Agro-Veto – Planche 1

Cours

Donner la définition de densité de probabilité.

Exercice préparé

1. (a) Soit f la fonction définie par $f : x \mapsto \dfrac{1}{x+1} + \dfrac{2}{x+2}$.
   Étudier son ensemble de définition, dresser son tableau de variations et déduire le nombre de solution de l'équation $f(x) = 1$.
   (b) Déterminer les solutions exactes de l'équation $f(x) = 1$.

Soit A_n la matrice de $\mathcal{M}_n(\mathbb{R})$ dont tous les coefficients diagonaux sont nuls et tels que les autres coefficients situés sur la colonne j sont égaux à j pour $j \in [\![1, n]\!]$.

2. (a) Écrire une fonction Python d'argument n renvoyant A_n.
   (b) En déduire une valeur approchée des valeurs propres de A_n pour $n \in \{2, 3, 4, 10\}$.
3. Pour $n \geq 2$, on pose $f_n : t \mapsto \displaystyle\sum_{k=1}^{n} \frac{k}{k+t}$ et on considère l'équation E_n d'inconnue $\lambda \in \mathbb{R}$: $f_n(\lambda) = 1$.
   (a) Montrer que E_n admet une unique solution dans chacun des intervalles $]-1, +\infty[$ et $]-k, -(k-1)[$ pour tout $k \in [\![2, n]\!]$.
   (b) On admet (pour l'instant) que les valeurs propres de A_n sont les solutions de l'équation E_n. La matrice A_n est-elle diagonalisable ?
4. Pour $n \geq 2$, on appelle λ_n la solution de E_n comprise entre -2 et -1.
   (a) Justifier :$\forall n \in \mathbb{N}^*,\ \forall t \in]-2, -1[,\ f_n(t) \leq f_{n+1}(t)$.
   En déduire la monotonie de la suite (λ_n).
   (b) Montrer que la suite (λ_n) converge vers un réel ℓ.
   Si l'on suppose $\ell \in]-2, -1[$, comparer $f_n(\lambda_n)$ et $f_n(\ell)$.
   (c) Pour $\lambda \in]-2, -1[$, calculer $\lim\limits_{n\to+\infty} f_n(\lambda)$ et conclure sur la valeur de ℓ.
   (d) Montrer que pour $k \in \mathbb{N}^*$, $\dfrac{1}{k} \leq 2(\sqrt{k} - \sqrt{k-1})$ et en déduire $\lim\limits_{n\to+\infty} \dfrac{1}{n} \displaystyle\sum_{k=1}^{n} \frac{1}{k}$.
   (e) Déterminer un équivalent de $\dfrac{1}{1+\lambda_n}$ puis de $\lambda_n + 1$.
   On pourra utiliser un encadrement de $\displaystyle\sum_{k=3}^{n} \frac{k}{k+\lambda_n}$.
5. On souhaite montrer ce qui a été admis en 3.(b).
   Montrer à l'aide d'un système, que les valeurs propres de A_n sont les solutions de E_n.

Agro-Veto – Planche 1

Exercice sans préparation

1. On considère des listes non-vides ne contenant que une ou deux valeurs différentes.
   Par exemple : `['Marwa', 'Ambre', ' Marwa','Ambre','Ambre']`.
   Écrire, en langage Python, une fonction nommée `election` qui prend en entrée une liste `L` de cette forme et qui renvoie l'élément majoritaire. La fonction doit renvoyer `None` en cas d'égalité.
2. On considère maintenant des listes non-vides mais pouvant contenir plus de deux valeurs différentes.
   La liste représente une urne et on veut savoir qui a obtenu le plus de voix.
   Écrire, en langage Python, une fonction nommée `election` qui prend en entrée une liste `L` de cette forme et qui renvoie la liste des personnes ayant obtenu le maximum de voix (la liste contient plusieurs personnes en cas d'égalité).

Éléments de correction–Planche 1

Exercice avec préparation

1. (a) La fonction f est définie et dérivable sur $\mathbb{R}\backslash\{-1,-2\}$ et

$$\forall x \in \mathbb{R}\backslash\{-1,-2\}, \quad f'(x) = -\frac{1}{(x+1)^2} - \frac{2}{(x+2)^2} < 0.$$

Ainsi :

x	$-\infty$		-2		-1		$+\infty$
f	0	$\searrow$	$-\infty$ / $+\infty$	$\searrow$	$-\infty$ / $+\infty$	$\searrow$	0

En appliquant le TVI, on obtient deux solutions à l'équation $f(x) = 1$: une sur $]-2,-1[$ et une sur $]-1,+\infty[$.

(b) Soit $x \in \mathbb{R}\backslash\{-1,-2\}$.

$$f(x) = 1 \iff (x+1)(x+2) = x+2+2(x+1) \iff x^2 = 2$$
$$\iff x = \sqrt{2} \quad \text{ou} \quad x = -\sqrt{2}.$$

Soit A_n la matrice de $\mathcal{M}_n(\mathbb{R})$ dont tous les coefficients diagonaux sont nuls et tels que les autres coefficients situés sur la colonne j sont égaux à j pour $j \in [\![1,n]\!]$.

2. (a) Écrire une fonction Python d'argument n renvoyant A_n.

```
def A(n):
        mat = np.zeros([n,n])
        for j in range(n):
                for i in range(n):
                        if i !=j :
                                mat[i,j]=j+1
        return mat
```

(b) En déduire une valeur approchée des valeurs propres de A_n pour $n \in \{2,3,4,10\}$.

```
import numpy.linalg as la
for n in [2,3,4,10]:
        print(la.eigvals(A(n))
```

3. Pour $n \geq 2$, on pose $f_n : t \mapsto \sum_{k=1}^{n} \frac{k}{k+t}$ et on considère l'équation E_n d'inconnue $\lambda \in \mathbb{R}$: $f_n(\lambda) = 1$.

(a) La fonction f_n est définie et dérivable sur $\mathbb{R}\backslash\{-1,\dots,-n\}$ et

$$\forall x \in \mathbb{R}\backslash\{-1,\dots,-n\}, \quad f_n'(x) = -\sum_{k=1}^{n} \frac{n}{(x+n)^2} < 0.$$

Ainsi f_n est strictement décroissante sur $]-\infty,-n[$, $]-1,+\infty[$ et $]-k,-(k-1)[$ pour $k \in [\![2,n]\!]$.

En considérant les limites aux bords de chacun des intervalles ci-dessus et en appliquant le TVI, on obtient une unique solution dans chacun des intervalles $]-1,+\infty[$ et $]-k,-(k-1)[$ pour tout $k \in [\![2,n]\!]$.

(b) On admet (pour l'instant) que les valeurs propres de A_n sont les solutions de l'équation E_n.

D'après la question précédente, A_n possède n valeurs propres distinctes donc elle est diagonalisable.

4. Pour $n \geq 2$, on appelle λ_n la solution de E_n comprise entre -2 et -1.

(a) Soit $n \in \mathbb{N}^*$ et $t \in]-2,-1[$. Alors :

$$f_{n+1}(t) = f_n(t) + \frac{n+1}{n+1+t}.$$

Or $\dfrac{n+1}{n+1+t} > 0$ donc :

$$f_{n+1}(t) \geq f_n(t).$$

On en déduit :

$$f_{n+1}(\lambda_n) \geq f_n(\lambda_n) = 1 = f_{n+1}(\lambda_{n+1})$$

puis par stricte décroissance de f_{n+1} sur $]-2,-1[$:

$$\lambda_n \leq \lambda_{n+1}.$$

La suite (λ_n) est donc croissante.

(b) La suite (λ_n) est croissante et majorée par -1 donc elle converge vers un réel ℓ.

Si l'on suppose $\ell \in]-2,-1[$, par croissance de la suite alors :

$$\forall n \in \mathbb{N}^*, \quad \lambda_n \leq \ell$$

puis par décroissance de f_n sur $]-2,-1[$:

$$\forall n \in \mathbb{N}^*, \quad f_n(\lambda_n) \geq f_n(\ell).$$

(c) Soit $\lambda \in]-2,-1[$. Pour tout $k \geq 3$, on a : $0 < k+\lambda < k$ donc $1 \leq \dfrac{k}{k+\lambda}$.

Ainsi :

$$f_n(\lambda) \geq \frac{1}{1+\lambda} + \frac{2}{2+\lambda} + n - 3.$$

D'où : $\lim\limits_{n\to+\infty} f_n(\lambda) = +\infty$.

Par passage à la limite dans les inégalités, on sait que $\ell \in [-2,-1]$.

Si $\ell \in]-2,-1[$ alors par ce qui précède on aurait :

$$\forall n \in \mathbb{N}^*, \quad f_n(\ell) \leq f_n(\lambda_n) = 1$$

ce qui contredit le fait que $\lim_{n\to+\infty} f_n(\ell) = +\infty$.

Ainsi $\ell \in \{-1, -2\}$. Or par croissance de la suite :

$$-2 < \lambda_1 \leq \ell$$

ce qui permet de conclure : $\ell = -1$.

(d) Soit $k \in \mathbb{N}^*$.

$$\frac{1}{k} \leq 2(\sqrt{k} - \sqrt{k-1}) \iff \frac{1}{k} \leq \frac{2}{\sqrt{k} + \sqrt{k-1}} \iff \sqrt{k} + \sqrt{k-1} \leq 2k.$$

Cette dernière inégalité est trivialement vraie d'où le résultat.

On en déduit, pour tout $n \in \mathbb{N}^*$:

$$0 \leq \frac{1}{n}\sum_{k=1}^{n} \frac{1}{k} \leq \frac{2}{n}\sum_{k=1}^{n}(\sqrt{k} - \sqrt{k-1}) = \frac{2\sqrt{n}}{n} \xrightarrow[n\to+\infty]{} 0.$$

Donc $\lim_{n\to+\infty} \frac{1}{n}\sum_{k=1}^{n} \frac{1}{k} = 0$.

(e) On a :

$$(*)1 = f_n(\lambda_n) = \frac{1}{1+\lambda_n} + \frac{2}{2+\lambda_n} + \sum_{k=3}^{n} \frac{k}{k+\lambda_n}.$$

Or comme $\lambda_n \in]-2, -1[$ on a pour tout $k \geq 3$:

$$k - 2 \leq k + \lambda_n \leq k$$

d'où

$$n - 3 \leq \sum_{k=3}^{n} \frac{k}{k+\lambda_n} \leq n - 3 + 2\sum_{k=3}^{n} \frac{1}{k+\lambda_n} \leq n - 3 + 2\sum_{k=3}^{n} \frac{1}{k-2}$$

d'où :

$$1 - \frac{3}{n} \leq \frac{1}{n}\sum_{k=3}^{n} \frac{k}{k+\lambda_n} \leq n - 3 + 2\sum_{k=3}^{n} \frac{1}{k+\lambda_n} \leq 1 - \frac{3}{n} + \frac{1}{n}\sum_{k=3}^{n} \frac{1}{k-2}.$$

Or avec la question précédente, le membre de gauche et de droite convergent vers 1 donc par encadrement on en déduit :

$$\lim_{n\to+\infty} \frac{1}{n}\sum_{k=3}^{n} \frac{k}{k+\lambda_n} = 1$$

c'est-à-dire : $\sum_{k=3}^{n} \frac{k}{k+\lambda_n} \underset{n\to+\infty}{\sim} n$.

Enfin $(*)$ donne

$$\begin{aligned}\frac{1}{1+\lambda_n} &= 1 - \frac{2}{2+\lambda_n} - \sum_{k=3}^{n} \frac{k}{k+\lambda_n} \\ &= -n + \underset{n\to+\infty}{o}(n) \\ &\underset{n\to+\infty}{\sim} -n.\end{aligned}$$

Par compatibilité avec l'inverse :

$$1+\lambda_n \underset{n\to+\infty}{\sim} -\frac{1}{n}.$$

5. Soit $\lambda \in \mathbb{R}$ et $X = \begin{pmatrix} x_1 \\ \dots \\ x_n \end{pmatrix}$ non nul.

$$A_nX = \lambda X \iff \left\{ \begin{array}{rcl} -\lambda x_1 + 2x_2 + \cdots + nx_n &=& 0 \\ x_1 - \lambda x_2 + \cdots + nx_n &=& 0 \\ \vdots & \vdots & \vdots \\ x_1 + 2x_2 + \cdots - \lambda x_n &=& 0. \end{array} \right.$$

En effectuant $L_2 \leftarrow L_2 - L_1$,..., $L_n \leftarrow L_n - L_1$:

$$A_nX = \lambda X \iff \left\{ \begin{array}{rcl} -\lambda x_1 + 2x_2 + \cdots + nx_n &=& 0 \\ (1+\lambda)x_1 - (\lambda+2)x_2 &=& 0 \\ \vdots & \vdots & \vdots \\ (1+\lambda)x_1 - (\lambda+n)x_n &=& 0. \end{array} \right.$$

On remarque déjà que $\lambda \neq -1$ car sinon les lignes 2 à n donnent $x_2 = \cdots = x_n = 0$ puis la ligne 1 donne $x_1 = 0$; ce qui contredit le fait que X est non nul.
De même, $\lambda \notin \{-2, \dots, -n\}$. En effet, si $\lambda = i$, $i \in \{-2, \dots, -n\}$ alors la ligne i donne $x_1 = 0$ (car on a déjà vu que $\lambda + 1 \neq 0$) puis les lignes $j \in [\![2, n]\!]$, $j \neq i$ donnent $x_j = 0$ puis la ligne 1 donne $x_i = 0$; cela contredit X non nul.
Ainsi on obtient

$$A_nX = \lambda X \iff \lambda \notin [\![-n, -1]\!] \text{ et } \left\{ \begin{array}{rcl} -\lambda x_1 + 2x_2 + \cdots + nx_n &=& 0 \\ \forall i \in [\![2, n]\!] \quad x_i = \dfrac{1+\lambda}{i+\lambda} x_1 & & \end{array} \right.$$

$$\iff \lambda \notin [\![-n, -1]\!] \text{ et } \left\{ \begin{array}{rcl} -\lambda x_1 + \displaystyle\sum_{i=2}^{n} i\frac{1+\lambda}{i+\lambda} x_1 &=& 0 \\ \forall i \in [\![2, n]\!] \quad x_i = \dfrac{1+\lambda}{i+\lambda} x_1 & & \end{array} \right.$$

$$\iff \lambda \notin [\![-n, -1]\!] \text{ et } \left\{ \begin{array}{rcl} -\lambda + \displaystyle\sum_{i=2}^{n} i\frac{1+\lambda}{i+\lambda} &=& 0 \\ \forall i \in [\![2, n]\!] \quad x_i = \dfrac{1+\lambda}{i+\lambda} x_1 & & \end{array} \right. \qquad (x_1 \neq 0 \quad \text{car } X \neq 0)$$

$$\iff \lambda \notin [\![-n, -1]\!] \text{ et } \left\{ \begin{array}{rcl} -\dfrac{\lambda}{1+\lambda} + \displaystyle\sum_{i=2}^{n} \frac{1}{i+\lambda} &=& 0 \\ \forall i \in [\![2, n]\!] \quad x_i = \dfrac{1+\lambda}{i+\lambda} x_1 & & \end{array} \right. \qquad (L_1 \leftarrow \frac{1}{1+\lambda} L_1).$$

$$\iff \lambda \notin [\![-n, -1]\!] \text{ et } \left\{ \begin{array}{rcl} -1 + \dfrac{1}{1+\lambda} + \displaystyle\sum_{i=2}^{n} \frac{1}{i+\lambda} &=& 0 \\ \forall i \in [\![2, n]\!] \quad x_i = \dfrac{1+\lambda}{i+\lambda} x_1 & & \end{array} \right. \qquad (L_1 \leftarrow \frac{1}{1+\lambda} L_1).$$

Ainsi les valeurs propres sont les solutions de E_n.

Exercice sans préparation

1.
```
def election(L):
    D = {}
    for elt in L:
        if elt in D.keys():
            D[elt] += 1
        else:
            D[elt]=1
    maxi = max([D[key] for key in D.keys()])
    L = []
    for key in D.keys():
        if D[key] == maxi:
            L.append(key)
    if len(L)>1:
                return 'None'

    return L[0]
```

2.
```
def election(L):
    D = {}
    for elt in L:
        if elt in D.keys():
            D[elt] += 1
        else:
            D[elt]=1
    maxi = max([D[key] for key in D.keys()])
    L = []
    for key in D.keys():
        if D[key] == maxi:
            L.append(key)
    return L
```

Agro-Veto – Planche 2

Cours

Donner la définition d'une matrice inversible et de l'inverse d'une matrice inversible.

Exercice préparé

On cherche à trouver des individus au sein d'une population possédant une propriété détectable par une analyse de sang. On fixe $q \in]0,1[$ et l'on suppose que les individus ont, indépendamment les uns des autres, une probabilité q de ne pas posséder la propriété recherchée.
Le résultat de l'analyse d'un échantillon de sang est dit positif si la propriété est présente, négatif si elle ne l'est pas. On va étudier divers protocoles de test.
On désire dans un premier temps trouver toutes les personnes qui ont la propriété dans un ensemble de n personnes, où n est un entier tel que $n \geq 2$.

1. Dans cette question, on étudie le protocole A, qui consiste à mélanger le sang des n personnes et analyser ce mélange. Si le résultat est négatif, on s'arrête (car alors personne ne possède la propriété recherchée). S'il est positif, on analyse alors individuellement le sang de chacune des n personnes. On note A_n la variable aléatoire qui compte le nombre d'analyses effectuées en appliquant ce protocole A pour n personnes.
   (a) Déterminer $A_n(\Omega)$. A_n admet-elle une espérance ?
   (b) Déterminer la loi de A_n.
   (c) Écrire une fonction Python qui prend en argument une liste L de n booléens et renvoie la valeur de A_n, en considérant qu'un `True` en k^e position signifie que la k^e personne possède la propriété recherchée et un `False` qu'elle ne le possède pas.
   (d) Utiliser la fonction précédente pour estimer numériquement $\mathbb{E}(A_{10})$ avec $q = 0.9$.
   (e) Prouver que $\mathbb{E}(A_n) = n + 1 - nq^n$.
   (f) On considère un entier naturel $k \in [\![1, n-1]\!]$. Calculer la probabilité que les k premières personnes testées soient toutes négatives sachant que le résultat de l'analyse du mélange est positif.
2. Dans cette question, on étudie le protocole B qui consiste à directement analyser individuellement le sang de chacune des n personnes.
   (a) À quelle condition sur q fait-on, en moyenne, moins de tests avec le protocole A qu'avec le protocole B ? On exprimera le résultat en fonction de n.
   (b) Étudier les variations et calculer la limite à droite en 0 de $x \mapsto x^x$.
   (c) Justifier que pour n assez grand, l'un des deux protocoles (que l'on déterminera) est préférable à l'autre (c'est-à-dire donne lieu à moins d'analyses en moyenne).
3. Dans cette question, on étudie un procédé *par regroupement* : on mélange le sang des n premières personnes de la population puis l'on teste ce mélange. Si le résultat est négatif, on procède de même avec les n personnes suivantes.
   Dès lors qu'un groupe de n personnes est testé positivement, on teste alors individuellement les n personnes de ce groupe, jusqu'à trouver la première personne possédant la propriété recherchée.
   On note G la variable aléatoire représentant le numéro du premier groupe positif. Ainsi, $G = 1$ si c'est le premier groupe qui a donné un test positif,$G = 2$ si c'est le second, etc. On considère k un entier strictement positif.
   (a) Calculer la probabilité $\mathbb{P}(G > k)$.
   (b) Déterminer la loi de G.

Agro-Veto – Planche 2

Exercice sans préparation

Soit $n \in \mathbb{N}^*$.
On parle de diviseur *strict* de n pour désigner tout diviseur k de n tel que $1 \leq k < n$.
On note s_n la somme des diviseurs stricts de n et on dit qu'un entier n est *parfait* si $s_n = n$.
En Python, la commande `n%k` renvoie le reste de la division de n par k (ce reste est nul si et seulement si k divise n).

1. Écrire une fonction en Python, nommée `s` qui prend en argument un entier n non nul et qui renvoie la somme de ses diviseurs stricts.
2. Écrire une fonction en Python, nommée `prop_parf` qui prend en argument un entier N non nul et qui renvoie la proportion de nombres parfaits dans l'intervalle $[\![1, N]\!]$.

Éléments de correction–Planche 2

Exercice avec préparation

1. (a) Soit on effectue un seul test (lorsque le mélange est négatif) soit on effectue $n+1$ tests (celui du mélange et celui des n individus). Donc $A_n(\Omega) = \{1, n+1\}$.
   Comme A_n est de support fini elle possède une espérance.

   (b) Par indépendance des individus : $\mathbb{P}(A_n = 1) = q^n$ donc $P(A_n = n+1) = 1 - q^n$.

   (c)
```
def A(L):
        n = len(L)
        k = 0
        test = L[k]
        while test == False and k<len(L):
                k = k+1
                test=L[k]
        if test == False:
                return 1
        return n+1
```

   ou mieux :

```
def A(L):
        n = len(L)
        for elt in L:
                if elt :
                        return n+1
        return 1
```

   (d) On commence par faire une fonction qui crée une liste de 10 personnes chacune ayant la probabilité q de ne pas avoir la propriété.

```
def echantillon(n,q):
        L = []
        for k in range(n):
                if rd.rand()<1-q:
                        L.append(True)
                else:
                        L.append(False)
        return L
```

   La commande `A(echantillon(10,0.9))` simule une réalisation de A_{10}. D'après la loi des grands nombres, le programme suivant permet l'estimation voulue :

```
moy = 0
for k in range(1000):
        moy += A(echantillon(10,0.9))
print(moy/1000)
```

   (e) D'après la question 1.(b) :

$$\mathbb{E}(A_n) = q^n + (n+1)(1-q^n) = n+1-nq^n.$$

(f) Soit X la variable aléatoire donnant le rang du premier individu positif dans un échantillon de n individus possédant un positif. X est à valeurs dans $[\![1, n]\!]$.
On a alors pour tout $i \in [\![1, n-1]\!]$, $\mathbb{P}(X = i) = q^{i-1}(1-q)$ et $\mathbb{P}(X = n) = q^{n-1}$.
La probabilité recherchée est alors :

$$\mathbb{P}(X \geq k+1) = \sum_{i=k+1}^{n} \mathbb{P}(X = i) = \sum_{i=k+1}^{n-1} q^{i-1}(1-q) + q^{n-1} = q^k.$$

2. (a) On cherche les valeurs de q telles que :

$$n + 1 - nq^n \leq n$$

ce qui équivaut à $\left(\frac{1}{n}\right)^{\frac{1}{n}} \leq q$.

(b) Notons $g : x \mapsto x^x = e^{x\ln(x)}$ définie et dérivable sur $\mathbb{R}_+^*$.
Par croissances comparées, $\lim\limits_{x\to 0^+} x\ln(x) = 0$ donc par continuité de la fonction exponentielle, $\lim\limits_{x\to 0^+} e^{x\ln(x)} = 1$.
Pour tout $x > 0$, $g'(x) = (1 + \ln(x))e^{x\ln(x)}$ donc :
— g est strictement décroissante sur $]0, e^{-1}]$;
— g est strictement croissante sur $[e^{-1}, +\infty[$.

(c) D'après la question précédente, $\lim\limits_{n\to+\infty} \left(\frac{1}{n}\right)^{\frac{1}{n}} = 1$ donc pour n grand on a (à q fixé) :

$$q < \left(\frac{1}{n}\right)^{\frac{1}{n}}.$$

Le protocole B est donc meilleur.

3. (a) La probabilité qu'un groupe soit négatif est, par indépendance des individus, q^n.
Alors, (cf question suivante et le cours) :

$$\mathbb{P}(G > k) = q^{nk}.$$

(b) G suit donc une loi géométrique de paramètre $1 - q^n$.

Exercice sans préparation

Soit $n \in \mathbb{N}^*$.
On parle de diviseur *strict* de n pour désigner tout diviseur k de n tel que $1 \leq k < n$.
On note s_n la somme des diviseurs stricts de n et on dit qu'un entier n est *parfait* si $s_n = n$.
En Python, la commande `n%k` renvoie le reste de la division de n par k (ce reste est nul si et seulement si k divise n).

1. Écrire une fonction en Python, nommée `s` qui prend en argument un entier n non nul et qui renvoie la somme de ses diviseurs stricts.

```
def s(n):
        somme = 0
        for k in range(1,n):
                if n%k == 0:
                        somme += k
        return somme
```

2. Écrire une fonction en Python, nommée `prop_parf` qui prend en argument un entier N non nul et qui renvoie la proportion de nombres parfaits dans l'intervalle $[\![1, N]\!]$.

```
def prop_parf(N):
        comp = 0
        for k in range(1,N+1):
                if s(k)==k :
                        comp +=1
        return comp/N
```

Agro-Veto – Planche 3

Cours

Énoncer le théorème de changement de variable pour les intégrales sur un segment.

Exercice préparé

Tous les vecteurs et matrices sont à coefficients réels.

1. Soit D une matrice diagonale d'ordre $n \geq 1$ dont les éléments diagonaux sont $(d_1, \dots, d_n)$.
   (a) Soit $X = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$ un vecteur colonne. Vérifier que $X^T D X = \sum_{i=1}^{n} d_i x_i^2$.
   (b) En déduire que les coefficients diagonaux de D sont strictement positifs si et seulement si $X^T D X > 0$ pour tout vecteur colonne X non nul.
2. (a) Écrire une fonction en langage Python nommée `f` qui prend en entrée une matrice carrée M et qui renvoie $X^T M X$ où X est un vecteur colonne dont les coefficients sont des variables aléatoires indépendantes suivant la loi uniforme sur $[0, 1]$.
   (b) Écrire un script qui affiche le nombre de fois où l'inégalité $X^T(A - 3I)X > 0$ est vérifiée après 100 exécutions de la fonction où A est la matrice ci-dessous et I la matrice identité d'ordre 3.
3. Soit $A = \begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix}$.
   On considère les vecteurs $U_1 = \frac{1}{\sqrt{3}} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$, $U_2 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix}$ et $U_3 = \frac{1}{\sqrt{6}} \begin{pmatrix} 1 \\ 1 \\ -2 \end{pmatrix}$.
   (a) Montrer que U_1 est un vecteur propre de A et donner la valeur propre associée.
   (b) Montrer que l'ensemble des vecteurs X tels que $AX = X$ est un sous-espace propre de A admettant (U_2, U_3) pour base orthonormée.
   (c) Déterminer une matrice P et une matrice diagonale D telles que $A = PD^2P^T$ et $P^TP = I$.
   (d) En déduire qu'il existe une matrice inversible L telle que $A = LL^T$.
4. Soit $B \in \mathcal{M}_3(\mathbb{R})$ une matrice symétrique.
   (a) Vérifier que $C = L^{-1}B(L^{-1})^T$ est symétrique (où L est la matrice de la question précédente).
   En déduire une matrice diagonale $\Delta \in \mathcal{M}_3(\mathbb{R})$ et une matrice orthogonale $Q \in \mathcal{M}_3(\mathbb{R})$ telles que $B = LQ\Delta(LQ)^T$.
   (b) On pose $R = LQ$ de sorte que $B = R\Delta R^T$. Calculer RR^T.

Agro-Veto – Planche 3

Exercice sans préparation

1. Écrire une fonction en Python, nommée `alea` qui prend en argument un entier positif k et renvoie 1 avec la probabilité $\dfrac{k+1}{k+2}$ et 0 avec la probabilité $\dfrac{1}{k+2}$.
2. Un mobile se déplace sur les points à coordonnées entières d'un axe selon les règles suivantes :
   — à l'instant $n = 0$, le mobile est au point d'abscisse 0 ;
   — si à l'instant $n \in \mathbb{N}$, le mobile est au point d'abscisse k alors à l'instant $n+1$, il est au point d'abscisse $k+1$ avec probabilité $\dfrac{k+1}{k+2}$ ou au point d'abscisse 0 avec la probabilité $\dfrac{1}{k+2}$.

   On note X_n l'abscisse du mobile à l'instant n.
   (a) Écrire une fonction en Python, nommée `simulX` qui prend en argument un entier naturel non nul n et qui simule n déplacement du mobile et renvoie la valeur de X_n.
   (b) Écrire une fonction en Python, nommée `attend` qui ne prend pas d'argument, qui simule les déplacements du mobile jusqu'au premier retour au point d'abscisse 0 et renvoie le plus petit n strictement positif pour lequel $X_n = 0$.

Éléments de correction–Planche 3

Exercice avec préparation

Tous les vecteurs et matrices sont à coefficients réels.

1. Soit D une matrice diagonale d'ordre $n \geq 1$ dont les éléments diagonaux sont $(d_1, \dots, d_n)$.
   (a) Simple calcul.
   (b) $\Rightarrow$ Supposons que les coefficients diagonaux de D sont strictement positifs et soit X vecteur colonne non nul.
   Alors : $X^T D X = \sum_{i=1}^{n} d_i x_i^2$ est une somme de nombres positifs dont au moins l'un est non nul (car X non nul) donc est strictement positif.
   $\Leftarrow$ Supposons que pour tout vecteur colonne X non nul $X^T D X > 0$.
   Alors pour tout $i \in [\![1, n]\!]$, un prenant pour X le i-ème vecteur de la base canonique, on a d'après la question précédente :
   $$0 < X^T D X = d_i.$$

2. (a)
```
def f(M):
        n = np.shape(M)[0]
        xt = rd.rand(n)
        mx = np.dot(M,xt.transpose())
        return np.dot(xt,mx)
```
   (b)
```
A = np.array([[2,1,1],[1,2,1],[1,1,2]])
B = A-3*np.eye(3)
nb = 0
for k in range(100):
        if f(B)>0:
                nb +=1
print(nb)
```

3. Soit $A = \begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix}$.
   On considère les vecteurs $U_1 = \frac{1}{\sqrt{3}} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$, $U_2 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix}$ et $U_3 = \frac{1}{\sqrt{6}} \begin{pmatrix} 1 \\ 1 \\ -2 \end{pmatrix}$.
   (a) On a $AU_1 = 4U_1$ donc U_1 est un vecteur propre associé à la valeur propre 4.
   (b) On vérifie facilement que $AU_2 = U_2$ et $AU_3 = U_3$ de sorte que 1 est bien valeur propre de A.
   De plus, il est facile de voir que la famille (U_2, U_3) est orthonormée donc libre. Ainsi (U_2, U_3) est une famille libre de $E_1(A)$ donc $\dim(E_1(A)) \geq 2$.
   Or on sait que $\dim(E_1(A)) \leq 2$ (car la somme des sous-espace propre est inférieure à 3 et $\dim(E_4(A)) \geq 1$).
   Ainsi $\dim(E_1(A)) = 2$ et (U_2, U_3) en est une base orthonormée.

(c) Soit $P = (U_1|U_2|U_3)$. Alors par les formules de changement de base :

$$A = P \begin{pmatrix} 4 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} P^{-1}.$$

De plus, comme la famille (U_1, U_2, U_3) est une base orthonormée de $\mathcal{M}_{3,1}(\mathbb{R})$ (cela se vérifie facilement) alors

$$P^T P = I \quad ie \quad P^{-1} = P^T.$$

En posant $D = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ on a alors le résultat.

(d) On pose $L = PD$ de sorte que $L^T = D^T P^T = DP^T$.
D'une part, $LL^T = PD^2P^T = A$.
D'autre part, L est inversible comme produit de deux matrices inversibles.

4. Soit $B \in \mathcal{M}_3(\mathbb{R})$ symétrique.

(a) Soit $C = L^{-1}B(L^{-1})^T$.

$$C^T = ((L^{-1})^T)^T B^T (L^{-1})^T = L^{-1}B^T(L^{-1})^T = L^{-1}B(L^{-1})^T = C.$$

C est symétrique à coefficients réels donc diagonalisable en base orthonormée d'après le théorème spectrale : il existe une matrice diagonale $\Delta \in \mathcal{M}_3(\mathbb{R})$ et une matrice orthogonale $Q \in \mathcal{M}_3(\mathbb{R})$ telles que

$$L^{-1}B(L^{-1})^T = C = Q\Delta Q^T.$$

En multipliant à gauche par L et à droite par L^T membre à membre :

$$B = LQ\Delta Q^T L^T = LQ\Delta(LQ)^T.$$

(b) On pose $R = LQ$ de sorte que $B = R\Delta R^T$. On a :

$$\begin{aligned} RR^T = LQ(LQ)^T &= LQQ^T L^T \\ &= LIL^T \quad \text{car } Q \text{ est orthogonale} \\ &= LL^T \\ &= A \end{aligned}$$

Exercice sans préparation

1.
```
import numpy.random as rd

def alea(k):
        if rd.rand()<(k+1)/(k+2):
                return 1
        else:
                return 0
```

2. Un mobile se déplace sur les points à coordonnées entières d'un axe selon les règles suivantes :
   — à l'instant $n = 0$, le mobile est au point d'abscisse 0 ;
   — si à l'instant $n \in \mathbb{N}$, le mobile est au point d'abscisse k alors à l'instant $n+1$, il est au point d'abscisse $k+1$ avec probabilité $\dfrac{k+1}{k+2}$ ou au point d'abscisse 0 avec la probabilité $\dfrac{1}{k+2}$.

   On note X_n l'abscisse du mobile à l'instant n.

   (a)
```
def simulX(n):
        X = 0
        for k in range(n):
                mouv = alea(X)
                if mouv == 1:
                        X += 1
                else:
                        X = 0
        return X
```

   (b)
```
def attend():
        X = alea(0)
        if X == 0 :
                return 1
        else:
                n = 1
                while X != 0:
                        mouv = alea(X)
                        if mouv == 1:
                                X += 1
                        else:
                                X = 0
                        n += 1
        return n
```

Agro-Veto – Planche 4

Cours

Énoncer le lemme des coalitions.

Exercice préparé

1. On considère les équations différentielles suivantes d'inconnue $y : \mathbb{R} \to \mathbb{R}$:

$$(H) \quad y'' - 4y' + 5y = 0,$$

$$(E) \quad y'' - 4y' + 5y = 2 - e^{2x}.$$

   (a) Déterminer l'ensemble des solutions de l'équation (H).
   (b) En déduire l'ensemble des solutions de l'équation (E).
   On pourra chercher une solution particulière y_0 de l'équation différentielle

$$y'' - 4y' + 5y = -e^{2x}$$

   sous la forme $y_0 : x \mapsto ce^{2x}$ où c est un réel à déterminer.

2. Pour tout réel x, on note $C(x) = \displaystyle\int_0^x e^{2t}\cos(t)dt$ et $S(x) = \displaystyle\int_0^x e^{2t}\sin(t)dt$.
   (a) Montrer que pour tout réel x, $C(x) = \dfrac{e^{2x}\cos(x) - 1}{2} + \dfrac{1}{2}S(x)$ et $S(x) = \dfrac{e^{2x}\sin(x)}{2} - \dfrac{1}{2}C(x)$.
   (b) En déduire une primitive de $x \mapsto e^{2x}\cos(x)$ et de $x \mapsto e^{2x}\sin(x)$ sur $\mathbb{R}$.

3. Écrire une fonction Python nommée `intC` prenant en paramètres un réel x et un entier `nb_pas` et qui renvoie une valeur approchée de $C(x)$ par la méthode des rectangles en découpant $[0, x]$ en `nb_pas` intervalles de même longueur.

On note $E = \mathcal{C}^\infty(\mathbb{R}, \mathbb{R})$ et on considère les éléments de E suivants :

$$f_1 : x \mapsto 1 \quad ; \quad f_2 : x \mapsto e^{2x} \quad ; \quad f_3 : x \mapsto e^{2x}\cos(x) \quad ; \quad f_4 : x \mapsto e^{2x}\sin(x).$$

On note F le sous-espace vectoriel de E engendré par $\mathcal{B} = (f_1, \dots, f_4)$.

4. Montrer que $\mathcal{B}$ est une base de F.
5. On note u l'application linéaire définie sur F par $u(f) = f'$ pour tout $f \in F$.
   (a) Montrer que u est linéaire.
   (b) Calculer l'image par u de $f_1, \dots, f_4$.
   (c) En déduire que u est un endomorphisme de F et déterminer $A = \mathrm{Mat}_{\mathcal{B}}(u)$.
6. Résoudre l'équation $AX = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$ d'inconnue $X \in \mathcal{M}_{4,1}(\mathbb{R})$.
   Quel résultat des questions précédentes retrouve-t-on ainsi ? Justifier.
7. Résoudre l'équation $(A^2 - 4A + 5I_4)X = \begin{pmatrix} 2 \\ -1 \\ 0 \\ 0 \end{pmatrix}$ d'inconnue $X \in \mathcal{M}_{4,1}(\mathbb{R})$.
   Quel résultat des questions précédentes retrouve-t-on ainsi ? Justifier.

Agro-Veto – Planche 4

Exercice sans préparation

On appelle *vecteurs creux* des vecteurs de grande dimension dont la grande majorité des coordonnées sont nulles.
Pour coder un vecteur creux, on utilise une liste de deux listes – celle de ses coordonnées non nulles et celle des indices correspondants triée par ordre croissant – suivies de la taille du vecteur.
Ainsi, le vecteur de $\mathbb{R}^9$ suivant $[1, 0, 3, 0, 0, 0, 0, -1, 0]$ est représenté par $[[1, 3, -1], [0, 2, 7], 9]$.
Les vecteurs usuels sont eux codés par des vecteurs `numpy`.

1. Écrire une fonction en Python, nommée `coder` renvoyant la représentation creuse d'un vecteur.
   Écrire une fonction `decoder` faisant le travail inverse.
   Par exemple :
   `coder([1,0,3,0,0,0,0,-1,0])` doit renvoyer `[[1,3,-1],[0,2,7],9]`.
   `decoder([[1,3,-1],[0,2,7],9])` doit renvoyer `[1,0,3,0,0,0,0,-1,0]` .
2. Pour de tels vecteurs, il serait trop coûteux, tant en temps qu'en mémoire, d'écrire toutes les coordonnées nulles et de calculer habituellement dans $\mathbb{R}^n$. C'est pourquoi on utilise le codage de la question précédente.
   Écrire une fonction `prod_scal(V1,V2)` prenant un argument deux vecteurs creux appartenant au même espace $\mathbb{R}^n$, codés comme dans la question 1 et qui calcule leur produit scalaire.
   Il n'est pas autorisé de repasser par la forme usuelle en décodant `V1` et `V2`.

Éléments de correction–Planche 4

Exercice avec préparation

1. On considère les équations différentielles suivantes d'inconnue $y : \mathbb{R} \to \mathbb{R}$:

$$(H) \quad y'' - 4y' + 5y = 0,$$

$$(E) \quad y'' - 4y' + 5y = 2 - e^{2x}.$$

   (a) Il s'agit d'une équation différentielle linéaire homogène d'ordre 2 à coefficients constants. Son équation caractéristique est :

$$r^2 - 4r + 5 = 0$$

   dont le discriminant est $\Delta = -4$. Les racines (complexes) sont donc :

$$r_1 = \frac{4 + 2i}{2} = 2 + i \quad ; \quad r_2 = 2 - i.$$

   Ainsi, les solutions de (H) sont les fonctions de la forme :

$$y : x \longmapsto ae^{2x}\cos(x) + be^{2x}\sin(x) \quad , (a, b) \in \mathbb{R}^2.$$

   (b) L'équation homogène associée à (E) est (H).

   — Une solution particulière de $y'' - 4y' + 5y = 2$ est la fonction constante $x \mapsto \frac{2}{5}$.

   — Cherchons une solution particulière y_0 de l'équation différentielle

$$y'' - 4y' + 5y = -e^{2x}$$

   sous la forme $y_0 : x \mapsto ce^{2x}$ où c est un réel à déterminer.
   La fonction y_0 ci-dessus est solution de l'équation si et seulement si pour tout $x \in \mathbb{R}$

$$(4c - 8c + 5c)e^{2x} = -e^{2x}$$

   si et seulement si $c = -1$.

   Par principe de superposition, les solutions de l'équation (E) sont les fonctions de la forme :

$$y : x \longmapsto ae^{2x}\cos(x) + be^{2x}\sin(x) - e^{2x} + \frac{2}{5} \quad , (a, b) \in \mathbb{R}^2.$$

2. Pour tout réel x, on note $C(x) = \displaystyle\int_0^x e^{2t}\cos(t)dt$ et $S(x) = \displaystyle\int_0^x e^{2t}\sin(t)dt$.

   (a) Soit $x \in \mathbb{R}$.

   Les fonctions $t \mapsto \frac{1}{2}e^{2t}$ et $t \mapsto \cos(t)$ sont de classe $\mathcal{C}^1$ sur $\mathbb{R}$ donc par IPP :

$$C(x) = \left[\frac{e^{2t}\cos(t)}{2}\right]_0^x + \frac{1}{2}\int_0^x e^{2t}\sin(t)dt = \frac{e^{2x}\cos(x) - 1}{2} + \frac{1}{2}S(x).$$

   De même, les fonctions $t \mapsto \frac{1}{2}e^{2t}$ et $t \mapsto \sin(t)$ sont de classe $\mathcal{C}^1$ sur $\mathbb{R}$ donc par IPP :

$$S(x) = \left[\frac{e^{2t}\sin(t)}{2}\right]_0^x - \frac{1}{2}\int_0^x e^{2t}\cos(t)dt = \frac{e^{2x}\sin(x)}{2} - \frac{1}{2}C(x).$$

(b) Par le théorème fondamental de l'analyse, C et S sont des primitives de $x \mapsto e^{2x}\cos(x)$ et $x \mapsto e^{2x}\sin(x)$ respectivement.
Or, d'après la question précédente, pour tout $x \in \mathbb{R}$

$$C(x) = \frac{e^{2x}\cos(x)-1}{2} + \frac{1}{2}S(x) = \frac{e^{2x}\cos(x)-1}{2} + \frac{1}{2}\left(\frac{e^{2x}\sin(x)}{2} - \frac{1}{2}C(x)\right)$$

d'où

$$C(x) = \frac{1}{5}(2e^{2x}\cos(x) - 2 + e^{2x}\sin(x)).$$

De même,

$$S(x) = \frac{e^{2x}\sin(x)}{2} - \frac{1}{2}C(x) = \frac{e^{2x}\sin(x)}{2} - \frac{1}{2}\left(\frac{e^{2x}\cos(x)-1}{2} + \frac{1}{2}S(x)\right)$$

d'où

$$S(x) = \frac{1}{5}\left(2e^{2x}\sin(x) - e^{2x}\cos(x) + 1\right).$$

3.
```python
def intC(x,nb_pas):
    s = 0
    for k in range(nb_pas):
        s += np.exp(2*k*x/nb_pas)*np.cos(k*x/nb_pas)
    return s*x/nb_pas
```

On note $E = \mathcal{C}^\infty(\mathbb{R},\mathbb{R})$ et on considère les éléments de E suivants :

$$f_1 : x \mapsto 1 \quad ; \quad f_2 : x \mapsto e^{2x} \quad ; \quad f_3 : x \mapsto e^{2x}\cos(x) \quad ; \quad f_4 : x \mapsto e^{2x}\sin(x).$$

On note F le sous-espace vectoriel de E engendré par $\mathcal{B} = (f_1, \dots, f_4)$.

4. Par définition de F, $\mathcal{B}$ est une famille génératrice de F. Il suffit donc de montrer qu'elle est libre.
Soit $(\lambda_1, \dots, \lambda_4) \in \mathbb{R}^4$ telle que $\sum_{i=1}^{4} \lambda_i f_1 = 0$, c'est-à-dire :

$$\forall x \in \mathbb{R}, \quad \lambda_1 + \lambda_2 e^{2x} + \lambda_3 e^{2x}\cos(x) + \lambda_4 e^{2x}\sin(x) = 0.$$

En faisant tendre x vers $+\infty$, il vient : $\lambda_1 = 0$.
En simplifiant par $e^{2x} \neq 0$, on a :

$$\forall x \in \mathbb{R}, \quad \lambda_2 + \lambda_3\cos(x) + \lambda_4\sin(x) = 0.$$

d'où en dérivant :

$$\forall x \in \mathbb{R}, \quad -\lambda_3\sin(x) + \lambda_4\cos(x) = 0.$$

En évaluant en $x = 0$ et $x = \frac{\pi}{2}$ on obtient $\lambda_3 = \lambda_4 = 0$.
Finalement, en réinjectant, on a aussi $\lambda_2 = 0$.
La famille est donc libre.

5. On note u l'application linéaire définie sur F par $u(f) = f'$ pour tout $f \in F$.
(a) C'est la linéarité de la dérivation.

(b) On a

$$u(f_1) = 0 \quad ; \quad u(f_2) = 2f_2 \quad ; \quad u(f_3) = 2f_3 - f_4 \quad ; \quad u(f_4) = 2f_4 + f_3.$$

(c) En particulier, $u(f_1), \dots, u(f_4) \in F$ donc $u(F) \subset F$ et u est bien un endomorphisme de F.

De plus, la question précédente montre que :

$$A = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 1 \\ 0 & 0 & -1 & 2 \end{pmatrix}.$$

6. Soit $X = \begin{pmatrix} x \\ y \\ z \\ t \end{pmatrix} \in \mathcal{M}_{4,1}(\mathbb{R})$.

$$AX = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \iff \left\{ \begin{array}{rcl} 2y & = & 0 \\ 2z + t & = & 1 \\ -z + 2t & = & 0 \end{array} \right. \iff \left\{ \begin{array}{rcl} y & = & 0 \\ z & = & 2t \\ t & = & \frac{1}{5} \end{array} \right. \iff \left\{ \begin{array}{rcl} y & = & 0 \\ t & = & \frac{1}{5} \\ z & = & \frac{2}{5} \end{array} \right.$$

Les solutions de l'équation $AX = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$ sont les vecteurs de la forme $\begin{pmatrix} x \\ 0 \\ \frac{2}{5} \\ \frac{1}{5} \end{pmatrix}$, $x \in \mathbb{R}$.

Cela signifie que les antécédents de f_3 par u, c'est-à-dire les primitives de f_3 sont les fonctions de la forme :

$$x f_1 + \frac{2}{5} f_3 + \frac{1}{5} f_4 \quad , x \in \mathbb{R}.$$

On retrouve (à la constante x près) le résultat de la question 2.(b).

7. On a

$$A^2 - 4A + 5I_4 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 3 & 4 \\ 0 & 0 & -4 & 3 \end{pmatrix} - \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 8 & 0 & 0 \\ 0 & 0 & 8 & 4 \\ 0 & 0 & -4 & 8 \end{pmatrix} + 5I_4 = \begin{pmatrix} 5 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

Soit $X = \begin{pmatrix} x \\ y \\ z \\ t \end{pmatrix} \in \mathcal{M}_{4,1}(\mathbb{R})$.

$$(A^2 - 4A + 5I_4)X = \begin{pmatrix} 2 \\ -1 \\ 0 \\ 0 \end{pmatrix} \iff 5x = 2 \quad \text{et} \quad y = -1.$$

Les solutions sont les vecteurs $\begin{pmatrix} 2/5 \\ -1 \\ z \\ t \end{pmatrix}$, $(z,t) \in \mathbb{R}^2$.

Les solutions d'inconnue $y \in F$ de :

$$u^2(y) - 4u(y) + 5y = f_1 - 2f_2$$

c'est-à-dire les solutions de (E) sont les fonctions :

$$\frac{2}{5}f_1 - f_2 + xf_3 + yf_4 \quad , (x,y) \in \mathbb{R}^2.$$

On retrouve le résultat de 1.(b).

Exercice sans préparation

1.
```
import numpy as np

def coder(L):
    coeff = []
    indice = []
    for k in range(len(L)):
        if L[k] != 0:
            coeff.append(L[k])
            indice.append(k)
    return [coeff,indice,len(L)]

def decoder(L):
    coeff,indice,taille = L[0],L[1],L[2]
    vect = np.zeros(taille)
    for k in range(len(indice)):
        vect[indice[k]]=coeff[k]
    return vect
```

2.
```
def prod_scal(V1,V2):
    s = 0
    for k in range(len(V1[1])):
        if V1[1][k]==V2[1][k]:
            s += V1[0][k]*V2[0][k]
    return s
```

Agro-Veto – Planche 5

Cours

Donner la définition de $\cap_{n=0}^{+\infty} A_n$ où pour tout $n \in \mathbb{N}$, A_n est un ensemble.

Exercice préparé

1. Soient f et g les fonctions définies sur $\mathbb{R}$ par :

$$\forall x \in \mathbb{R}, \quad f(x) = \begin{cases} x\ln(|x|) & \text{si } x \neq 0 \\ 0 & \text{si } x = 0 \end{cases} \quad ; \quad g(x) = xf(x).$$

   (a) Représenter en Python sur un même graphe les fonctions f et g sur l'intervalle $[-2, 2]$.
   (b) Montrer que f est continue sur $\mathbb{R}$.
   (c) Montrer que g est de classe $\mathcal{C}^1$ sur $\mathbb{R}$ et déterminer g'.
   (d) La fonction g est-elle de classe $\mathcal{C}^2$ sur $\mathbb{R}$?

2. Soit E l'espace vectoriel des fonctions continues de $\mathbb{R}$ dans $\mathbb{R}$.
   On note F le sous-espace vectoriel de E engendré par les fonctions $f_0 : x \mapsto 1$, $f_1 : x \mapsto x$ et la fonction f de la question 1 :

$$F = \text{Vect}(f_0, f_1, f).$$

   Montrer que (f_0, f_1, f) est une base de F.

3. Pour toute fonction φ de F, on note $\Phi(\varphi)$ la fonction dérivée de la fonction $x \mapsto x\varphi(x)$.
   (a) Montrer que $\Phi : \varphi \mapsto \Phi(\varphi)$ est linéaire.
   (b) Vérifier que $\Phi(f_0) = f_0$, $\Phi(f_1) = 2f_1$ et calculer $\Phi(f)$.
   (c) En déduire que Φ est un endomorphisme de F et préciser sa matrice dans la base (f_0, f_1, f).
   (d) L'endomorphisme Φ est-il bijectif de F vers F ? Si oui, préciser la matrice de Φ^{-1} dans la base (f_0, f_1, f).
   (e) L'endomorphisme Φ est-il diagonalisable ?

Agro-Veto – Planche 5

Exercice sans préparation

Les candidat·es pourront choisir de représenter les matrices sous la forme d'une liste de listes ou sous la forme d'un `array NumPy`.

1. Écrire une fonction en Python, nommée `somme_ligne(M,i)` renvoyant la somme des éléments de la ligne i de la matrice carrée M.

   Faire de même avec les colonnes.

2. On dit qu'une matrice M est une matrice magique si la somme de chaque ligne et la somme de chaque colonne de M sont toutes égales.

   Par exemple, la matrice $\begin{pmatrix} 11 & 15 & 19 \\ 21 & 7 & 17 \\ 13 & 23 & 9 \end{pmatrix}$ est magique alors que la matrice $\begin{pmatrix} 7 & 1 \\ 1 & 15 \end{pmatrix}$ ne l'est pas.

   Écrire une fonction `est_magique(M)` qui renvoie `True` si la matrice carrée M est magique et `False` sinon.

Éléments de correction–Planche 5

Exercice avec préparation

1. Soient f et g les fonctions définies sur $\mathbb{R}$ par :

$$\forall x \in \mathbb{R}, \quad f(x) = \begin{cases} x\ln(|x|) & \text{si } x \neq 0 \\ 0 & \text{si } x = 0 \end{cases} \quad ; \quad g(x) = xf(x).$$

(a)
```
import matplotlib.pyplot as plt
import numpy as np

def f(x):
        if x == 0:
                return 0
        else:
                return x*np.log(np.abs(x))

def g(x):
        return x*f(x)

abscisses = np.linspace(-2,2,100)
ordonnees_f=f(abscisses)
ordonnees_g=g(abscisses)
plt.plot(abscisses, ordonnees_f)
plt.plot(abscisses, ordonnees_g)
plt.show()
```

(b) La fonction f est continue sur $\mathbb{R}^*$ par opérations sur les fonctions continues.
Par croissances comparées, $\lim\limits_{x\to 0} x\ln(|x|) = 0$ donc f est continue en 0.

(c) La fonction f est de classe $\mathcal{C}^1$ sur $\mathbb{R}^*$ par opérations sur les fonctions de classe $\mathcal{C}^1$.
Pour tout $x \in \mathbb{R}^*$, on a :

$$g'(x) = 2x\ln(|x|) + x.$$

Étudions la dérivabilité en 0 : soit $x \neq 0$.

$$\frac{g(x) - g(0)}{x} = f(x) \xrightarrow[x\to 0]{} 0$$

donc g est dérivable en 0 et $g'(0) = 0$.
Étudions la continuité en 0 de g' : par croissances comparées

$$\lim_{x\to 0} 2x\ln(|x|) + x = 0 = g'(0).$$

Ainsi g est dérivable sur $\mathbb{R}$ de dérivée continue sur $\mathbb{R}$ donc est de classe $\mathcal{C}^1$ sur $\mathbb{R}$.

(d) On a, pour tout $x \neq 0$

$$\frac{g'(x) - g'(0)}{x} = 2\ln(|x|) + 1 \xrightarrow[x\to 0]{} -\infty.$$

Donc la fonction g' n'est pas dérivable en 0.
A fortiori, g n'est pas de classe $\mathcal{C}^2$ sur $\mathbb{R}$.

2. Comme (f_0, f_1, f) est génératrice de F par construction, il suffit de vérifier qu'elle est libre.
   Soit $(a, b, c) \in \mathbb{R}^3$ tel que $af_0 + bf_1 + cf = 0$.
   En évaluant en 0, on trouve $a = 0$.
   En évaluant en 1, on trouve $b = 0$.
   En évaluant en e, on trouve $c = 0$.
   Ainsi (f_0, f_1, f) est libre donc est une base de F.
3. Pour toute fonction φ de F, on note $\Phi(\varphi)$ la fonction dérivée de la fonction $x \mapsto x\varphi(x)$.
   (a) Soit $u, v \in F$ et $\lambda \in \mathbb{R}$.
   Pour tout $x \in \mathbb{R}$, on a :

$$\begin{aligned}\Phi(u + \lambda v)(x) = (x \mapsto x(u(x) + \lambda v(x))'(x) &= u(x) + \lambda v(x) + x(u'(x) + \lambda v'(x)) \\ &= u(x) + xu'(x) + \lambda(v(x) + xv'(x)) \\ &= \Phi(u)(x) + \lambda\Phi(v)(x).\end{aligned}$$

   Ainsi : $\Phi(u + \lambda v) = \Phi(u) + \lambda\Phi(v)$.
   Donc Φ est linéaire.
   (b) On a pour tout $x \in \mathbb{R}$:

$$\Phi(f_0)(x) = (x \mapsto x)' = 1 = f_0(x) \quad ; \quad \Phi(f_1)(x) = (x \mapsto x^2)' = 2x = 2f_1(x)$$

   donc

$$\Phi(f_0) = f_0 \quad ; \quad \Phi(f_1) = 2f_1.$$

   De même d'après la question 1 :

$$\Phi(f) = g' = 2f + f_1.$$

   (c) Comme $\Phi(f_0), \Phi(f_1)$ et $\Phi(f)$ sont dans F alors Φ est bien un endomorphisme de F.
   De plus, d'après la question précédente, sa matrice dans la base (f_0, f_1, f), notée A est :

$$A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 1 \\ 0 & 0 & 2 \end{pmatrix}.$$

   (d) La matrice A est triangulaire à coefficients diagonaux non nuls donc elle est inversible. Par conséquent Φ est bijective.
   La matrice de Φ^{-1} dans la base (f_0, f_1, f) est égale à A^{-1}.

$$A^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1/2 & -1/4 \\ 0 & 0 & 1/2 \end{pmatrix}.$$

   (e) L'endomorphisme Φ est diagonalisable si et seulement si A l'est. Comme A est triangulaire, ses valeurs propres sont ses coefficients diagonaux : $\mathrm{Spec}(A) = \{1, 2\}$.
   Déterminons les sous-espaces propres : soit $X = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \in \mathcal{M}_{3,1}(\mathbb{R})$.

—

$$AX = X \iff \left\{ \begin{array}{rcl} x & = & x \\ 2y+z & = & y \\ 2z & = & z \end{array} \right. \iff y = z = 0.$$

Ainsi $E_1(A) = \text{Vect}\left(\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}\right)$.

—

$$AX = 2X \iff \left\{ \begin{array}{rcl} x & = & 2x \\ 2y+z & = & 2y \\ 2z & = & 2z \end{array} \right. \iff x = z = 0.$$

Ainsi $E_1(A) = \text{Vect}\left(\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}\right)$.

Comme $\dim(E_1(A)) + \dim(E_2(A)) = 2 < 3$ alors A n'est pas diagonalisable.

Exercice sans préparation

1. ```
   def somme_ligne(M,i):
       n = np.shape(M)[0]
       s = 0
       for j in range(n):
           s += M[i,j]
       return s

   def somme_colonne(M,j):
       n = np.shape(M)[0]
       s = 0
       for i in range(n):
           s += M[i,j]
       return s
   ```

2. ```
   def est_magique(M):
       n = np.shape(M)[0]
       s = somme_ligne(M,0)
       for i in range(1,n):
           if somme_ligne(M,i) !=s:
               return False
       for j in range(n):
           if somme_colonne(M,j) !=s:
               return False
       return True
   ```

Agro-Veto – Planche 6

Cours

Expliquer comment résoudre l'équation $y' + ay = f$ où a, f sont continues sur un intervalle I.

Exercice préparé

Deux pièces A et B sont reliées entre elles mais seule B possède une issue vers l'extérieur. À l'instant $t = 0$, la pièce A contient une guêpe dont le comportement est le suivant :

- si la guêpe sort à l'extérieur, elle ne rentre plus jamais ;
- si à l'instant $n \in \mathbb{N}$ elle se trouve en A, elle y reste avec probabilité $2/5$ l'instant suivant ou passe en B avec probabilité $3/5$;
- si à l'instant $n \in \mathbb{N}$ elle se trouve en B, elle y reste avec probabilité $1/5$ l'instant suivant ou passe en A avec probabilité $2/5$ ou sort à l'extérieur avec probabilité $2/5$.

On note pour tout $n \in \mathbb{N}$:

- A_n : « l'instant n, la guêpe est dans la pièce A » et $a_n = \mathbb{P}(A_n)$
- B_n : « l'instant n, la guêpe est dans la pièce B » et $b_n = \mathbb{P}(B_n)$
- S_n : « l'instant n, la guêpe sort » et $s_n = \mathbb{P}(s_n)$.

1. (a) Calculer $a_0, b_0, s_0, a_1, b_1, s_1$.
   (b) Calculer s_2.
2. (a) Écrire une fonction Python `deplace(i)` d'argument un entier $i \in \{0, 1, 2\}$ représentant la position de la guêpe à un instant (0 : en A ; 1 : en B ; 2 : dehors) et qui renvoie la position de la guêpe l'instant suivant.
   (b) Écrire une fonction Python `sortie()` modélisant le comportement de la guêpe et renvoyant le temps de sortie.
   En déduire une estimation du temps moyen de sortie.
3. Soit $n \in \mathbb{N}$.
   (a) Exprimer a_{n+1} et b_{n+1} en fonction de a_n et b_n.
   (b) On pose $X_n = \begin{pmatrix} a_n \\ b_n \end{pmatrix}$. Trouver une matrice $M \in \mathcal{M}_2(\mathbb{R})$ telle que $X_n = M^n X_0$.
4. Soit $A = \begin{pmatrix} 2 & 2 \\ 3 & 1 \end{pmatrix}$.
   (a) Trouver une matrice inversible P et une matrice diagonale D telle que $A = PDP^{-1}$.
   (b) En déduire une expression de X_n en fonction de n.
5. On note T la variable aléatoire donnant le temps de sortie de la guêpe.
   (a) Justifier que pour tout $n \in \mathbb{N}$, $\mathbb{P}(T \leq n) = s_n$ et en déduire $P(T = n)$ pour tout $n \in \mathbb{N}$.
   (b) La variable T possède-t-elle une espérance ? Si oui, la calculer et comparer avec le résultat de 2.(b).

Agro-Veto – Planche 6

Exercice sans préparation

1. On considère des listes non vides ne contenant que une ou deux valeurs différentes.

   Par exemple : `['Marwa', 'Ambre', ' Marwa','Ambre','Ambre']`.

   Écrire, en langage Python, une fonction nommée `election` qui prend en entrée une liste `L` de cette forme et qui renvoie l'élément majoritaire. La fonction doit renvoyer `None` en cas d'égalité.

2. On considère maintenant des listes non vides mais pouvant contenir plus de deux valeurs différentes.

   La liste représente une urne et on veut savoir qui a obtenu le plus de voix.

   Écrire, en langage Python, une fonction nommée `election` qui prend en entrée une liste `L` de cette forme et qui renvoie la liste des personnes ayant obtenu le maximum de voix (la liste contient plusieurs personnes en cas d'égalité).

Éléments de correction–Planche 6

Exercice avec préparation

Deux pièces A et B sont reliées entre elles mais seule B possède une issue vers l'extérieur. À l'instant $t=0$, la pièce A contient une guêpe dont le comportement est le suivant :

- si la guêpe sort à l'extérieur, elle ne rentre plus jamais ;
- si à l'instant $n \in \mathbb{N}$ elle se trouve en A, elle y reste avec probabilité $2/5$ l'instant suivant ou passe en B avec probabilité $3/5$;
- si à l'instant $n \in \mathbb{N}$ elle se trouve en B, elle y reste avec probabilité $1/5$ l'instant suivant ou passe en A avec probabilité $2/5$ ou sort à l'extérieur avec probabilité $2/5$.

On note pour tout $n \in \mathbb{N}$:

- A_n : « l'instant n, la guêpe est dans la pièce A » et $a_n = \mathbb{P}(A_n)$
- B_n : « l'instant n, la guêpe est dans la pièce B » et $b_n = \mathbb{P}(B_n)$
- S_n : « l'instant n, la guêpe sort » et $s_n = \mathbb{P}(s_n)$.

1. (a) À l'instant initial, la guêpe est en A donc $a_0 = 1$ et $b_0 = s_0 = 0$.

   Puisqu'elle se situe initialement en A, alors $a_1 = \dfrac{2}{5}$, $b_1 = \dfrac{3}{5}$ et $s_1 = 0$.

   (b) Les événements A_1, B_1, S_1 forment un système complet d'événements. D'après la formule des probabilités totales :

$$\begin{aligned} s_2 &= \mathbb{P}_{A_1}(S_2)a_1 + \mathbb{P}_{B_1}(S_2)b_1 + \mathbb{P}_{S_1}(S_2)s_1 \\ &= \frac{2}{5}\mathbb{P}_{A_1}(S_2) + \frac{3}{5}\mathbb{P}_{B_1}(S_2) \quad (\text{car } s_1 = 0) \\ &= \frac{3}{5}\mathbb{P}_{B_1}(S_2) \quad \text{car } \mathbb{P}_{A_1}(S_2) = 0 \text{ (A n'a pas de sortie vers l'extérieur)} \\ &= \frac{3}{5} \times \frac{2}{5} \\ &= \frac{6}{25}. \end{aligned}$$

2. (a)
```
def deplace(i):
    x = rd.rand()
    if i == 0:
        if x<3/5:
            return 1
        else:
            return 0
    elif i == 1:
        if x<2/5:
            return 0
        elif 2/5<x<3/5:
            return 1
        else:
            return 2
    else:
        return 2
```

(b)
```
def sortie():
    i = 0
    n = 0
    while i != 2:
        i = deplace(i)
        n = n+1
    return n
```

Sous réserve que la variable aléatoire T donnant le temps de sortie de la guêpe possède une espérance alors, d'après la loi des grands nombres, la fonction suivante donne une estimation de $\mathbb{E}(T)$, c'est-à-dire du temps de sortie moyen :

```
def ET(N=10000):
    s = 0
    for k in range(N):
        s += sortie()
    return s/N
```

On trouve (pour $N = 10000$) un temps moyen d'environ 5.8.

3. Soit $n \in \mathbb{N}$.

(a) La famille (A_n, B_n, S_n) forme un système complet d'événements. D'après la formule des probabilités totales, on a donc :

$$a_{n+1} = \mathbb{P}_{A_n}(A_{n+1})a_n + \mathbb{P}_{B_n}(A_{n+1})b_n + \mathbb{P}_{S_n}(A_{n+1})s_n$$
$$b_{n+1} = \mathbb{P}_{A_n}(B_{n+1})a_n + \mathbb{P}_{B_n}(B_{n+1})b_n + \mathbb{P}_{S_n}(B_{n+1})s_n.$$

Sachant qu'une fois sortie, la guêpe ne rentre plus, on a :

$$\mathbb{P}_{S_n}(A_{n+1}) = \mathbb{P}_{S_n}(B_{n+1}) = 0.$$

Les autres données de l'énoncé donnent :

$$\mathbb{P}_{A_n}(A_{n+1}) = \frac{2}{5} = \mathbb{P}_{B_n}(A_{n+1}) \quad ; \quad \mathbb{P}_{A_n}(B_{n+1}) = \frac{3}{5} \quad ; \quad \mathbb{P}_{B_n}(B_{n+1}) = \frac{1}{5}.$$

D'où :

$$a_{n+1} = \frac{2}{5}(a_n + b_n) \quad ; \quad b_{n+1} = \frac{3}{5}a_n + \frac{1}{5}b_n.$$

(b) D'après la question précédente, pour tout $n \in \mathbb{N}$, on a :

$$X_{n+1} = MX_n \quad \text{où} \quad M = \frac{1}{5}\begin{pmatrix} 2 & 2 \\ 3 & 1 \end{pmatrix}.$$

Une récurrence immédiate donne alors :

$$\forall n \in \mathbb{N}, \quad X_n = M^n X_0.$$

4. Soit $A = \begin{pmatrix} 2 & 2 \\ 3 & 1 \end{pmatrix}$.

(a) On commence par chercher les valeurs propres de A.
Soit $\lambda \in \mathbb{R}$.

$$\begin{aligned}\lambda \in \mathrm{Spec}(A) \iff \det(A - \lambda I_2) = 0 &\iff \det\left(\begin{pmatrix} 2-\lambda & 2 \\ 3 & 1-\lambda \end{pmatrix}\right) = 0 \\ &\iff (2-\lambda)(1-\lambda) - 6 = 0 \\ &\iff \lambda = 4 \quad ; \quad \lambda = -1.\end{aligned}$$

Comme A est d'ordre 2 avec deux valeurs propres distinctes, on sait qu'elle est diagonalisable.
On cherche maintenant une base de chaque sous-espace propre. Soit $X = \begin{pmatrix} x \\ y \end{pmatrix} \in \mathcal{M}_{2,1}(\mathbb{R})$.

— Sous-espace propre associé à la valeur propre 4.

$$AX = 4X \iff \left\{ \begin{array}{rcl} 2x + 2y & = & 4x \\ 3x + y & = & 4y \end{array} \right. \iff x = y.$$

Ainsi $E_4(A) = \mathrm{Vect}\left(\begin{pmatrix} 1 \\ 1 \end{pmatrix}\right)$.

— Sous-espace propre associé à la valeur propre -1.

$$AX = -X \iff \left\{ \begin{array}{rcl} 2x + 2y & = & -x \\ 3x + y & = & -y \end{array} \right. \iff 3x = 2y.$$

Ainsi $E_{-1}(A) = \mathrm{Vect}\left(\begin{pmatrix} -2 \\ 3 \end{pmatrix}\right)$.

Ainsi les matrices $P = \begin{pmatrix} 1 & -2 \\ 1 & 3 \end{pmatrix}$ et $D = \begin{pmatrix} 4 & 0 \\ 0 & -1 \end{pmatrix}$ conviennent.

(b) Soit $n \in \mathbb{N}$. Comme $M = \frac{1}{5}A$, on a : $X_n = \frac{1}{5^n} A^n X_0$.
Une récurrence classique montre que :

$$\forall n \in \mathbb{N}, \quad A^n = PD^nP^{-1} = P\begin{pmatrix} 4^n & 0 \\ 0 & (-1)^n \end{pmatrix} P^{-1}.$$

Ainsi :

$$X_n = \frac{1}{5^n} P \begin{pmatrix} 4^n & 0 \\ 0 & (-1)^n \end{pmatrix} P^{-1} X_0$$

où $P^{-1} = \frac{1}{5}\begin{pmatrix} 3 & 2 \\ -1 & 1 \end{pmatrix}$ et $X_0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ (voir question 1).
Le calcul matriciel donne alors :

$$X_n = \frac{1}{5}\begin{pmatrix} 3(4/5)^n + 2(-1/5)^n \\ 3(4/5)^n - 3(-1/5)^n \end{pmatrix}.$$

5. On note T la variable aléatoire donnant le temps de sortie de la guêpe.

(a) Soit $n \in \mathbb{N}$. Comme une fois sortie, la guêpe ne rentre plus, S_n est réalisé si et seulement si la guêpe est sortie à l'instant n ou avant l'instant n. Ainsi :

$$S_n = [T \leq n]$$

D'où $\mathbb{P}(T \leq n) = s_n$.
Soit $n \in \mathbb{N}^*$. On a :

$$P(T = n) = P(T \leq n) - \mathbb{P}(T \leq n-1) = s_n - s_{n-1}.$$

Or (A_n, B_n, S_n) étant un système complet d'événements, on a : $s_n + a_n + b_n = 1$ et de même $s_{n-1} + a_{n-1} + b_{n-1} = 1$, donc :

$$P(T = n) = P(T \leq n) - \mathbb{P}(T \leq n-1) = a_{n-1} + b_{n-1} - a_n - b_n.$$

Enfin avec la question précédente, on obtient :

$$\begin{aligned}\mathbb{P}(T = n) &= \frac{1}{5}\left(3\left(\frac{4}{5}\right)^{n-1} + 2\left(-\frac{1}{5}\right)^{n-1} + 3\left(\frac{4}{5}\right)^{n-1} - 3\left(-\frac{1}{5}\right)^{n-1} - 3(4/5)^n - 2(-1/5)^n - 3\left(\frac{4}{5}\right.\right. \\ &= \frac{6}{25}\left(\left(\frac{4}{5}\right)^{n-1} - \left(-\frac{1}{5}\right)^{n-1}\right).\end{aligned}$$

et $P(T = 0) = 0$.

(b) La variable T possède une espérance si et seulement si $\sum_{n\geq 1} n\mathbb{P}(T = n)$ est absolument convergente.
Or par la question précédente et l'inégalité triangulaire :

$$\forall n \geq 1, \quad |n\mathbb{P}(T = n)| \leq \frac{6}{25}\left(n\left(\frac{4}{5}\right)^{n-1} + n\left(\frac{1}{5}\right)^{n-1}\right).$$

Le membre de droite est le terme général d'une série convergente (combinaison linéaire de séries géométriques dérivées premières de raison dans $]-1, 1[$) donc par comparaison pour les séries à termes positifs, $\sum_{n\geq 1} |n\mathbb{P}(T = n)|$ est convergente.
Ainsi, T possède une espérance et :

$$\begin{aligned}\mathbb{E}(T) = \sum_{n=1}^{+\infty} n\mathbb{P}(T = n) &= \frac{6}{25}\left(\frac{1}{\left(1 - \frac{4}{5}\right)^2} + \frac{1}{\left(1 + \frac{1}{5}\right)^2}\right) \\ &= 6 - \frac{1}{6} \\ &\simeq 5.833\end{aligned}$$

Le résultat est cohérent avec l'approximation de 2.(b).

Exercice sans préparation

1.
```
def election(L):
    D = {}
    for elt in L:
        if elt in D.keys():
            D[elt] += 1
        else:
            D[elt]=1
    maxi = max([D[key] for key in D.keys()])
    L = []
    for key in D.keys():
        if D[key] == maxi:
            L.append(key)
    if len(L)>1:
                return 'None'

    return L[0]
```

2.
```
def election(L):
    D = {}
    for elt in L:
        if elt in D.keys():
            D[elt] += 1
        else:
            D[elt]=1
    maxi = max([D[key] for key in D.keys()])
    L = []
    for key in D.keys():
        if D[key] == maxi:
            L.append(key)
    return L
```

Agro-Veto – Planche 7

Cours

Définition de deux suites adjacentes.

Exercice préparé

Pour tout $y \in \mathbb{R}$, on note $A(y) = \begin{pmatrix} 0 & y & 1 \\ y & 1 & 0 \\ y & 0 & 1 \end{pmatrix}$.

1. (a) Pour $y = 1$ montrer que $A(y)$ est diagonalisable.
   (b) Montrer que pour tout $y \in \mathbb{R}$, $\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$ est un vecteur propre de $A(y)$ et donner la valeur propre associée.
   (c) Déterminer le spectre de $A(y)$ en fonction de y.
   (d) Déterminer les valeurs de y pour lesquelles $A(y)$ est diagonalisable.
2. Soit Y une variable aléatoire discrète telle que $Y(\Omega) = \mathbb{Z}$ et vérifiant :
   — $|Y|$ suit la loi de Poisson de paramètre $\lambda > 0$.
   — $\forall k \in \mathbb{Z}$, $\mathbb{P}(Y = k) = \mathbb{P}(Y = -k)$.

   (a) Rappeler, pour tout $k \in \mathbb{N}$, la valeur de $\mathbb{P}(|Y| = k)$.
   (b) Déterminer $\mathbb{P}(Y = 0)$ et $\mathbb{P}(Y = k)$ pour tout $k \in \mathbb{Z}^*$.
   (c) Écrire une fonction Python `simuleY(lambdaa)` qui prend en argument un réel `lambdaa` représentant le paramètre λ et qui simule la variable aléatoire Y.
3. On considère désormais la matrice aléatoire $A(Y)$.
   (a) Déterminer avec Python une valeur approchée de $\mathbb{P}(\text{rg}(A(Y)) = 2)$. On pourra utiliser la commande `numpy.linalg.matrix_rank`.
   (b) Déterminer la probabilité que $A(Y)$ soit diagonalisable.
   (c) Déterminer la probabilité que $A(Y)$ soit inversible.
   (d) Déterminer la probabilité que $\text{rg}(A(Y)) = 2$.

Agro-Veto – Planche 7

Exercice sans préparation

On considère une élection avec $n \geq 2$ candidat·es numéroté·es de 0 à $n-1$.
Un bulletin est représenté par une liste classant les candidats par ordre de préférence. Par exemple, le bulletin $[1, 0, 3, 4, 2]$ signifie que le votant préfère le candidat·e 1 puis le 0 ...
Le suffrage est représenté par une liste dont les éléments sont les bulletins.

1. Écrire, en langage Python, une fonction nommée `pref` qui prend en entrée un bulletin `B` et deux candidat·es `a`, `b` et qui renvoie le préféré des deux dans le bulletin `B`.
2. On dit qu'un·e candidat·e `a` bat le candidat·e `b` si plus de la moitié (strictement) des bulletins préfèrent le candidat· `a` au candidat· `b`.

   Écrire, en langage Python, une fonction nommée `duel` qui prend en entrées le suffrage `L` et deux candidat·es `a`, `b` et qui renvoie `a` si `a` bat `b` ; `bb` bat `a` et `None` s'ils sont ex-æquo.

Éléments de correction–Planche 7

Exercice avec préparation

Pour tout $y \in \mathbb{R}$, on note $A(y) = \begin{pmatrix} 0 & y & 1 \\ y & 1 & 0 \\ y & 0 & 1 \end{pmatrix}$.

1. (a) On a $A(1) = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}$ est symétrique à coefficients réels donc d'après le théorème spectral, elle est diagonalisable.

   (b) Soit $y \in \mathbb{R}$.
   $$A(y)\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = (1+y)\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}.$$
   Donc $\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$ est un vecteur propre de $A(y)$ associer à la valeur propre $1+y$.

   (c) Soit $\lambda \in \mathbb{R}$.

$$\begin{aligned}
\operatorname{rg}(A(y) - \lambda I_3) &= \operatorname{rg}\left(\begin{pmatrix} -\lambda & y & 1 \\ y & 1-\lambda & 0 \\ y & 0 & 1-\lambda \end{pmatrix}\right) \\
&= \operatorname{rg}\left(\begin{pmatrix} 1+y-\lambda & y & 1 \\ 1+y-\lambda & 1-\lambda & 0 \\ 1+y-\lambda & 0 & 1-\lambda \end{pmatrix}\right) \quad (C_1 \to C_1 + C_2 + C_3) \\
&= \operatorname{rg}\left(\begin{pmatrix} 1+y-\lambda & y & 1 \\ 0 & 1-y-\lambda & -1 \\ 0 & -y & -\lambda \end{pmatrix}\right) \quad (L_i \to L_i - L_1,\ i = 2,3) \\
&= \operatorname{rg}\left(\begin{pmatrix} 1+y-\lambda & 1 & y \\ 0 & -1 & 1-y-\lambda \\ 0 & -\lambda & -y \end{pmatrix}\right) \quad (C_2 \leftrightarrow C_3) \\
&= \operatorname{rg}\left(\begin{pmatrix} 1+y-\lambda & 1 & y \\ 0 & -1 & 1-y-\lambda \\ 0 & 0 & \lambda^2-(1-y)\lambda-y \end{pmatrix}\right) \quad (L_3 \leftarrow L_3 - \lambda L_2).
\end{aligned}$$

   Ainsi :
   $$\lambda \in \operatorname{Spec}(A(y)) \iff 1+y-\lambda = 0 \quad \text{ou} \quad \lambda^2-(1-y)\lambda - y = 0.$$
   Le discriminant de $\lambda^2-(1-y)\lambda-y=0$ est $\Delta = (1-y)^2+4y = (1+y)^2$ donc l'équation a deux solutions 1 et $-y$.
   Donc finalement : $\operatorname{Spec}(A(y)) = \{1+y, -y, 1\}$.

   (d) — Si $y \notin \{0, -1, -\frac{1}{2}\}$ alors les valeurs propres $1+y$, $-y$ et 1 sont deux à deux distinctes : $A(y)$ possède trois valeurs propres distinctes donc est diagonalisable.

— Si $y = 0$, le spectre est $\{0, 1\}$ et en reprenant $1.(c)$:

$$\mathrm{rg}(A(0) - I_3) = \mathrm{rg}\left(\begin{pmatrix} 0 & 1 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{pmatrix}\right) = 1$$

donc par le théorème du rang :

$$\dim(E_1(A(0)) = 3 - 1 = 2.$$

Ainsi $\dim(E_1(A(0)) + \dim(E_0(A(0)) = 3$ et $A(0)$ est diagonalisable.

— Si $y = -1$ alors le spectre est $\{1, 0\}$ et en reprenant $1.(c)$:

$$\mathrm{rg}(A(-1) - \lambda I_3) = \mathrm{rg}\left(\begin{pmatrix} -\lambda & 1 & -1 \\ 0 & -1 & 2 - \lambda \\ 0 & 0 & \lambda^2 - 2\lambda + 1 \end{pmatrix}\right).$$

Donc

$$\mathrm{rg}(A(-1) - I_3) = \mathrm{rg}\left(\begin{pmatrix} -1 & 1 & -1 \\ 0 & -1 & 1 \\ 0 & 0 & 0 \end{pmatrix}\right) = 2$$

et par le théorème du rang $\dim(E_1(A(-1)) = 1$.

De même,

$$\mathrm{rg}(A(-1)) = \mathrm{rg}\left(\begin{pmatrix} 0 & 1 & -1 \\ 0 & -1 & 2 \\ 0 & 0 & 1 \end{pmatrix}\right) = 2$$

et par le théorème du rang $\dim(E_1(A(-1)) = 1$.

La somme des dimensions des sous-espaces propres n'est pas égale à 3 donc $A(-1)$ n'est pas diagonalisable.

— Si $y = -\dfrac{1}{2}$ le spectre est $\{1, 1/2\}$ et :

$$\mathrm{rg}(A(y) - \lambda I_3) = \mathrm{rg}\left(\begin{pmatrix} 1/2 - \lambda & 1 & -1/2 \\ 0 & -1 & 3/2 - \lambda \\ 0 & 0 & \lambda^2 - 3/2\lambda + 1/2 \end{pmatrix}\right).$$

D'où :

— $\mathrm{rg}(A(y) - I_3) = \mathrm{rg}\left(\begin{pmatrix} -1/2 & 1 & -1/2 \\ 0 & -1 & 1/2 \\ 0 & 0 & 0 \end{pmatrix}\right) = 2$ et par le théorème du rang $\dim(E_1(A(y)) = 1$;

— $\mathrm{rg}(A(y) - 1/2I_3) = \mathrm{rg}\left(\begin{pmatrix} 0 & 1 & -1/2 \\ 0 & -1 & 1 \\ 0 & 0 & 0 \end{pmatrix}\right) = 2$ et par le théorème du rang $\dim(E_{1/2}(A(y)) = 1$.

La somme des dimensions des sous-espaces propres n'est pas égale à 3 donc $A(-1/2)$ n'est pas diagonalisable.

Finalement $A(y)$ est diagonalisable si et seulement si $y \notin \{-1, -1/2\}$.

2. Soit Y une variable aléatoire discrète telle que $Y(\Omega) = \mathbb{Z}$ et vérifiant :

— $|Y|$ suit la loi de Poisson de paramètre $\lambda > 0$.
— $\forall k \in \mathbb{Z}$, $\mathbb{P}(Y = k) = \mathbb{P}(Y = -k)$.

(a) Pour tout $k \in \mathbb{N}$, la valeur de $\mathbb{P}(|Y| = k) = e^{-\lambda}\dfrac{\lambda^k}{k!}$.

(b) On a :

$$\mathbb{P}(Y = 0) = \mathbb{P}(|Y| = 0) = e^{-\lambda}.$$

Soit $k \in \mathbb{Z}^*$.

$$\mathbb{P}(|Y| = |k|) = \mathbb{P}(Y = k) + \mathbb{P}(Y = -k) = 2\mathbb{P}(Y = k).$$

Donc

$$\mathbb{P}(Y = k) = \frac{1}{2}\mathbb{P}(|Y| = |k|) = e^{-\lambda}\frac{\lambda^{|k|}}{2(|k|)!}.$$

(c)
```
def simuleY(lambdaa):
        x = rd.rand()
        if x<1/2:
                return rd.poisson(lambdaa)
        else:
                return -rd.poisson(lambdaa)
```

3. On considère désormais la matrice aléatoire $A(Y)$.

(a) D'après la loi des grands nombres :

```
def A(y):
        return np.array([[0,y,1],[y,1,0],[y,0,1]])

def rang(lambdaa, N =10000):
        s = 0
        for k in range(N):
                if np.linalg.matrix_rank(A(simuleY(lambdaa))) ==2 :
                        s += 1
        return s/N
```

(b) D'après 1.(d) et comme Y est à valeur dans $\mathbb{Z}$:

$$\mathbb{P}(A(Y) \text{ diagonalisable}) = 1 - \mathbb{P}(Y = -1) = 1 - \frac{\lambda}{2}e^{-\lambda}.$$

(c) On sait que $A(Y)$ est inversible si et seulement si son spectre ne contient pas 0. D'après 1.(c), on a donc :

$$\mathbb{P}(A(Y) \text{ inversible}) = 1 - \mathbb{P}(Y = -1) - \mathbb{P}(Y = 0) = 1 - \left(\frac{\lambda}{2} + 1\right)e^{-\lambda}.$$

(d) D'après 1.(c) :

$$\operatorname{rg}(A(Y)) = \operatorname{rg}\left(\begin{pmatrix} 1+Y & 1 & Y \\ 0 & -1 & 1-Y \\ 0 & 0 & -Y \end{pmatrix}\right)$$

Donc :

$$\mathbb{P}(\operatorname{rg}(A(Y)) = 2) = \mathbb{P}(Y = -1) + \mathbb{P}(Y = 0) = \left(\frac{\lambda}{2} + 1\right)e^{-\lambda}.$$

Exercice sans préparation

1. ```
   def pref(B,a,b):
       rang_a = -1
       rang_b = -1
       for k in range(len(B)):
           if B[k] == a:
               rang_a = k
           if B[k] == b:
               rang_b=k
       if rang_a<rang_b:
           return a
       else:
           return b
   ```

2. ```
   def duel(L,a,b):
       res = {a : 0, b:0}
       for bulletin in L:
           res[pref(bulletin,a,b)]+=1
       if res[a]>len(L)/2:
           return a
       elif res[b]>len(L)/2:
           return b
       else:
       return 'none'
   ```

Agro-Veto – Planche 8

Cours

Donner la loi de la somme de deux variables aléatoires indépendantes de lois normales.

Exercice préparé

1. Soit $(u_n)_{n\in\mathbb{N}^*}$ une suite de réels positifs, décroissante et convergeant vers 0. Pour tout $n \in \mathbb{N}^*$, on pose :
$$S_n = \sum_{k=1}^{n} (-1)^k u_k.$$
   (a) Montrer que $(S_{2n})_{n\in\mathbb{N}^*}$ et $(S_{2n+1})_{n\in\mathbb{N}}$ sont adjacentes.
   (b) En déduire que $(S_n)_{n\in\mathbb{N}^*}$ est convergente.
   (c) Montrer que $\sum_{n\geq 1} \frac{(-1)^n}{n}$ est convergente.. Est-elle absolument convergente ?
   (d) Écrire une fonction Python prenant en entrée un entier `N` et traçant $\left(\sum_{k=1}^{n} \frac{(-1)^k}{k}\right)_{n\in[\![1,N]\!]}$.
2. (a) Montrer que pour tout $n \in \mathbb{N}$, $\int_0^1 \frac{x^n}{1+x}dx \leq \frac{1}{n+1}$.
   (b) En déduire $\lim\limits_{n\to+\infty} \int_0^1 \frac{x^n}{1+x}dx$.
   (c) Montrer que $\lim\limits_{n\to+\infty} \sum_{k=0}^{n} \frac{(-1)^k}{k+1} = \int_0^1 \frac{1}{1+x}dx$ puis calculer cette intégrale.
   (d) En déduire la valeur de $\sum_{n=1}^{+\infty} \frac{(-1)^n}{n}$.
   (e) Déterminer de même $\sum_{n=1}^{+\infty} \frac{(-x)^n}{n}$ pour tout $x \in [0,1]$.
3. Pour tout $n \in \mathbb{N}^*$ on pose $u_n = \frac{1}{n+(-1)^n}$ et $v_n = (-1)^n u_n$.
   (a) Soit $n \in \mathbb{N}^*$. Montrer que $u_{2n+1} - u_{2n}$ et $u_{2n+2} - u_{2n+1}$ sont de signes opposés. Peut-on appliquer le résultat de la question 1 ?
   (b) Montrer que $v_n = \frac{(-1)^n}{n} - \frac{1}{n^2} + \underset{n\to+\infty}{o}\left(\frac{1}{n^2}\right)$.
   (c) En déduire que $\sum_{n\geq 1} v_n$ converge.

Agro-Veto – Planche 8

Exercice sans préparation

1. Écrire, en langage Python, une fonction qui prend en entrée un entier $n \geq 2$ et qui :
   — crée la liste $L = [0, 1, \ldots, n-1]$;
   — pour tout $i \in [\![0, n-2]\!]$ tire au hasard un $j \in [\![i, n-1]\!]$ et échange $L[i]$ et $L[j]$.
2. Écrire, en langage Python, une fonction nommée `melange` qui prend en entrée une chaîne de caractères et qui mélange les caractères de façon aléatoire.

Éléments de correction–Planche 8

Exercice avec préparation

1. (a) Soit $n \in \mathbb{N}^*$.
   — Par décroissance de (u_n) :
   $$S_{2(n+1)} - S_{2n} = u_{2n+2} - u_{2n+1} \leq 0 \quad \text{et} \quad S_{2n+1} - S_{2(n-1)+1} = u_{2n+1} - u_{2n} \geq 0.$$
   Donc (S_{2n}) est décroissante et (S_{2n+1}) est croissante.
   — Comme (u_n) converge vers 0 alors :
   $$S_{2n+1} - S_{2n} = -u_{2n+1} \xrightarrow[n\to+\infty]{} 0.$$
   Les suites sont donc adjacentes.

   (b) En particulier, elles convergent vers une même limite. Les suites extraites $(S_{2n})_{n\in\mathbb{N}^*}$ et $(S_{2n+1})_{n\in\mathbb{N}}$ convergent vers une même limite donc $(S_n)_{n\in\mathbb{N}^*}$ converge.

   (c) La suite $\left(\frac{1}{n}\right)_n$ est à termes positifs, décroissante et de limite 0 donc d'après ce qui précède, la suite des sommes partielles de la série $\sum_{n\geq 1} \frac{(-1)^n}{n}$ est convergente.
   Ainsi $\sum_{n\geq 1} \frac{(-1)^n}{n}$ est convergente. Elle n'est pas absolument convergente car la série harmonique diverge.

   (d)
```
def serie(N):
    termes = []
    s = 0
    for k in range(1,N+1):
        s+=(-1)**k/k
        termes.append(s)
    plt.plot(range(1,N+1),termes)
    plt.show()
```

2. (a) Soit $n \in \mathbb{N}$. Pour tout $x \in [0,1]$, $\frac{x^n}{1+x} \leq x^n$ donc par croissance de l'intégrale :
   $$\int_0^1 \frac{x^n}{1+x}dx \leq \int_0^1 x^n dx = \frac{1}{n+1}.$$

   (b) Comme $x \mapsto \frac{x^n}{1+x}$ est positive sur $[0,1]$ alors par positivité de l'intégrale et avec la question précédente :
   $$\forall n \in \mathbb{N}, \quad 0 \leq \int_0^1 \frac{x^n}{1+x}dx \leq \frac{1}{n+1}.$$
   Le théorème des gendarmes permet de conclure que $\lim\limits_{n\to+\infty} \int_0^1 \frac{x^n}{1+x}dx = 0$.

(c) Pour tout $x \in [0,1]$ et tout $n \in \mathbb{N}$, on a :

$$\frac{1-(-x)^{n+1}}{1+x} = \sum_{k=0}^{n}(-x)^k \quad ie \quad \frac{1}{1+x} = \frac{(-x)^{n+1}}{1+x} + \sum_{k=0}^{n}(-x)^k.$$

D'où, par linéarité de l'intégrale :

$$\begin{aligned}\int_0^1 \frac{1}{1+x}dx &= \int_0^1 \frac{(-x)^{n+1}}{1+x}dx + \sum_{k=0}^{n}\int_0^1 (-x)^k dx \\ &= (-1)^{n+1}\int_0^1 \frac{x^{n+1}}{1+x}dx + \sum_{k=0}^{n}\frac{(-1)^k}{k+1}.\end{aligned}$$

Or d'après la question précédente $\lim\limits_{n\to+\infty}\int_0^1 \frac{x^{n+1}}{1+x}dx = 0$ donc :

$$\lim_{n\to+\infty}\sum_{k=0}^{n}\frac{(-1)^k}{k+1} = \int_0^1 \frac{1}{1+x}dx = \ln(2).$$

(d) Soit $n \in \mathbb{N}^*$. En effectuant le changement d'indice $k = i-1$ on a :

$$\sum_{i=1}^{n}\frac{(-1)^i}{i} = \sum_{k=0}^{n-1}\frac{(-1)^{k+1}}{k+1} = -\sum_{k=0}^{n-1}\frac{(-1)^k}{k+1}.$$

On déduit de la question précédente que la suite des sommes partielles de $\sum\limits_{n\geq 1}\frac{(-1)^n}{n}$ converge vers $-\ln(2)$.

Ainsi $\sum\limits_{n=1}^{+\infty}\frac{(-1)^n}{n} = -\ln(2)$.

(e) Soit $x \in [0,1]$. En utilisant la relation de 2.(c) puis en intégrant entre 0 et x on a :

$$\begin{aligned}\int_0^x \frac{1}{1+t}dt &= \int_0^x \frac{(-t)^{n+1}}{1+t}dt + \sum_{k=0}^{n}\int_0^x (-t)^k dt \\ &= (-1)^{n+1}\int_0^x \frac{t^{n+1}}{1+t}dt + \sum_{k=0}^{n}\frac{(-x)^k}{k+1}.\end{aligned}$$

Or l'intégrande étant positive sur $[0,1]$ on a :

$$0 \leq \int_0^x \frac{t^{n+1}}{1+t}dt \leq \int_0^1 \frac{t^{n+1}}{1+t}dt \leq \frac{1}{n+1}$$

ce qui permet de conclure comme en 2.(c) que

$$\sum_{k=0}^{+\infty}\frac{(-x)^k}{k+1} = \int_0^x \frac{1}{1+t}dt = \ln(1+x).$$

En effectuant le même changement d'indice qu'en 2.(d) on en déduit alors :

$$\sum_{n=1}^{+\infty}\frac{(-x)^n}{n} = -\ln(1+x).$$

3. Pour tout $n \in \mathbb{N}^*$ on pose $u_n = \dfrac{1}{n+(-1)^n}$ et $v_n = (-1)^n u_n$.

(a) Soit $n \in \mathbb{N}^*$.

$$(u_{2n+1} - u_{2n})(u_{2n+2} - u_{2n+1}) = \left(\frac{1}{2n} - \frac{1}{2n+1}\right)\left(\frac{1}{2n+3} - \frac{1}{2n+2}\right) < 0.$$

Donc $u_{2n+1} - u_{2n}$ et $u_{2n+2} - u_{2n+1}$ sont de signes opposés.
La suite (u_n) n'est donc pas décroissante et on ne peut pas appliquer le résultat de la question 1.

(b) Soit $n \in \mathbb{N}^*$.

$$v_n = \frac{(-1)^n}{n+(-1)^n} = \frac{(-1)^n}{n} \times \frac{1}{1+\frac{(-1)^n}{n}}.$$

Comme $\lim\limits_{n\to+\infty} \dfrac{(-1)^n}{n} = 0$ on peut composer avec le DL à l'ordre 1 en 0 de $x \mapsto \dfrac{1}{1+x}$:

$$\begin{aligned} v_n = \frac{(-1)^n}{n} \times \frac{1}{1+\frac{(-1)^n}{n}} &= \frac{(-1)^n}{n}\left(1 - \frac{(-1)^n}{n} + \underset{n\to+\infty}{o}\left(\frac{1}{n}\right)\right) \\ &= \frac{(-1)^n}{n} - \frac{1}{n^2} + \underset{n\to+\infty}{o}\left(\frac{1}{n^2}\right). \end{aligned}$$

(c) On en déduit que :

$$v_n - \frac{(-1)^n}{n} \underset{n\to+\infty}{\sim} -\frac{1}{n^2} \quad \text{puis} \quad \left|v_n - \frac{(-1)^n}{n}\right| \underset{n\to+\infty}{\sim} \frac{1}{n^2}.$$

Ainsi la série $\displaystyle\sum_{n\geq 1}\left(v_n - \frac{(-1)^n}{n}\right)$ est absolument convergente donc convergente.

Enfin :

$$\sum_{n\geq 1} v_n = \sum_{n\geq 1}\left(v_n - \frac{(-1)^n}{n}\right) + \sum_{n\geq 1}\frac{(-1)^n}{n}$$

est combinaison linéaire de séries convergentes donc est convergente.

Exercice sans préparation

1.
```
def melange(n):
    L =[k for k in range(n)]
    for i in range(n-2):
        j = rd.randint(i+1,n)
        L[i],L[j]=L[j],L[i]
    return L
```

2.
```
def melange2(chaine):
    n = len(chaine)
    L = melange(n)
    copie = ''
    for k in range(n):
        copie+=chaine[L[k ]]
    return copie
```

Agro-Veto – Planche 9

Cours

Donner la définition des fonctions partielles d'une fonction $f : \mathbb{R}^2 \to \mathbb{R}$.

Exercice préparé

Soit $N \geq 2$ et $p \in]0,1[$.
On dispose d'une urne initialement remplie avec N boules et d'une pièce donnant « Pile » avec probabilité p.
On répète l'expérience suivante :

— on lance de façon indépendante la pièce autant de fois qu'il y a de boules dans l'urne ;

— on retire de l'urne autant de boules que de « Faces » obtenues.

Pour tout $n \in \mathbb{N}$, on note X_n la variable aléatoire donnant le nombre de boules dans l'urne après n répétitions de l'expérience. En particulier, $X_0 = N$.

1. Donner la loi de X_1.
2. Écrire une fonction Python qui prend en argument p, n, N et qui simule X_n.
3. Justifier que pour tout $n \in \mathbb{N}^*$, $X_n(\Omega) \subset [\![0, N]\!]$.
   La variable X_n possède-elle une espérance ?
4. Écrire une fonction Python qui prend en argument p, n, N et qui donne une valeur approchée de $\mathbb{E}(X_n)$.
5. Soit $n \in \mathbb{N}$.
   (a) Pour tout $(j,k) \in [\![0, N]\!]^2$ calculer $\mathbb{P}_{[X_n=j]}(X_{n+1} = k)$.
   (b) En déduire que pour tout $k \in [\![0, N]\!]$:
   $$\mathbb{P}(X_{n+1} = k) = \sum_{j=k}^{N} \binom{j}{k} p^k (1-p)^{j-k} \mathbb{P}(X_n = j).$$
   (c) En déduire que $\mathbb{E}(X_{n+1}) = p\mathbb{E}(X_n)$ puis donner une expression de $\mathbb{E}(X_n)$ en fonction de p, n et N.
6. Soit $n \in \mathbb{N}$. On souhaite retrouver le résultat de 5.(c) par une autre méthode. Pour tout $t \in \mathbb{R}$, on pose $G_n(t) = \sum_{k=0}^{N} \mathbb{P}(X_n = k) t^k$.
   (a) Calculer G_1.
   (b) Montrer que $\mathbb{E}(X_n) = G_n'(1)$.
   (c) Montrer que pour tout $t \in \mathbb{R}$, $G_{n+1}(t) = G_n(pt + 1 - p)$.
   (d) Retrouver le résultat de 5.(c).

Agro-Veto – Planche 9

Exercice sans préparation

On considère une matrice carrée comme la liste de ses lignes.

Par exemple, $M = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$ est représentée en Python par $[[1,2,3],[4,5,6],[7,8,9]]$.

1. Écrire, en langage Python, une matrice carrée M et qui renvoie `True` si elle est symétrique et `False` sinon.
2. Écrire, en langage Python, une fonction qui renvoie la transposée d'une matrice carrée.

Éléments de correction–Planche 9

Exercice avec préparation

Soit $N \geq 2$ et $p \in]0,1[$.
On dispose d'une urne initialement remplie avec N boules et d'une pièce donnant « Pile » avec probabilité p.
On répète l'expérience suivante :

— on lance de façon indépendante la pièce autant de fois qu'il y a de boules dans l'urne ;

— on retire de l'urne autant de boules que de « Faces » obtenues.

Pour tout $n \in \mathbb{N}$, on note X_n la variable aléatoire donnant le nombre de boules dans l'urne après n répétitions de l'expérience. En particulier, $X_0 = N$.

1. À l'issue de la première expérience on a retiré un nombre de boules égal au nombre de « Faces » obtenues. Il reste donc un nombre de boules égal au nombre de « Piles » obtenues.
   La variable X_1 compte donc le nombre de « Piles » lors de N répétitions indépendantes d'une épreuve de Bernoulli de paramètre p.
   Donc $X_1 \hookrightarrow \mathcal{B}(N,p)$.

2. ```
   def simuleX(p,n,N):
       urne = N
       for i in range(n):
           s = 0
           for k in range(urne):
               x = rd.rand()
               if x>p :
                   s += 1
           urne = urne - s
       return urne
   ```

3. C'est évident : on part de N boules et à chaque tour on retire entre 0 et N boules.
   La variable X_n possède une espérance car elle est de support fini.

4. D'après la loi faible des grands nombres, le programme suivant convient.
   ```
   def Esp(p,n,N,nb = 10000):
       s = 0
       for k in range(nb):
           s += simuleX(p,n,N)
       return s/nb
   ```

5. Soit $n \in \mathbb{N}$.
   (a) Soit $(j,k) \in [\![0,N]\!]^2$.
   Sachant que $[X_n = j]$ est réalisé, on lance la pièce j fois et X_{n+1} est égal au nombre de « Piles » obtenues. Les lancers étant indépendants, on en déduit que la loi de X_{n+1} sachant $[X_n = j]$ est une $\mathcal{B}(j,p)$.
   Ainsi
   $$\mathbb{P}_{[X_n=j]}(X_{n+1}=k) = \begin{cases} 0 & \text{si } k > j \\ \binom{j}{k} p^k (1-p)^{j-k} & \text{si } k \leq j \end{cases}$$
   ```

(b) Soit $k \in [\![0, N]\!]$. D'après la formule des probabilités totales appliquées avec le système complet d'événements $([X_n = j])_{j\in[\![0,N]\!]}$:

$$\begin{aligned}\mathbb{P}(X_{n+1} = k) &= \sum_{j=0}^{N} \mathbb{P}_{[X_n=j]}(X_{n+1} = k)\mathbb{P}(X_n = j)\\ &= \sum_{j=0}^{k-1} \mathbb{P}_{[X_n=j]}(X_{n+1} = k)\mathbb{P}(X_n = j) + \sum_{j=k}^{N} \mathbb{P}_{[X_n=j]}(X_{n+1} = k)\mathbb{P}(X_n = j)\\ &= 0 + \sum_{j=k}^{N} \binom{j}{k} p^k(1-p)^{j-k}\mathbb{P}(X_n = j).\end{aligned}$$

(c) On a en faisant une interversion de somme double triangulaire :

$$\begin{aligned}\mathbb{E}(X_{n+1}) = \sum_{k=0}^{N} k\mathbb{P}(X_{n+1} = k) &= \sum_{k=0}^{N} k \sum_{j=k}^{N} \binom{j}{k} p^k(1-p)^{j-k}\mathbb{P}(X_n = j)\\ &= \sum_{j=0}^{N} \mathbb{P}(X_n = j) \sum_{k=0}^{j} k\binom{j}{k} p^k(1-p)^{j-k}\\ &= \sum_{j=0}^{N} \mathbb{P}(X_n = j) \left(0 + \sum_{k=1}^{j} k\binom{j}{k} p^k(1-p)^{j-k}\right)\\ &= \sum_{j=0}^{N} \mathbb{P}(X_n = j) \sum_{k=1}^{j} j\binom{j-1}{k-1} p^k(1-p)^{j-k}\\ &= \sum_{j=0}^{N} j\mathbb{P}(X_n = j) \sum_{\ell=0}^{j-1} \binom{j-1}{\ell} p^{\ell+1}(1-p)^{j-1-\ell}\\ &= p\sum_{j=0}^{N} j\mathbb{P}(X_n = j) \sum_{\ell=0}^{j-1} \binom{j-1}{\ell} p^{\ell}(1-p)^{j-1-\ell}\\ &= p\sum_{j=0}^{N} j\mathbb{P}(X_n = j)(1-p+p)^{j-1}\\ &= p\mathbb{E}(X_n)\end{aligned}$$

où on a utilisé : $\forall k \in [\![1, j]\!]$, $k\binom{j}{k} = j\binom{j-1}{k-1}$, fait le changement d'indice $\ell = k-1$ puis utilisé la formule du binôme de Newton.

On peut aller beaucoup plus vite en reconnaissant à la ligne 2 l'espérance d'une variable Z de loi binomiale $\mathcal{B}(j, p)$:

$$\begin{aligned}\mathbb{E}(X_{n+1}) = \sum_{k=0}^{N} k\mathbb{P}(X_{n+1}=k) &= \sum_{k=0}^{N} k \sum_{j=k}^{N} \binom{j}{k} p^k (1-p)^{j-k} \mathbb{P}(X_n = j) \\ &= \sum_{j=0}^{N} \mathbb{P}(X_n=j) \underbrace{\sum_{k=0}^{j} k\binom{j}{k} p^k (1-p)^{j-k}}_{=\mathbb{E}(Z)=pj} \\ &= p\sum_{j=0}^{N} j\mathbb{P}(X_n=j) \\ &= p\mathbb{E}(X_n).\end{aligned}$$

En particulier, la suite $(\mathbb{E}(X_n))_{n\geq 1}$ est géométrique de raison p donc :

$$\forall n \geq 1, \quad \mathbb{E}(X_n) = p^{n-1}\mathbb{E}(X_1) = p^n N.$$

6. Soit $n \in \mathbb{N}$. On souhaite retrouver le résultat de 5.(c) par une autre méthode. Pour tout $t \in \mathbb{R}$, on pose $G_n(t) = \sum_{k=0}^{N} \mathbb{P}(X_n = k)t^k$.

   (a) Soit $t \in \mathbb{R}$. D'après la question 1 et la formule du binôme de Newton

   $$G_1(t) = \sum_{k=0}^{N} \mathbb{P}(X_1=k)t^k = \sum_{k=0}^{N} \binom{N}{k} p^k (1-p)^{N-k} t^k = (pt+1-p)^N.$$

   (b) La fonction G_n est polynomiale donc dérivable et :

   $$\forall t \in \mathbb{R}, \quad G_n'(t) = \sum_{k=0}^{N} k\mathbb{P}(X_1=k)t^{k-1}.$$

   Donc :

   $$G_n'(1) = \sum_{k=0}^{N} k\mathbb{P}(X_1=k) = \mathbb{E}(X_n).$$

   (c) Soit $t \in \mathbb{R}$. En utilisant 5.(a) et 5.(b)

   $$\begin{aligned}G_{n+1}(t) &= \sum_{k=0}^{N}\sum_{j=k}^{N} \binom{j}{k} p^k (1-p)^{j-k} \mathbb{P}(X_n=j) t^k \\ &= \sum_{j=0}^{N} \mathbb{P}(X_n=j) \sum_{k=0}^{j} \binom{j}{k} p^k (1-p)^{j-k} t^k \\ &= \sum_{j=0}^{N} \mathbb{P}(X_n=j)(pt+1-p)^j \\ &= G_n(pt+1-p).\end{aligned}$$

   (d) En particulier : $\forall t \in \mathbb{R}$, $G_{n+1}'(t) = pG_n'(pt+1-p)$.
   Avec la question 6.(b) on a donc : $\mathbb{E}(X_{n+1})G_{n+1}'(1) = pG_n'(p+1-p) = pG_n'(1) = p\mathbb{E}(X_n)$.

Exercice sans préparation

1. ```
   def sym(M):
       n = len(M)
       for i in range(n):
           for j in range(n):
               if M[i][j] != M[j][i]:
                   return False
       return True
   ```

2. Pas du tout optimal mais fait le job :

   ```
   def transpose(M):
           n = len(M)
           trM = [ [0 for k in range(n)] for i in range(n)]
           for k in range(n):
                   for i in range(n):
                           trM[k][i]= M[i][k]
           return trM
   ```
   ```

Agro-Veto – Planche 10

Cours

Relation de Chasles pour les intégrales sur un segment.

Exercice préparé

Pour tout $x \in \mathbb{R}$, on pose :

$$\sinh(x) = \frac{e^x - e^{-x}}{2} \quad \text{et} \quad \cosh(x) = \frac{e^x + e^{-x}}{2}.$$

1. Que dire d'une fonction à la fois paire et impaire ?
2. (a) Tracer le graphe de sinh et cosh entre -4 et 4 sur Python.
   (b) Déterminer la parité de ces deux fonctions et leurs limites en $+\infty$ et $-\infty$.
   (c) Déterminer le domaine de dérivabilité et calculer leurs dérivées.
3. On note E l'espace vectoriel des fonctions de classe C^∞ sur $\mathbb{R}$ et $F = \text{Vect}(\sin, \cos, \sinh, \cosh)$.
   Enfin, on considère l'endomorphisme d de E définie par :
   $$\forall f \in E, \quad d(f) = f'.$$
   (a) Montrer que $(\sin, \cos, \sinh, \cosh)$ est libre.
   On pourra s'aider de la question 1.
   (b) En déduire la dimension de F.
   (c) Montrer que $d(F) \subset F$.
4. On définit φ l'endomorphisme de F induit par d :
   $$\forall f \in F, \quad \varphi(f) = f'.$$
   (a) Écrire la matrice M de φ dans la base $(\sin, \cos, \sinh, \cosh)$.
   (b) Calculer M^2, M^3 et M^4 et en déduire l'expression de M^n pour tout $n \in \mathbb{N}$.
   (c) En déduire que φ est un automorphisme de F.
5. Calculer les valeurs propres de M à l'aide de Python. M est-elle diagonalisable ?
6. Déterminer $\text{Im}(\varphi^2 - \text{Id}_F)$.
   L'équation $y'' - y = \cosh$ admet-elle des solutions dans F ?
7. Déterminer $\text{Ker}(\varphi - \text{Id}_F)$ et $\text{Im}(\varphi - \text{Id}_F)$.
   En déduire l'ensemble des solutions dans F de $y' - y = e^{-t} + \sin(t)$.

Agro-Veto – Planche 10

Exercice sans préparation

1. Écrire une fonction Python qui prend en arguments une liste L, un réel x et renvoyant le nombre de x dans L.
2. Écrire une fonction Python qui prend en arguments un entier m, un entier n, un réel $p \in [0,1]$ et renvoyant, sous forme de tableau à une ligne, m valeurs tirées suivant une loi $\mathcal{B}(n,p)$.
3. Écrire une fonction Python qui prend en arguments un entier m, un entier n, un réel $p \in [0,1]$ et renvoyant, sous forme de tableau à deux lignes, les modalités (dans la première ligne) et le nombres d'occurrences d'une série de m valeurs tirées suivant une loi $\mathcal{B}(n,p)$.

Éléments de correction–Planche 10

Exercice avec préparation

Pour tout $x \in \mathbb{R}$, on pose :

$$\sinh(x) = \frac{e^x - e^{-x}}{2} \quad \text{et} \quad \cosh(x) = \frac{e^x + e^{-x}}{2}.$$

1. Soit f une fonction définie sur un intervalle I symétrique par rapport à l'origine. Si f est à la fois paire et impaire alors :

   $$\forall x \in I, \quad f(x) = f(-x) \quad \text{et} \quad -f(x) = f(-x).$$

   Ainsi :

   $$\forall x \in I, \quad f(x) = -f(x) \quad i.e. \quad f(x) = 0.$$

   La fonction f est donc la fonction nulle sur I.

2. (a)

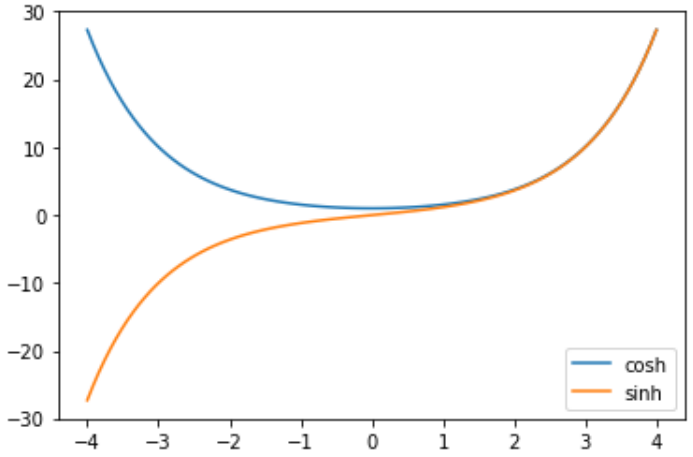
```
import matplotlib.pyplot as plt
import numpy as np

def sinh(x):
        return (np.exp(x)-np.exp(-x))/2

    def cosh(x):
        return (np.exp(x)+np.exp(-x))/2

    abs = np.linspace(-4,4,100)
    ord_cosh = cosh(abs)
    ord_sinh = sinh(abs)
    plt.plot(abs, ord_cosh, label='cosh')
    plt.plot(abs, ord_sinh, label='sinh')
    plt.legend()
    plt.show()
```

On obtient la figure suivante :



(b) D'après les graphes obtenus, on peut conjecturer que cosh est paire et sinh est impaire. Prouvons-le : pour tout $x \in \mathbb{R}$ on a :

$$\cosh(-x) = \frac{e^{-x}+e^{x}}{2} = \cosh(x) \quad ; \quad \sinh(-x) = \frac{e^{-x}-e^{x}}{2} = -\sinh(x).$$

Par opération sur les limites, on a aussi :

$$\lim_{x\to+\infty} \cosh(x) = \lim_{x\to-\infty} \cosh(x) = +\infty$$

et

$$\lim_{x\to+\infty} \sinh(x) = +\infty \quad ; \quad \lim_{x\to-\infty} \sinh(x) = -\infty.$$

(c) Par opération sur les fonctions dérivables, cosh et sinh sont dérivables sur $\mathbb{R}$ et pour tout réel x on a :

$$\cosh'(x) = \frac{e^{x}-e^{-x}}{2} = \sinh(x) \quad ; \quad \sinh'(x) = \frac{e^{x}+e^{-x}}{2} = \cosh(x).$$

3. On note E l'espace vectoriel des fonctions de classe C^∞ sur $\mathbb{R}$ et $F = \mathrm{Vect}(\sin, \cos, \sinh, \cosh)$. Enfin, on considère l'endomorphisme d de E définie par :

$$\forall f \in E, \quad d(f) = f'.$$

(a) Soit $(a,b,c,d) \in \mathbb{R}^4$ tel que

$$a\sin + b\cos + c\sinh + d\cosh = 0.$$

Alors on a :

$$a\sin + c\sinh = -b\cos - d\cosh$$

et les fonctions $a\sin + c\sinh$ et $-b\cos - d\cosh$ sont donc paires et impaires donc nulles :

$$a\sin + c\sinh = 0 \quad \text{et} \quad -b\cos - d\cosh = 0.$$

Pour tout $x > 0$, on a

$$0 = a\sin(x) + c\sinh(x) = \sinh(x)\left(\frac{a\sin(x)}{\sinh(x)} + c\right)$$

d'où ($\sinh(x) \neq 0$ pour $x > 0$) :

$$0 = \frac{a\sin(x)}{\sinh(x)} + c.$$

En faisant tendre x vers $+\infty$ on obtient $c = 0$. De même, on trouve $d = 0$.
Il reste donc :

$$a\sin = 0 \quad \text{et} \quad b\cos = 0$$

ce qui implique $a = b = 0$.
Finalement $(\sin, \cos, \sinh, \cosh)$ est libre.

(b) La dimension de F est donc 4.

(c) On a :

$$d(F) = \mathrm{Vect}(d(\sin), d(\cos), d(\sinh), d(\cosh)) = \mathrm{Vect}(\cos, -\sin, \cosh, \sinh) = F.$$

4. On définit φ l'endomorphisme de F induit par d :

$$\forall f \in F, \quad \varphi(f) = f'.$$

(a) On a déjà vu que :

$$\varphi(\sin) = \cos \quad ; \quad \varphi(\cos) = -\sin \quad ; \quad \varphi(\sinh) = \cosh \quad ; \quad \varphi(\cosh) = \sinh$$

donc

$$M = \begin{pmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

(b) On trouve :

$$M^2 = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} ; \quad M^3 = \begin{pmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \quad ; \quad M^4 = I_4.$$

On en déduit, par une récurrence immédiate :

— Si $n = 4k$ avec $k \in \mathbb{N}$ alors $M^n = I_4$.
— Si $n = 4k+1$ avec $k \in \mathbb{N}$ alors $M^n = M$.
— Si $n = 4k+2$ avec $k \in \mathbb{N}$ alors $M^n = M^2$.
— Si $n = 4k+3$ avec $k \in \mathbb{N}$ alors $M^n = M^3$.

(c) Comme $M^4 = I_3$ alors M est inversible, d'inverse M^3.
On en déduit que φ est un isomorphisme de F dans F donc un automorphisme de F.

5.
```
import numpy as np
import numpy.linalg as la

M = np.array([[0,-1,0,0],[1,0,0,0],[0,0,0,1],[0,0,1,0]])
print(la.eigvals(M))
```

On trouve comme valeurs propres -1, 1 i et $-i$. Donc M est diagonalisable sur $\mathbb{C}$ mais pas sur $\mathbb{R}$.

6. On a :

$$M^2 - I_4 = \begin{pmatrix} -2 & 0 & 0 & 0 \\ 0 & -2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

Donc

$$\mathrm{Im}(\varphi^2 - \mathrm{Id}_F) = \mathrm{Vect}(\cos, \sin).$$

En particulier, $\cosh \notin \mathrm{Im}(\varphi^2 - \mathrm{Id}_F)$ donc il ne possède pas d'antécédent par $\varphi^2 - \mathrm{Id}_F$ c'est-à-dire que l'équation $y'' - y = \cosh$ n'a pas de solution dans F.

7. On a :

$$M - I_4 = \begin{pmatrix} -1 & -1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 1 & -1 \end{pmatrix}.$$

Soit $X = \begin{pmatrix} x \\ y \\ z \\ t \end{pmatrix}$. Alors

$$(M - I_4)X = 0 \iff \begin{cases} -x - y &= 0 \\ x - y &= 0 \\ -z + t &= 0 \\ z - t &= 0 \end{cases} \iff x = y = 0 \quad \text{et} \quad z = t.$$

Ainsi : $\mathrm{Ker}(M - I_4) = \mathrm{Vect}\left(\begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}\right)$.

On en déduit

$$\mathrm{Ker}(\varphi - \mathrm{Id}_F) = \mathrm{Vect}(\cosh + \sinh).$$

De plus,

$$\begin{aligned} \mathrm{Im}(\varphi - \mathrm{Id}_F) &= \mathrm{Vect}((\varphi - \mathrm{Id}_F)(\sin), (\varphi - \mathrm{Id}_F)(\cos), (\varphi - \mathrm{Id}_F)(\sinh), (\varphi - \mathrm{Id}_F)(\cosh)) \\ &= \mathrm{Vect}(\cos - \sin, -\sin - \cos, \cosh - \sinh, \sinh - \cosh) \\ &= \mathrm{Vect}(\cos - \sin, \sin, t \mapsto e^{-t}). \end{aligned}$$

Résoudre dans F l'équation $y' - y = e^{-t} + \sin(t)$ revient à trouver les antécédents de $t \mapsto e^{-t} + \sin(t)$ par $\varphi - \mathrm{Id}_F$.
Déjà, il a bien des solutions car $t \mapsto e^{-t} + \sin(t)$ appartient à $\mathrm{Im}(\varphi - \mathrm{Id}_F)$.
Ensuite, si on dispose d'une solution y_0 alors pour toute autre solution y on a :

$$(\varphi - \mathrm{Id}_F)(y - y_0) = 0$$

c'est-à-dire $y - y_0 \in \mathrm{Ker}(\varphi - \mathrm{Id}_F) = \mathrm{Vect}(\cosh + \sinh)$.
L'ensemble des solutions sera alors :

$$\{y_0 + a(\cosh + \sinh) \ ; \ a \in \mathbb{R}\}.$$

Pour déterminer une solution particulière y_0, c'est-à-dire un antécédent de $t \mapsto e^{-t} + \sin(t)$ par $\varphi - \mathrm{Id}_F$, on raisonne matriciellement.
Remarquons d'abord que $t \mapsto e^{-t} + \sin(t)$ appartient à F et que ses coordonnées dans la base $(\sin, \cos, \sinh, \cosh)$ sont $(1, 0, -1, 1)$. On cherche donc (x, y, z, t) (les coordonnées de y_0 dans la

base $(\sin, \cos, \sinh, \cosh))$ tel qie :

$$(M - I_4) \begin{pmatrix} x \\ y \\ z \\ t \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ -1 \\ 1 \end{pmatrix}.$$

On trouve :

$$x = y = -\frac{1}{2} \quad ; \quad z = 1 + t.$$

Ainsi avec $t = 0$ on trouve une solution particulière :

$$y_0 = -\frac{\cos + \sin}{2} + \sinh.$$

Exercice sans préparation

1.
```
def comptage(L,x):
    s = 0
    for elt in L:
        if elt == x:
            s += 1
    return s
```

2.
```
def bin(n,p):
    s = 0
    for k in range(n):
        if rd.rand()<p:
            s+=1
    return s

def serie_bin(m,n,p):
    S = np.zeros(m)
    for k in range(m):
        S[k]=bin(n,p)
    return S
```

3.
```
def tableau(m,n,p):
    L = serie_bin(m,n,p)
    Tab = np.zeros([2,n+1])
    for k in range(n+1):
        Tab[0,k]=k
        Tab[1,k]=comptage(L,k)
    return Tab
```

Agro-Veto – Planche 11

Cours : Donner la définition des fonctions partielles d'une fonction définie sur $\mathbb{R}^2$.

Exercice préparé

Rappel : soit S, T deux variables aléatoires de densités respectives f_S et f_T, indépendantes ; alors $S+T$ est à densité et une densité est donnée par : $\forall t \in \mathbb{R}, \quad f_{S+T}(t) = \displaystyle\int_{-\infty}^{+\infty} f_S(s) f_T(t-s) ds$.

1. Soit $U \hookrightarrow \mathcal{U}(]0,1[)$ et $\lambda > 0$. Montrer que $X = -\dfrac{1}{\lambda}\ln(1-U)$ suit une loi exponentielle dont on précisera le paramètre.
2. Écrire un programme Python qui simule une loi exponentielle.
3. On considère une suite $(X_n)_{n\in\mathbb{N}^*}$ de variables aléatoires indépendantes et de même loi que X. On définit la suite de variables aléatoires $(S_n)_{n\in\mathbb{N}^*}$ par $S_0 = 0$ et $\forall n \in \mathbb{N}^*$, $S_n = X_1 + \cdots + X_n$.
   (a) À l'aide d'une récurrence, montrer que la fonction f_n définie ci-dessous est une densité.
   $$\forall n \in \mathbb{N}^*, \quad f_n(t) = \frac{\lambda e^{-\lambda t}(\lambda t)^{n-1}}{(n-1)!}\mathbf{1}_{[0,+\infty[}(t).$$
   (b) En utilisant le rappel et une récurrence, montrer que f_n est une densité de S_n pour tout $n \geq 1$.
4. On suppose qu'à un arrêt, les différences entre les horaires de passage successifs d'un bus sont indépendantes et de même loi exponentielle de paramètre λ.
   On définit un instant $S_0 = 0$ puis on note S_1, S_2 etc, les horaires de passages successifs des bus. On note alors, pour tout $t > 0$, N_t le nombre de bus passés à l'arrêt entre l'instant 0 et l'instant t. Autrement dit : $\forall n \geq 1$, $[N_t = n] = [S_n \leq t < S_{n+1}]$.
   (a) Pour $n \geq 0$, exprimer avec soin l'événement $[N_t \geq n]$ à l'aide de S_n.
   (b) Justifier alors que : $\forall n \geq 0$, $\mathbb{P}(N_t = n) = \mathbb{P}(S_n \leq t) - \mathbb{P}(S_{n+1} \leq t)$.
   (c) En déduire que N_t suit une loi de Poisson de paramètre λt.
5. On suppose plus précisément que les horaires de passages successifs d'un bus sont, en moyenne, de 10 minutes. Un individu arrive à l'arrêt à l'instant $T = 100$ min pour prendre le bus et on se pose alors deux questions :
   — Combien de temps en moyenne va-t-il attendre le prochain bus ?
   — Combien de temps en moyenne s'écoule-t-il entre le prochain bus et celui qui l'a précédé ?
   On considère le programme suivant :

```
import math as m
import random as rd

def autobus():
    a,b,N=0,0,10000
    for k in range(N):
        s = 0
            while s<100:
                r = s
                s-= 10*m.log(1-rd.rand())
                u,v=s-100,s-r
                a,b=a+u,b+v
    return a/N, b/N
```

   (a) Expliquer ce que représentent les variables r, s, u et v dans le programme.
   (b) Le programme affiche les valeurs suivantes 10.062252 20.315494. Pourquoi les valeurs affichées sont-elles paradoxales vis à vis de la situation ?

Agro-Veto – Planche 11

Exercice sans préparation

1. Écrire une fonction d'argument une liste L qui teste si la liste L vérifie l'hypothèse $\mathcal{H}$ suivante :
   $\mathcal{H}$: les éléments de la liste L sont des entiers qui sont compris au sens large entre 0 et $n-1$ avec n la longueur de la liste.
2. Écrire une fonction d'argument une liste L vérifiant l'hypothèse $\mathcal{H}$ et qui teste si la liste L vérifie l'hypothèse $\mathcal{H}'$ suivante :
   $\mathcal{H}'$: la liste L contient exactement une fois chaque valeur entre 0 et $n-1$ avec n la longueur de la liste.

Éléments de correction–Planche 11

Exercice avec préparation

On rappelle que si S et T sont deux variables aléatoires réelles de densités respectives f_S et f_T et indépendantes alors $S+T$ est une variable aléatoire à densité dont une densité est donnée par la formule de convolution :

$$\forall t \in \mathbb{R}, \quad f_{S+T}(t) = \int_{-\infty}^{+\infty} f_S(s) f_T(t-s) ds \qquad (E).$$

1. Soit U une variable de loi uniforme sur $]0,1[$ et λ un réel strictement positif. On note $X = -\frac{1}{\lambda}\ln(1-U)$.
   Soit $x \in \mathbb{R}$. On a :
$$\begin{aligned} F_X(x) = P(X \leq x) = P(-\frac{1}{\lambda}\ln(1-U) \leq x) &= P(1-U \geq e^{-\lambda x}) \\ &= P(U \leq 1-e^{-\lambda x}) \\ &= \begin{cases} 0 & \text{si } 1-e^{-\lambda x} \leq 0 \\ 1-e^{-\lambda x} & \text{si } 0 < 1-e^{-\lambda x} < 1 \\ 1 & \text{sinon} \end{cases} \\ &= \begin{cases} 0 & \text{si } x \leq 0 \\ 1-e^{-\lambda x} & \text{si } x > 0 \end{cases} \\ &= \begin{cases} 0 & \text{si } x < 0 \\ 1-e^{-\lambda x} & \text{si } x \geq 0 \end{cases} \end{aligned}$$
   On reconnaît la fonction de répartition d'une loi exponentielle de paramètre λ.
   Ainsi X suit la loi exponentielle de paramètre λ.

2.
```
def simule_exp(lambda_):
    return -1/lambda_*np.log(1-rd.rand())
```

3. On considère une suite $(X_n)_{n\in\mathbb{N}^*}$ de variables aléatoires indépendantes et de même loi que X.
   On définit la suite de variables aléatoires $(S_n)_{n\in\mathbb{N}^*}$ par $S_0 = 0$ et $\forall n \in \mathbb{N}^*$, $S_n = X_1 + \cdots + X_n$.
   (a) — **Initialisation** : f_1 est une densité d'une loi exponentielle de paramètre λ donc il n'y a rien à faire.
   — **Hérédité** : soit $n \in \mathbb{N}^*$ et supposons que f_n est une densité.
   On remarque déjà que f_{n+1} est continue sur $\mathbb{R}$ et positive. Il suffit d'étudier l'intégrale de f_{n+1} sur $\mathbb{R}$.
   — Comme f_{n+1} est nulle sur $]-\infty,0[$, $\int_{-\infty}^{0} f_{n+1}(t)dt$ converge et vaut 0.
   — Soit $A > 0$. On effectue une IPP
$$\begin{aligned} \int_0^A f_{n+1}(t)dt &= \frac{1}{n!}\int_0^A \lambda e^{-\lambda t}(\lambda t)^n dt \\ &= \frac{1}{n!}\left([-e^{-\lambda t}(\lambda t)^n]_0^A + \int_0^A e^{-\lambda t} n\lambda(\lambda t)^{n-1} dt\right) \\ &= \frac{-e^{-\lambda A}(\lambda A)^n}{n!} + \int_0^A f_n(t)dt. \end{aligned}$$

Par hypothèse de récurrence, $\lim\limits_{A\to+\infty}\int_0^A f_n(t)dt = 1$.

Par croissances comparées $\lim\limits_{A\to+\infty}\dfrac{-e^{-\lambda A}(\lambda A)^n}{n!} = 0$.

Finalement, $\lim\limits_{A\to+\infty}\int_0^A f_{n+1}(t)dt = 1$.

— On en conclut que $\int_{\mathbb{R}} f_{n+1}(t)dt$ converge et vaut 1.

Ainsi, f_{n+1} est bien une densité de probabilité.

— **Conclusion** : par principe de récurrence on a montré que pour tout $n \in \mathbb{N}^*$ la fonction f_n est une densité de probabilité.

(b) — **Initialisation :** pour $n = 1$ c'est immédiat.

— **Hérédité :** soit $n \geq 1$ et supposons que f_n est une densité de S_n.

On a $S_{n+1} = S_n + X_{n+1}$ avec, d'après le lemme des coalitions, S_n et X_{n+1} indépendants.

Donc une densité de S_{n+1} est donnée par :

$$\forall t \in \mathbb{R}, \quad g(t) = \int_{-\infty}^{+\infty} f_n(s)f_1(t-s)ds.$$

Comme f_n est nulle sur $]-\infty, 0[$ on a :

$$\forall t \in \mathbb{R}, \quad g(t) = \int_{0}^{+\infty} f_n(s)f_1(t-s)ds.$$

Soit $t \in \mathbb{R}$. Le facteur $f_1(t-s)$ est nul pour les s dans l'intervalle $[t, +\infty[$.

— Si $t < 0$ alors $[0, +\infty[\subset [t, +\infty[$ donc $g(t) = 0$ (l'intégrande est nul sur $[0, +\infty[$).

— Si $t \geq 0$ alors l'intégrande est nul sur $[t, +\infty[$ donc

$$\begin{aligned} g(t) = \int_0^t f_n(s)f_1(t-s)ds &= \int_0^t \lambda e^{-\lambda s}\frac{(\lambda s)^{n-1}}{(n-1)!}\lambda e^{-\lambda(t-s)}ds \\ &= \frac{\lambda^{n+1}}{(n-1)!}e^{-\lambda t}\int_0^t s^{n-1}ds \\ &= \frac{\lambda^{n+1}}{(n-1)!}e^{-\lambda t}\frac{t^n}{n} \\ &= \frac{(\lambda t)^n}{n!}\lambda e^{-\lambda t}. \end{aligned}$$

Ainsi on a bien $g = f_{n+1}$.

— **Conclusion :** par principe de récurrence, on a montré que pour tout $n \geq 1$, f_n est une densité de S_n.

4. On suppose qu'à un arrêt, les différences entre les horaires de passage successifs d'un bus sont indépendantes et de même loi exponentielle de paramètre λ.

On définit un instant $S_0 = 0$ puis on note S_1, S_2 etc, les horaires de passages successifs des bus.

On note alors, pour tout $t > 0$, N_t le nombre de bus passés à l'arrêt entre l'instant 0 et l'instant t.

Autrement dit : $\forall n \geq 1$, $[N_t = n] = [S_n \leq t < S_{n+1}]$.

(a) Soit $n \geq 0$. L'événement $[N_t \geq n]$ est réalisé si et seulement si il y a eu au moins n bus de passés entre les instants 0 et t , c'est-à-dire si et seulement si $[S_1 \leq t]$,..., $[S_n \leq t]$ sont réalisés. Comme $S_1 \leq S_2 \leq \cdots \leq S_n$ alors on obtient :

$$[N_t \geq n] = \bigcap_{i=1}^{n} [S_n \leq t] = [S_n \leq t].$$

(b) Soit $\forall n \geq 0$. On a :

$$[S_n \leq t] = [N_t \geq n] = [N_t \geq n+1] \cup [N_t = n] = [S_{n+1} \leq t] \cup [N_t = n].$$

Les événements $[S_{n+1} \leq t]$ et $[N_t = n]$ étant disjoints, on a donc :

$$P(S_n \leq t) = P(N_t = n) + P(S_{n+1} \leq t)$$

d'où :

$$P(N_t = n) = P(S_n \leq t) - P(S_{n+1} \leq t).$$

(c) Soit $n \geq 1$. Avec la question précédente et connaissant une densité de S_n et S_{n+1} on a :

$$\begin{aligned} P(N_t = n) = P(S_n \leq t) - P(S_{n+1} \leq t) &= \int_0^t \lambda e^{-\lambda t} \left(\frac{(\lambda t)^{n-1}}{(n-1)!} - \frac{(\lambda t)^n}{n!} \right) dt \\ &= \frac{e^{-\lambda t}(\lambda t)^n}{n!} \quad \text{(voir l'IPP de 3.(a))} \end{aligned}$$

On a alors comme N_t est à valeurs dans $\mathbb{N}$:

$$P(N_t = 0) = 1 - \sum_{n=1}^{+\infty} \frac{e^{-\lambda t}(\lambda t)^n}{n!} = \frac{e^{-\lambda t}(\lambda t)^0}{0!}.$$

Ainsi,

$$\forall n \in \mathbb{N}, \quad P(N_t = n) = \frac{e^{-\lambda t}(\lambda t)^n}{n!}.$$

Donc N_t suit une loi de Poisson de paramètre λt.

5. On suppose que les horaires de passages successifs d'un bus sont, en moyenne, de 10 minutes. Comme l'espérance d'une loi de Poisson de paramètre λ est $\frac{1}{\lambda}$, on a donc ici :

$$\frac{1}{\lambda} = 10 \quad i.e \quad \lambda = \frac{1}{10}.$$

Un individu arrive à l'arrêt à l'instant $T = 100$ min pour prendre le bus.
On se pose alors deux questions :

— Combien de temps en moyenne va-t-il attendre le prochain bus ? Il s'agit de la valeur moyenne (espérance) de $S_{N_T+1} - T$.
— Combien de temps en moyenne s'écoule-t-il entre le prochain bus et celui qui l'a précédé ? Il s'agit de la valeur moyenne (espérance) de $S_{N_T+1} - S_{N_T}$.

On considère le programme suivant :

```
import math as m
import random as rd

def autobus():
        a,b,N=0,0,10000
        for k in range(N):
                s = 0
                while s<100:
                        r = s
                        s-= 10*m.log(1-rd.random())
                u,v=s-100,s-r
                a,b=a+u,b+v
        return a/N, b/N
```

(a) — La variable r donne les temps de passage successifs des bus avant l'arrivée de l'individu : la durée entre deux passages successifs étant donnée par `10*m.log(1-rd.rand())` qui suit bien une loi exponentielle de paramètre $\frac{1}{10}$.

À la fin de la boucle, r est l'heure du dernier passage avant l'arrivée de l'individu et s l'heure du passage suivant.

— La variable $u = s - 100$ représente alors l'attente de l'individu avant le prochain bus.

— La variable $v = s - r$ représente le temps entre le bus précédent et le bus suivant l'arrivée de l'individu.

(b) Le programme affiche les valeurs suivantes :

10.062252 20.315494

D'après la loi des grands nombres, a/N est une valeur approchée du temps moyen d'attente du prochain bus et b/N est une valeur approchée du temps moyen d'attente entre le bus précédent et le bus suivant l'arrivée de l'individu.

Ce résultat est paradoxal car en moyenne, le temps entre le bus précédent et le bus suivant l'arrivée de l'individu semble avoisiner les 20 min alors qu'on a fait l'hypothèse que les bus passent en moyenne toutes les 10 minutes.

On pourrait également s'attendre, vu que les bus passent toutes les 10 minutes, à ce que le temps d'attente moyen de l'individu soit de 5 min. Or la simulation montre qu'il est plutôt de 10 min.

Exercice sans préparation

1.
```
def testH(L):
    n = len(L)
    for elt in L:
        if elt != int(elt) or elt<0 or elt>n-1:
            return False
    return True
```

2.
```
def testHprime(L):
    if not testH(L):
        return 'L ne verifie pas H'
    else :
        n= len(L)
        if sum(L)== n*(n-1)/2:
            return True
        else:
            return False
```

Agro-Veto – Planche 12

Cours

Donner la définition d'un sous-espace vectoriel F d'un espace vectoriel E.

Exercice préparé

Soit f une fonction de $\mathbb{R}$ dans $\mathbb{R}$. On dit qu'une suite $u = (u_n)_{n\in\mathbb{N}^*}$ est adaptée à f si et seulement si

$$\forall n \in \mathbb{N}^*,\ \forall x \in \mathbb{R}, \quad u_n f(nx) = \sum_{k=0}^{n-1} f\left(x + \frac{k}{n}\right) \quad (*).$$

On admet que si une fonction non identiquement nulle f possède une suite adaptée, alors cette suite est unique.
On note E l'ensemble des fonctions de $\mathbb{R}$ dans $\mathbb{R}$ possédant une suite adaptée.

1. Montrer que la suite constante égale à 1 est adaptée à la fonction $x \mapsto x - \frac{1}{2}$.
2. Montrer que si f est une fonction dérivable admettant une suite adaptée (u_n) alors la suite (nu_n) est adaptée à f'.
3. On admet dans la suite que l'on peut définir une suite de polynômes $(B_p)_{p\in\mathbb{N}}$ de la façon suivante :
$$B_0 = 1 \quad \text{et} \quad \forall p \in \mathbb{N}^*, \quad B_p' = pB_{p-1} \text{ et } \int_0^1 B_p(t)dt = 0.$$
L'objectif est alors de montrer que pour tout $p \in \mathbb{N}$, B_p appartient à E.
   (a) Écrire une fonction Python qui prend en argument une liste de réels $L = [a_0, \dots, a_n]$ et renvoie la valeur de l'intégrale $\int_0^1 (\sum_{k=0}^{n} a_k x^k)dx$.
   (b) Calculer B_1, B_2 et vérifier que B_0 et B_1 appartiennent à E.
   (c) Déterminer, pour tout $p \in \mathbb{N}$, le degré et le coefficient dominant de B_p.
   (d) Soit $p \in \mathbb{N}^*$. On suppose que B_{p-1} appartient à E et on cherche à montrer que B_p appartient aussi à B_p.
      i. Montrer que si B_p appartient à E alors la suite $\left(\frac{1}{n^{p-1}}\right)_{n\in\mathbb{N}^*}$ est adaptée à B_p.
      ii. Montrer que la fonction $\varphi : x \mapsto \frac{1}{n^{p-1}} B_p(nx) - \sum_{k=0}^{n-1} B_p\left(x + \frac{k}{n}\right)$ est constante.
      iii. Calculer $\int_0^{\frac{1}{n}} \varphi(x)dx$.
      iv. Conclure.

Agro-Veto – Planche 12

Exercice sans préparation

Soit a et b deux entiers naturels avec $a < b$.

1. Écrire une fonction `verif(L,a,b)` qui a pour paramètres une liste `L` d'entiers naturels et deux entiers a et b et qui renvoie `True` si tous les éléments de `L` sont dans l'intervalle $[\![a, b]\!]$ et `False` sinon.
2. Soit `L` une liste dont chaque élément est dans l'intervalle $[\![a, b]\!]$.

   Écrire une fonction `denombre(L,a,b)` qui détermine le nombre d'apparitions de chacune des valeurs possibles. On affichera le résultat sous forme d'une liste dont le premier élément représentera le nombre de a, le deuxième élément le nombre de $a + 1$, etc.

   Par exemple, `denombre([1,3,0,4,1,3,1],0,4)` renverra `[1,3,0,2,1]`.

Éléments de correction–Planche 12

Exercice avec préparation

1. Soit $n \in \mathbb{N}^*$ et $x \in \mathbb{R}$. Alors d'une part
$$1 \times \left(nx - \frac{1}{2}\right) = nx - \frac{1}{2}$$
et d'autre part
$$\sum_{k=0}^{n-1} \left(x + \frac{k}{n} - \frac{1}{2}\right) = nx + \frac{n(n-1)}{2n} - \frac{n}{2} = nx - \frac{1}{2}.$$
Ainsi, pour tout $n \in \mathbb{N}^*$, pour tout $x \in \mathbb{R}$:
$$1 \times \left(nx - \frac{1}{2}\right) = \sum_{k=0}^{n-1} \left(x + \frac{k}{n} - \frac{1}{2}\right).$$

2. Soit f une fonction dérivable et soit (u_n) une suite adaptée à f.
Soit $n \in \mathbb{N}^*$.
Comme (u_n) est adaptée à f alors pour tout $x \in \mathbb{R}$ on a :
$$u_n f(nx) = \sum_{k=0}^{n-1} f\left(x + \frac{k}{n}\right)$$
Comme f est dérivable sur $\mathbb{R}$ on obtient en dérivant :
$$nu_n f'(nx) = \sum_{k=0}^{n-1} f'\left(x + \frac{k}{n}\right).$$
Cela montre le résultat voulu.

3. (a) Par linéarité, l'intégrale vaut :
$$\int_0^1 (\sum_{k=0}^{n} a_k x^k)dx = \sum_{k=0}^{n} a_k \int_0^1 x^k dx = \sum_{k=0}^{n} \frac{a_k}{k+1}.$$
D'où le programme suivant :
```
def int(L):
        n = len(L)-1
        s = 0
        for k in range(n+1):
                s += L[k]/(k+1)
        return s
```

(b) Par construction $B_0 = 1$ et on cherche B_1 tel que

$$B_1' = B_0 \quad \text{et} \quad \int_0^1 B_1(t)dt = 0.$$

La première condition donne $B_1 = X + c$ où $c \in \mathbb{R}$ et la deuxième condition impose :

$$0 = \int_0^1 (t + c)dt = \frac{1}{2} + c.$$

Ainsi $B_1 = x - \frac{1}{2}$.
De même, on cherche B_2 tel que

$$B_2' = 2B_1 = 2X - 1 \quad \text{et} \quad \int_0^1 B_2(t)dt = 0.$$

La première condition donne $B_2 = X^2 - X + c$ où $c \in \mathbb{R}$ et la deuxième condition impose :

$$0 = \int_0^1 (x^2 - x + c)dt = \frac{1}{3} - \frac{1}{2} + c.$$

Ainsi $B_2 = x^2 - x + \frac{1}{6}$.
D'après la question 1, B_1 appartient bien à E et d'après la question 2, $B_0 = B_1'$ aussi.

(c) Montrons par récurrence que pour tout $p \in \mathbb{N}^*$, B_p est de degré p et de coefficient dominant 1.

— **Initialisation** : c'est la question précédente.
— **Hérédité** : soit $p \in \mathbb{N}^*$ et supposons que B_p est de degré p et de coefficient dominant 1. Alors :

$$B_p = x^p + Q \quad \text{avec} \quad \deg(Q) \leq p - 1.$$

Donc

$$B_{p+1}' = (p + 1)X^p + pQ$$

et en notant R une primitive de Q :

$$B_{p+1} = X^{p+1} + pR \quad \text{avec} \quad \deg(R) \leq p.$$

Ainsi le degré de B_{p+1} est $p + 1$ et son coefficient dominant est 1.
— **Conclusion** : par principe de récurrence, on a montré que pour tout $p \in \mathbb{N}^*$, B_p est de degré p et de coefficient dominant 1.

(d) i. On suppose que B_p appartient à E donc qu'il existe une suite (u_n) adaptée à B_p :

$$\forall n \in \mathbb{N}^*,\ \forall x \in \mathbb{R}, \quad u_n B_p(nx) = \sum_{k=0}^{n-1} B_p\left(x + \frac{k}{n}\right).$$

Soit $n \in \mathbb{N}^*$ et écrivons $B_p = X^p + Q$ avec $\deg(Q) \leq p - 1$. Pour tout $x \in \mathbb{R}$ on a :

$$\begin{aligned} u_n((nx)^p + Q(nx)) &= \sum_{k=0}^{n-1} \left(\left(x + \frac{k}{n}\right)^p + Q\left(x + \frac{k}{n}\right)\right) \\ &= nx^p + H_n(x) \end{aligned}$$

où H_n est un polynôme de degré au plus $p-1$. Ainsi pour tout $x \neq 0$:

$$x^p n^p u_n \left((1 + \frac{Q(nx)}{x^p n^p} \right) = nx^p \left(1 + \frac{H_n(x)}{nx^p} \right).$$

d'où

$$n^p u_n \left((1 + \frac{Q(nx)}{x^p n^p} \right) = n \left(1 + \frac{H_n(x)}{nx^p} \right).$$

Comme Q et H_n sont de degré inférieur ou égal à $p-1$, en faisant tendre x vers $+\infty$ on obtient :

$$n^p u_n = n \quad \text{i.e} \quad u_n = \frac{1}{n^{p-1}}.$$

ii. Soit $n \in \mathbb{N}^*$. La fonction $\varphi : x \mapsto \frac{1}{n^{p-1}} B_p(nx) - \sum_{k=0}^{n-1} B_p \left(x + \frac{k}{n} \right)$ est dérivable sur $\mathbb{R}$ car polynomiale. Pour tout $x \in \mathbb{R}$:

$$\begin{aligned}
\varphi'(x) &= \frac{n}{n^{p-1}} B_p'(nx) - \sum_{k=0}^{n-1} B_p' \left(x + \frac{k}{n} \right) \\
&= \frac{p}{n^{p-2}} B_{p-1}(nx) - p \sum_{k=0}^{n-1} B_{p-1} \left(x + \frac{k}{n} \right) \quad \text{car } B_p' = pB_{p-1} \\
&= p \left(\frac{1}{n^{p-2}} B_{p-1}(nx) - \sum_{k=0}^{n-1} B_{p-1} \left(x + \frac{k}{n} \right) \right).
\end{aligned}$$

Or, on a supposé que B_{p-1} appartient à E donc, d'après la question précédente appliqué à B_{p-1}, la suite $\left(\frac{1}{n^{p-2}} \right)$ est adaptée à B_{p-1}. Ainsi :

$$\forall x \in \mathbb{R}, \quad \varphi'(x) = p \left(\frac{1}{n^{p-2}} B_{p-1}(nx) - \sum_{k=0}^{n-1} B_{p-1} \left(x + \frac{k}{n} \right) \right) = 0.$$

Donc φ est constante sur $\mathbb{R}$.

iii. Soit $n \in \mathbb{N}^*$.

$$\int_0^{\frac{1}{n}} \varphi(x) dx = \frac{1}{n^{p-1}} \int_0^{\frac{1}{n}} B_p(nx) dx - \sum_{k=0}^{n-1} \int_0^{\frac{1}{n}} B_p \left(x + \frac{k}{n} \right) dx.$$

En effectuant le changement de variable $s = nx$ dans la première intégrale et $s = x + \frac{k}{n}$ dans chaque terme de la somme on obtient :

$$\begin{aligned}
\int_0^{\frac{1}{n}} \varphi(x) dx &= \frac{1}{n^{p-1}} \int_0^{\frac{1}{n}} B_p(nx) dx - \sum_{k=0}^{n-1} \int_0^{\frac{1}{n}} B_p \left(x + \frac{k}{n} \right) dx \\
&= \frac{1}{n^p} \int_0^1 B_p(s) ds - \sum_{k=0}^{n-1} \int_{\frac{k}{n}}^{\frac{k+1}{n}} B_p(s) ds \\
&= \frac{1}{n^p} \int_0^1 B_p(s) ds - \int_0^1 B_p(s) ds \quad \text{(Chasles)} \\
&= 0
\end{aligned}$$

par hypothèse sur B_p.

iv. La fonction φ est constante ; notons c cette constante.
D'après la question précédente :

$$0 = \int_0^{\frac{1}{n}} \varphi(x)dx = \frac{c}{n}.$$

Donc $c = 0$.
Cela signifie que pour tout $n \in \mathbb{N}^*$, pour tout $x \in \mathbb{R}$:

$$\frac{1}{n^{p-1}} B_p(nx) = \sum_{k=0}^{n-1} B_p\left(x + \frac{k}{n}\right).$$

Ainsi B_p appartient à E.
On a ainsi montré :

— **Initialisation :** B_0 et B_1 sont dans E.
— **Hérédité :** pour tout $p \in \mathbb{N}^*$, si B_{p-1} est dans E alors B_p aussi.
— **Conclusion :** par principe de récurrence, pour tout $p \in \mathbb{N}$, B_p appartient à E.

Exercice sans préparation

1. ```
   def verif(L,a,b):
       for elt in L:
           if elt >b or elt<a :
               return False
       return True
   ```

2. ```
   def denombre(L,a,b):
       n = b-a+1
       if verif(L,a,b):
           tab = [0 for k in range(n)]
           for elt in L:
               tab[elt-a]+=1
           return tab
       return 'verif(L,a,b) faux'
   ```

Agro-Veto – Planche 13

Cours

Donner la définition d'une densité de probabilité.

Exercice préparé

Soit $n \in \mathbb{N}^*$. On considère l'espace vectoriel $\mathbb{R}^n$ muni de sa base canonique $\mathcal{B} = (e_1, \dots, e_n)$, de son produit scalaire canonique noté $(\cdot|\cdot)$ et de sa norme usuelle notée $\|.\|$.
Soit $p \in [\![1, n]\!]$. Pour toute famille $(u_1, \dots, u_p)$ de vecteurs de $\mathbb{R}^n$, on définit G la matrice de Gram de $(u_1, \dots, u_p)$ par

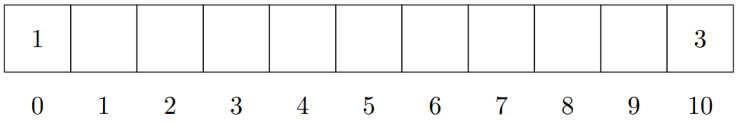
$$G = \begin{pmatrix} (u_1|u_1) & (u_1|u_2) & \cdots & (u_1|u_p) \\ (u_2|u_1) & (u_2|u_2) & \cdots & (u_2|u_p) \\ \vdots & \vdots & & \vdots \\ (u_p|u_1) & (u_p|u_2) & \cdots & (u_p|u_p) \end{pmatrix}.$$

1. (a) Écrire une fonction Python `ps` prenant une argument deux vecteurs `u` et `v` sous forme de liste de même taille et renvoyant $(u|v)$.
   (b) Écrire une fonction Python `Gram` prenant une argument $(u_1, \dots, u_p)$ sous forme d'une liste de listes et renvoyant la matrice G.
   (c) Tester votre fonction avec les vecteurs $u_1 = (1, -1, 0)$, $u_2 = (1, 0, -1)$ et $u_3 = (1, 1, 1)$.
2. Justifier que la matrice de Gram d'une famille de vecteurs de $\mathbb{R}^n$ est diagonalisable.
3. Soit $(u_1, \dots, u_p)$ une famille de $\mathbb{R}^n$ et G sa matrice de Gram. On cherche à montrer que la famille est libre si et seulement si G est inversible.
   (a) On suppose que G est inversible et soient $\alpha_1, \dots, \alpha_p \in \mathbb{R}$ tels que $\sum_{k=1}^{p} \alpha_k u_k = 0$. Montrer que $G \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_p \end{pmatrix} = 0$ puis en déduire que $(u_1, \dots, u_p)$ est libre.
   (b) On suppose $(u_1, \dots, u_p)$ libre et soit $X = \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_p \end{pmatrix}$ tel que $GX = 0$.
      i. Montrer que pour tout $i \in [\![1, p]\!]$, $\left(u_i \middle| \sum_{k=1}^{p} \alpha_k u_k \right) = 0$.
      ii. En déduire que $\sum_{k=1}^{p} \alpha_k u_k = 0$.
      iii. Montrer que $X = 0$ et en déduire que G est inversible.
4. Soit $(v_1, \dots, v_n)$ une famille de $\mathbb{R}^n$ telle que :
$$\forall i \in [\![1, n]\!], \quad \|v_i\| = 1 \quad \text{et} \quad \forall (i, j) \in [\![1, n]\!]^2,\ i \neq j \Rightarrow \|v_i - v_j\| = 1.$$
   (a) Pour tout $a, b \in \mathbb{R}^n$, montrer que $\|a - b\|^2 = \|a\|^2 + \|b\|^2 - 2(a|b)$.
   (b) En déduire la matrice de Gram de la famille $(v_1, \dots, v_n)$ que l'on notera toujours G.
   (c) On pose $A = 2G$. Exprimer A^2 en fonction de n, A et I_n.
   En déduire que A est inversible.
   (d) Montrer que $(v_1, \dots, v_n)$ est une base de $\mathbb{R}^n$.

Agro-Veto – Planche 13

Exercice sans préparation

1. Écrire une fonction Python d'arguments d'entrée `i` et `N`, qui simule une marche aléatoire sur $\mathbb{Z}$: le marcheur démarre de l'entier `i` et à chaque pas il avance de 1 ou recule de 1 avec probabilité $\frac{1}{2}$. La marche s'arrête après le `N`-ième pas et la fonction renvoie l'entier sur lequel le marcheur s'est arrêté.
2. Un plateau de jeu est constitué de $n+1$ cases numérotées de 0 à n avec $n > 1$, les cases 0 et n contenant des valeurs a et b. Par exemple avec $n = 10$, $a = 1$ et $b = 3$:



Pour $i \in [\![0, n]\!]$, on considère le jeu J_i suivant : on place un pion dans la case numéro i et tant que l'on n'est pas dans la case 0 ou n, on lance une pièce équilibrée. Si elle tombe sur face, on se déplace à gauche, si elle tombe sur pile, on se déplace vers la droite.

On admet que la partie se termine presque surement et on appelle gain du joueur la valeur a ou b contenue dans la case finale.

Écrire une fonction qui prend en paramètres les valeurs n, a, b et i, qui simule ce jeu et qui renvoie le gain.

Éléments de correction–Planche 13

Exercice avec préparation

1. (a)
```
def ps(u,v):
    n = len(u)
    s = 0
    for i in range(n):
        s += u[i]*v[i]
    return s
```

   (b)
```
def Gram(L):
    p = len(L)
    G = np.zeros([p,p])
    for i in range(p):
        for j in range(p):
            G[i,j]=ps(L[i],L[j])
    return G
```

   (c) On obtient la matrice $\begin{pmatrix} 2 & 1 & 0 \\ 1 & 2 & 0 \\ 0 & 0 & 3 \end{pmatrix}$.

2. Par symétrie du produit scalaire, G est une matrice symétrique à coefficients réels donc elle est diagonalisable d'après le théorème spectral.

3. (a) On suppose que G est inversible. Soient $\alpha_1, \dots, \alpha_p \in \mathbb{R}$ tels que $\sum_{k=1}^{p} \alpha_k u_k = 0$.

$$G \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_p \end{pmatrix} = \begin{pmatrix} \sum_{j=1}^{n} \alpha_j (u_1 | u_j) \\ \vdots \\ \sum_{j=1}^{n} \alpha_j (u_p | u_j) \end{pmatrix} = \begin{pmatrix} \left(u_1 \,|\, \sum_{j=1}^{n} \alpha_j u_j \right) \\ \vdots \\ \left(u_p \,|\, \sum_{j=1}^{n} \alpha_j u_j \right) \end{pmatrix} = 0.$$

   Comme par hypothèse G est inversible, cela implique que $\begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_p \end{pmatrix} = 0$.

   Ainsi, on a montré : $\sum_{k=1}^{p} \alpha_k u_k = 0 \Longrightarrow \alpha_1 = \cdots = \alpha_p = 0$.

   La famille $(u_1, \dots, u_p)$ est libre.

   (b) On suppose que $(u_1, \dots, u_p)$ est libre.

i. D'après le calcul précédent :

$$GX = \begin{pmatrix} \left(u_1 \,|\, \sum_{j=1}^{p} \alpha_j u_j\right) \\ \vdots \\ \left(u_p \,|\, \sum_{j=1}^{p} \alpha_j u_j\right) \end{pmatrix}.$$

L'hypothèse $GX = 0$ donne donc :

$$\forall i \in [\![1, p]\!] \quad \left(u_i \,|\, \sum_{j=1}^{p} \alpha_j u_j\right) = 0.$$

ii. On a par linéarité du produit scalaire :

$$\|\sum_{k=1}^{p} \alpha_k u_k\|^2 = \left(\sum_{i=1}^{p} \alpha_i u_i \,|\, \sum_{j=1}^{p} \alpha_j u_j\right) = \sum_{i=1}^{p} \alpha_i \left(u_i \,|\, \sum_{j=1}^{p} \alpha_j u_j\right) = 0.$$

Donc $\sum_{k=1}^{p} \alpha_k u_k = 0$.

iii. Comme la famille $(u_1, \ldots, u_p)$ est supposée libre, cela entraîne que $\alpha_1 = \cdots = \alpha_p = 0$ c'est-à-dire $X = 0$.

On a donc montré que $GX = 0 \Rightarrow X = 0$: $\mathrm{Ker}(G) = \{\}$. Donc G est inversible.

4. (a) Soit $a, b \in \mathbb{R}^n$. Par bilinéarité et symétrie du produit scalaire :

$$\begin{aligned} \|a - b\|^2 = (a - b|a - b) &= (a|a) + (b|b) - (a|b) - (b|a) \\ &= \|a\|^2 + \|b\|^2 - 2(a|b). \end{aligned}$$

(b) On en déduit :

$$(v_i|v_j) = \frac{1}{2}(\|v_i\|^2 + \|v_j\|^2 - \|v_i - v_j\|^2.$$

D'après les hypothèses sur la famille $(v_1, \ldots, v_n)$ on obtient donc :

$$\forall i \neq j, \quad (v_i|v_j) = \frac{1}{2}$$

et

$$\forall i, \quad (v_i|v_i) = 1.$$

Donc

$$G = \frac{1}{2}(J_n + I_n)$$

où J_n est la matrice dont tous les coefficients valent 1.

(c) On a donc $A = J_n + I_n$ et comme I_n et J_n commutent :

$$A^2 = J_n^2 + 2J_n + I_n^2 = J_n^2 + 2J_n + I_n.$$

Un simple calcul permet de voir que $J_n^2 = nJ_n$ de sorte que :

$$A^2 = nJ_n + 2J_n + I_n = (n+2)J_n + I_n = (n+2)(A - I_n) + I_n = (n+2)A - (n+1)I_n.$$

On en déduit :

$$A(A - (n+2)I_n) = -(n+1)I_n$$

ce qui permet de conclure que A est inversible d'inverse $-\dfrac{1}{n+1}(A - (n+2)I_n)$.

(d) Comme $A = 2G$ est inversible, alors G aussi et la question 3 permet de déduire que $(v_1, \dots, v_n)$ est une famille libre de $\mathbb{R}^n$.
Comme de plus $\text{Card}(v_1, \dots, v_n) = \dim(\mathbb{R}^n)$, c'est une base de $\mathbb{R}^n$.

Exercice sans préparation

1.
```
def marche(i,N):
    x0=i
    for k in range(N):
        if rd.rand()<0.5:
            x0 = x0+1
        else:
            x0=x0-1
    return x0
```

2.
```
def jeu(n,a,b,i):
    x0=i
    while x0 !=0 and x0 != n:
        if rd.rand()<0.5:
            x0 = x0+1
        else:
            x0=x0-1
        print(x0)
    if x0 ==0:
        return a
    else:
        return b
```

Agro-Veto – Planche 14

Cours

Énoncer le théorème d'intégration par parties pour les intégrales sur un segment.

Exercice préparé

Soit $n \in \mathbb{N}^*$. On dispose de n jetons et trois urnes numérotées de 1 à 3.
Pour chaque jeton, on choisit une des trois urnes au hasard et avec équiprobabilité et on place le jeton dans cette urne. Le placement de chaque jeton est indépendant du placement des autres jetons.
On note X la variable aléatoire égale au nombre de jetons contenus dans l'urne 1 à la fin de l'expérience et Y le nombre d'urne restées vides.

1. Écrire une fonction Python qui prend en argument un entier $n \in \mathbb{N}^*$, simule l'expérience aléatoire et renvoie les valeurs de X et Y obtenues.
2. Dans cette question $n = 10$. Utiliser la fonction précédente pour simuler un grand nombre de fois l'expérience et obtenir une valeur approchée de $E(XY)$, $E(X)$ et $E(Y)$.
   Que peut-on conjecturer sur la valeur de la covariance du couple (X, Y) ?
3. Dans cette question $n = 2$.
   (a) Déterminer $X(\Omega)$ et $Y(\Omega)$ puis donner la loi conjointe du couple (X, Y) sous forme de tableau.
   (b) Donner la loi de X puis celle de Y.
   (c) Calculer la covariance du couple (X, Y).
4. Dans cette question, on revient au cas général où n est un entier naturel non nul quelconque.
   Pour $i \in \{1, 2, 3\}$, on note Y_i la variable aléatoire qui vaut 1 si l'urne numéro i est vide à la fin de l'expérience et qui vaut 0 sinon.
   (a) Déterminer la loi de X, et donner la valeur de son espérance.
   (b) En remarquant que $Y = Y_1 + Y_2 + Y_3$, calculer $E(Y)$.
   (c) Démontrer : $\forall i \in \{2, 3\},\ \forall j \in [\![0, n]\!], \quad P([X = j] \cap [Y_i = 1]) = \binom{n}{j} \frac{1}{3^n}$.
   (d) Calculer alors $E(XY_i)$ pour $i \in \{2, 3\}$. Que vaut cette espérance si $i = 1$?
   (e) Calculer la covariance du couple (X, Y).

Agro-Veto – Planche 14

Exercice sans préparation

On souhaite exploiter le suivi d'une randonnée en effectuant des mesures lors de différentes points de passage : latitude, longitude, altitude et temps (date).
Ces données sont contenues dans une liste `coords`, où `coords[i]` désigne la liste des 4 mesures effectuées au point de passage numéro i. Ainsi `coords[i][2]` renvoie l'altitude relevée au point de passage numéro `i`.

1. Écrire une fonction `temps(coords)` qui renvoie la liste des temps relevés lors de la randonnée.
2. Sans utiliser la fonction `max`, écrire une fonction `plus_haut(coords)` qui renvoie la liste `[lat,long]` du point le plus haut.

Éléments de correction–Planche 14

Exercice avec préparation

Soit $n \in \mathbb{N}^*$. On dispose de n jetons et trois urnes numérotées de 1 à 3.
Pour chaque jeton, on choisit une des trois urnes au hasard et avec équiprobabilité et on place le jeton dans cette urne. Le placement de chaque jeton est indépendant du placement des autres jetons.
On note X la variable aléatoire égale au nombre de jetons contenus dans l'urne 1 à la fin de l'expérience et Y le nombre d'urne restées vides.

1.
```
import numpy.random as rd

def experience(n):
    L = [0,0,0]
    for k in range(n):
        L[rd.randint(0,3)]+=1
    Y = 0
    for elt in L:
        if elt ==0:
            Y+=1
    return L[0],Y
```

2. Dans cette question $n = 10$.

```
n = 10
N = 100000

X,Y,XY = 0,0,0
for k in range(N):
    L=experience(n)
    X+= L[0]
    Y+= L[1]
    XY += L[0]*L[1]
print('E(XY)='+str(XY/N))
print('E(X)='+str(X/N))
print('E(Y)='+str(Y/N))
print('Cov(X,Y)='+str(XY/N-X/N*Y/N))
```

   La loi des grands nombres assure que l'on obtient bien des valeurs approchées des espérances.
   On obtient : `Cov(X,Y)=0.0004177760000000086`.
   On conjecture que la covariance du couple (X, Y) est nulle.

3. Dans cette question $n = 2$.
   (a) Il y a deux jetons donc $X(\Omega) = [\![0, 2]\!]$ et $Y(\Omega) = [\![1, 2]\!]$.

$j \in Y(\Omega)$ \ $i \in X(\Omega)$	0	1	2
1	2/9	4/9	0
2	2/9	0	1/9

En effet, par exemple pour $[X=1]\cap[Y=1]$: l'urne 1 contient 1 jeton exactement et une urne est vide. Donc :

— soit le premier jeton est mis dans l'urne 1 et le deuxième dans l'urne 2, soit l'inverse ;

— soit le premier jeton est mis dans l'urne 1 et le deuxième dans l'urne 3, soit l'inverse.

Donc par indépendance des jetons :

$$P([X=1]\cap[Y=1]) = \frac{1}{3}\times\frac{1}{3}+\frac{1}{3}\times\frac{1}{3}+\frac{1}{3}\times\frac{1}{3}+\frac{1}{3}\times\frac{1}{3} = \frac{4}{9}.$$

(b) Avec la formule des probabilités totales appliquées avec le système complet d'événements $([Y=1],[Y=2])$:

$$\forall i\in X(\Omega),\quad P(X=i) = P([X=i]\cap[Y=1]) + P([X=i]\cap[Y=2]).$$

D'où :

$i\in X(\Omega)$	0	1	2
$P(X=i)$	4/9	4/9	1/9

De même :

$j\in Y(\Omega)$	1	2
$P(Y=j)$	2/3	1/3

(c) On a : $E(X)=2/3$, $E(Y)=4/3$ et $E(XY)=4/9+4/9=8/9$. Par la formule de Koenig-Huygens :

$$\mathrm{Cov}(X,Y) = E(XY)-E(X)E(Y) = 0.$$

4. Dans cette question, on revient au cas général où n est un entier naturel non nul quelconque. Pour $i\in\{1,2,3\}$, on note Y_i la variable aléatoire qui vaut 1 si l'urne numéro i est vide à la fin de l'expérience et qui vaut 0 sinon.

(a) On répète n fois indépendantes l'épreuve de Bernoulli dont le succès est « le jeton est mis dans l'urne 1 ». La variable X donne le nombre de succès donc X suit la loi binomiale de paramètre n et $\frac{1}{3}$. Son espérance vaut $n/3$.

(b) Les variables Y_i suivent des lois de Bernoulli. Si on répète n fois indépendantes l'épreuve de Bernoulli dont le succès est « le jeton est mis dans une urne différente de la i ». La variable Z_i donnant le nombre de succès suit la loi binomiale de paramètre n et $\frac{2}{3}$ et on a :

$$P(Y_i=1) = P(Z_i=n) = \left(\frac{2}{3}\right)^n.$$

Ainsi Y_i suit la loi de Bernoulli de paramètre $\left(\frac{2}{3}\right)^n$.

Par linéarité :

$$E(Y) = E(Y_1)+E(Y_2)+E(Y_3) = 3\left(\frac{2}{3}\right)^n.$$

(c) Soit $i \in \{2,3\}$ et $j \in [\![0,n]\!]$. L'événement $[X=j] \cap [Y_i = 1]$ signifie que l'urne 1 contient j jetons et l'urne i aucune (donc la troisième contient les autres) : il y a $\binom{n}{j}$ issues et chacune est de probabilité $\frac{1}{3^j} \times \frac{1}{3^{n-j}} = \frac{1}{3^n}$. Ainsi

$$\forall i \in \{2,3\},\ \forall j \in [\![0,n]\!], \quad P([X=j] \cap [Y_i=1]) = \binom{n}{j}\frac{1}{3^n}.$$

(d) Soit $i \in \{2,3\}$. Par le théorème de transfert :

$$\begin{aligned} E(XY_i) &= 0 \times \sum_{j=0}^{n} jP(X=j, Y_i=0) + 1 \times \sum_{j=0}^{n} jP(X=j, Y_i=1) \\ &= \sum_{j=0}^{n} \binom{n}{j}\frac{j}{3^n} \\ &= \frac{1}{3^n}\sum_{j=0}^{n} j\binom{n}{j} \\ &= \frac{n2^{n-1}}{3^n}. \end{aligned}$$

Si $Y_i = 1$ alors $X = 0$ donc $XY_i = 0$. Ainsi XY_i est la loi certain égale à 0.

(e) On a :

$$E(XY) = E(XY_1) + E(XY_2) + E(XY_3) = \frac{n2^n}{3^n}$$

et

$$E(X)E(Y) = \frac{n}{3} \times 3\left(\frac{2}{3}\right)^n = \frac{n2^n}{3^n}.$$

Par la formule de Koenig-Huygens :

$$\mathrm{Cov}(X,Y) = E(XY) - E(X)E(Y) = 0.$$

Exercice sans préparation

On souhaite exploiter le suivi d'une randonnée en effectuant des mesures lors de différentes points de passage : latitude, longitude, altitude et temps (date).
Ces données sont contenues dans une liste `coords`, où `coords[i]` désigne la liste des 4 mesures effectuées au point de passage numéro i. Ainsi `coords[i][2]` renvoie l'altitude relevée au point de passage numéro `i`.

1.
```
def temps(coords):
    L = []
    for elt in coords :
        L.append(elt[3])
    return L
```

2.
```
def plus_haut(coords):
    max = coords[0][2]
    [lat,long]=[coords[0][0],coords[0][1]]
    for elt in coords :
        if elt[2]>max:
            max = elt[2]
            [lat,long]=[elt[0],elt[1]]
    return [lat,long]
```

Agro-Veto – Planche 15

Cours

Donner la définition du gradient d'une fonction définie sur $\mathbb{R}^2$.

Exercice préparé

Soit $n \in \mathbb{N}^*$. On note $\mathbb{R}_n[X]$ l'espace vectoriel des fonctions polynomiales de degré inférieur ou égal à n.
On définit sur $\mathbb{R}_n[X]$ l'application D par :

$$\forall P \in \mathbb{R}_n[X], \quad D(P) = P(X+1) - P(X).$$

1. (a) Montrer que D est un endomorphisme de $\mathbb{R}_n[X]$.
   (b) Déterminer $D(1)$ puis $D(X^k)$ pour tout $k \in [\![1, n]\!]$.
   (c) Donner la matrice de D dans la base canonique de $\mathbb{R}_n[X]$.
   (d) Déterminer le spectre de D. D est-il diagonalisable ?
2. On pose $H_0(X) = 1$ et pour tout $k \in \mathbb{N}^*$, $H_k(X) = \prod_{i=0}^{k-1}(X - i)$.
   (a) Montrer que $\mathcal{B} = (H_0, \dots, H_n)$ est une base de $\mathbb{R}_n[X]$.
   (b) Calculer $D(H_0)$. Montrer que pour tout $k \in [\![1, n]\!]$, $D(H_k) = kH_{k-1}$.
   (c) Déterminer la matrice représentative de D dans la base $\mathcal{B}$.
3. En Python, un polynôme de $\mathbb{R}_n[X]$ est codé en listant ses $n+1$ coefficients par ordre croissant de degré. Par exemple, dans $\mathbb{R}_4[X]$ le polynôme $P = 5X^3 - 2X + 3$ est représenté par la liste `[3,-2,0,5,0]`.
   (a) Programmer une fonction Python qui prend en arguments une liste de longueur $n+1$ modélisant un polynôme $P \in \mathbb{R}_n[X]$ et un réel a et qui renvoie la liste modélisant $(X - a)P \in \mathbb{R}_{n+1}[X]$.
   (b) Programmer une fonction Python qui prend en argument $n \in \mathbb{N}^*$ et renvoie la liste modélisant H_n.
4. Soit Y une variable aléatoire suivant une loi de Poisson de paramètre $\lambda > 0$.
   (a) Montrer que $H_2(Y)$ possède une espérance et donner sa valeur.
   (b) Déterminer les coordonnées de 1, X et X^2 dans la base (H_0, H_1, H_2).
   (c) Retrouver la valeur de la variance de Y.
5. On note $\mathcal{C}$ l'espace vectoriel des fonctions continues de $\mathbb{R}$ dans $\mathbb{R}$.
   À tout élément $f \in E$, on associe la fonction $g = \tilde{D}(f)$ définie par :

$$\forall x \in \mathbb{R}, \quad g(x) = \tilde{D}(f)(x) = f(x+1) - f(x).$$

6. (a) On dit qu'un réel λ est une valeur propre de $\tilde{D}$ s'il existe une fonction non nulle f de E telle que $\tilde{D}(f) = \lambda f$.
   En considérant les fonctions $h_a : x \mapsto e^{ax}$ et $k_a : x \mapsto \sin(\pi x)e^{ax}$ où a est un réel, déterminer les valeurs propres de $\tilde{D}$.
   (b) Si F désigne la fonction de répartition d'une variable aléatoire à densité X, montrer que $g = \tilde{D}(F)$ est une densité de probabilité.
   (c) Expliciter g si X suit la loi uniforme sur $[0, 1]$.

Agro-Veto – Planche 15

Exercice sans préparation

1. Écrire une fonction Python qui simule une série de N lancers d'une pièce équilibrée et qui renvoie la liste des résultats de ces lancers (Pile est codé par 1 et Face par 0).
2. Écrire une fonction Python qui simule une série de lancers d'une pièce équilibrée jusqu'à l'obtention de la configuration Pile – Pile – Face et renvoie le nombre de lancers nécessaires à l'apparition de cette configuration.

   Évaluer le temps moyenne d'attente de cette configuration.

Éléments de correction–Planche 15

Exercice avec préparation

Soit $n \in \mathbb{N}^*$. On note $\mathbb{R}_n[X]$ l'espace vectoriel des fonctions polynomiales de degré inférieur ou égal à n.
On définit sur $\mathbb{R}_n[X]$ l'application D par :

$$\forall P \in \mathbb{R}_n[X], \quad D(P) = P(X+1) - P(X).$$

1. (a) Montrons que D est une application linéaire. Soit $(P,Q) \in \mathbb{R}_n[X]$ et soit $\lambda \in \mathbb{R}$. On a alors :

$$\begin{aligned} D(P+\lambda Q) &= (P+\lambda Q)(X+1) - (P+\lambda Q)(X) \\ &= P(X+1) + \lambda Q(X+1) - (P(X) + \lambda Q(X)) \\ &= P(X+1) - P(X) + \lambda(Q(X) - Q(X+1)) \\ &= D(P) + \lambda D(Q). \end{aligned}$$

Ainsi : $\forall (P,Q) \in \mathbb{R}_n[X]^2\ \forall \lambda \in \mathbb{R},\ D(P+\lambda Q) = D(P) + \lambda D(Q)$. Donc D est linéaire.
Par ailleurs il est clair que $\deg(P(X+1)) = \deg(P)$ donc

$$\deg(D(P)) \leq \max(\deg(P), \deg(P(X+1)) \leq n.$$

L'application D est une application linéaire définie sur $\mathbb{R}_n[X]$ à valeurs dans $\mathbb{R}_n[X]$, c'est donc un endomorphisme de $\mathbb{R}_n[X]$.

(b) Soit $k \in [\![1, n]\!]$. D'après la formule du binôme de Newton on a :

$$(X+1)^k = \sum_{i=0}^{k} \binom{k}{i} X^i$$

et

$$D(X^k) = (X+1)^k - X^k = \sum_{i=0}^{k-1} \binom{k}{i} X^i.$$

De plus,

$$D(1)(X) = 1 - 1 = 0.$$

(c) On a vu que pour tout $k \in [\![1, n]\!]$, on a

$$D(X^k) = (X+1)^k - X^k = \sum_{i=0}^{k-1} \binom{k}{i} X^i.$$

Ainsi la matrice recherchée est :

$$\begin{pmatrix} 0 & 1 & 1 & \cdots & 1 \\ 0 & 0 & \binom{2}{1} & \cdots & \binom{n}{1} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \binom{n}{n-1} \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

(d) La matrice représentative de D dans la base canonique est triangulaire donc ses valeurs propres, qui sont aussi celles de D, sont égales à ses coefficients diagonaux. D'où :

$$\mathrm{Sp}(D) = \{0\}.$$

Si D était diagonalisable, ce serait donc l'endomorphisme nul. Ce n'est manifestement pas le cas donc D n'est pas diagonalisable.

2. On pose $H_0(X) = 1$ et pour tout $k \in \mathbb{N}^*$, $H_k(X) = \prod_{i=0}^{k-1}(X-i)$.

(a) Pour tout $i \in [\![0, n]\!]$, $\deg(H_i) = i$. La famille $\mathcal{B}$ est une famille échelonnée de $n+1$ polynômes non nuls de $\mathbb{R}_n[X]$. Comme $\dim(\mathbb{R}_n[X]) = n+1$, c'est donc une base de $\mathbb{R}_n[X]$

(b) On a : $D(H_0) = 0$ par la question **1.(b)**.
Soit i un entier entre 2 et n. On a

$$\begin{aligned}
D(H_i)(X) = H_i(X+1) - H_i(X) &= \prod_{k=0}^{i-1}(X+1-k) - \prod_{k=0}^{i-1}(X-k) \\
&= \prod_{k=0}^{i-1}(X-(k-1)) - \prod_{k=0}^{i-1}(X-k) \\
&= (X+1)\prod_{k=1}^{i-1}(X-(k-1)) - \prod_{k=0}^{i-1}(X-k) \\
&= (X+1)\prod_{k=0}^{i-2}(X-k) - \prod_{k=0}^{i-1}(X-k) \\
&= (X+1-(X-(i-1)))\prod_{k=1}^{i-2}(X-k) \\
&= i\prod_{k=1}^{i-2}(X-k) \\
&= iH_{i-1}(X).
\end{aligned}$$

De plus, $D(H_1) = D(X) = 1 = H_0$.

(c) D'après la question précédente, la matrice recherchée est :

$$\begin{pmatrix}
0 & 1 & 0 & \cdots & 0 \\
0 & 0 & 2 & \cdots & 0 \\
\vdots & \vdots & \ddots & \ddots & \vdots \\
0 & 0 & 0 & 0 & n \\
0 & 0 & 0 & 0 & 0
\end{pmatrix}.$$

3. En Python, un polynôme de $\mathbb{R}_n[X]$ est codé en listant ses $n+1$ coefficients par ordre croissant de degré. Par exemple, dans $\mathbb{R}_4[X]$ le polynôme $P = 5X^3 - 2X + 3$ est représenté par la liste `[3,-2,0,5,0]`.

(a) Soit $P = \sum_{k=0}^{n} a_k X^k$. Alors :

$$\begin{aligned}(X-a)P &= \sum_{k=0}^{n} a_k X^{k+1} - a\sum_{k=0}^{n} a_k X^k \\ &= \sum_{k=1}^{n+1} a_{k-1} X^k - a\sum_{k=0}^{n} a_k X^k \\ &= a_n X^{n+1} + \sum_{k=1}^{n}(a_{k-1} - aa_k)X^k - aa_0.\end{aligned}$$

On en déduit le programme :

```
def poly(L,a):
        Lnew = [0 for k in range(len(L)+1)]
        Lnew[0]=-a*L[0]
        Lnew[len(L)]=L[len(L)-1]
        for k in range(1,len(L)):
                Lnew[k]=L[k-1]-a*L[k]
        return Lnew
```

(b) On utilise le fait que $H_{n+1} = (X-n)H_n$ pour implémenter un programme récursif.

```
def H(n):
        if n == 0:
                return [1]
        else :
                return poly(H(n-1),n)
```

4. Soit Y une variable aléatoire suivant une loi de Poisson de paramètre $\lambda > 0$.

(a) On a $H_2 = \frac{1}{2}X(X-1)$. D'après le théorème de transfert $H_2(Y)$ possède une espérance si et seulement si la série $\sum_{k\geq 0} \frac{k(k-1)}{2}\frac{e^{-\lambda}\lambda^k}{k!}$ est absolument convergente.

Comme la série est à termes positifs, la convergence absolue et la convergence sont équivalentes.

Soit $n \in \mathbb{N}$. On a

$$\begin{aligned}\sum_{k=0}^{n} \frac{k(k-1)}{2}\frac{e^{-\lambda}\lambda^k}{k!} = \sum_{k=2}^{n} \frac{k(k-1)}{2}\frac{e^{-\lambda}\lambda^k}{k!} &= \frac{e^{-\lambda}}{2}\sum_{k=2}^{n}\frac{\lambda^k}{(k-2)!} \\ &= \frac{e^{-\lambda}}{2}\sum_{\ell=0}^{n-2}\frac{\lambda^{\ell+2}}{\ell!} \\ &= \frac{\lambda^2 e^{-\lambda}}{2}\sum_{\ell=0}^{n-2}\frac{\lambda^\ell}{\ell!}.\end{aligned}$$

On reconnaît une somme partielle d'une série exponentielle de paramètre λ convergente. Ainsi la suite des sommes partielles de $\sum_{k\geq 0} \frac{k(k-1)}{2}\frac{e^{-\lambda}\lambda^k}{k!}$ est convergente donc $\sum_{k\geq 0} \frac{k(k-1)}{2}\frac{e^{-\lambda}\lambda^k}{k!}$ est (absolument) convergente.

La variable $H_2(Y)$ possède donc une espérance et :

$$\mathbb{E}(H_2(Y)) = \frac{\lambda^2 e^{-\lambda}}{2} \sum_{\ell=0}^{+\infty} \frac{\lambda^\ell}{\ell!} = \frac{\lambda^2}{2}.$$

(b) Comme $H_0 = 1$ les coordonnées de 1 dans la base (H_0, H_1, H_2) sont $(1, 0, 0)$.
Comme $H_1 = X$ les coordonnées de X dans la base (H_0, H_1, H_2) sont $(0, 1, 0)$.
Comme $H_2 = \frac{1}{2}X(X-1)$ les coordonnées de X^2 dans la base (H_0, H_1, H_2) sont $(0, 1, 2)$.

(c) On a par la formule de Koenig-Huygens :

$$\mathbb{V}(Y) = \mathbb{E}(Y^2) - \mathbb{E}(Y)^2 = \mathbb{E}(Y + 2H_2(Y)) - E(Y)^2 = \lambda + 2\frac{\lambda^2}{2} - \lambda^2 = \lambda.$$

5. On note $\mathcal{C}$ l'espace vectoriel des fonctions continues de $\mathbb{R}$ dans $\mathbb{R}$.
À tout élément $f \in E$, on associe la fonction $g = \tilde{D}(f)$ définie par :

$$\forall x \in \mathbb{R}, \quad g(x) = \tilde{D}(f)(x) = f(x+1) - f(x).$$

(a) Pour tout $x \in \mathbb{R}$, on a :

$$\tilde{D}(h_a)(x) = h_a(x+1) - h_a(x) = e^{ax+a} - e^{ax} = (e^a - 1)e^{ax} = (e^a - 1)h_a(x).$$

Ainsi pour tout $a \in \mathbb{R}$, $e^a - 1$ est une valeur propre de $\tilde{D}$. D'où

$$]-1, +\infty[= \{e^a - 1 \ ; \ a \in \mathbb{R}\} \subset \mathrm{Sp}(\tilde{D}).$$

De même, pour tout $x \in \mathbb{R}$:

$$\begin{aligned} \tilde{D}(k_a)(x) = k_a(x+1) - k_a(x) &= e^{ax+a}\sin(\pi x + \pi) - e^{ax}\sin(\pi x) \\ &= -e^{ax+a}\sin(\pi x) - e^{ax}\sin(\pi x) \\ &= -(e^a + 1)k_a(x). \end{aligned}$$

Ainsi pour tout $a \in \mathbb{R}$, $-(e^a + 1)$ est une valeur propre de $\tilde{D}$. D'où

$$]-\infty, -1[= \{-(e^a + 1) \ ; \ a \in \mathbb{R}\} \subset \mathrm{Sp}(\tilde{D}).$$

Donc on a :

$$\mathbb{R}\backslash\{-1\} \subset \mathrm{Sp}(\tilde{D}).$$

Il ne reste plus qu'à déterminer si -1 est une valeur propre. Soit $f \in \mathcal{C}$:

$$\tilde{D}(f) = -f \Rightarrow \forall x \in \mathbb{R}, \ f(x+1) - f(x) = -f(x) \Rightarrow \forall x \in \mathbb{R}, \ f(x+1) = 0 \Rightarrow f = 0.$$

Donc -1 n'est pas valeur propre et finalement

$$\mathbb{R}\backslash\{-1\} = \mathrm{Sp}(\tilde{D}).$$

(b) Soit F la fonction de répartition d'une variable aléatoire à densité X et posons $g = \tilde{D}(F)$.
Comme F est continue et croissante alors g est continue et positive.

Soit $A > 0$. Alors :

$$\begin{aligned}\int_0^A g(t)dt &= \int_0^A (F(t+1) - F(t))dt \\ &= \int_0^A F(t+1)dt - \int_0^A F(t)dt \\ &= \int_1^{A+1} F(s)ds - \int_0^A F(t)dt \quad (s = t+1) \\ &= \int_1^A F(s)ds + \int_A^{A+1} F(t)dt - \int_0^1 F(t)dt - \int_1^A F(t)dt \quad \text{(Chasles)} \\ &= \int_A^{A+1} F(t)dt - \int_0^1 F(t)dt.\end{aligned}$$

Or, comme F est une fonction de répartition : $\lim\limits_{x\to+\infty} F(x) = 1$.

Soit $\varepsilon > 0$, il existe donc $A_\varepsilon > 0$ tel que :

$$\forall x \geq A_\varepsilon, \quad |F(x) - 1| \leq \varepsilon.$$

Ainsi, pour tout $A \geq A_\varepsilon$, on a :

$$\begin{aligned}\left|\int_A^{A+1} F(t)dt - 1\right| &= \left|\int_A^{A+1} (F(t) - 1)dt\right| \\ &\leq \int_A^{A+1} |F(t) - 1|dt \\ &\leq \varepsilon.\end{aligned}$$

Ainsi :

$$\forall \varepsilon > 0 \ \exists A_\varepsilon \ \forall A > 0 \quad A \geq A_\varepsilon \Rightarrow \left|\int_A^{A+1} F(t)dt - 1\right| \leq \varepsilon.$$

Cela signifie que :

$$\lim_{A\to+\infty} \int_A^{A+1} F(t)dt = 1.$$

En particulier, $\displaystyle\int_0^{+\infty} g(t)dt$ converge et vaut :

$$1 - \int_0^1 F(t)dt.$$

De même soit $A < 0$. Alors :

$$\begin{aligned}\int_A^0 g(t)dt &= \int_A^0 (F(t+1) - F(t))dt \\ &= \int_A^0 F(t+1)dt - \int_A^0 F(t)dt \\ &= \int_{A+1}^1 F(s)ds - \int_A^0 F(t)dt \quad (s = t+1) \\ &= \int_0^1 F(s)ds + \int_{A+1}^A F(t)dt\end{aligned}$$

Or, comme F est une fonction de répartition : $\lim_{x\to-\infty} F(x) = 0$.

Soit $\varepsilon > 0$, il existe donc $A_\varepsilon < 0$ tel que :

$$\forall x \leq A_\varepsilon, \quad |F(x)| \leq \varepsilon.$$

Ainsi, pour tout $A \leq A_\varepsilon$, on a :

$$\left|\int_{A+1}^{A} F(t)dt\right| \leq \int_{A}^{A+1} |F(t)|dt \leq \varepsilon.$$

Ainsi :

$$\forall \varepsilon > 0 \ \exists A_\varepsilon \ \forall A < 0 \quad A \leq A_\varepsilon \Rightarrow \left|\int_{A}^{A+1} F(t)dt\right| \leq \varepsilon.$$

Cela signifie que :

$$\lim_{A\to -infty} \int_{A+1}^{A} F(t)dt = 0.$$

En particulier, $\displaystyle\int_{-\infty}^{0} g(t)dt$ converge et vaut :

$$\int_{0}^{1} F(t)dt.$$

Finalement, l'intégrale $\displaystyle\int_{-\infty}^{+\infty} g(t)dt$ converge et vaut :

$$1 - \int_{0}^{1} F(t)dt + \int_{0}^{1} F(t)dt = 1.$$

Cela finit de prouver que g est une densité de probabilité.

(c) Si X suit la loi uniforme sur $[0,1]$ alors

$$\forall x \in \mathbb{R}, F(x) = \begin{cases} 0 & \text{si } x < 0 \\ x & \text{si } x \in [0,1] \\ 1 & \text{si } x > 1 \end{cases}$$

Donc pour $x \in \mathbb{R}$:

$$\begin{aligned} g(x) = F(x+1) - F(x) &= \begin{cases} 0 & \text{si } x+1 < 0 \\ x+1 & \text{si } x+1 \in [0,1] \\ 1 & \text{si } x+1 > 1 \end{cases} - \begin{cases} 0 & \text{si } x < 0 \\ x & \text{si } x \in [0,1] \\ 1 & \text{si } x > 1 \end{cases} \\ &= \begin{cases} 0 & \text{si } x < -1 \\ x+1 & \text{si } x \in [-1,0] \\ 1 & \text{si } x > 0 \end{cases} - \begin{cases} 0 & \text{si } x < 0 \\ x & \text{si } x \in [0,1] \\ 1 & \text{si } x > 1 \end{cases} \\ &= \begin{cases} 0 & \text{si } x < -1 \\ x+1 & \text{si } x \in [-1,0] \\ 1-x & \text{si } x \in]0,1] \\ 0 & \text{si } x > 1 \end{cases} \end{aligned}$$

Exercice sans préparation

1.
```
def simulation(N):
    L = []
    for k in range(N):
        if rd.rand()<1/2:
            L.append(1)
        else:
            L.append(0)
    return N
```

2.
```
def piece ():
    if rd.rand()<1/2:
        return 1
    else:
        return 0

def attente():
    s1 = piece()
    s2 = piece()
    s3 = piece()
    n=3
    while s1!= 1 or s2!=1 or s3!=0:
        s1 = s2
        s2 = s3
        s3 = piece()
        n +=1
    return n

N = 10000
moy = 0
for k in range(N):
    moy+=attente()
print(moy/N)
```

C'est la loi des grands nombres qui justifie que la valeur affichée est une valeur approchée du temps moyen d'attente.

On trouve environ 8.

Agro-Veto – Planche 16

Cours

Énoncer le théorème de transfert dans le cas d'une variable aléatoire admettant une densité.

Exercice préparé

Pour tout $n \geqslant 1$, on considère la matrice $K_n \in \mathcal{M}_{n+1}(\mathbb{R})$ telle que pour tout $i \in [\![1, n]\!]$, $(K_n)_{i,i+1} = i$, pour tout $j \in [\![1, n]\!]$, $(K_n)_{j+1,j} = -n-1+j$ et dont tous les autres coefficients sont nuls. On a donc :

$$K_1 = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \quad \text{et} \quad K_2 = \begin{pmatrix} 0 & 1 & 0 \\ -2 & 0 & 2 \\ 0 & -1 & 0 \end{pmatrix}$$

1. Déterminer les valeurs propres et les vecteurs propres de K_1. Cette matrice est-elle diagonalisable sur $\mathbb{R}$? Sur $\mathbb{C}$?
2. Écrire une fonction `K` en langage Python qui prend en entrée un entier n et qui renvoie la matrice K_n.
3. Utiliser la fonction `K` et la fonction `eigvals` du module `numpy.linalg` pour déterminer les valeurs propres de K_n pour $n \in [\![1, 10]\!]$. Que peut-on conjecturer ?
4. On se propose de montrer la conjecture faite dans la question précédente. On note $\mathcal{F}(\mathbb{R}, \mathbb{C})$ l'espace vectoriel des fonctions de $\mathbb{R}$ vers $\mathbb{C}$ et V_n le $\mathbb{C}$-sous-espace vectoriel engendré par la famille de fonctions $\mathcal{B}_n = (f_k)_{k \in [\![0,n]\!]}$ définies par

$$\forall x \in \mathbb{R},\ f_k(x) = \cos^{n-k}(x)\sin^k(x).$$

   On considère l'application φ_n définie pour tout $f \in V_n$ par $\varphi_n(f) = f'$.

   (a) Soient $(\lambda_0, \dots, \lambda_n) \in \mathbb{C}^{n+1}$ et $x \in]-\pi/2, \pi/2[$. Montrer que

$$\lambda_0 f_0(x) + \cdots + \lambda_n f_n(x) = 0 \iff \lambda_0 + \lambda_1 \tan(x) + \cdots + \lambda_n \tan(x)^n = 0.$$

   (b) En déduire que la famille $\mathcal{B}_n$ est une base de V_n et la dimension de V_n.

   (c) Monter que φ_n est un endomorphisme de V_n et déterminer sa matrice dans la base $\mathcal{B}_n$.

   (d) Pour tout $k \in [\![0, n]\!]$ on note g_k la fonction définie par

$$\forall x \in \mathbb{R}, \quad g_k(x) = \exp(i(n-2k)x).$$

   Justifier que pour tout $x \in \mathbb{R}, g_k(x) = (\cos(x) + i\sin(x))^{n-k}(\cos(x) - i\sin(x))^k$.

   (e) En déduire que pour tout $k \in [\![0, n]\!]$, g_k appartient à V_n.

   On pourra utiliser sans le justifier que $\left(\sum_{j=0}^{n-k} a_j\right).\left(\sum_{l=0}^{k} b_l\right) = \sum_{j=0}^{n-k}\sum_{l=0}^{k} a_j.b_l.$

   (f) En déduire les valeurs propres de φ_n puis celle de K_n.

   (g) La matrice K_n est-elle diagonalisable sur $\mathbb{C}$?

   (h) Déterminer pour quelle valeur de n, la matrice K_n est inversible.

   (i) Lorsque K_n n'est pas inversible, déterminer une base du noyau.

Exercice sans préparation

1. Écrire une fonction `gene(n)` qui prend en entrée un entier naturel non nul n et retourne une chaîne de n caractères formée aléatoirement, de façon équiprobable, des caractères `'A','C','G','T'`.
2. Écrire une fonction `nbAC(g)` qui prend en entrée une chaine de caractères formée des caractères `'A','C','G','T'` et qui retourne le nombre de fois où la séquence `'AC'` est présente.

   Par exemple, `nbAC('GAGCACCCTACTTGGCGCGA')` retournera 2.

Éléments de correction–Planche 16

Exercice avec préparation

1. Soit $\lambda \in \mathbb{C}$. C'est valeur propre de K_1 si et seulement si $\det(K_1 - \lambda I_2) = 0$.
   Or
   $$K_1 - \lambda I_2 = \begin{pmatrix} -\lambda & 1 \\ -1 & -\lambda \end{pmatrix}$$
   et $\det(K_1 - \lambda I_2) = \lambda^2 + 1$ donc
   $$\det(K_1 - \lambda I_2) = 0 \iff \lambda^2 = -1 \iff \lambda = \pm i.$$
   Les valeurs propres de K_1 dans $\mathbb{C}$ sont i et $-i$: $\text{Spec}(K_1) = \{-i, i\}$.
   K_1 admettant deux valeurs propres distinctes dans $\mathbb{C}$ et K_1 étant carrée d'ordre 2, on peut dire que K_1 est diagonalisable sur $\mathbb{C}$.
   En revanche, l'équation $\lambda^2+1 = 0$ d'inconnue λ n'admettant aucune solution réelle, K_1 n'admet aucune valeur propre dans $\mathbb{R}$: K_1 n'est donc pas diagonalisable sur $\mathbb{R}$.
   Recherchons des sous-espaces propres $E_i(K_1)$ et $E_{-i}(K_1)$ associés respectivement aux valeurs propres i et $-i$.
   Soit $X = \begin{pmatrix} x \\ y \end{pmatrix} \in \mathcal{M}_{2,1}(\mathbb{C})$.
   - $X \in E_i(K_1) \iff (K_1 - iI_2)X = 0 \iff \begin{cases} -ix + y = 0 \\ -x - iy = 0 \end{cases} \iff y = ix.$
     D'où $E_i(K_1) = \text{Vect}\left(\begin{pmatrix} 1 \\ i \end{pmatrix}\right)$.
   - $X \in E_{-i}(K_1) \iff (K_1 + iI_2)X = 0 \iff \begin{cases} ix + y = 0 \\ -x + iy = 0 \end{cases} \iff y = -ix.$
     D'où $E_{-i}(K_1) = \text{Vect}\left(\begin{pmatrix} 1 \\ -i \end{pmatrix}\right)$.
2. Fonction `K` qui renvoie la matrice K_n :
   ```
   import numpy as np
   import numpy.linalg as la

   # attention au décalage d'indice pour les tableaux numpy
   def K(n):
       Kn = np.zeros([n+1,n+1])
       for i in range(n):
           Kn[i,i+1] = i+1
       for j in range(n):
           Kn[j+1,j] = -n-1+(j+1)
       return Kn
   ```
3. Rappel : `eigvals(M)` affiche la liste des valeurs propres de M.
   On propose le code suivant :
   ```
   for n in range(1,11):
       print("valeurs␣propres␣de␣K" + str(n)+ ":")
       print(la.eigvals(K(n)),'\n')
   ```

Rappel :

— Un nombre complexe $z = a + ib$ se note `a+bj` en langage Python.

— Il y a des arrondis à faire : par exemple `1.08420217e-19` doit être considéré comme nul dans ce contexte.

On peut alors conjecturer que :

— si n est pair : $\mathrm{Spec}(K_n) = \{-ni, (-n+2)i, \dots, -2i, 0, 2i, 4i, \dots, (n-2)i, ni\}$

— si n est impair : $\mathrm{Spec}(K_n) = \{-ni, (-n+2)i, \dots, -3i, -i, i, 3i, \dots, (n-2)i, ni\}$

Dans les deux cas, le spectre de K_n peut s'écrire sous la forme :

$$\mathrm{Spec}(K_n) = \{(n-2k)i,\ k \in [\![0, n]\!]\}\,.$$

4. (a) Soient $(\lambda_0, \dots, \lambda_n) \in \mathbb{C}^{n+1}$ et $x \in]-\pi/2, \pi/2[$.
La fonction cos ne s'annule pas sur l'intervalle ouvert $]-\pi/2, \pi/2[$ donc $\cos(x) \neq 0$.

$$\begin{aligned}
\lambda_0 f_0(x) + \cdots + \lambda_n f_n(x) = 0 &\iff \sum_{k=0}^{n} \lambda_k \cos^{n-k}(x)\sin^k(x) = 0 \\
&\iff \sum_{k=0}^{n} \lambda_k \frac{\cos^{n-k}(x)\sin^k(x)}{\cos^n(x)} = 0 \quad \text{en divisant par } \cos^n(x) \neq 0 \\
&\iff \sum_{k=0}^{n} \lambda_k \frac{\sin^k(x)}{\cos^k(x)} = 0 \\
&\iff \sum_{k=0}^{n} \lambda_k \tan^k(x) = 0
\end{aligned}$$

Ainsi pour tout $x \in]-\pi/2, \pi/2[$ on a l'équivalence suivante :

$$\lambda_0 f_0(x) + \cdots + \lambda_n f_n(x) = 0 \iff \lambda_0 + \lambda_1 \tan(x) + \cdots + \lambda_n \tan^n(x) = 0.$$

(b) Montrons que $\mathcal{B}_n$ est une base de V_n :

— La famille $\mathcal{B}_n = (f_k)_{k \in [\![0,n]\!]}$ engendre V_n par définition.

— Reste à démontrer que cette famille $\mathcal{B}_n$ est libre : soit $(\lambda_0, \lambda_1, \dots, \lambda_n) \in \mathbb{C}^{n+1}$ tel que

$$\underbrace{\lambda_0 f_0 + \lambda_1 f_1 + \cdots + \lambda_n f_n = 0}_{(*)}$$

(fonction nulle).
La relation $(*)$ signifie que $\lambda_0 f_0(x) + \lambda_1 f_1(x) + \cdots + \lambda_n f_n(x) = 0$ pour tout réel x.
En particulier, pour tout $x \in]-\pi/2, \pi/2[$, on a

$$\lambda_0 f_0(x) + \lambda_1 f_1(x) + \cdots + \lambda_n f_n(x) = 0.$$

D'après la question précédente, on a donc, pour tout réel $x \in]-\pi/2, \pi/2[$:

$$\lambda_0 + \lambda_1 \tan(x) + \cdots + \lambda_n \tan^n(x) = 0.$$

Considérons alors le polynôme $P(X) = \lambda_0 + \lambda_1 X + \cdots + \lambda_n X^n$. On vient d'établir que pour tout réel $x \in]-\pi/2, \pi/2[$, $P(\tan(x)) = 0$. Or $\tan(]-\pi/2, \pi/2[) = \mathbb{R}$. Ainsi le polynôme P admet une infinité de racines. Il s'agit donc du polynôme nul : $P(X) = 0$.

Or, un polynôme est nul si, et seulement si, tous ses coefficients sont nuls.
Ainsi $\lambda_0 = \lambda_1 = \ldots = \lambda_n = 0$.
On a donc montré que $\lambda_0 f_0 + \lambda_1 f_1 + \cdots + \lambda_n f_n = 0$ implique $\lambda_0 = \lambda_1 = \ldots = \lambda_n = 0$.
Ainsi, la famille $\mathcal{B}_n$ est libre.

Conclusion : $\mathcal{B}_n$ est une base de V_n et $\dim(V_n) = n+1$.

(c) Montrons que l'application $\varphi_n : f \mapsto f'$ est un endomorphisme de V_n.

— La linéarité de φ_n résulte de la linéarité de la dérivation :
$\forall (\lambda, \mu) \in \mathbb{C}^2, \forall (f,g) \in V_n^2 : \quad (\lambda f + \mu g)' = \lambda f' + \mu g'$.

— Reste à prouver que pour tout $f \in V_n$, $\varphi_n(f) \in V_n$.
Soit $f \in V_n$; alors f s'écrit comme combinaison linéaire des vecteurs $f_0, f_1, \ldots, f_n$ car la famille $\mathcal{B}_n$ engendre V_n.
La fonction φ_n étant linéaire, $\varphi_n(f)$ s'écrit alors comme combinaison linéaire des $\varphi_n(f_k)$, $k \in [\![0,n]\!]$.
Pour montrer que $\varphi_n(f)$ appartient à V_n, il suffit donc de montrer que pour tout $k \in [\![0,n]\!]$, $\varphi_n(f_k) \in V_n$.

— Pour $k = 0$, $f_0(x) = \cos^n$ et $f_0' = -n f_1$.

— Pour $k \in [\![1, n-1]\!]$: $f_k = \cos^{n-k} \sin^k$ donc

$$\begin{aligned} f_k' &= -(n-k)\cos^{n-k-1}\sin^{k+1} + k\cos^{n-k+1}\sin^{k-1} \\ &= -(n-k) f_{k+1} + k f_{k-1}. \end{aligned}$$

— Pour $k = n$, $f_n = \sin^n$ et $f_n' = n\cos\sin^{n-1} = n f_{n-1}$.

On constate que pour tout $k \in [\![0,n]\!]$, on a bien $\varphi_n(f_k) \in V_n$.
Conclusion : $\varphi_n : f \mapsto f'$ est un endomorphisme de V_n.

D'après les calculs précédents,

$$\mathrm{Mat}_{B_n}(\varphi_n) = \begin{pmatrix} 0 & 1 & 0 & \ldots & \ldots & 0 \\ -n & 0 & 2 & 0 & & \vdots \\ 0 & -n+1 & 0 & 3 & \ddots & \vdots \\ \vdots & 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & -2 & 0 & n \\ 0 & \ldots & \ldots & 0 & -1 & 0 \end{pmatrix}.$$

On reconnaît la matrice K_n. Ainsi, $\mathrm{Mat}_{B_n}(\varphi_n) = K_n$.

(d) Pour tout $k \in [\![0,n]\!]$, $\forall x$, $g_k(x) = \exp(i(n-2k)x) = e^{i(n-2k)x}$.
Par ailleurs, pour tout x :

$$\begin{aligned} (\cos(x) + i\sin(x))^{n-k}(\cos(x) - i\sin(x))^k &= (e^{ix})^{n-k} \times (e^{-ix})^k \\ &= e^{i(n-k)x} \times e^{-ikx} \\ &= e^{i(n-2k)x}. \end{aligned}$$

Ainsi on a bien, pour tout $x \in \mathbb{R}$

$$g_k(x) = (\cos(x) + i\sin(x))^{n-k}(\cos(x) - i\sin(x))^k.$$

(e) En utilisant la formule du binôme, pour tout x

$$\begin{aligned} g_k(x) &= (\cos(x) + i\sin(x))^{n-k}(\cos(x) - i\sin(x))^k \\ &= \left(\sum_{j=0}^{n-k} \binom{n-k}{j} i^j \sin^j(x)\cos^{n-k-j}(x)\right)\left(\sum_{l=0}^{k} \binom{k}{l}(-i)^l \sin^l(x)\cos^{k-l}(x)\right) \\ &= \sum_{j=0}^{n-k}\sum_{l=0}^{k} \binom{n-k}{j}\binom{k}{l}(-1)^l i^{j+l}\cos^{n-j-l}(x)\sin^{j+l}(x) \end{aligned}$$

Pour $0 \leq j \leq n-k$ et $0 \leq l \leq k$ on a $0 \leq j+l \leq n$ donc $j+l \in [\![0,n]\!]$ ce qui permet d'écrire $\cos^{n-(j+l)}(x)\sin^{j+l}(x) = f_{j+l}(x)$.

Ainsi,

$$\forall x, \quad g_k(x) = \sum_{j=0}^{n-k}\sum_{l=0}^{k} \binom{n-k}{j}\binom{k}{l}(-1)^l i^{j+l} f_{j+l}(x)$$

et donc la fonction g_k est combinaison linéaire des fonction $f_0, f_1, \ldots, f_n$.

On en déduit que pour tout $k \in [\![0,n]\!]$, g_k appartient à V_n.

(f) Soit $k \in [\![0,n]\!]$. $\forall x$, $g_k(x) = e^{i(n-2k)x}$ donc

$$\forall x, \quad g_k'(x) = i(n-2k)e^{i(n-2k)x}$$

et comme $g_k \in V_n$, cela s'écrit

$$\varphi_n(g_k) = i(n-2k)g_k.$$

Ainsi, pour tout $k \in [\![0,n]\!]$, $\varphi_n(g_k) = i(n-2k)g_k$ avec $g_k \neq 0$.

Cela prouve que les nombres complexes $i(n-2k)$, avec $k \in [\![0,n]\!]$, sont des valeurs propres de l'endomorphisme φ_n.

Ces $n+1$ valeurs propres étant deux à deux distinctes, comme $\dim(V_n) = n+1$, φ_n n'admet pas d'autres valeurs propres. Ainsi

$$\mathrm{Spec}(\varphi_n) = \{(n-2k)i,\ k \in [\![0,n]\!]\}.$$

Comme K_n est la matrice de φ_n dans la base $\mathcal{B}_n$, elle possède les mêmes valeurs propres. On a ainsi démontré la conjecture de la question 3 :

$$\mathrm{Spec}(K_n) = \{(n-2k)i,\ k \in [\![0,n]\!]\}.$$

(g) On peut répondre rapidement à cette question en remarquant que l'on a trouvé $n+1$ valeurs propres distinctes de K_n. La matrice K_n est donc diagonalisable sur $\mathbb{C}$.

Pluis précisémment (ce niveau de précision n'est pas demandé) :

Les $n+1$ vecteurs $g_0, g_1, \ldots g_n$ étant des vecteurs propres de φ_n associés à des valeurs propres distinctes deux à deux, ils forment une famille libre de V_n.

La famille $(g_0, g_1, \ldots g_n)$ est une famille libre de $n+1$ vecteurs de V_n et $\dim(V_n) = n+1$, on conclut que la famille $\mathcal{B}_n' = (g_0, g_1, \ldots g_n)$ est une base de V_n diagonalisant ϕ_n.

La matrice de φ_n dans cette base est diagonale, notons-la D_n :

$$D_n = \mathrm{Mat}_{B'_n}(\varphi_n) = \begin{pmatrix} ni & 0 & \dots & \dots & \dots & 0 \\ 0 & (n-2)i & 0 & & & \vdots \\ \vdots & 0 & (n-4)i & 0 & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & 0 \\ \vdots & & & 0 & -(n-2)i & 0 \\ 0 & \dots & \dots & \dots & 0 & -ni \end{pmatrix}.$$

Ainsi φ_n est diagonalisable et les sous-espaces propres sont tous de dimension 1 : $E_{(n-2k)i}(\varphi_n) = \mathrm{Vect}(g_k)$ pour tout $k \in [\![0, n]\!]$.

Notons P la matrice de passage de la base $\mathcal{B}_n$ à la base $\mathcal{B}'_n$. Alors $K_n = PD_nP^{-1}$.

(h) La matrice K_n est inversible si et seulement si 0 n'est pas valeur propre de K_n.

Or $0 \in \mathrm{Spec}(K_n) \iff \exists k \in [\![0, n]\!], (n-2k)i = 0$.

Donc 0 est valeur propre de K_n si et seulement si il existe $k \in [\![0, n]\!]$ tel que $n = 2k$, (ce qui signifie que n est pair).

Conclusion : K_n est inversible si et seulement si n est impair.

(i) On suppose que n est pair : K_n n'est donc pas inversible car 0 est valeur propre de K_n (et de φ_n).

De plus, le sous-espace propre associé à la valeur propre 0 correspond au noyau :

$$E_0(\varphi_n) = \mathrm{Ker}(\varphi_n).$$

D'après l'étude précédente, $E_{i(n-2k)}(\varphi_n) = \mathrm{Vect}(g_k)$ pour tout $k \in [\![0, n]\!]$.

Comme n est pair, prenons en particulier $k = n/2$ donc $n - 2k = 0$.

On a donc $E_0(\varphi_n) = \mathrm{Vect}(g_k)$ pour $k = n/2$.

Or $\forall x, \quad g_k(x) = e^{i(n-2k)x} = e^{i0)x} = 1$.

Donc $\mathrm{Ker}(\varphi_n) = \mathrm{Vect}(1)$ (ensemble des fonctions constantes).

Il faut ensuite faire le lien entre le noyau de φ_n et le noyau de K_n.

Soit un vecteur $g \in V_n$ et X la matrice colonne des coordonnées de g dans la base $\mathcal{B}_n = (f_0, f_1, \dots, f_n)$.

Sachant que $K_n = \mathrm{Mat}_{\mathcal{B}_n}(\varphi_n)$, on a l'équivalence suivante :

$$\varphi_n(g) = 0 \iff K_nX = 0$$

Ainsi $g \in \mathrm{Ker}(\varphi_n) \iff X \in \mathrm{Ker}(K_n)$.

Sachant que $\mathrm{Ker}(\varphi_n) = \mathrm{Vect}(g_k)$ avec $k = n/2$, déterminer une base de $\mathrm{Ker}(K_n)$ revient donc à calculer les coordonnées de g_k dans la base $\mathcal{B}_n$.

On reprend le calcul du 4.(e) pour $k = n/2$ mais de façon plus simple, comme $n - k = k$:

$$\begin{aligned} g_k(x) &= (\cos(x) + i\sin(x))^{n-k}(\cos(x) - i\sin(x))^k \\ &= (\cos(x) + i\sin(x))^k(\cos(x) - i\sin(x))^k \\ &= (\cos^2(x) + \sin^2(x))^k \\ &= \sum_{j=0}^{k} \binom{k}{j} \sin^{2j}(x)\cos^{2k-2j}(x) \\ &= \sum_{j=0}^{k} \binom{k}{j} f_{2j}(x) \end{aligned}$$

Ainsi $g_k = \sum_{j=0}^{k} \binom{k}{j} f_{2j} + \sum_{j=0}^{k-1} 0 f_{2j+1}$.
On en déduit que les coordonnées de g_k dans la base $\mathcal{B}_n$ sont :

$$(\binom{k}{0}, 0, \binom{k}{1}, 0, \binom{k}{2}, 0, \ldots, \binom{k}{k-1}, 0, \binom{k}{k}).$$

Donc

$$\mathrm{Ker}(K_n) = \mathrm{Vect}(X_k)$$

où $X_k = \begin{pmatrix} 1 \\ 0 \\ k \\ 0 \\ \binom{k}{2} \\ 0 \\ \vdots \\ 0 \\ \binom{k}{2} \\ 0 \\ k \\ 0 \\ 1 \end{pmatrix}$ et $n = 2k$

Exercice sans préparation

1. ```
   import random as rd

   def gene(n):
       c = "ACGT"
       ch = ""
       for k in range(n):
           ch += rd.choice(c)
       return ch
   ```

2. ```
   def nbAC(g):
       compteur = 0
       for k in range(len(g)-1):
           if g[k:k+2] == 'AC':
               compteur += 1
       return compteur
   ```

Agro-Veto – Planche 17

Cours

Lien(s) entre l'indépendance de deux variables aléatoires discrètes et leur covariance.

Exercice préparé

1. On considère φ l'endomorphisme de $\mathbb{R}^3$, dont la matrice représentative dans la base canonique est la matrice A de $\mathcal{M}_3(\mathbb{R})$ suivante :

$$A = \begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 0 & 0 & 3 \end{pmatrix}.$$

   (a) Montrer que le spectre de l'endomorphisme φ est : $\text{Spec}(\varphi) = \{1, 3\}$. L'endomorphisme φ est-il diagonalisable ?

   (b) On note $a_1 = (1, 1, 0)$, $a_2 = (0, 0, 1)$ et $a_3 = (1, -1, 0)$.
   Montrer que la famille $\mathcal{B} = (a_1, a_2, a_3)$ est une base de $\mathbb{R}^3$ et déterminer la matrice M de l'endomorphisme φ dans la base $\mathcal{B}$.

   (c) Déterminer une matrice carrée P telle que $A = PMP^{-1}$ et expliciter P^{-1} à l'aide de la fonction `inv` de Python.
   La commande `inv` *du module* `linalg` *de la bibliothèque* `numpy` *permet de calculer l'inverse d'une matrice carrée de type* `matrix`*.*

2. Soient f, g et h trois fonctions dérivables sur $\mathbb{R}$ vérifiant :

$$\forall t \in \mathbb{R},\ \begin{cases} f'(t) &= 2f(t) + g(t) + h(t) \\ g'(t) &= f(t) + 2g(t) + h(t) \\ h'(t) &= 3h(t) \end{cases} \quad \text{et} \quad f(0) = g(0) = h(0) = 1.$$

   (a) Déterminer l'expression de $h(t)$ pour tout $t \in \mathbb{R}$, puis tracer à l'aide de Python l'allure de la courbe représentative de h sur l'intervalle $[0, 1]$.

   (b) On note $X(t) = \begin{pmatrix} f(t) \\ g(t) \\ h(t) \end{pmatrix}$ et $X'(t) = \begin{pmatrix} f'(t) \\ g'(t) \\ h'(t) \end{pmatrix}$.

   On note $Y(t) = P^{-1}X(t) = \begin{pmatrix} u(t) \\ v(t) \\ w(t) \end{pmatrix}$ et $Y'(t) = P^{-1}.X'(t) = \begin{pmatrix} u'(t) \\ v'(t) \\ w'(t) \end{pmatrix}$.

   Vérifier qu'on a : $\forall t \in \mathbb{R},\ u'(t) = 3u(t) + e^{3t}$.

   (c) En déduire l'expression de $u(t)$ pour tout $t \in \mathbb{R}$.

   (d) Déterminer alors l'expression de $f(t)$ et $g(t)$ en fonction de t.

Exercice sans préparation

1. Écrire une fonction `somme(f, a, b, N)` qui retourne $\sum_{k=0}^{N-1} f\left(a + k\frac{b-a}{N}\right)$ où f est une fonction, a et b deux réels ($a < b$), et N un entier supérieur à 1.
2. Écrire une fonction prenant en entrée une fonction f, deux réels a, b qui retourne une valeur approchée de $\int_a^b f(t)\ dt$. Utiliser cette fonction pour donner une valeur approchée de $\int_0^{+\infty} \exp\left(-t^2\right)\ dt$ à l'aide de l'égalité :

$$\int_0^{+\infty} f(t)\mathrm{d}t = \int_0^1 \frac{1}{(1-t)^2} f\left(\frac{t}{1-t}\right)\ dt.$$

Éléments de correction–Planche 17

Exercice avec préparation

1. (a) Il suffit d'échelonner $A-\lambda I_3$ pour trouver le spectre de A et donc celui de φ. On cherchera ensuite la dimension de ses sous-espaces propres.

Soit $\lambda \in \mathbb{C}$. Les matrices suivantes ont toutes même rang (via les mouvements de l'algorithme du pivot de Gauss) :

$$\begin{aligned} rg\begin{pmatrix} 2-\lambda & 1 & 1 \\ 1 & 2-\lambda & 1 \\ 0 & 0 & 3-\lambda \end{pmatrix} &= rg\begin{pmatrix} 1 & 2-\lambda & 1 \\ 2-\lambda & 1 & 1 \\ 0 & 0 & 3-\lambda \end{pmatrix} \\ &= rg\begin{pmatrix} 1 & 2-\lambda & 1 \\ 0 & 1-(2-\lambda)^2 & 1-(2-\lambda) \\ 0 & 0 & 3-\lambda \end{pmatrix} \\ &= rg\begin{pmatrix} 1 & 2-\lambda & 1 \\ 0 & (3-\lambda)(\lambda-1) & \lambda-1 \\ 0 & 0 & 3-\lambda \end{pmatrix} \end{aligned}$$

Cette dernière matrice n'est pas de rang 3 si et seulement si $\lambda = 1$ ou $\lambda = 3$ et donc

$$\mathrm{Spec}(A) = \{1, 3\}.$$

— Pour $\lambda = 3$, $X = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \in \mathrm{Ker}(A - \lambda I_3)$ si et seulement si

$$\begin{pmatrix} 1 & 2-\lambda & 1 \\ 0 & (3-\lambda)(\lambda-1) & \lambda-1 \\ 0 & 0 & 3-\lambda \end{pmatrix} X = \begin{pmatrix} 1 & -1 & 1 \\ 0 & 0 & 2 \\ 0 & 0 & 0 \end{pmatrix} X = 0 \iff (x - y = 0 \quad \text{et} \quad z = 0)$$

et donc $\mathrm{Ker}(A - 3I_3) = \mathrm{Vect}\left(\begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}\right)$ et $\dim \mathrm{Ker}(A - 3I_3) = 1$.

— Pour $\lambda = 1$, $X = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \in \mathrm{Ker}(A - \lambda I_3)$ si et seulement si

$$\begin{pmatrix} 1 & 2-\lambda & 1 \\ 0 & (3-\lambda)(\lambda-1) & \lambda-1 \\ 0 & 0 & 3-\lambda \end{pmatrix} .X = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 2 \end{pmatrix} .X = 0 \iff (x + y = 0 \quad \text{et} \quad z = 0)$$

et donc $\mathrm{Ker}(A - I_3) = \mathrm{Vect}\left(\begin{pmatrix} (1 \\ -1 \\ 0 \end{pmatrix}\right)$ et $\dim \mathrm{Ker}(A - I_3) = 1$.

L'endomorphisme n'est pas diagonalisable, la somme des dimensions de ses sous-espaces propres étant 2.

(b) En considérant la matrice

$$P = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}$$

on montre facilement qu'elle est inversible (un coup de pivot) et donc que la famille $\mathcal{B}$ est une base de $\mathbb{R}^3$, la matrice P étant la matrice de passage de la base canonique de $\mathbb{R}^3$ vers la base $\mathcal{B}$.

On a alors $\varphi(a_1) = 3a_1$, $\varphi(a_2) = a_1 + 3a_2$ et $\varphi(a_3) = a_3$, d'où la matrice

$$M = \begin{pmatrix} 3 & 1 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

(c) Par ce qui a été dit sur la matrice de passage P, on obtient donc $M = P^{-1}MP$.

On trouve l'inverse avec Python :

$$P^{-1} = \frac{1}{2}\begin{pmatrix} 1 & 1 & 0 \\ 0 & 0 & 2 \\ 1 & -1 & 0 \end{pmatrix}.$$

2. (a) On a une équation différentielle linéaire du premier ordre et homogène, et, sachant que $h(0) = 1$, on a donc directement la solution

$$\forall t \in \mathbb{R},\ h(t) = e^{3t}.$$

(b) On note d'abord que $X'(t) = AX(t)$, et donc

$$Y'(t) = P^{-1}APY(t) = MY(t).$$

La première ligne du système donne donc :

$$\forall t \in \mathbb{R}, \quad u'(t) = 3u(t) + v(t)$$

mais comme $Y = P^{-1}X$ on a

$$\forall t \in \mathbb{R}, \quad v(t) = h(t) = e^{3t}$$

ce qui est la réponse attendue.

(c) On note que $u(t) = \frac{1}{2}(f(t) + g(t))$, et donc $u(0) = 1$.

On trouve alors pour tout t : $u(t) = e^{3t}(t + 1)$.

(d) On a pour tout t :

$$\begin{pmatrix} f(t) \\ g(t) \\ h(t) \end{pmatrix} = PY(t) = \begin{pmatrix} u(t) + w(t) \\ u(t) - w(t) \\ v(t) \end{pmatrix}.$$

De même qu'en 2.(a), à la dernière ligne de l'identité $Y' = MY$, on trouve $w'(t) = w(t)$, avec $w(0) = \frac{1}{2}(f(0) - g(0)) = 0$. Ainsi, on a $w = 0$, puis

$$\forall t \in \mathbb{R}, \quad f(t) = g(t) = u(t).$$

Exercice sans préparation

1.
```
import numpy as np

def somme(f,a,b,N):
    S = 0
    x = a
    dx = (b-a)/N
    for k in range(N):
        S += f(x)
        x += dx
    return S
```

2. On écrit une variation du code précédent calculant une somme de Riemann

```
def sommeR(f,a,b,N):
    S = 0
    x = a
    dx = (b-a)/N
    for k in range(N):
        S += f(x)
        x += dx
    return S*dx
```

Pour donner une valeur approchée de $\displaystyle\int_0^{+\infty} \exp\left(-t^2\right)\ dt$ (qui vaut $\frac{1}{2}\sqrt{\pi}$ à l'aide de l'égalité :

$$\int_0^{+\infty} f(t)\ dt = \int_0^1 \frac{1}{(1-x)^2} f\left(\frac{x}{1-x}\right)\ dx,$$

on utilise la fonction tout juste écrite mais appliquée à la fonction adéquate. On peut remarquer que l'intégrale de droite est une intégrale généralisée ayant une singularité en 1 si f est continue sur $[0,+\infty[$ et donc que l'application d'une somme de Riemann est sujette à caution. Ce problème est levé en remarquant que dans le cas où $f : t \mapsto \exp(-t^2)$, la fonction $g : x \mapsto 1/(1-x)^2.f\left(\frac{x}{1-x}\right)$ se prolonge par continuité en 1.

En effet, si $t = \frac{x}{1-x}$, $t = \frac{1}{1-x} - 1$ et $\frac{1}{(1-x)^2} = (t+1)^2$. La limite lorsque $x \to 1^-$ de $g(x)$ est celle lorsque $t \to +\infty$ de $(1+t)^2.e^{-t^2}$, c'est-à-dire 0 par croissances comparées.

```
def f(t) :
    return np.exp(-t*t)
def g(x) :
    q = 1/(1-x)
    return f(x*q)*q*q

print(sommeR(g,0,1,1000))
print(np.sqrt(np.pi)/2)
```

Agro-Veto – Planche 18

Cours

Définition de la dérivée d'une fonction f en un point a.

Exercice préparé

On rappelle que, si X et Y sont deux variables aléatoires réelles indépendantes admettant respectivement les densités f et g, alors la variable aléatoire $X+Y$ admet une densité $f*g$ définie par

$$(f*g)(x) = \int_{-\infty}^{+\infty} f(x-t)g(t)dt.$$

1. On considère deux variables aléatoires indépendantes U et V suivant la loi uniforme sur $]0,1[$. Soient λ, μ deux réels strictement positifs.
   (a) Déterminer les lois des variables aléatoires $-\frac{1}{\lambda}\ln(U)$ et $-\frac{1}{\mu}\ln(V)$.
   (b) On considère X et Y deux variables aléatoires indépendantes, suivant la loi exponentielle de paramètres respectifs λ et μ.
   Écrire une fonction Python qui prend en argument λ et μ et qui simule la variable aléatoire $\min(X,Y)$.
   (c) Déterminer la loi de la variable aléatoire $\min(X,Y)$ et vérifier qu'il s'agit d'une loi exponentielle dont on précisera le paramètre.
   (d) Déterminer la loi de $-Y$.
   (e) Montrer qu'une densité de $X-Y$ est la fonction h définie sur $\mathbb{R}$ par :

$$h : x \mapsto \begin{cases} \dfrac{\lambda\mu}{\lambda+\mu}e^{-\lambda x} & \text{si } x > 0, \\ \dfrac{\lambda\mu}{\lambda+\mu}e^{\mu x} & \text{si } x \leqslant 0. \end{cases}.$$

   (f) Calculer alors la probabilité de l'événement $[X \leqslant Y]$.
2. Soit $(X_i)_{i\in\mathbb{N}^*}$ une suite de variables aléatoires mutuellement indépendantes telles que :
   - pour tout $n \in \mathbb{N}$, X_{2n+1} suit la loi exponentielle de paramètre 1 ;
   - pour tout $n \in \mathbb{N}^*$ X_{2n} suit la loi exponentielle de paramètre 2.

   Si $i \geqslant 2$, on dit que l'événement « X_i est un creux » est réalisé si $[X_i \leqslant X_{i-1}]$ et $[X_i \leqslant X_{i+1}]$ sont réalisés tous les deux.
   (a) Avec Python, estimer $\mathbb{P}$(« X_2 est un creux ») et $\mathbb{P}$(« X_3 est un creux »).
   (b) Calculer la probabilité des deux événements précédents.
3. (a) Que vaut la probabilité de l'événement « X_2 et X_3 sont des creux » ?
   (b) Les événements « X_4 est un creux » et « X_8 est un creux » sont-ils indépendants ?
   (c) Déterminer la loi du nombre de creux parmi les 10 variables aléatoires $X_4, X_8, X_{12}, \ldots, X_{40}$.

Exercice sans préparation

Soit $n \in \mathbb{N}$ et $(a_0, \dots, a_n) \in \mathbb{R}^{n+1}$. On considère le polynôme $P(X) = a_0 + a_1 X + \dots + a_n X^n$.
Dans Python, on représente ce polynôme P par la liste de ses coefficients `[`$a_0, \dots, a_n$`]`.

1. Écrire une fonction Python `eval` qui prend en argument un polynôme P et un réel a et qui renvoie $P(a)$.
2. Écrire une fonction Python `deriv` qui prend en argument un polynôme P et qui renvoie P'.
3. On considère l'application linéaire f définie sur $\mathbb{R}[X]$ par $f(P)(X) = 2XP(X) - P'(X)$.
   Écrire une fonction Python `f` qui prend en argument un polynôme P et qui renvoie $f(P)$.

Éléments de correction–Planche 18

Exercice avec préparation

1. (a) Posons $X = -\dfrac{1}{\lambda}\ln(U)$ et $Y = -\dfrac{1}{\mu}\ln(U)$. On commence par calculer la fonction de répartition F_X de X. Soit $x \in \mathbb{R}$. On a

$$F_X(x) = \mathbb{P}(X \leq x) = \mathbb{P}\left(-\frac{1}{\lambda}\ln(U) \leq x\right) = \mathbb{P}\big(\ln(U) \geq -\lambda x\big),$$

la dernière égalité étant obtenue par le fait que $-\dfrac{1}{\lambda} < 0$ (car $\lambda > 0$). Par stricte croissance de la fonction exponentielle sur $\mathbb{R}$, on a [1]

$$F_X(x) = \mathbb{P}\big(U \geq \exp(-\lambda x)\big) = 1 - F_U\big(\exp(-\lambda x)\big)$$

où F_U est la fonction de répartition de U donnée par

$$F_U : x \longmapsto \begin{cases} 0 & \text{si } x \leq 0 \\ x & \text{si } 0 < x < 1 \\ 1 & \text{si } x \geq 1 \end{cases}$$

On a donc

$$F_X(x) = \begin{cases} 1 & \text{si } \exp(-\lambda x) \leq 0 \\ 1 - \exp(-\lambda x) & \text{si } 0 < \exp(-\lambda x) < 1 \\ 0 & \text{si } \exp(-\lambda x) \geq 1 \end{cases}$$

Le premier cas n'arrive jamais par stricte positivité de l'exponentielle. Le deuxième cas est équivalent à $-\lambda x < 0$ c'est-à-dire $x > 0$ (car $\lambda > 0$) et le troisième cas est équivalent à $x \leq 0$ pour la même raison. Finalement, on obtient

$$\forall x \in \mathbb{R}, \quad F_X(x) = \begin{cases} 0 & \text{si } x \leq 0 \\ 1 - \exp(-\lambda x) & \text{si } x > 0 \end{cases}$$

On reconnaît la fonction de répartition associée à la loi exponentielle de paramètre λ. La fonction de répartition caractérisant la loi, cela montre que $X = -\dfrac{1}{\lambda}\ln(U)$ suit la loi exponentielle de paramètre λ.

De même, $Y = -\dfrac{1}{\mu}\ln(U)$ suit la loi exponentielle de paramètre μ.

(b)
```
def Exp(lbda) : # lambda est un mot-clé réservé du langage
    return -m.log(rd.random())/lbda

def Z(lbda, mu):
    X = Exp(lbda)
    Y = Exp(mu)
    return min(X, Y)
```

1. Pour être tout-à-fait précis, on a $\mathbb{P}\big(U \geq \exp(-\lambda x)\big) = 1 - \mathbb{P}\big(U < \exp(-\lambda x)\big) = 1 - \mathbb{P}\big(U \leq \exp(-\lambda x)\big)$ puisque $\mathbb{P}\big(U = \exp(-\lambda x)\big) = 0$, U étant à densité.

(c) On détermine la fonction de répartition F_Z de $Z = \min(X, Y)$. Soit $z \in \mathbb{R}$. On a

$$F_Z(z) = \mathbb{P}(Z \leq z) = 1 - \mathbb{P}(Z > z) = 1 - \mathbb{P}\big(\min(X, Y) > z\big).$$

Or, pour tous nombres réels x et y, $\min(x, y) > z$ si et seulement si $x > z$ et $y > z$ donc

$$F_Z(z) = 1 - \mathbb{P}(X > z, Y > z) = 1 - \mathbb{P}(X > z)\,\mathbb{P}(Y > z) = 1 - \big(1 - F_X(z)\big)\big(1 - F_Y(z)\big)$$

où l'avant dernière égalité est obtenue par indépendance de X et de Y. Puisque X et Y suivent toutes les deux des lois exponentielles, on a

$$F_Z(z) = \begin{cases} 0 & \text{si } z \leq 0 \\ 1 - \exp(-\lambda\, x)\exp(-\mu\, x) & \text{si } z > 0 \end{cases} = \begin{cases} 0 & \text{si } z \leq 0 \\ 1 - \exp\big(-(\lambda + \mu)\, x\big) & \text{si } z > 0 \end{cases}$$

On reconnaît la fonction de répartition associée à la loi exponentielle de paramètre $\lambda + \mu > 0$ donc $Z = \min(X, Y)$ suit la loi exponentielle de paramètre $\lambda + \mu$.

(d) Déterminons la fonction de répartition F_{-Y} de $-Y$. Soit $y \in \mathbb{R}$. On a

$$F_{-Y}(y) = \mathbb{P}(-Y \leq y) = \mathbb{P}(Y \geq -y) = 1 - F_Y(-y) = \begin{cases} 1 & \text{si } -y \leq 0 \ (\Leftrightarrow y \geq 0) \\ \exp(\mu\, y) & \text{si } -y > 0 \ (\Leftrightarrow y < 0) \end{cases}$$

La fonction F_{-Y} est clairement continue sur $\mathbb{R}^*$. De plus,

$$\lim_{y \to 0^-} F_{-Y}(y) = \lim_{y \to 0^-} \exp(\mu\, y) = 1 = F_{-Y}(0)$$

ce qui montre que F_{-Y} est continue en 0. Elle est donc continue sur $\mathbb{R}$.
Elle est de plus clairement de classe $\mathcal{C}^1$ sur $\mathbb{R}^*$.
D'après le cours, on en déduit que $-Y$ est une variable aléatoire à densité dont une densité f_{-Y} est donnée par $f_{-Y} = (F_{-Y})'$ sur $\mathbb{R}^*$ et en attribuant une valeur arbitraire à $f_{-Y}(0)$. En choisissant 0 pour cette valeur arbitraire, on obtient :

$$f_{-Y} : y \longmapsto \begin{cases} \mu\, \exp(\mu\, y) & \text{si } y < 0 \\ 0 & \text{si } y \geq 0 \end{cases}$$

(e) Puisque X et Y sont indépendantes alors X et $-Y$ sont indépendantes. D'après le rappel, $X - Y = X + (-Y)$ admet donc une densité h donnée par

$$\forall x \in \mathbb{R}, \quad h(x) = \int_{-\infty}^{+\infty} f_X(x - u)\, f_{-Y}(u)\, \mathrm{d}u$$

où f_X est une densité de X, par exemple :

$$F_X : u \longmapsto \begin{cases} 0 & \text{si } u \leq 0 \\ \lambda\, e^{\lambda\, u} & \text{si } u > 0 \end{cases}.$$

Soit $x \in \mathbb{R}$. Pour tout $u \in \mathbb{R}$,

$$\begin{aligned} f_X(x-u).f_{-Y}(u) &= \mathbf{1}_{x-u\geq 0}.\mathbf{1}_{u<0}\lambda.\mu.\exp(-\lambda(x-u)+\mu.u) \\ &= \mathbf{1}_{u\leq x}.\mathbf{1}_{u<0}\lambda.\mu.\exp(-\lambda(x-u)+\mu.u) \\ &= \begin{cases} \mathbf{1}_{u<0}\lambda.\mu.\exp(-\lambda.x).\exp((\lambda+\mu).u) & \text{si } x \geq 0 \\ \mathbf{1}_{u<x}\lambda.\mu.\exp(-\lambda.x).\exp((\lambda+\mu).u) & \text{si } x \leq 0 \end{cases}. \end{aligned}$$

Distinguons donc deux cas :

— Si $x > 0$ alors

$$\begin{aligned} h(x) &= \lambda.\mu.\exp(-\lambda.x).\int_{-\infty}^{0}\exp((\lambda+\mu).u)\ du \\ &= \lambda.\mu.e^{-\lambda.x}\lim_{r\to-\infty}\left[\frac{e^{(\lambda+\mu).u}}{\lambda+\mu}\right]_r^0 \\ &= \lambda.\mu.e^{-\lambda.x}\lim_{r\to-\infty}\left(\frac{1-e^{(\lambda+\mu).r}}{\lambda+\mu}\right) \\ &= \frac{\lambda.\mu}{\lambda+\mu}.e^{-\lambda.x}. \end{aligned}$$

— Si $x \leq 0$ alors, de manière similaire au calcul précédent ,

$$\begin{aligned} h(x) &= \lambda.\mu.\exp(-\lambda.x).\int_{-\infty}^{x}\exp((\lambda+\mu).u)\ du \\ &= \lambda.\mu.e^{-\lambda.x}\lim_{r\to-\infty}\left[\frac{e^{(\lambda+\mu).u}}{\lambda+\mu}\right]_r^x \\ &= \lambda.\mu.e^{-\lambda.x}\lim_{r\to-\infty}\left(\frac{e^{(\lambda+\mu).r}-e^{(\lambda+\mu).r}}{\lambda+\mu}\right) \\ &= \frac{\lambda.\mu}{\lambda+\mu}e^{\mu.x}. \end{aligned}$$

Une densité de $X-Y$ est $h : x \mapsto \begin{cases} \dfrac{\lambda\,\mu}{\lambda+\mu}\,e^{-\lambda\,x} & \text{si } x > 0\,; \\ \dfrac{\lambda\,\mu}{\lambda+\mu}\,e^{\mu\,x} & \text{si } x \leq 0. \end{cases}$

(f) On a

$$\begin{aligned} \mathbb{P}(X \leq Y) &= \mathbb{P}(X-Y \leq 0) \\ &= \int_{-\infty}^{0} h(x)\ dx = \lim_{r\to-\infty}\int_r^0 \frac{\lambda.\mu}{\lambda+\mu}e^{\mu.x}\ dx \\ &= \frac{\lambda.\mu}{\lambda+\mu}\lim_{r\to-\infty}\left[\frac{1}{\mu}e^{\mu.x}\right]_r^0 = \frac{\lambda.\mu}{\lambda+\mu}\lim_{r\to-\infty}\left(\frac{1}{\mu}-\frac{1}{\mu}e^{\mu.r}\right) \\ &= \frac{\lambda.\mu}{\lambda+\mu}\cdot\frac{1}{\mu} \end{aligned}$$

car $\mu > 0$, d'où

$$\mathbb{P}(X \leq Y) = \frac{\lambda}{\lambda+\mu}.$$

2. (a) On utilise l'approche fréquentielle par moyenne empirique, ce qui se justifie par la loi faible des grands nombres.

```
NS = 10**5                          # nombre de réalisations
M_X2creux, M_X3creux = 0, 0         # frequence empirique des événements
                                    # = moyenne empirique d'indicatrice d'évèneme
for _ in range(NS):
    X1, X3 = Exp(1), Exp(1)
```

```
        X2, X4 = Exp(2), Exp(2)
        if X2 <= X1 and X2 <= X3:   # X2 est un creux
            M_X2creux += 1
        if X3 <= X2 and X3 <= X4:   # X3 est un creux
            M_X3creux += 1
    print('Probabilité␣que␣X2␣soit␣un␣creux␣:', M_X2creux/NS)
    print('Probabilité␣que␣X3␣soit␣un␣creux␣:', M_X3creux/NS)
```

On obtient, en exécutant ce script,

```
Probabilité que X2 soit un creux : 0.50062
Probabilité que X3 soit un creux : 0.20073
```

(b) L'évènement « X_2 est un creux » est l'évènement $[X_2 \leq X_1, X_2 \leq X_3] = [X_2 \leq \min(X_1, X_3)]$. D'après la question 1.(c), $\min(X_1, X_3)$ suit la loi exponentielle de paramètre $1+1=2$ (X_1 et X_3 sont bien indépendantes et suivent la loi exponentielle de paramètre 1).
D'après le lemme des coalitions, puisque X_1, X_2 et X_3 sont mutuellement indépendantes, X_2 et $\min(X_1, X_3)$ sont indépendantes. La question 1.(f) montre alors que

$$\mathbb{P}\big(X_2 \leq \min(X_1, X_3)\big) = \frac{2}{2+2} = \frac{1}{2}.$$

Ainsi, « X_2 est un creux » a pour probabilité $\frac{1}{2}$.

Le même raisonnement montre que « X_3 est un creux » a pour probabilité $\frac{1}{5}$.

3. (a) Si X_2 et X_3 sont tous les deux des creux, en particulier $X_2 \leq X_3$ et $X_3 \leq X_2$ d'où $X_2 = X_3$. Autrement dit, on a l'inclusion

$$[\text{« } X_2 \text{ et } X_3 \text{ sont des creux »}] \subset [X_2 = X_3]$$

d'où

$$0 \leq \mathbb{P}(\text{« } X_2 \text{ et } X_3 \text{ sont des creux »}) \leq \mathbb{P}(X_2 = X_3).$$

Or, $\mathbb{P}(X_2 = X_3) = \mathbb{P}(X_2 - X_3 = 0) = 0$ car $X_2 - X_3$ est à densité d'après la question 1.(e) (X_2 et X_3 sont indépendantes et suivent toutes les deux une loi exponentielle). On en déduit donc par encadrement que la probabilité que X_2 et X_3 soient des creux est nulle.

(b) L'évènement « X_4 est un creux » ne dépend que des variables aléatoires X_3, X_4 et X_5. L'évènement « X_8 est un creux » ne dépend que des variables aléatoires X_7, X_8 et X_9. Or, les variables aléatoires X_3, X_4, X_5, X_7, X_8 et X_9 sont mutuellement indépendantes. Le lemme des coalitions assure alors que :

les évènements « X_4 est un creux » et « X_8 est un creux » sont indépendants.

(c) Par le même argument qu'à la question précédente, les évènements « X_{4i} est un creux » pour $i \in \{1, \dots, 10\}$ sont mutuellement indépendants et par le même calcul qu'à la question 2.(b), chaque évènement « X_{4i} est un creux » pour $i \in \{1, \dots, 10\}$ a la même probabilité $\frac{1}{2}$. Ainsi :

la loi du nombre de creux parmi $X_4, \dots, X_{40}$ suit la loi binomiale de paramètres 10 et $\frac{1}{2}$.

Exercice sans préparation

1. On propose ici deux solutions, avec boucle et avec utilisation de liste en compréhension.

```
def eval1(P, a):
    resultat = 0
    for k in range(len(P)):
        resultat += P[k]*a**k
    return resultat

def eval2(P, a):
    return sum([P[k]*a**k for k in range(len(P))])
```

2. Il faut faire attention ici au cas des polynômes constants, pour ne pas se retrouver avec une liste vide... On propose encore ici deux solutions, avec boucle et avec utilisation de liste en compréhension.
   Si un polynôme $P \in \mathbb{R}[X]$ est donné par

$$P = \sum_{k=0}^{n} a_k X^k,$$

   ce qui correspond à la liste $[a_0, \dots, a_n]$, alors son polynôme dérivé

$$P' = \sum_{k=1}^{n} k a_k X^{k-1}$$

   (avec la convention que cette somme est le polynôme nul si $n = 0$, c'est-à-dire si P est constant) correspond à la liste $[0]$ si P est constant et $[a_1, 2a_2, \dots, n\,a_n]$ (de longueur n) si P n'est pas constant.

```
def deriv1(P):
    if len(P) == 1:   # si P est constant
        return [0]
    else:
        Pprime = []
        for k in range(1, len(P)):
            Pprime.append(k*P[k])
        return Pprime

def deriv2(P):
    if len(P) == 1:   # si P est constant
        return [0]
    else:
        return [k*P[k] for k in range(1, len(P))]
```

3. En reprenant les notations de la question précédente, on a $f(P) = 2a_0\,X$ si P est constant et

$$\begin{aligned} f(P) &= 2XP - P' \\ &= -a_1 + (2a_0 - 2a_2)\,X + (2a_1 - 3a_3)\,X^2 + \cdots + (2a_{n-2} + n\,a_n)\,X^{n-1} + 2a_{n-1}\,X^n + 2a_n X^{n+1} \end{aligned}$$

   sinon. Cela conduit alors à la fonction suivante :

```
def f(P):
    n = len(P) - 1
    if n == 0:
        return [0, 2*P[0]]
    else:
        resultat = [P[1]]   # le terme constant est a1
        for k in range(1, n):
            resultat.append(2*P[k-1] - (k+1)*P[k+1])
        resultat.extend([2*P[n-1], 2*P[n]])
        return resultat
```

Cours

Énoncer le théorème de Rolle.

Exercice préparé

Soit $r \geq 2$ un entier nature et X_r une variable aléatoire uniformément distribuée sur l'ensemble fini $[\![1, r]\!]$ et $(X_{r,n})_{n\in\mathbb{N}^*}$ une suite de variables aléatoires indépendantes de même loi que X_r.
On pose, pour $n \in \mathbb{N}^*$, $Y_{r,n} = \min(X_{r,1}, \dots X_{r,n})$.

1. Simulation informatique.
   (a) Écrire une fonction Python d'entête `SY(r, n, NS)` retournant une liste de `NS` valeurs simulées de $Y_{r,n}$ pour une valeur de $r =$ `r` donnée.
   (b) Décrire ce que retourne la fonction `PY(r, n, NS)` écrite dans l'extrait de code suivant :

```
def PY(r, n, NS):
    L = SY(r, n, NS)
    P = [0] * r
    for i in range(len(L)):
        P[L[i] - 1] += 1
    for k in range(len(P)):
        P[k] = P[k]/NS
    return P
```

   (c) En supposant que `P` a été calculée par `P = PY(r, n, NS)` pour certaines valeurs de `r, n, NS` décrire ce que calcule (et retourne) la fonction `FY` écrite dans l'extrait de code suivant :

```
def FY(P) :
    F = []
    s = 0
    for k in range(len(P)) :
        s += P[k]
        F.append(s)
    return F
```

2. Loi de $Y_{r,n}$.
   (a) Calculer, pour tout $n \in \mathbb{N}^*$, tout k, $\mathbb{P}(Y_{r,n} \geq k)$ puis $\mathbb{P}(Y_{r,n} = k)$.
   (b) En déduire que lorsque $n \to +\infty$ (r est ici fixé), $Y_{r,n}$ converge en loi vers une variable à préciser.
   (c) Montrer que lorsque $r \to +\infty$, $Y_{r,r}$ converge en loi vers une variable aléatoire dont on précisera la loi.

3. (a) Soit Z une v.a à valeurs dans $[\![1, r]\!]$. Démontrer que $\mathbb{E}(Z) = \sum_{k=1}^{r} \mathbb{P}(Z \geq k)$.
   (b) En déduire que $\mathbb{E}(Y_{r,n}) = \dfrac{1}{r^n} \sum_{\ell=1}^{r} \ell^n$. Combien vaut $\mathbb{E}(Y_{r,2})$?

4. Soit $n \in \mathbb{N}^*$ et $(e_r)_{r\geq 1}$ la suite définie par $e_r = \sum_{\ell=1}^{r} \left(\dfrac{\ell}{r}\right)^n$.

   Montrer $e_r \underset{r\to+\infty}{\sim} \dfrac{r}{n+1}$.

Exercice sans préparation

1. Afficher les 10 premiers termes de la suite définie par $u_0 = 1/2$ et pour tout $n \in \mathbb{N}$, $u_{n+1} = u_n(1 - u_n)$.
2. Pour la même suite, trouver le plus petit indice n tel que $u_n < 1/100$.
3. Soit pour tout $n \in \mathbb{N}$, $v_n = nu_n$. Étudier informatiquement la monotonie de (v_n).
4. Trouver le premier n tel que $v_n > 0.99$.

Éléments de correction–Planche 19

Exercice avec préparation

Soit $r \geq 2$ un entier nature et X_r une variable aléatoire uniformément distribuée sur l'ensemble fini $[\![1, r]\!]$ et $(X_{r,n})_{n\in\mathbb{N}^*}$ une suite de variables aléatoires indépendantes de même loi que X_r.
On pose, pour $n \in \mathbb{N}^*$, $Y_{r,n} = \min(X_{r,1}, \dots X_{r,n})$.

1. Simulation informatique.

   (a)
   ```
   def X(r) :
       return np.random.randint(r) + 1

   def Y(r, n) :
       m = X(r)
       for _ in range(n-1): #calcul de minimum de liste
           m = min(X(r), m)
       return m

   def SY(r, n, NS):
       return [ Y(r, n) for _ in range(NS)]
   ```

   (b) Il s'agit de l'évaluation des fréquences d'apparition de chacun des nombres k entre 1 et r dans une liste simulée par la fonction de la question précédente. Par la loi faible des grands nombres, lorsque `NS` est grand, il s'agit d'une évaluation par simulation de la loi de $Y_{r,n}$: $\mathbb{P}(Y_{r,n} = k) \simeq$ `P[k-1]`.

   (c) La fonction `FY` retourne la fonction de répartition de la loi de probabilité décrite par `P`. Vu ce qui vient d'être dit, il s'agit d'un moyen d'évaluer la fonction de répartition de $Y_{r,n}$.

2. Loi de $Y_{r,n}$.

   (a) Soit $n \in \mathbb{N}^*$, $Y_{r,n}$ est à valeurs dans $[\![1, r]\!]$. Pour $k \in [\![1, r]\!]$:

   $$\mathbb{P}(Y_{r,n} \geq k) = \mathbb{P}(X_{r,1} \geq k, \dots, X_{r,n} \geq k) = \mathbb{P}(X_r \geq k)^n = \frac{(r-k+1)^n}{r^n}.$$

   L'avant-dernière égalité étant due à l'indépendance des $X_{r,k}$ et la dernière à la connaissance de la fonction de répartition d'une v.a. uniforme sur $[\![1, r]\!]$ (qui donne $\mathbb{P}(X_r \geq k) = \frac{r-k+1}{r}$).
   De plus,

   $$\mathbb{P}(Y_{r,n} = k) = \mathbb{P}(Y_{r,n} \geq k) - \mathbb{P}(Y_{r,n} \geq k+1) = \frac{(r-k+1)^n - (r-k)^n}{r^n}.$$

   (b) Si l'on fixe $r \in \mathbb{N}^*$: $\lim\limits_{n\to+\infty} \mathbb{P}(Y_{r,n} = k) = \begin{cases} 1 & \text{si } k = 1 \\ 0 & \text{sinon} \end{cases}$.
   Donc $(Y_{r,n})_n$ converge en loi vers une v.a. constante valant 1.

   (c) Soit $k \in \mathbb{N}$ est fixé. Pour tout $r \geq k$, on a (passage classique à la forme exponentielle) :

   $$\frac{(r-k+1)^r}{r^r} = \left(1 - \frac{k-1}{r}\right)^r \xrightarrow[r\to+\infty]{} e^{-(k-1)}.$$

On a donc, pour tout $k \in \mathbb{N}^*$,

$$\mathbb{P}(Y_{r,r} = k) = \frac{(r-k+1)^r - (r-k)^r}{r^r} \xrightarrow[r\to+\infty]{} e^{-(k-1)} - e^{-k} = (1-e^{-1})e^{-(k-1)}.$$

On en déduit que $Y_{r,r}$ converge en loi vers une variable géométrique de paramètre $1-e^{-1}$.

3. (a) Soit Z une v.a à valeurs dans $[\![1,r]\!]$. Démontrer que $\mathbb{E}(Z) = \sum_{k=1}^{r} \mathbb{P}(Z \geq k)$.

Deux façons de faire :

— Par technique de somme double :

$$\begin{aligned}
\sum_{k=1}^{r} \mathbb{P}(Z \geq k) &= \sum_{k=1}^{r} \sum_{\ell=k}^{r} \mathbb{P}(Z = \ell) \\
&= \sum_{\ell=1}^{r} \sum_{k=1}^{\ell} \mathbb{P}(Z = \ell) \quad \text{interversion somme double triangulaire} \\
&= \sum_{\ell=1}^{r} \ell \mathbb{P}(Z = \ell) = \mathbb{E}(Z)
\end{aligned}$$

— Par changement d'indice et télescopage :

$$\begin{aligned}
\mathbb{E}(Z) &= \sum_{k=1}^{r} k\mathbb{P}(Z = k) \\
&= \sum_{k=1}^{r} k(\mathbb{P}(Z \geq k) - \mathbb{P}(Z \geq k+1)) \\
&= \sum_{k=1}^{r} k\mathbb{P}(Z \geq k) - \sum_{k=1}^{r} k\mathbb{P}(Z \geq k+1) \\
&= \sum_{k=1}^{r} k\mathbb{P}(Z \geq k) - \sum_{\ell=2}^{r+1} (\ell-1)\mathbb{P}(Z \geq \ell) \quad (\ell = k+1) \\
&= \sum_{k=1}^{r} k\mathbb{P}(Z \geq k) - \sum_{k=1}^{r} (k-1)\mathbb{P}(Z \geq k) \\
&= \sum_{k=1}^{r} (k-(k-1))\mathbb{P}(Z \geq k) = \sum_{k=1}^{r} \mathbb{P}(Z \geq k)
\end{aligned}$$

(b) En appliquant cette formule à $Z = Y_{r,n}$, on en déduit (dernière égalité par réindexation $\ell = r-(k-1)$) que

$$\mathbb{E}(Y_{r,n}) = \frac{1}{r^n} \sum_{k=1}^{r} (r-k+1)^n = \frac{1}{r^n} \sum_{\ell=1}^{r} \ell^n.$$

On a donc $\mathbb{E}(Y_2) = \frac{1}{r^2} \frac{r(r+1)(2r+1)}{6} = \frac{(r+1)(2r+1)}{6r}$ (somme des r premiers carrés d'entiers).

4. Soit $n \in \mathbb{N}^*$.

   On peut écrire

$$e_r = r\frac{1}{r}\sum_{\ell=1}^{r}\left(\frac{\ell}{r}\right)^n$$

   On reconnait une somme de Riemann (d'indice r) de la fonction $x \mapsto x^n$ (continue) sur l'intervalle $[0,1]$. Donc

$$\frac{1}{r}\sum_{\ell=1}^{r}\left(\frac{\ell}{r}\right)^n \xrightarrow[r\to+\infty]{} \int_0^1 x^n\,dx = \frac{1}{n+1}$$

   et donc

$$e_r \underset{r\to+\infty}{\sim} \frac{r}{n+1}.$$

Exercice sans préparation

1. ```
   import numpy as np
   import matplotlib.pyplot as plt

   def u(u0=1/2,n=10) :
       un = u0
       for _ in range(n) :
           print(_,un)
           un *= 1-un
   ```

2. ```
   def minimum_un(u0=1/2) :
       un = u0
       compteur = 0
       while un >= 1/100 :
           un *= 1-un
           compteur += 1
       return compteur
   ```

   On trouve 94.

3. ```
   def v(u0=1/2,n=100) :
       un = u0
       L = [0]
       for k in range(1,n+1) :
           un *= 1-un
           L.append(k*un)
       return L

   plt.plot(v())
   plt.show()
   ```

   La suite semble croissante.

4. ```
   def minimum_v(u0=1/2) :
       un = u0
       k = 0
       while k*un <= 0.99 :
           un *= 1-un
           k += 1
       return k
   ```

   On trouve 843.

Agro-Veto – Planche 20

Cours

Fonction de répartition d'une variable aléatoire de loi uniforme sur $[-\pi, \pi]$.

Exercice préparé

Rappel : algorithme de dichotomie. *On considère une fonction f continue sur un segment $[a,b]$. On suppose que f s'annule exactement une fois sur $[a,b]$, en un point que l'on note γ.*
On définit les suites $(a_k)_{k\geqslant 0}$ et $(b_k)_{k\geqslant 0}$ de la façon suivante :

- *$a_0 = a$ et $b_0 = b$.*
- *Pour tout $k \in \mathbb{N}$, on note $c_k = \dfrac{a_k + b_k}{2}$ et :* $\begin{cases} a_{k+1} = a_k \text{ et } b_{k+1} = c_k & \text{si } f(a_k)f(c_k) \leqslant 0 \\ a_{k+1} = c_k \text{ et } b_{k+1} = b_k & \text{sinon} \end{cases}$

On sait alors que les suites (a_k) et (b_k) convergent toutes les deux vers γ, en vérifiant :

$$\forall k \in \mathbb{N},\ a_k \leqslant \gamma \leqslant b_k \quad \text{et} \quad b_k - a_k = \frac{b-a}{2^k}.$$

On peut montrer que si l'entier k est tel que $\dfrac{b-a}{2^k} \leqslant \epsilon$, alors a_k et b_k sont des valeurs approchées de γ à ϵ-près.

On définit la matrice $A = \begin{pmatrix} 7 & 3 & 4 \\ 6 & 0 & -1 \\ 6 & 1 & 2 \end{pmatrix}$

1. (a) Soit $\lambda \in \mathbb{C}$. Montrer que λ est valeur propre de A si et seulement si $\lambda^3 - 9\lambda^2 - 27\lambda + 23 = 0$.
   (b) On pose $f : x \mapsto x^3 - 9x^2 - 27x + 23$. Calculer $f(0)$ et $f(3)$. Donner son tableau de variations et ses limites en $\pm\infty$.
   (c) En déduire que A admet trois valeurs propres distinctes $\lambda_1 < \lambda_2 < \lambda_3$.
   (d) Implémenter l'algorithme de dichotomie et l'utiliser pour évaluer numériquement les valeurs propres de A. (On pourra calculer $f(-4)$ et $f(12)$.)
   (e) Montrer qu'il existe une matrice P inversible telle que $A = PDP^{-1}$ où $D = \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{pmatrix}$.
2. On s'intéresse maintenant à l'ensemble –noté $C(A)$– des matrices 3×3 qui commutent avec A, c'est à dire :
   $$C(A) = \{M \in \mathcal{M}_3(\mathbb{R}) \mid AM = MA\}.$$
   On définit de manière analogue l'ensemble $C(D)$ des matrices qui commutent avec D.
   (a) Montrer que $C(A)$ est un sous-espace vectoriel de $\mathcal{M}_3(\mathbb{R})$.
   (b) Montrer que $M \in C(D)$ si et seulement si M est diagonale.
   (c) Montrer que $M \in C(A)$ si et seulement si $P^{-1}MP$ est diagonale. Quelle est la dimension de $C(A)$?
3. Montrer qu'il existe un polynôme Q non nul, de degré ≤ 3, tel que $Q(A) = 0$.
4. Donner un exemple d'un tel polynôme Q.

Exercice sans préparation

1. Écrire une fonction Python calculant la valeur de $\frac{1}{n}\sum_{k=1}^{n} e^{-\frac{k^2}{2n^2}}$. Justifier que pour n suffisamment grand, cette somme est proche de $\int_0^1 e^{-\frac{1}{2}x^2}\,dx$.
2. La fonction `norm.rvs` de la bibliothèque `scipy.stats` retourne une valeur simulée d'une v.a X suivant une loi normale centrée réduite. Justifier que

$$\int_0^1 e^{-\frac{1}{2}x^2}\,dx = \sqrt{2\pi}\mathbb{P}(0 \leq X \leq 1)$$

   et utiliser cette identité pour évaluer –par simulation– l'intégrale de gauche.
3. Comparer les valeurs obtenues par les deux fonctions écrites. Quelle est la méthode la plus précise (à nombre d'opérations similaire) ?

Éléments de correction–Planche 20

Exercice avec préparation

On définit la matrice $A = \begin{pmatrix} 7 & 3 & 4 \\ 6 & 0 & -1 \\ 6 & 1 & 2 \end{pmatrix}$

1. (a) Soit $\lambda \in \mathbb{C}$.
Le nombre λ est valeur propre de A si et seulement si la matrice $A - \lambda I_3$ est de rang ≤ 2. Hors, en appliquant l'algorithme du pivot de Gauss, les matrices suivantes ont toutes même rang : $A - \lambda I_3$,

$$\begin{pmatrix} 6 & 1 & 2-\lambda \\ 6 & -\lambda & -1 \\ 7-\lambda & 3 & 4 \end{pmatrix} \quad (L_1 \leftrightarrow L_3)$$
$$\begin{pmatrix} 6 & 1 & 2-\lambda \\ 0 & -1-\lambda & -3+\lambda \\ 0 & 11+\lambda & 24-(7-\lambda)(2-\lambda) \end{pmatrix} \quad (L_2 \leftarrow L_2 - L_1, L_3 \leftarrow 6L_3 - (7-\lambda)L_1)$$
$$\begin{pmatrix} 6 & 1 & 2-\lambda \\ 0 & -1-\lambda & -3+\lambda \\ 0 & 10 & 7+10\lambda-\lambda^2 \end{pmatrix} \quad (L_3 \leftarrow L_3 + L_2).$$

En échangeant les lignes L_2 et L_3 puis en remplaçant L_3 par $10L_3 + (1+\lambda)L_2$ on obtient que le rang de $A - \lambda I_3$ est égal au rang de

$$\begin{pmatrix} 6 & 1 & 2-\lambda \\ 0 & 10 & 7+10\lambda-\lambda^2 \\ 0 & 0 & -23+27\lambda+9\lambda^2-\lambda^3 \end{pmatrix}.$$

Ainsi λ est valeur propre si et seulement si

$$\lambda^3 - 9\lambda^2 - 27\lambda + 23 = 0.$$

(b) On pose $f : x \mapsto x^3 - 9x^2 - 27x + 23$. Calculer $f(0)$ et $f(3)$.
La fonction f est polynomiale de degré 3, de coefficient dominant 1 et donc

$$\lim_{x\to+\infty} f(x) = +\infty \quad \text{et} \quad \lim_{x\to-\infty} f(x) = -\infty.$$

Par ailleurs, on a $f(0) = 23 > 0$, $f(3) = 27 - 3*27 - 3*27 + 23 = 4 - 4*27 = -4*26 < 0$ et finalement :

$$\forall x \in \mathbb{R}, \quad f'(x) = 3x^2 - 18x - 27 = 3(x^2 - 6x - 9).$$

Ce trinôme a pour discriminant $\Delta = 36 + 36 = 72$ et donc ses racines sont $x_\pm = (6 \pm 6\sqrt{2})/2 = 3(1 \pm \sqrt{2})$.
On est maintenant en mesure de dresser le tableau de variations de f :

x	$-\infty$		x_-		x_+		$-\infty$
$f'(x)$		$+$	0	$-$	0	$+$	
f	$-\infty$	↗		↘		↗	$+\infty$

(c) Comme $x_- < 0 < x_+ < 3$ et avec le tableau de variation, on remarque que

$$f(x_-) \geq f(0) > 0 \quad ; \quad f(x_+) \leq f(3) < 0.$$

En appliquant le théorème de la bijection à la fonction f, continue, strictement monotone sur chacun des intervalles $]-\infty, x_-[$, $]x_-, x-+[$ et $]x_+, +\infty[$, on en déduit que l'équation $f(\lambda) = 0$ admet exactement trois solutions (une dans chacun des intervalles précités) et ces solutions sont exactement les (trois) valeurs propres (distinctes) de A. La matrice A admet donc trois valeurs propres distinctes $\lambda_1 < \lambda_2 < \lambda_3$.

(d) La fonction effectuant la dichotomie :

```
def dichotomie(f,a,b,eps = 0.001):
    while True :
        c = (a+b)*0.5
        if f(a)*f(c) < 0 :
            b = c
        else :
            a = c
        if np.abs(b-a) < eps :
            return c
```

L'implémentation de la fonction f :

```
def f(x) :
    return 23 + x*(-27+x*(-9+x))
```

Évaluation numérique des valeurs propres de A. (On calcule, par la machine, que $f(-4) < 0$ et $f(12) > 0$.)

```
lambada = []
lambada.append(dichotomie(f,-4,2))
lambada.append(dichotomie(f,0,3))
lambada.append(dichotomie(f,3,12))
print(lambada)
```

Résultat :

```
[-2.924072265625, 0.700927734375, 11.22271728515625]
```

(e) La matrice A, de taille 3×3 admet 3 valeurs propres distinctes, elle est donc diagonalisable et il existe une matrice P inversible telle que $A = PDP^{-1}$ où $D = \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{pmatrix}$.

2. On s'intéresse maintenant à l'ensemble –noté $C(A)$– des matrices 3×3 qui commutent avec A, c'est à dire :

$$C(A) = \{M \in \mathcal{M}_3(\mathbb{R}) \mid AM = MA\}.$$

On définit de manière analogue l'ensemble $C(D)$ des matrices qui commutent avec D.

(a) Il est clair que $C(A)$ est un sous-ensemble de $\mathcal{M}_3(\mathbb{R})$, non vide car contenant la matrice nulle (ou, par exemple, la matrice A). Il reste à vérifier que $C(A)$ est stable par combinaisons linéaires :

Soient $M, N \in C(A)$, $\lambda, \mu \in \mathbb{C}$. Comme, en développant pour la première égalité et en factorisant pour la dernière, on a :

$$\begin{aligned}(\lambda.M + \mu.N)A &= \lambda MA + \mu NA \\ &= \lambda AM + \mu AN \quad (M, N \in C(A)) \\ &= A(\lambda M + \mu N).\end{aligned}$$

On en déduit que $(\lambda M + \mu N) \in C(A)$.

On en conclut que $C(A)$ est un sous-espace vectoriel de $\mathcal{M}_3(\mathbb{R})$.

(b) Pour une matrice $M \in \mathcal{M}_3(\mathbb{R})$, on note $M = (m_{ij})_{1\leq i,j\leq 3}$ ses composantes. On a

$$MD = (\lambda_j m_{ij})_{1\leq i,j\leq 3} \quad \text{et} \quad DM = (\lambda_i m_{ij})_{1\leq i,j\leq 3}$$

et donc $MD = DM$ si et seulement si

$$\forall i, j \in \{1, 2, 3\}, \quad \lambda_j m_{ij} = \lambda_i m_{ij}.$$

Si $M \in C(D)$, pour $i \neq j$, comme $\lambda_i \neq \lambda_j$, et que l'on doit avoir $(\lambda_j - \lambda_i)m_{ij} = 0$ alors forcément $m_{ij} = 0$ et donc M est diagonale.

Réciproquement, si M est diagonale, il est clair que $MD = DM$.

On a donc obtenu : $M \in C(D)$ si et seulement si M est diagonale. .

(c) On a $M \in C(A)$ si et seulement si $AM = MA$ si et seulement si $P^{-1}AMP = P^{-1}MAP$ si et seulement si $P^{-1}APP^{-1}MP = P^{-1}MPP^{-1}AP$ si et seulement si $P^{-1}MP \in C(D)$ si et seulement si $P^{-1}MP$ est diagonale (par la question précédente).

On en déduit que l'application $\psi : C(A) \to C(D)$ définie par $\psi(M) = P^{-1}MP$ est bien définie, bijective et il est facile de voir qu'elle est linéaire.

Or $C(D)$ est le sev des matrices diagonales qui est de dimension 3 et donc, comme ψ est un isomorphisme, $C(A)$ est de dimension 3.

3. La famille de matrices (I_3, A, A^2, A^3) est une famille de 4 vecteurs dans $C(A)$ qui est de dimension 3, elle est donc liée. Il existe donc des coefficients réels $q_i, i = 0, 1, 2, 3$, non tous nuls tels que

$$q_0 I_3 + q_1 A + q_2 A^2 + q_3 A^3 = 0$$

ce qui, en posant le polynôme $Q = q_0 + q_1 X + q_2 X^2 + q_3 X^3 \in \mathbb{C}[X]$, non nul, signifie que $Q(A) = 0$.

4. La fonction polynomiale f est associée à un tel polynôme Q : $Q = 23 - 27X - 9X^2 + X^3$. En effet

$$Q(A) = PP^{-1}Q(A)PP^{-1} = PQ(P^{-1}AP)P^{-1} = PQ(D)P^{-1} = 0.$$

Exercice sans préparation

1. On reconnaît que la somme $\dfrac{1}{n}\displaystyle\sum_{k=1}^{n} e^{-\frac{k^2}{2n^2}}$ est une somme de Riemann associée à la fonction $f : x \mapsto e^{-\frac{1}{2}x^2}$, continue sur l'intervalle $[0,1]$.
   Le théorème sur les sommes de Riemann justifie donc l'approximation.
   Le code de calcul est le suivant :

```
def S(n):
    s = 0
    for k in range(1,n+1):
        s += np.exp(-0.5*k**2/n**2)
    return s/n
```

2. Si la v.a X suit une loi normale centrée réduite, par définition de la densité de probabilité :

$$\frac{1}{\sqrt{2\pi}}\int_0^1 e^{-\frac{1}{2}x^2}\,dx = \mathbb{P}(0 \leq X \leq 1).$$

   On peut calculer par simulation la fréquence de l'événement $[0 \leq X \leq 1]$ sur un grand nombre de tirages. La loi faible des grands nombres appliquée à la v.a. de Bernoulli $\mathbf{1}_{[0\leq X\leq 1]}$ garantie alors que cette fréquence est une approximation $\mathbb{P}(0 \leq X \leq 1)$ d'où l'on déduit une évaluation de l'intégrale de gauche.

```
def P(NS = 10_000):
    cpt = 0
    for _ in range(NS):
        cpt += int( 0<= st.norm.rvs() <=1)
    return cpt/NS*np.sqrt(2*np.pi)
```

3. Affichage :

```
print(S(100),S(10_000),S(100_000))

print(P(NS = 100_000),P(NS = 100_000),P(NS = 100_000),P(NS = 100_000))
```

   et résultats obtenus :

```
0.8536519907516997 0.8556047179196932 0.8556224245403873
0.858720714323088 0.8550359707593805 0.8508749678234929 0.8545095788217
```

   La première ligne donne les valeurs données par la première méthode, la deuxième par la deuxième méthode (avec nombres de tirages différents).
   La valeur exacte cherchée vaut 0.8556 à 10^{-4} près.
   Les valeurs successives données par la première méthode sont de plus en plus précises à mesure que `n` augmente.
   En comparaison celles données par la deuxième méthode sont des valeurs précises seulement à 0.1 près pour `NS = 10_000` et 0.01 près pour `NS = 100_000`. Dans la deuxième méthode, la possibilité de faire une « grosse erreur » existe (mais la probabilité que celle-ci se produise est faible et contrôlée, c'est le sens de la LFGN).

Agro-Veto – Planche 21

Cours

Donner le développement limité à l'ordre 4 en 0 de $x \mapsto \ln(1+x)$.

Exercice préparé

— On dispose initialement d'une urne U_0 contenant 1 boule blanche et 2 boules rouges.

— Pour tout $n \in \mathbb{N}$, on remplit ensuite l'urne U_{n+1} avec 3 boules de la façon suivante. On effectue 3 tirages avec remise dans l'urne U_n, et pour chaque boule rouge (respectivement blanche) tirée, on place une nouvelle boule rouge (respectivement blanche) dans l'urne U_{n+1}.

Pour tout $n \in \mathbb{N}$, on note Y_n le nombre de boules blanches dans l'urne U_n. En particulier $Y_0 = 1$.

1. Identifier la loi de la variable aléatoire Y_1.
2. Soit $n \in \mathbb{N}$, et $k \in \{0;1;2;3\}$. Déterminer la loi de Y_{n+1} sous la probabilité conditionnelle $\mathbb{P}_{[Y_n=k]}$, c'est-à-dire calculer, pour tout $j \in \{0;1;2;3\}$: $\mathbb{P}_{[Y_n=k]}(Y_{n+1}=j)$.
3. Écrire une fonction Python prenant en argument un entier $n \in \mathbb{N}^*$ et simulant les variables aléatoires $Y_1, \dots, Y_n$. La fonction renverra le résultat sous la forme d'une liste $[Y_0, Y_1, \dots, Y_n]$.
4. (a) Soit $n \in \mathbb{N}$. Justifier que tout $k \in \{0;1;2;3\}$, $\displaystyle\sum_{j=0}^{3} j\mathbb{P}_{[Y_n=k]}(Y_{n+1}=j) = k$.
   (b) En déduire que $\mathbb{E}[Y_{n+1}] = \mathbb{E}[Y_n]$.
   (c) En déduire l'expression de $\mathbb{E}[Y_n]$ pour tout $n \in \mathbb{N}$.

Pour tout $n \in \mathbb{N}$, on note

$$a_n = \mathbb{P}(Y_n = 0) \quad ; \quad b_n = \mathbb{P}(Y_n = 1) \quad ; \quad c_n = \mathbb{P}(Y_n = 2) \quad ; \quad d_n = \mathbb{P}(Y_n = 3).$$

5. Montrer que pour tout $n \in \mathbb{N}, b_{n+1} + c_{n+1} = \dfrac{2}{3}(b_n + c_n)$.
6. En déduire la convergence et la limite des suites (b_n) et (c_n).
7. Montrer que la suite (a_n) et la suite (d_n) sont croissantes. Montrer qu'elles convergent.
8. À l'aide de la question 4, montrer que (d_n) converge vers $1/3$. Quelle est la limite de la suite (a_n) ? Interpréter le résultat.
9. On note T le numéro de la première urne ne contenant que des boules rouges ou que des boules blanches.
   (a) Pour tout $n \in \mathbb{N}$, calculer $\mathbb{P}(T > n)$.
   (b) En déduire la loi de T et son espérance.

Exercice sans préparation

1. Écrire une fonction `somme_cumul` qui prend en entrée une liste `L` d'entiers et qui renvoie une liste `M` des sommes cumulées, c'est à dire une liste `M`, de même longueur que `L`, telle que $M[i] = \sum_{k=0}^{i} L[k]$ pour tout indice i de `L`.
2. On propose à deux joueurs A et F la résolution d'une suite infinie de problèmes numérotés $0, 1, \ldots, n, \ldots$
   Ils débutent la résolution à l'instant 0 du problème numéro 0. Dès que l'un des joueurs termine la résolution du problème k il passe au problème $k+1$.
   Le numéro d'un problème est attribué au joueur qui le résout en premier. En cas de simultanéité de la résolution d'un problème par les deux joueurs celui-ci n'est pas attribué.
   On souhaite écrire une fonction Python `repartition(dureesA,dureesF)` qui, étant donnés les deux listes des durées de résolution des n premiers problèmes par les deux joueurs, renvoie les listes des numéros des problèmes attribués à A et F pour les n premiers problèmes.
   (a) `repartition([7,1,2,1,2],[4,2,4,3,1])` renvoie `([3,4],[0,1])`.
   Quel couple renvoie l'exécution de `repartition([3,2,5,1,1,1],[4,2,2,4,4,2])` ?
   (b) Écrire la fonction recherchée.

Éléments de correction–Planche 21

Exercice avec préparation

— On dispose initialement d'une urne U_0 contenant 1 boule blanche et 2 boules rouges.

— Pour tout $n \in \mathbb{N}$, on remplit ensuite l'urne U_{n+1} avec 3 boules de la façon suivante. On effectue 3 tirages avec remise dans l'urne U_n, et pour chaque boule rouge (respectivement blanche) tirée, on place une nouvelle boule rouge (respectivement blanche) dans l'urne U_{n+1}.

Pour tout $n \in \mathbb{N}$, on note Y_n le nombre de boules blanches dans l'urne U_n. En particulier $Y_0 = 1$.
Remarque préliminaire : pour $n \geq 1$, Y_n correspond au nombre de boules blanches tirées dans U_{n-1} et $Y_n(\Omega) = \{0, 1, 2, 3\}$.

1. La variable Y_1 correspond au nombre de boules blanches tirées dans U_0.
   Les 3 tirages se faisant avec remise, ils sont indépendants. À chaque tirage, la probabilité de tirer une blanche est $1/3$ donc

$$Y_1 \text{ suit une loi binomiale } \mathcal{B}\left(3, \frac{1}{3}\right).$$

   Ainsi on a :
   - $Y_1(\Omega) = \{0, 1, 2, 3\}$
   - pour tout $k \in \{0, 1, 2, 3\}$, $\mathbb{P}(Y_1 = j) = \binom{3}{j}\left(\frac{1}{3}\right)^j\left(\frac{2}{3}\right)^{3-j}$

   Ce qui donne le tableau de loi suivant :

j	0	1	2	3
$\mathbb{P}(Y_1 = j)$	$\frac{8}{27}$	$\frac{12}{27}$	$\frac{6}{27}$	$\frac{1}{27}$

2. Soit $k \in \{0, 1, 2, 3\}$.
   On suppose que l'événement $[Y_n = k]$ est réalisé : l'urne U_n contient alors k boules blanches et $3 - k$ boules rouges.
   On effectue 3 tirages avec remise dans cette urne. La probabilité de tirer une blanche à chaque tirage est $k/3$ donc

$$Y_{n+1} \text{ sachant } [Y_n = k] \text{ suit une loi binomiale } \mathcal{B}\left(3, \frac{k}{3}\right).$$

   Ainsi pour tout $j \in \{0, 1, 2, 3\}$,

$$\mathbb{P}_{[Y_n=k]}\left(Y_{n+1} = j\right) = \binom{3}{j}\left(\frac{k}{3}\right)^j\left(\frac{3-k}{3}\right)^{3-j}.$$

   On convient que $0^0 = 1$ car si $k = 0$, on doit avoir $\mathbb{P}_{[Y_n=0]}\left(Y_{n+1} = 0\right) = 1$ et si $k = 3$, on doit avoir $\mathbb{P}_{[Y_n=3]}\left(Y_{n+1} = 3\right) = 1$.
3. Fonction Python qui simule les variables aléatoires $Y_1, \ldots, Y_n$ et renvoie la liste $[Y_0, Y_1, \ldots, Y_n]$:

```
def tirages(n):
    L = [1]
    nb_bl = 1 # nombre de blanche Y0 = 1
    for k in range(1,n+1):
        p = nb_bl/3 #proba de tirer une blanche au tirage k
        nb_bl = 0 # initialisation de Yk
        for i in range(3):
            if rd.random() < p:
                nb_bl += 1
        L.append(nb_bl) # on ajoute Yk dans L
    return L
```

4. (a) Soient $n \in \mathbb{N}$ et $k \in \{0,1,2,3\}$;
$\displaystyle\sum_{j=0}^{3} j\mathbb{P}_{[Y_n=k]}(Y_{n+1}=j) = \sum_{j=0}^{3} j\binom{3}{j}\left(\frac{k}{3}\right)^j\left(\frac{3-k}{3}\right)^{3-j}$: c'est l'espérance de la loi $\mathcal{B}(3,k/3)$.
D'après le cours, cette espérance vaut $3\times\dfrac{k}{3}=k$. Ainsi on a bien

$$\sum_{j=0}^{3} j\mathbb{P}_{[Y_n=k]}(Y_{n+1}=j) = k.$$

(b) On a : $\mathbb{E}[Y_{n+1}] = \displaystyle\sum_{j=0}^{3} j\mathbb{P}(Y_{n+1}=j)$.
On considère alors le système complet d'événements $([Y_n=k])_{k\in[\![0,3]\!]}$.
En appliquant la formule des probabilités totales , on obtient que pour tout $j \in [\![0,3]\!]$:

$$\mathbb{P}(Y_{n+1}=j) = \sum_{k=0}^{3} \mathbb{P}_{[Y_n=k]}(Y_{n+1}=j)\,\mathbb{P}(Y_n=k).$$

Donc en remplaçant dans $\mathbb{E}[Y_{n+1}]$, il vient :

$$\mathbb{E}[Y_{n+1}] = \sum_{j=0}^{3} j\left(\sum_{k=0}^{3} \mathbb{P}_{[Y_n=k]}(Y_{n+1}=j)\,\mathbb{P}(Y_n=k)\right).$$

En inversant les sommes, on obtient :

$$\mathbb{E}[Y_{n+1}] = \sum_{k=0}^{3}\left(\sum_{j=0}^{3} j\mathbb{P}_{[Y_n=k]}(Y_{n+1}=j)\right)\mathbb{P}(Y_n=k).$$

En utilisant le résultat du 4.(a), on a donc

$$\mathbb{E}[Y_{n+1}] = \sum_{k=0}^{3} k\mathbb{P}(Y_n=k) = \mathbb{E}(Y_n).$$

(c) La suite $(\mathbb{E}[Y_n])$ est une suite constante et comme $\mathbb{E}[Y_0]) = 1$, on conclut que

$$\mathbb{E}[Y_n] = 1, \text{ pour tout } n \in \mathbb{N}.$$

5. D'après 2, pour tout $k \in [\![0,3]\!]$:

$$\mathbb{P}_{[Y_n=k]}(Y_{n+1}=1) = \binom{3}{1}\left(\frac{k}{3}\right)^1\left(\frac{3-k}{3}\right)^{3-1} = 3\frac{k}{3}\left(\frac{3-k}{3}\right)^2 = \frac{k(3-k)^2}{9}$$

et

$$\mathbb{P}_{[Y_n=k]}(Y_{n+1}=2) = \binom{3}{2}\left(\frac{k}{3}\right)^2\left(\frac{3-k}{3}\right)^{3-2} = 3\left(\frac{k}{3}\right)^2\frac{3-k}{3} = \frac{k^2(3-k)}{9}.$$

En appliquant la formule des probabilités totales, on a donc :

$$b_{n+1} = \mathbb{P}(Y_{n+1}=1) = \sum_{k=0}^{3}\mathbb{P}_{[Y_n=k]}(Y_{n+1}=1)\,\mathbb{P}(Y_n=k) = \sum_{k=0}^{3}\frac{k(3-k)^2}{9}\mathbb{P}(Y_n=k).$$

Or $k=0$ et pour $k=3$, $k(3-k)^2=0$ donc

$$b_{n+1} = \sum_{k=1}^{2}\frac{k(3-k)^2}{9}\mathbb{P}(Y_n=k).$$

De même,

$$c_{n+1} = \mathbb{P}(Y_{n+1}=2) = \sum_{k=0}^{3}\mathbb{P}_{[Y_n=k]}(Y_{n+1}=2)\,\mathbb{P}(Y_n=k) = \sum_{k=1}^{2}\frac{k^2(3-k)}{9}\mathbb{P}(Y_n=k).$$

Ainsi

$$\begin{aligned} b_{n+1}+c_{n+1} &= \frac{1}{9}\sum_{k=1}^{2}(k^2(3-k)+k(3-k)^2)\mathbb{P}(Y_n=k) \\ &= \frac{1}{9}\sum_{k=1}^{2}k(3-k)[k+(3-k)]\mathbb{P}(Y_n=k) \\ &= \frac{1}{3}\sum_{k=1}^{2}k(3-k)\mathbb{P}(Y_n=k) \\ &= \frac{1}{3}(2\mathbb{P}(Y_n=1)+2\mathbb{P}(Y_n=2)) \end{aligned}$$

Ainsi on a bien

$$b_{n+1}+c_{n+1} = \frac{2}{3}(b_n+c_n),\ \forall n\in\mathbb{N}.$$

6. La suite (b_n+c_n) est une suite géométrique de raison $2/3$.
On a donc

$$\forall n\in\mathbb{N},\quad b_n+c_n = \left(\frac{2}{3}\right)^n(b_0+c_0)$$

avec $b_0+c_0=1$ car $Y_0=1$.
Comme $2/3 \in]-1,1[$ donc $\lim\limits_{n\to+\infty}\left(\frac{2}{3}\right)^n = 0$.
Ainsi $\lim\limits_{n\to+\infty}(b_n+c_n)=0$.
Or :

$$\forall n\in\mathbb{N},\quad 0\le b_n\le b_n+c_n \quad\text{et}\quad 0\le c_n\le b_n+c_n.$$

On en déduit, par encadrement, que

les deux suites (b_n) et (c_n) convergent vers 0.

7. S'il n'y a pas de boules blanches dans l'urne U_n, le $(n+1)$-ème tirage ne donnera que des rouges. Donc il ne pourra y avoir de boules blanches dans l'urne U_{n+1}.
Il est donc clair que l'événement $[Y_n = 0]$ implique l'événement $[Y_{n+1} = 0]$:
$$[Y_n = 0] \subset [Y_{n+1} = 0].$$
On en déduit que pour tout $n \in \mathbb{N}$, $\mathbb{P}(Y_n = 0) \leq \mathbb{P}(Y_{n+1} = 0)$ i.e. $a_n \leq a_{n+1}$.
De même, $[Y_n = 3]$ implique l'événement $[Y_{n+1} = 3]$ donc $\mathbb{P}(Y_n = 3) \leq \mathbb{P}(Y_{n+1} = 3)$ i.e. $d_n \leq d_{n+1}$.
On a donc prouvé que
$$\text{les suites } (a_n) \text{ et } (d_n) \text{ sont croissantes.}$$
Ces suites étant croissantes et majorées par 1 (a_n et c_n sont des probabilités), on conclut que
$$\text{les suites } (a_n) \text{ et } (d_n) \text{ convergent.}$$

8. Par définition de l'espérance :
$$\forall n \in \mathbb{N}, \quad \mathbb{E}[Y_n] = \sum_{k=0}^{3} k\mathbb{P}(Y_n = k) = b_n + 2c_n + 3d_n.$$
D'après la question 4 , $\mathbb{E}[Y_n] = 1$ pour tout $n \in \mathbb{N}$.
Ainsi, pour tout $n \in \mathbb{N}$
$$b_n + 2c_n + 3d_n = 1 \quad \text{ie} \quad d_n = \frac{1}{3}(1 - b_n - 2c_n).$$
En passant à la limite dans cette égalité, comme $\lim\limits_{n\to+\infty} b_n = \lim\limits_{n\to+\infty} c_n = 0$, on en déduit que
$$\text{la suite } (d_n) \text{ converge vers } 1/3.$$
Par ailleurs, la famille $([Y_n = 0], [Y_n = 1], [Y_n = 2], [Y_n = 3])$ est un système complet d'événements donc $a_n + b_n + c_n + d_n = 1$ pour tout $n \in \mathbb{N}$.
Ainsi
$$\lim_{n\to+\infty} a_n = \lim_{n\to+\infty} (1 - b_n - c_n - d_n) = 1 - 0 - 0 - \frac{1}{3} = \frac{2}{3}.$$
Au bout d'un très grand nombre de tirages, il est quasi-certain que l'urne sera unicolore : 2 chances sur 3 qu'elle soit composée de 3 rouges, 1 chance sur 3 d'avoir 3 blanches.

9. On note T le numéro de la première urne ne contenant que des boules rouges ou que des boules blanches.
   (a) Soit $n \in \mathbb{N}$. L'événement $[T > n]$ signifie que l'urne U_n contient au moins une blanche et au moins une rouge.
   Donc $[T > n] = [Y_n = 1] \cup [Y_n = 2]$. Ces deux événements étant incompatibles, on a :
   $$\mathbb{P}(T > n) = \mathbb{P}(Y_n = 1) + \mathbb{P}(Y_n = 2) = b_n + c_n.$$
   D'après 6 : $b_n + c_n = \left(\frac{2}{3}\right)^n$.
   Ainsi,
   $$\forall n \in \mathbb{N}, \quad \mathbb{P}(T > n) = \left(\frac{2}{3}\right)^n.$$

(b) Loi de T :

— $T(\Omega) = \mathbb{N}^*$ car il faut au minimum un tirage pour qu'il ne reste qu'une seule couleur.

— Soit $n \in \mathbb{N}^*$ fixé. On a

$$[T > n-1] = [T = n] \cup [T > n]$$

réunion d'événements incompatibles donc

$$\mathbb{P}(T > n-1) = \mathbb{P}(T = n) + \mathbb{P}(T > n).$$

Ainsi

$$\mathbb{P}(T = n) = \mathbb{P}(T > n-1) - \mathbb{P}(T > n) = \left(\frac{2}{3}\right)^{n-1} - \left(\frac{2}{3}\right)^{n} = \left(\frac{2}{3}\right)^{n-1}\left(1 - \frac{2}{3}\right).$$

On observe que T suit une loi géométrique de paramètre $\dfrac{1}{3}$.

D'après le cours on a $\mathbb{E}(T) = \dfrac{1}{1/3}$ soit $\mathbb{E}(T) = 3$.

Exercice sans préparation

1.
```
def somme_cumul(L):
    M = []
    s = 0
    for elt in L:
        s += elt
        M.append(s)
    return M
```

2. (a) Si `dureeA = [3,2,5,1,1,1]` et `dureeF = [4,2,2,4,4,2]`, le calcul des durées cumulées de chacune de ces deux listes donne :
   - `[3,5,10,11,12,13]` pour le joueur A
   - `[4,6,8,12,16,18]` pour le joueur F.

   En position k, on obtient l'instant où le joueur a terminé la résolution du problème k.

   En comparant les termes en position k de ces deux listes, on sait à quel joueur doit être attribué le problème k.

   `repartition([3,2,5,1,1,1],[4,2,2,4,4,2])` renvoie le couple `([0,1,3,4,5],[2])` car le seul problème pour lequel F termine sa résolution avant A est le problème 2 (en effet 8 $<$10).

   (b)
```
def repartition(dureeA, dureeF):
    attrA, attrF = [], []
    duree_cumulA = somme_cumul(dureeA)
    duree_cumulF = somme_cumul(dureeF)
    n = len(dureeA)
    for k in range(n):
        if duree_cumulA[k] < duree_cumulF[k]: # A résout en premier
le pb k
            attrA.append(k)
        elif duree_cumulF[k] < duree_cumulA[k]: # F résout en premier
le pb k
            attrF.append(k)
    return attrA, attrF
```