Feuille Oraux_2 : Préparation à l'informatique de l'oral de l'agro.

Ex 1 : On considère φ l'endomorphisme de $\mathbb{R}^3$, dont la matrice représentative dans la base canonique est la matrice A de $\mathscr{M}_3(\mathbb{R})$ suivante :

$$A = \begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 0 & 0 & 3 \end{pmatrix}$$

1) Montrer que le spectre de l'endomorphisme φ est : $Sp(\varphi) = \{1, 3\}$.
   L'endomorphisme φ est-il diagonalisable ?
2) On note $a_1 = (1, 1, 0), a_2 = (0, 0, 1)$ et $a_3 = (1, -1, 0)$.
   Montrer que la famille $\mathscr{B} = (a_1, a_2, a_3)$ est une base de $\mathbb{R}^3$ et déterminer la matrice M de l'endomorphisme φ dans la base $\mathscr{B}$.
3) 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`.

Ex 2 : On considère une urne contenant n boules numérotées de 1 à n et on effectue n tirages successifs d'une boule avec remise. On note X la variable aléatoire représentant le nombre de numéros distincts obtenus.

1) Déterminer la loi de X dans les cas $n = 2$ et $n = 3$.
   Que vaut l'espérance de X dans les cas $n = 2$ et $n = 3$?
2) a. Ecrire une fonction Python d'argument n qui simule l'expérience et renvoie la liste des numéros tirés.
   b. Ecrire une fonction Python d'argument n qui simule la variable X.
   On pourra obtenir l'ensemble des valeurs d'une liste L avec la commande set (L) et obtenir le cardinal d'un ensemble s avec la commande len (s).
   c. Ecrire une fonction Python d'argument n qui calcule une valeur approchée de l'espérance de X.

Ex 3 : 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 :

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

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 Python qui prend en entrée un entier n et qui renvoie la matrice K_n.
3) Utiliser la fonction précédente 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 ?

Ex 4 : Soit $(\Omega, \mathcal{A}, \mathcal{P})$ un espace probabilisé et X une variable aléatoire sur cet espace suivant la loi normale centrée réduite.
On note φ la fonction densité continue de X, Φ sa fonction de répartition, f la fonction $x \longmapsto P(X > x)$ et (E) l'équation différentielle :

$$\forall x \in \mathbb{R}, \quad y'(x) + xy(x) = 0$$

On considère la fonction M définie par : $\forall x \in \mathbb{R}, M(x) = \dfrac{f(x)}{\varphi(x)}$.

1) Résoudre l'équation différentielle (E). Montrer que φ est solution de (E).
   [...]
5) Écrire un script Python permettant d'afficher sur un même graphique les courbes des fonctions

$$x \longmapsto M(x),\ x \longmapsto \frac{1}{x} \text{ et } x \longmapsto \frac{1}{x} - \frac{1}{x^3} \text{ sur l'intervalle }]0, 5]$$

   Pour calculer $f(x)$, on pourra utiliser la fonction `norm.sf()` du module `scipy.stats`

Ex 5 : Soient a_1, a_2 et a_3 trois réels distincts. Soit A la matrice défine par :

$$A = \begin{pmatrix} 1 & a_1 & a_1^2 \\ 1 & a_2 & a_2^2 \\ 1 & a_3 & a_3^2 \end{pmatrix}$$

Pour tout $i \in [\![1,3]\!]$, on pose :

$$s_i = \sum_{\substack{j=1 \\ j\neq i}}^{3} a_j, \quad p_i = \prod_{\substack{j=1 \\ j\neq i}}^{3} a_j \quad \text{et } d_i = \prod_{\substack{j=1 \\ j\neq i}}^{3} (a_i - a_j).$$

1) Ecrire une fonction Python qui prend en argument une liste contenant les valeurs de a_1, a_2 et a_3 et renvoie la matrice A, sous forme de liste de listes.
2) Ecrire une fonction Python qui prend en argument un entier i et une liste contenant les valeurs de a_1, a_2 et a_3, et renvoie une liste contenant les valeurs de s_i, p_i et d_i.

Ex 6 : Soient $n \in \mathbb{N}^*$ et $(T_1, \ldots, T_n)$ une famille de variables aléatoires mutuellement indépendantes suivant la loi normale centrée réduite $\mathcal{N}(0,1)$. On note pour tout $i \in [\![1,n]\!], X_i = T_i^2$ et $H_n = \sum_{i=1}^{n} X_i$.

On souhaite montrer que pour tout $t > 0$:

$$P\left(H_n - E(H_n) \geqslant 2\sqrt{nt} + 2t\right) \leqslant \mathrm{e}^{-t}$$

1) Écrire une fonction simule en Python qui prend en entrée un entier n et qui simule la variable aléatoire H_n.
2) Montrer que pour tout $i \in [\![1,n]\!], X_i$ admet une espérance et déterminer sa valeur.
3) En déduire que H_n admet une espérance et qu'elle vaut n.
4) À l'aide de la question 1a donner une estimation pour $t = 1, 2, 3$ et $n = 100$ de $P\left(H_n - E(H_n) \geqslant 2\sqrt{nt} + 2t\right)$ ainsi que la valeur $\exp(-t)$.

Ex 7 : Soit N un entier supérieur ou égal à 2 . On considère une urne contenant N boules indiscernables au toucher, numérotées de 1 à N. On procède à des tirages successifs d'une boule avec remise de la boule dans l'urne avant le tirage suivant. On note pour tout $k \geqslant 1, X_k$ le numéro obtenu au k-ième tirage, et Z_k le nombre de numéros distincts obtenus au cours des k premiers tirages.

1) Écrire une fonction Python `NBDiff(L)` prenant en argument une liste L et qui renvoie le nombre d'éléments distincts présents dans cette liste.
2) Écrire une fonction Python `Z(N,k)` qui, prenant en arguments les valeurs de N et k, et renvoie une simulation de Z_k.
3) Estimer l'espérance de Z_k à l'aide de votre programme, et conjecturer son comportement lorsque :
   a. $N = 10$ et $k \to +\infty$
   b. $k = 10$ et $N \to +\infty$
   c. $N = k$ et $N \to +\infty$

Ex 8 : On lance indéfiniment une pièce équilibrée.

On s'intéresse au rang du lancer auquel on obtient pour la première fois la succession des résultats «Pile,Pile,Face», dans cet ordre. On note alors X la variable aléatoire égale au rang du lancer où, pour la première fois, on obtient cette configuration. Si celle-ci n'est jamais obtenue, on conviendra que X vaut -1 .

Par exemple, si on obtient dans cet ordre : Pile, Face, Face, Pile, Pile, Pile, Face, alors X prend la valeur 7.

[...]

1) Écrire une fonction Python sans argument qui simule les lancers de dés jusqu'à l'apparition de la séquence « Pile, Pile, Face »et qui renvoie sous forme de liste les résultats de tous les lancers réalisés.
2) Utiliser la fonction précédente pour émettre une conjecture quant à l'existence et la valeur éventuelle de l'espérance de X.

Ex 9 : On note $\mathcal{B} = (e_1, e_2, e_3)$ la base canonique de $\mathbb{R}^3$. On considère l'endomorphisme f de $\mathbb{R}^3$ défini par

$$\forall (x, y, z) \in \mathbb{R}^3, f(x, y, z) = (x + y - z, 2y, -x + y + z).$$

On considère aussi l'endomorphisme g de $\mathbb{R}^3$ dont la matrice dans $\mathcal{B}$ est

$$B = \begin{pmatrix} 0 & -2 & -5 \\ -2 & 0 & 4 \\ 1 & 1 & 0 \end{pmatrix}$$

On pose $u = e_1 - e_2 = (1, -1, 0)$ et $v = g(e_1) + e_1$.

1) a. Déterminer la matrice A de f dans la base $\mathcal{B}$.

   b. À l'aide de Python, déterminer les valeurs propres de g et conjecturer la dimension de chaque sous-espace propre de f. L'endomorphisme f semble-t-il diagonalisable ?

   On rappelle que, dans la bibliothèque Python `numpy`*, la fonction* `linalg.eig(A)` *renvoie les valeurs propres (réelles et complexes) de A et la matrice dont les colonnes sont des vecteurs propres associés à ces valeurs propres (dans le même ordre).*

   [...]

3) On note $E = \{M \in \mathcal{M}_3(\mathbb{R}) / AM = MB\}$.

   Écrire une fonction Python E(M) qui prend en argument une matrice M de $\mathcal{M}_3(\mathbb{R})$ qui renvoie `True` si $M \in E$ et `False` sinon.

   On rappelle que, si N est une matrice contenant des booléens, l'instruction `N.all()` *renvoie* `True` *si* `N` *ne contient que des* `True` *et renvoie* `False` *sinon.*

   On rappelle aussi que, dans la bibliothèque Python `numpy`*, la fonction* `dot(N, P)` *renvoie le produit matriciel NP.*

Ex 10 : ***Rappel*** *: la méthode d'Euler permet d'approcher la solution φ d'une équation différentielle au voisinage d'un point connu, en utilisant, de proche en proche, l'approximation affine de la fonction au voisinage de chaque point :*

en tout point a, $\quad \varphi(a + h) \simeq \varphi(a) + h\varphi'(a)$, $\quad$ lorsque h est petit (positif ou négatif)

Soit $I =]1, +\infty[$, et on considère l'équation différentielle (E) :

$$\forall t \in I, -t^2 y'(t) + ty(t) = (y(t))^2$$

ayant pour inconnue une fonction $y : I \to \mathbb{R}$ dérivable sur I.

1) a. Montrer que la fonction $f : t \mapsto \dfrac{t}{\ln(t)}$ est une solution de l'équation (E) sur $]1, +\infty[$.

   b. Représenter en Python la fonction f sur l'intervalle $[2, 4]$.

2) On cherche une solution y de l'équation différentielle (E) vérifiant $y(e) = 3$.

   Sous réserve d'existence de y, utiliser la méthode d'Euler pour représenter en Python un tracé approximatif de la courbe représentative de y sur l'intervalle $[2, 4]$ en partant du point $(e; 3)$.

   On pourra tracer successivement une solution sur $[e, 4]$ puis une solution sur $[2, e]$.

   Comparer avec le graphe obtenu à la question 1.

Ex 11 : Soit N un entier supérieur ou égal à 3 . Une urne contient N boules dont $N - 2$ sont blanches et 2 sont noires. On tire au hasard, une par une et sans remise, les N boules de cette urne.

Les tirages étant numérotés de 1 à N, on note X_1 la variable aléatoire égale au numéro du tirage qui a fourni, pour la première fois, une boule noire et X_2 la variable aléatoire égale au numéro du tirage qui a fourni, pour la deuxième fois, une boule noire.

1) Lorsque $N = 10$, simuler informatiquement une expérience et afficher les valeurs prises par X_1 et X_2.

   On rappelle à cet effet que la fonction `random()` *de la bibliothèque Python* `random` *renvoie un nombre pseudo-aléatoire que l'on peut supposer uniformément distribué entre 0 et 1 .*

2) Soient i et j deux entiers de l'intervalle $[\![1, N]\!]$. Montrer que la loi du couple (X, Y) est donnée par :

$$P((X_1 = i) \cap (X_2 = j)) = \begin{cases} 0 & \text{si} \quad 1 \leqslant j \leqslant i \leqslant N \\ \dfrac{2}{N(N-1)} & \text{si} \quad 1 \leqslant i < j \leqslant N \end{cases}$$

Ex 12 : **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 entier naturel k, on note $c_k = \dfrac{a_k+b_k}{2}$ et :* $\left|\begin{array}{l} \text{si } f(a_k)\,f(c_k) \leqslant 0, \text{ alors } a_{k+1} = a_k \text{ et } b_{k+1} = c_k \\ \text{sinon, on pose } a_{k+1} = c_k \text{ et } b_{k+1} = b_k. \end{array}\right.$

On sait alors que les suites (a_k) et (b_k) convergent toutes les deux vers γ, en vérifiant :

$$\forall k \in \mathbb{N}, a_k \leqslant \gamma \leqslant b_k \text{ et } \forall k \in \mathbb{N}, b_k - a_k = \frac{b-a}{2^k}$$

On peut montrer que si l'entier k est tel que $\dfrac{b-a}{2^k} \leqslant \varepsilon$, alors a_k et b_k sont des valeurs approchées de γ à ε-près.

1) On considère pour tout entier $n \geqslant 2$, la fonction f_n définie sur $\mathbb{R}_+^*$ par : $f_n(x) = x^n - \ln x - n$.
   a. Dresser le tableau de variations de f_n.
   b. *On rappelle et on admet l'inégalité suivante :* $\forall x \in \mathbb{R}_+^*, \quad \ln(x) \leqslant x - 1 \quad (*)$.
   En déduire que pour tout entier $n \geqslant 2$, il existe un unique réel $u_n \in]\,0, n^{-\frac{1}{n}}\,[$ tel que $f_n(u_n) = 0$ et un unique réel $v_n \in]\,n^{-\frac{1}{n}}, +\infty\,[$ tel que $f_n(v_n) = 0$.
2) **Étude de la suite (v_n)**
   La suite $(v_n)_{n\geqslant 2}$ vérifie donc l'égalité : $\forall n \geqslant 2, \quad v_n^n - \ln(v_n) - n = 0$.
   a. Justifier en utilisant si besoin $(*)$ que $f_n\left((2n)^{\frac{1}{n}}\right) > 0$, puis en déduire que : $\forall n \geqslant 2, v_n \leqslant (2n)^{\frac{1}{n}}$.
   b. En utilisant l'algorithme de dichotomie, déterminer des valeurs approchées à 10^{-3} près des termes v_n pour n allant de 2 à 30 .
   Représenter la suite (v_n) graphiquement.

Ex 13 : On définit la fonction numérique f sur $\mathbb{R}_+^*$ par la relation :

$$\forall x \in \mathbb{R}_+^*, \quad f(x) = \int_0^1 \frac{\cos(t)}{x+t}\,\mathrm{d}t$$

1) Proposer une fonction Python prenant en argument un réel $x > 0$ et retournant une approximation de $f(x)$.
2) Proposer une approximation du graphe de la fonction f à l'aide de l'outil informatique.
   Conjecturer un résultat sur la monotonie de la fonction f et sur les limites au bord de son domaine de définition.

Ex 14 : Un joueur dispose de N dés équilibrés à 6 faces. Il lance une première fois ceux-ci et on note X_1 le nombre de 6 obtenus. Il met de côté les dés correspondants et relance les autres dés (s'il en reste). On note X_2 le nombre de 6 obtenus et on répète l'expérience définissant ainsi une suite de variables aléatoires $X_1, X_2, \ldots$. S'il ne reste plus de dés au m-ème lancer, on a alors, pour tout $k \geqslant m, X_k = 0$.
Pour tout entier naturel non nul n, on définit la variable $S_n = X_1 + X_2 + \cdots + X_n$, qui correspond alors au nombre de 6 obtenus après n lancers.

1) Écrire une fonction Python X(N) qui prend en argument le nombre de dés N et renvoie la valeur de X_1.
2) En déduire une fonction Python S(N, n) qui prend en arguments les nombre de dés N et n le nombre de lancers effectués et renvoie la valeur de S_n.

Ex 15 : On considère la suite $(u_n)_{n\geqslant 1}$ telle que $u_1 \in]\,0;\pi\,[$ et pour tout $n \in \mathbb{N}^*$: $\quad u_{n+1} = \left(1 + \dfrac{1}{n}\right)\sin(u_n)$.

1) Montrer que pour tout $n \geqslant 3$: $\quad 0 < u_n < \dfrac{\pi}{2}$.
2) Déterminer le seul réel vers lequel la suite (u_n) peut converger.
3) Représenter graphiquement u_n en fonction de n pour plusieurs valeurs de u_1, puis émettre une conjecture sur la monotonie de la suite (u_n).