est une variable aléatoire X à valeurs strictement positives suivant la loi exponentielle de paramètre λ (avec $\lambda > 0$).
On suppose qu'en cas de panne, le composant électronique est immédiatement remplacé par un composant neuf dont la durée de vie est indépendante et de même loi que celle des composants précédents.

Pour $j \in \mathbb{N}^*$, on note X_j la variable aléatoire égale à la durée de vie du j-ième composant. Pour n entier de $\mathbb{N}^*$, on constitue ainsi un n-échantillon $(X_1, X_2, ..., X_n)$ de variables aléatoires à valeurs strictement positives, indépendantes et de même loi que X.

On note $(x_1, x_2, ..., x_n)$ la réalisation du n-échantillon $(X_1, X_2, ..., X_n)$ et on pose : $\overline{x}_n = \dfrac{1}{n}\sum_{i=1}^{n} x_i$.

5. Donner une interprétation de $\dfrac{1}{\lambda}$. Dans quelle unité s'exprime $\dfrac{1}{\lambda}$?

6. **On admet** que si T et Z sont deux variables aléatoires à densité indépendantes définies sur le même espace probabilisé, de densités respectives f_T et f_Z telles que f_T ou f_Z soit bornée sur $\mathbb{R}$, alors la variable aléatoire $T+Z$ est à densité et admet pour densité la fonction f_{T+Z} définie pour tout x réel par :

$$f_{T+Z}(x) = \int_{-\infty}^{+\infty} f_T(t) f_Z(x-t)dt$$

Démontrer que pour tout $n \in \mathbb{N}^*$, la variable aléatoire Y_n est à densité et admet pour densité la fonction

$$f_{Y_n} : x \longmapsto \begin{cases} \dfrac{\lambda^n x^{n-1}}{(n-1)!}\mathrm{e}^{-\lambda x} & \text{si } x > 0 \\ 0 & \text{si } x \leqslant 0 \end{cases}$$

7. 7.a. Soit $n \in [\![2; +\infty[\![$. Considérons la fonction $g_n : x \longmapsto \begin{cases} 1 - \mathrm{e}^{-\lambda x}\displaystyle\sum_{k=0}^{n-1}\frac{(\lambda x)^k}{k!} & \text{si } x > 0 \\ 0 & \text{sinon} \end{cases}$.

Démontrer que la fonction g est dérivable sur $\mathbb{R}$ et calculer, pour tout $x \in \mathbb{R}$, $g'(x)$.

7.b. Démontrer alors que pour tout $n \in \mathbb{N}^*$, la fonction de répartition F_{Y_n} de la variable aléatoire Y_n est donnée par :

$$F_{Y_n} : x \longmapsto \begin{cases} 1 - \mathrm{e}^{-\lambda x}\displaystyle\sum_{k=0}^{n-1}\frac{(\lambda x)^k}{k!} & \text{si } x > 0 \\ 0 & \text{sinon} \end{cases}$$

8. Soit $t > 0$ un réel fixé et N_t la variable aléatoire égale au nombre de pannes dans l'intervalle $[0, t]$.

8.a. Calculer, pour tout $n \in \mathbb{N}$, la probabilité $\mathbb{P}\big([N_t \geqslant n]\big)$.

8.b. Déterminer la loi de la variable aléatoire N_t.

8.c. Donner une estimation "naturelle" du nombre moyen de pannes dans l'intervalle $[0, t]$.

9. L'objectif de cette question est de déterminer un intervalle de confiance asymptotique du paramètre inconnu $\dfrac{1}{\lambda}$ au niveau de confiance $1-\alpha$ (où $0 < \alpha < 1$).

9.a. Soit Φ la fonction de répartition de la loi normale centrée réduite. Démontrer l'existence d'un unique réel t_α tel que $\Phi(t_\alpha) = 1 - \dfrac{\alpha}{2}$.

Établir : $t_\alpha > 0$.

9.b. On pose : $R_n = \sqrt{n}\left(\lambda \overline{X}_n - 1\right)$. Justifier que la suite de variables aléatoires $(R_n)_{n\in\mathbb{N}^*}$ converge en loi vers une variable aléatoire suivant la loi normale centrée réduite.

9.c. Déterminer alors un intervalle de confiance asymptotique de $\dfrac{1}{\lambda}$ au niveau de confiance $1-\alpha$.

9.d. Que se passe-t-il lorsque α est proche de 1 ?

Partie III. Un résultat asymptotique

Les notations et le contexte sont ceux des parties précédentes.

Pour tout $n \in \mathbb{N}^*$, on pose $T_n = M_n - \dfrac{1}{\lambda}\ln(n)$ et on note F_{T_n} la fonction de répartition de T_n.

10. On considère la fonction F définie sur $\mathbb{R}$ par :

$$\forall x \in \mathbb{R},\ F(x) = \exp\left(-\mathrm{e}^{-\lambda x}\right)$$

10.a. Démontrer que F est la fonction de répartition d'une variable aléatoire à densité T dont on donnera une densité. *On dire que T suit la loi de Gumbel de paramètre λ.*

10.b. Démontrer que la suite $(T_n)_{n\in\mathbb{N}^*}$ converge en loi vers T.

11. 11.a. On pose : $Z = -\dfrac{1}{\lambda}\ln(\lambda X)$. Montrer que les variables aléatoires Z et T suivent la même loi.

11.b. Justifier que Z admet une espérance et que $\mathbb{E}(Z) = -\dfrac{1}{\lambda}\displaystyle\int_0^{+\infty} \mathrm{e}^{-t}\ln(t)dt$.

11.c. À l'aide de la concavité de la fonction ln sur $\mathbb{R}_*^+$, établir l'inégalité : $\mathbb{E}(T) \geqslant 0$.

12. 12.a. Démontrer que F est bijective de $\mathbb{R}$ dans $]0; 1[$ et expliciter sa bijection réciproque notée G.

12.b. Écrire une fonction `Python` prenant en argument un réel a et permettant d'afficher les courbes de F et G sur le même graphique dans le cas où $\lambda = a$.

12.c. Soit U une variable aléatoire suivant la loi uniforme sur l'intervalle $]0, 1[$. Quelle est la loi de la variable aléatoire $G(U)$?

12.d. En déduire l'inégalité : $\mathbb{E}(T) \leqslant \dfrac{1}{\lambda}$.

12.e. Écrire en `Python` une fonction telle que l'exécution de `simule_T(a)` renvoie une réalisation de la variable aléatoire T dans le cas où $\lambda = a$.

12.f. Écrire une fonction `Python` qui permet de renvoyer une valeur numérique approchée de $\mathbb{E}(T)$.

★★★★★★★ Fin ★★★★★★★

Aide-mémoire Python

Listes

- `[]` : Créer une liste vide
- `L[i:j]` : Désigne la liste composée des éléments de `L` dont les indices vont de `i` à `j-1`
- `L[i:j:pas]` : Désigne la liste composée des éléments de `L` d'indices `i, i+pas, i+2*pas,...` compris entre `i` et `j-1`
- `[a]*n` ou `n*[a]` : Créer une liste avec `n` fois l'élément `a`
- `L.append(a)` : Ajoute l'élément `a` à la fin de la liste `L`
- `L1 + L2` : Concatène les deux listes `L1` et `L2`
- `len(L)` : Renvoie le nombre d'éléments de la liste `L`
- `del(L[k])` : Supprime l'élément d'indice `k` de la liste `L`
- `L.count(a)` : Renvoie le nombre d'occurrences de `a` dans la liste `L`
- `L.remove(a)` : Enlève la première occurrence de la valeur `a` de la liste `L`
- `max(L)` : Renvoie le plus grand élément de la liste `L`
- `min(L)` : Renvoie le plus petit élément de la liste `L`
- `sum(L)` : Renvoie la somme de tous les éléments de la liste `L`
- `a in L` : Vaut `True` si `a` se trouve au moins une fois dans `L` et `False` sinon

Modules mathématiques

numpy

`import numpy as np`

- `np.array(L)` : Transforme la liste `L` en vecteur ou matrice numpy
- `np.linspace(a,b,n)` : Crée un vecteur de n valeurs uniformément réparties entre a et b (inclus)
- `np.zeros([n,m])` : Crée la matrice nulle de taille $n \times m$
- `np.zeros(n)` : Crée le vecteur nul de taille n
- `np.ones([n,m])` : Crée la matrice de taille $n \times m$ dont tous les coefficients valent 1
- `np.ones(n)` : Crée le vecteur de taille n dont tous les coefficients valent 1
- `np.eye(n)` : Crée la matrice identité de taille n
- `np.diag(L)` : Crée la matrice diagonale dont les termes diagonaux sont les éléments de la liste `L`
- `np.transpose(M)` : Renvoie la transposée de M
- `np.dot(M,P)` : Renvoie le produit matriciel MP
- `np.sum(M)` : Renvoie la somme de tous les éléments de M
- `np.prod(M)` : Renvoie le produit de tous les éléments de M
- `np.max(M)` : Renvoie le plus grand élément de M
- `np.min(M)` : Renvoie le plus petit élément de M
- `np.shape(M)` : Renvoie dans un couple le format de la matrice M
- `np.mean(M)` : Renvoie la moyenne des éléments de M
- `np.var(M)` : Renvoie la variance des éléments de M
- `np.std(M)` : Renvoie l'écart-type des éléments de M
- `np.median(M)` : Renvoie la médiane des éléments de M
- `np.cumsum(M)` : Renvoie la matrice des sommes cumulées (gauche à droite, haut à bas) des éléments de M
- `np.arange(a,b,eps)` : Renvoie la liste des flottants de a à b de pas constant eps
- `np.abs(x)` : Renvoie $|x|$
- `np.floor(x)` : Renvoie $\lfloor x \rfloor$
- `np.sqrt(x)` : Renvoie $\sqrt{x}$ si $x \geqslant 0$
- `np.log(x)` : Renvoie $\ln(x)$ si $x > 0$
- `np.exp(x)` : Renvoie e^x
- `np.e` : Renvoie e
- `np.pi` : Renvoie π

Sous module d'algèbre linéaire `linalg` de numpy

`import numpy.linalg as al`

- `al.inv(M)` : Renvoie l'inverse de la matrice carrée M si elle est inversible
- `al.eig(M)` : Si la matrice M est diagonalisable, renvoie un couple (D, P) où D est le vecteur des coefficients diagonaux d'une matrice diagonale semblable à M et P une matrice de passage associée
- `al.matrix_power(M,n)` : Renvoie la puissance `n`-ième de la matrice carrée M
- `al.matrix_rank(M)` : Renvoie le rang de la matrice M
- `al.solve(M,Y)` : Renvoie une solution de l'équation matricielle $MX = Y$ lorsque M est inversible

Sous module `random` de numpy pour la simulation probabiliste

`import numpy.random as rd`

- `rd.random([r,s])` : Simule une réalisation d'une matrice (`r`, `s`) dont les coefficients sont des variables aléatoires indépendantes qui suivent la loi uniforme $\mathscr{U}\left([0;1]\right)$.
- `rd.randint(a,b,[r,s])` : Simule une réalisation d'une matrice (`r`, `s`) dont les coefficients sont des variables aléatoires indépendantes qui suivent la loi uniforme discrète $\mathscr{U}\left(\llbracket a;b-1\rrbracket\right)$.
- `rd.binomial(n,p,[r,s])` : Simule une réalisation d'une matrice (`r`, `s`) dont les coefficients sont des variables aléatoires indépendantes qui suivent la loi binomiale $\mathscr{B}(n;p)$
- `rd.geometric(p,[r,s])` : Simule une réalisation d'une matrice (`r`, `s`) dont les coefficients sont des variables aléatoires indépendantes qui suivent la loi géométrique $\mathscr{G}(p)$
- `rd.poisson(a,[r,s])` : Simule une réalisation d'une matrice (`r`, `s`) dont les coefficients sont des variables aléatoires indépendantes qui suivent la loi de Poisson $\mathscr{P}(a)$
- `rd.exponential(a,[r,s])` : Simule une réalisation d'une matrice (`r`, `s`) dont les coefficients sont des variables aléatoires indépendantes qui suivent la loi exponentielle $\mathscr{E}\left(\dfrac{1}{a}\right)$
- `rd.normal(m,d,[r,s])` : Simule une réalisation d'une matrice (`r`, `s`) dont les coefficients sont des variables aléatoires indépendantes qui suivent la loi normale $\mathscr{N}(m,d^2)$.

Si le paramètre `[r,s]` est remplacé par `r`, ces fonctions renvoient la réalisation d'un vecteur de longueur `r` correspondant à la loi en question, et si ce paramètre est omis, elles renvoient un seul coefficient suivant les mêmes contraintes.

Sous module graphique `pyplot` de `matplotlib`

`import matplotlib.pyplot as plt`

- `plt.plot(X,Y,options)` : Crée la courbe des points définis par les listes `X`, abscisses, et `Y`, ordonnées suivant les options graphiques définies par la chaîne de caractères `options`
- `plt.bar(X,Y)` : Crée le diagramme en bâtons défini par les listes ou vecteurs `X`, abscisses et `Y`, ordonnées
- `plt.hist(X,Y,density=True)` : Crée l'histogramme de fréquences des valeurs définies par la liste `X`, `Y` étant soit le nombre de classes, soit les bornes des classes
- `plt.boxplot(X)` : Génère le diagramme en boîte basé sur le vecteur de données `X`
- `plt.axis('equal')` : Rend le repère orthonormé
- `plt.xlim(xmin,xmax)` : Fixe les bornes de l'axe des abscisses
- `plt.ylim(ymin,ymax)` : Fixe les bornes de l'axe des ordonnées
- `plt.show()` : Affiche le graphique