



PRÉPASIMPLE

prepasimple.fr

EDHEC

Mathématiques

2021

Corrigé de mathématiques approfondies

Mathématiques approfondies — Prépa ECG

Annales de concours · Corrigé

Exercice 1

1a) La croissance de (a_n) permet de comparer chaque terme à a_n . Pour $0 \leq k \leq n-1$, on a $a_k \leq a_n$. En sommant puis en divisant par $n > 0$:

$$b_n = \frac{1}{n} \sum_{k=0}^{n-1} a_k \leq a_n.$$

Pour étudier la croissance des moyennes, on utilise la relation $(n+1)b_{n+1} = nb_n + a_n$. Elle donne :

$$b_{n+1} - b_n = \frac{a_n - b_n}{n+1} \geq 0.$$

Ainsi, $b_n \leq a_n$ pour tout $n \geq 1$, et (b_n) est croissante.

1b) Une suite croissante de limite réelle ℓ vérifie $a_n \leq \ell$ pour tout n . En effet, pour $m \geq n$, $a_n \leq a_m$, et le passage à la limite en m donne cette majoration.

La question précédente fournit donc $b_n \leq \ell$. La suite (b_n) est croissante et majorée : le théorème de la limite monotone assure qu'elle converge vers un réel ℓ' . En passant à la limite dans $b_n \leq \ell$, on obtient $\ell' \leq \ell$.

1c) Rectification locale : le PDF annonce une « égalité » alors que la formule imprimée est une inégalité. C'est bien cette inégalité qui doit être démontrée.

On partage la somme en deux blocs de même longueur :

$$2nb_{2n} = \sum_{k=0}^{n-1} a_k + \sum_{k=n}^{2n-1} a_k = nb_n + \sum_{k=n}^{2n-1} a_k.$$

Dans le second bloc, chaque a_k est supérieur ou égal à a_n . Ainsi, $2nb_{2n} \geq nb_n + na_n$. En divisant par $2n > 0$, on conclut :

$$b_{2n} \geq \frac{b_n + a_n}{2}.$$

1d) Méthode 1 — Utiliser l'inégalité précédente.

Les suites (b_n) et (a_n) convergent respectivement vers ℓ' et ℓ . La sous-suite (b_{2n}) converge également vers ℓ' . Le passage à la limite dans 1c donne :

$$\ell' \geq \frac{\ell' + \ell}{2}.$$

Donc $\ell' \geq \ell$. En réunissant cette inégalité et celle de 1b, on obtient $\ell' = \ell$. Ainsi

$$\lim_{n \rightarrow +\infty} b_n = \ell.$$

Méthode 2 — Isoler un nombre fini de termes.

Cette preuve directe explique pourquoi une moyenne a la même limite que ses termes, même sans monotonie. Soit $\varepsilon > 0$. Comme $a_k \rightarrow \ell$, il existe $N \geq 1$ tel que $|a_k - \ell| \leq \varepsilon/2$ pour $k \geq N$. Posons

$$C = \sum_{k=0}^{N-1} |a_k - \ell|,$$

qui est fini. Pour $n > N$:

$$|b_n - \ell| \leq \frac{1}{n} \sum_{k=0}^{n-1} |a_k - \ell| \leq \frac{C}{n} + \frac{n-N}{n} \frac{\varepsilon}{2} \leq \frac{C}{n} + \frac{\varepsilon}{2}.$$

Pour n assez grand, $C/n \leq \varepsilon/2$, d'où $|b_n - \ell| \leq \varepsilon$. Cela établit aussi $b_n \rightarrow \ell$. La première méthode est celle attendue ici ; la seconde s'applique plus largement aux moyennes de suites convergentes.

2a) On raisonne par récurrence sur n .

Au rang 0, $u_0 = 1$ est défini et supérieur ou égal à 1. Supposons u_n défini et $u_n \geq 1$.

Alors $u_n^2 + u_n \geq 2 > 0$, donc sa racine carrée est définie et $u_{n+1} = \sqrt{u_n^2 + u_n} \geq \sqrt{2} \geq 1$.

La récurrence établit que tous les termes sont définis et que $u_n \geq 1$ pour tout $n \in \mathbb{N}$.

2b) Puisque $u_n > 0$, on a $u_{n+1}^2 = u_n^2 + u_n > u_n^2$. Les deux termes étant positifs, $u_{n+1} > u_n$: la suite est strictement croissante.

Méthode 1 — Exclure une limite finie.

Si (u_n) était majorée, sa croissance entraînerait sa convergence vers un réel $L \geq 1$. Par continuité de $x \mapsto \sqrt{x^2 + x}$ sur $[1, +\infty[$, la relation de récurrence donnerait $L = \sqrt{L^2 + L}$, puis $L^2 = L^2 + L$ et $L = 0$. C'est incompatible avec $L \geq 1$.

La suite est donc croissante et non majorée, ce qui entraîne $u_n \rightarrow +\infty$.

Méthode 2 — Construire une minoration explicite.

On a $u_{k+1}^2 - u_k^2 = u_k \geq 1$. En sommant pour $k = 0, \dots, n-1$, on obtient $u_n^2 - u_0^2 \geq n$, donc $u_n \geq \sqrt{n+1}$. Cette minoration tend vers $+\infty$, ce qui prouve directement $u_n \rightarrow +\infty$.

La première méthode utilise les propriétés des suites monotones ; la seconde donne en plus un encadrement utilisable numériquement.

2c) Question adaptée au programme actuel. Code adapté en Python. Au début de chaque test de boucle, les variables représentent respectivement n , u_{n-1} et S_n . Pour passer au rang suivant, on calcule d'abord u_n , puis on l'ajoute à S_n .

Python

```
import numpy as np

n = 1
u = 1.0
S = 1.0
```

```
while S <= 1000:
    u = np.sqrt(u**2 + u)
    S = S + u
    n = n + 1
print(n)
```

Après une itération, les variables représentent le nouveau rang n , u_{n-1} et S_n : l'invariant est conservé. Comme $S_n \geq n$, la boucle termine. Tous les rangs sont parcourus dans l'ordre, et le test $S \leq 1000$ conserve le cas d'égalité : la sortie est donc le premier rang pour lequel $S_n > 1000$.

Ce code a été exécuté : il affiche 64. On obtient numériquement $S_{63} \approx 999,982$ et $S_{64} \approx 1031,620$. Ces valeurs constituent un contrôle numérique du script ; les arguments précédents établissent sa logique.

3a) On rationalise la différence pour éliminer la soustraction de deux termes qui tendent vers l'infini. Comme $u_n > 0$:

$$u_{n+1} - u_n = \frac{u_n}{\sqrt{u_n^2 + u_n} + u_n} = \frac{1}{\sqrt{1 + 1/u_n} + 1}.$$

D'après 2b, $u_n \rightarrow +\infty$, donc $1/u_n \rightarrow 0$. Par continuité de la racine carrée et du quotient dont le dénominateur tend vers $2 \neq 0$, on conclut :

$$\lim_{n \rightarrow +\infty} (u_{n+1} - u_n) = \frac{1}{2}.$$

3b) Méthode 1 — Étudier la dérivée.

La fonction f est dérivable sur $[1, +\infty[$, avec dérivée à droite en 1, et :

$$f'(x) = \frac{2x + 1}{2\sqrt{x^2 + x}} - 1.$$

Les quantités $2x + 1$ et $2\sqrt{x^2 + x}$ sont positives. Or $(2x + 1)^2 - 4(x^2 + x) = 1 > 0$, donc $2x + 1 > 2\sqrt{x^2 + x}$ et $f'(x) > 0$. La fonction f est strictement croissante.

Puisque (u_n) est strictement croissante et $u_{n+1} - u_n = f(u_n)$, la suite $(u_{n+1} - u_n)$ est strictement croissante.

Méthode 2 — Utiliser une expression sans dérivation.

Pour $x \geq 1$, la rationalisation donne $f(x) = 1/(\sqrt{1+1/x} + 1)$. Si $1 \leq x < y$, alors $1/x > 1/y$, donc $\sqrt{1+1/x} + 1 > \sqrt{1+1/y} + 1 > 0$. En prenant les inverses, $f(x) < f(y)$.

On retrouve la stricte croissance de f , puis celle de $f(u_n) = u_{n+1} - u_n$. Cette seconde méthode exploite directement la formule utile pour la limite en 3a.

3c) Posons $a_k = u_{k+1} - u_k$ pour tout $k \in \mathbb{N}$. D'après 3a et 3b, (a_k) est croissante et converge vers $1/2$. La question 1 s'applique donc :

$$\frac{1}{n} \sum_{k=0}^{n-1} (u_{k+1} - u_k) \rightarrow \frac{1}{2}.$$

La somme est télescopique et vaut $u_n - u_0 = u_n - 1$. Par conséquent $(u_n - 1)/n \rightarrow 1/2$, puis $u_n/n \rightarrow 1/2$ car $1/n \rightarrow 0$.

Ainsi $u_n/(n/2) \rightarrow 1$, autrement dit $u_n \sim_{n \rightarrow +\infty} n/2$.

4a) La récurrence fournit $u_k = u_{k+1}^2 - u_k^2$. En sommant pour $k = 0, \dots, n-1$:

$$S_n = \sum_{k=0}^{n-1} (u_{k+1}^2 - u_k^2) = u_n^2 - u_0^2 = u_n^2 - 1.$$

D'après 3c, $u_n^2 \sim n^2/4$. Comme $u_n^2 \rightarrow +\infty$, on a $(u_n^2 - 1)/u_n^2 \rightarrow 1$. Par produit des rapports, on conclut :

$$S_n = u_n^2 - 1 \quad \text{et} \quad S_n \sim_{n \rightarrow +\infty} \frac{n^2}{4}.$$

4b) Question adaptée au programme actuel. Code adapté en Python. Comme $u_n > 0$, l'identité précédente donne les équivalences

$$S_n > 1000 \iff u_n^2 > 1001 \iff u_n > \sqrt{1001}.$$

Python

```
import numpy as np

n = 0
u = 1.0
while u <= np.sqrt(1001):
    u = np.sqrt(u**2 + u)
    n = n + 1
print(n)
```

Au début de chaque test, u vaut u_n . La boucle parcourt les rangs successifs jusqu'au premier franchissement strict de $\sqrt{1001}$. Elle termine puisque $u_n \rightarrow +\infty$.

Ce code a été exécuté et affiche lui aussi 64. L'initialisation au rang 0 est valide : même si S_0 n'a pas été défini dans le sujet, $u_0 = 1$ est sous le seuil, et le premier rang répondant à la condition est strictement positif.

Exercice 2

1a) Puisque l'exponentielle est strictement positive, $Y > 0$ et donc $F_Y(x) = 0$ si $x \leq 0$.

Pour $x > 0$, la croissance de l'exponentielle et celle du logarithme donnent $\{e^Z \leq x\} = \{Z \leq \ln x\}$. Par définition de Φ :

$$F_Y(x) = \begin{cases} 0 & \text{si } x \leq 0, \\ \Phi(\ln x) & \text{si } x > 0. \end{cases}$$

1b) Sur $]0, +\infty[$, la fonction F_Y est dérivable par composition, et la densité normale vérifie $\Phi'(z) = \frac{1}{\sqrt{2\pi}} e^{-z^2/2}$. Ainsi :

$$F'_Y(x) = \frac{1}{x\sqrt{2\pi}} \exp\left(-\frac{(\ln x)^2}{2}\right).$$

Sur $] -\infty, 0[$, sa dérivée est nulle. De plus, $\ln x \rightarrow -\infty$ lorsque $x \rightarrow 0^+$, et $\Phi(z) \rightarrow 0$

lorsque $z \rightarrow -\infty$. Donc F_Y est continue en 0. Puisque Y est admise à densité, la dérivée de F_Y sur les deux intervalles, avec valeur choisie 0 au point 0, fournit la densité demandée.

Pour vérifier directement sa normalisation, on utilise la primitive $x \mapsto \Phi(\ln x)$ sur un segment $[a, b] \subset]0, +\infty[$, puis on passe aux bornes impropres :

$$\int_a^b \frac{1}{x\sqrt{2\pi}} e^{-(\ln x)^2/2} dx = \Phi(\ln b) - \Phi(\ln a) \rightarrow 1 \text{ lorsque } a \rightarrow 0^+ \text{ et } b \rightarrow +\infty.$$

Cette limite utilise $\Phi(z) \rightarrow 0$ lorsque $z \rightarrow -\infty$ et $\Phi(z) \rightarrow 1$ lorsque $z \rightarrow +\infty$. Ainsi la fonction positive f_Y donnée dans l'énoncé est bien normalisée. La valeur d'une densité en un point isolé ne modifie pas la loi.

2a) Chaque X_n est une variable finie, donc ses deux premiers moments existent. Par la formule de l'espérance :

$$E(X_n) = p - (1 - p) = 2p - 1.$$

Comme $X_n^2 = 1$, la formule de Koenig-Huygens donne :

$$V(X_n) = E(X_n^2) - E(X_n)^2 = 1 - (2p - 1)^2 = 4p(1 - p).$$

Ces valeurs ne dépendent pas de n .

2b) Un produit de nombres appartenant à $\{-1, 1\}$ appartient à $\{-1, 1\}$. Ces deux valeurs ont une probabilité strictement positive : toutes les variables égales à 1 donnent $T_n = 1$, et $X_1 = -1, X_2 = \dots = X_n = 1$ donnent $T_n = -1$, avec probabilité $(1 - p)p^{n-1} > 0$.

La variable T_n est bornée, donc son espérance existe. L'indépendance mutuelle des X_k permet de factoriser l'espérance de leur produit :

$$E(T_n) = \prod_{k=1}^n E(X_k) = (2p - 1)^n.$$

En utilisant les deux valeurs de T_n , on obtient la relation :

$$P(T_n = 1) - P(T_n = -1) = (2p - 1)^n.$$

2c) Méthode 1 — Résoudre le système issu de l'espérance.

Les événements $\{T_n = 1\}$ et $\{T_n = -1\}$ forment un système complet, donc

$$P(T_n = 1) + P(T_n = -1) = 1.$$

En ajoutant et en soustrayant cette relation à celle de 2b, on résout le système :

$$P(T_n = 1) = \frac{1 + (2p - 1)^n}{2}, \quad P(T_n = -1) = \frac{1 - (2p - 1)^n}{2}.$$

Les expressions sont bien comprises entre 0 et 1 car $|2p - 1| < 1$.

Méthode 2 — Compter les facteurs négatifs.

Pour $k = 1, \dots, n$, posons $B_k = 1$ si $X_k = -1$ et $B_k = 0$ sinon. Les B_k sont indépendantes par le lemme des coalitions et suivent la loi de Bernoulli de paramètre $1 - p$. Leur somme

$$N = \sum_{k=1}^n B_k$$

suit donc la loi binomiale de paramètres n et $1 - p$.

Le produit T_n vaut 1 si N est pair, et -1 si N est impair. Notons A la somme des probabilités binomiales de rang pair et B celle des rangs impairs. Le binôme de Newton donne :

$$A + B = \sum_{j=0}^n \binom{n}{j} (1-p)^j p^{n-j} = 1,$$

$$A - B = \sum_{j=0}^n (-1)^j \binom{n}{j} (1-p)^j p^{n-j} = (p - (1-p))^n = (2p - 1)^n.$$

Ainsi $A = [1 + (2p - 1)^n]/2$ et $B = [1 - (2p - 1)^n]/2$. On retrouve la loi de T_n . La première méthode utilise directement 2b ; la seconde explique le rôle de la parité du nombre de facteurs négatifs.

2d) On utilise les fonctions de répartition, afin de vérifier directement la définition de la convergence en loi. Comme $0 < p < 1$, $|2p - 1| < 1$ et $(2p - 1)^n \rightarrow 0$. Pour tout réel x :

$$F_{T_n}(x) = \begin{cases} 0 & \text{si } x < -1, \\ \frac{1 - (2p - 1)^n}{2} & \text{si } -1 \leq x < 1, \\ 1 & \text{si } x \geq 1. \end{cases}$$

Cette fonction converge vers la fonction de répartition de la loi donnant probabilité $1/2$ à chacune des valeurs -1 et 1 . En particulier, la convergence a lieu en tous ses points de continuité, c'est-à-dire pour $x \notin \{-1, 1\}$.

Ainsi T_n converge en loi vers une variable T de loi de Rademacher de paramètre $1/2$: $P(T = 1) = P(T = -1) = 1/2$.

3a) Soit une issue appartenant aux deux événements à gauche. L'inégalité triangulaire fournit, pour cette issue :

$$|T_{n+1} - T_n| \leq |T_{n+1} - T'| + |T' - T_n| < \frac{1}{2} + \frac{1}{2} = 1.$$

Cette issue appartient donc à l'événement à droite, ce qui prouve l'inclusion.

3b) En prenant les complémentaires dans 3a et en utilisant les lois de Morgan :

$$\{|T_{n+1} - T_n| \geq 1\} \subset \left\{|T_{n+1} - T'| \geq \frac{1}{2}\right\} \cup \left\{|T_n - T'| \geq \frac{1}{2}\right\}.$$

La croissance de la probabilité puis l'inégalité $P(A \cup B) \leq P(A) + P(B)$ donnent

exactement :

$$P(|T_{n+1} - T_n| \geq 1) \leq P\left(|T_{n+1} - T'| \geq \frac{1}{2}\right) + P\left(|T_n - T'| \geq \frac{1}{2}\right).$$

Aucune indépendance avec T' n'est nécessaire.

3c) L'identité $T_{n+1} = T_n X_{n+1}$ donne $T_{n+1} - T_n = T_n(X_{n+1} - 1)$. Comme $|T_n| = 1$, la valeur absolue de cette différence est 0 si $X_{n+1} = 1$, et 2 si $X_{n+1} = -1$.

Par conséquent $\{|T_{n+1} - T_n| \geq 1\} = \{X_{n+1} = -1\}$, et :

$$P(|T_{n+1} - T_n| \geq 1) = 1 - p.$$

3d) Supposons que T_n converge en probabilité vers une variable T' sur le même espace. Alors les deux probabilités à droite dans 3b tendent vers 0.

Pour préciser le lien avec la définition qui peut être formulée avec une inégalité stricte, on utilise $P(|T_n - T'| \geq 1/2) \leq P(|T_n - T'| > 1/4) \rightarrow 0$. La même majoration s'applique à $n + 1$.

Les questions 3b et 3c entraîneraient $1 - p \leq 0$ par passage à la limite, alors que $1 - p > 0$. Cette contradiction prouve que (T_n) ne converge en probabilité vers aucune variable aléatoire. Sa convergence en loi établie en 2d n'impose donc pas une convergence en probabilité.

4a) Ici $p = 1/2$, donc $E(X_k) = 0$ et $V(X_k) = 1$. Par linéarité de l'espérance et indépendance des variables :

$$E(\bar{X}_n) = 0, \quad V(\bar{X}_n) = \frac{1}{n^2} \sum_{k=1}^n V(X_k) = \frac{1}{n}.$$

La variance est strictement positive, donc la variable centrée réduite est définie et vaut :

$$\bar{X}_n^* = \frac{\bar{X}_n - E(\bar{X}_n)}{\sqrt{V(\bar{X}_n)}} = \sqrt{n} \bar{X}_n.$$

4b) Les X_k sont indépendantes, de même loi, et possèdent une espérance finie 0 ainsi qu'une variance finie strictement positive 1. Le théorème limite central s'applique et donne $\sqrt{n} \bar{X}_n \xrightarrow{\mathcal{L}} Z$, où Z suit la loi normale centrée réduite.

Comme $U_n > 0$, les règles de calcul des puissances réelles donnent l'identité exacte :

$$U_n^{1/\sqrt{n}} = \exp\left(\frac{n\bar{X}_n}{\sqrt{n}}\right) = \exp(\sqrt{n} \bar{X}_n).$$

La fonction exponentielle est continue sur $\mathbb{R}$. Le résultat du programme sur la composition d'une convergence en loi par une fonction continue donne donc :

$$U_n^{1/\sqrt{n}} \xrightarrow{\mathcal{L}} e^Z.$$

La variable limite a bien la même loi que Y étudiée en question 1. Il s'agit d'une composition par une fonction fixe et continue.

Exercice 3

1) Pour $x \in E$, l'antisymétrie de f donne $\langle f(x), x \rangle = -\langle x, f(x) \rangle$. Le produit scalaire réel est symétrique, donc $\langle x, f(x) \rangle = \langle f(x), x \rangle$.

Ainsi $2\langle f(x), x \rangle = 0$, d'où $\langle f(x), x \rangle = 0$ pour tout $x \in E$.

2) Méthode 1 — Orthogonalité et dimensions.

Pour $x \in \text{Ker}(f)$ et $y \in \text{Im}(f)$, il existe $z \in E$ tel que $y = f(z)$. Alors :

$$\langle x, y \rangle = \langle x, f(z) \rangle = -\langle f(x), z \rangle = 0.$$

Le noyau et l'image sont donc orthogonaux. Leur intersection est réduite à $\{0_E\}$: un vecteur v de l'intersection serait orthogonal à lui-même, donc vérifierait $\|v\|^2 = 0$ et $v = 0_E$.

Le théorème du rang fournit $\dim \text{Ker}(f) + \dim \text{Im}(f) = \dim E$. La somme directe des deux sous-espaces a donc la dimension de E , ce qui prouve $E = \text{Ker}(f) \oplus \text{Im}(f)$. Cette somme est même orthogonale.

Méthode 2 — Identifier l'orthogonal de l'image.

Pour $x \in E$, on a :

$$x \in \text{Im}(f)^\perp \iff \forall z \in E, \langle x, f(z) \rangle = 0 \iff \forall z \in E, \langle f(x), z \rangle = 0.$$

La dernière condition équivaut à $f(x) = 0_E$: il suffit de prendre $z = f(x)$ pour l'implication non immédiate. Ainsi $\text{Im}(f)^\perp = \text{Ker}(f)$. Le théorème du supplémentaire orthogonal dans un espace euclidien donne alors $E = \text{Im}(f) \oplus \text{Ker}(f)$.

La première méthode met en avant le rang ; la seconde identifie directement un supplémentaire orthogonal, ce qui servira pour construire la base finale.

3) La composée $s = f \circ f$ est un endomorphisme. Pour tous $x, y \in E$, deux applications de l'antisymétrie donnent :

$$\langle s(x), y \rangle = \langle f(f(x)), y \rangle = -\langle f(x), f(y) \rangle = \langle x, f(f(y)) \rangle = \langle x, s(y) \rangle.$$

Donc s est symétrique. De plus, pour tout $x \in E$:

$$\langle s(x), x \rangle = -\|f(x)\|^2 \leq 0.$$

Si $x \neq 0_E$ est un vecteur propre de s associé à μ , alors $\mu\|x\|^2 = -\|f(x)\|^2$. Comme $\|x\|^2 > 0$, $\mu = -\|f(x)\|^2/\|x\|^2 \leq 0$.

Le théorème spectral assure que s , symétrique sur un espace euclidien, possède une base orthonormale de vecteurs propres réels. Toutes ses valeurs propres sont dans $\mathbb{R}_-$.

4a) Posons $F = \text{Im}(f)$, muni du produit scalaire induit par celui de E . Si $x \in F$, alors $f(x) \in \text{Im}(f) = F$: l'application g est donc bien à valeurs dans F .

Comme restriction de l'application linéaire f à un sous-espace stable, g est un

endomorphisme de F . Pour $x, y \in F$, la relation $\langle g(x), y \rangle = -\langle x, g(y) \rangle$ résulte directement de celle satisfaite par f .

Ainsi g est un endomorphisme antisymétrique de F .

4b) Le raisonnement de la question 3 appliqué à g montre que $t = g \circ g$ est symétrique et que $\langle t(x), x \rangle = -\|g(x)\|^2$ pour $x \in F$.

La question 2 donne $F \cap \text{Ker}(f) = \{0_E\}$. Si $x \in F$ et $g(x) = 0_E$, alors $x \in F \cap \text{Ker}(f)$, donc $x = 0_E$. Ainsi g est injectif et $\|g(x)\| > 0$ pour $x \neq 0_E$.

Pour un vecteur propre non nul x de t associé à λ , on en déduit :

$$\lambda\|x\|^2 = -\|g(x)\|^2 < 0.$$

Donc $\lambda < 0$. Le théorème spectral s'applique à t sur F , et toutes ses valeurs propres appartiennent à $\mathbb{R}_-^*$.

5a) Les applications g et $t = g^2$ commutent car $g \circ t = t \circ g = g^3$. Puisque $t(e_1) = \lambda e_1$:

$$t(g(e_1)) = g(t(e_1)) = \lambda g(e_1).$$

Ainsi $g(e_1) \in E_\lambda(t)$. Le vecteur e_1 appartient déjà à cet espace et est non nul. L'injectivité de g montre que $g(e_1)$ est également non nul. Enfin l'antisymétrie donne $\langle e_1, g(e_1) \rangle = 0$ par le calcul de la question 1.

Deux vecteurs non nuls orthogonaux sont libres : dans une relation $\alpha e_1 + \beta g(e_1) = 0_E$, prendre successivement le produit scalaire avec chacun des deux vecteurs impose $\alpha = 0$ et $\beta = 0$.

La famille $(e_1, g(e_1))$ est donc une famille orthogonale et libre de $E_\lambda(t)$.

5b) Notons $W_1 = \text{Vect}(e_1, g(e_1))$ et $F_2 = E_\lambda(t) \cap W_1^\perp$. Puisque $g^2(e_1) = \lambda e_1$, l'espace W_1 est stable par g . On montre que F_2 l'est également, sans supposer de résultat supplémentaire sur les endomorphismes antisymétriques.

Si $x \in F_2$, la commutation utilisée en 5a donne $g(x) \in E_\lambda(t)$. En outre :

$$\langle g(x), e_1 \rangle = -\langle x, g(e_1) \rangle = 0,$$

$$\langle g(x), g(e_1) \rangle = -\langle x, g^2(e_1) \rangle = -\lambda \langle x, e_1 \rangle = 0.$$

Donc $g(x) \in F_2$. Le supplémentaire orthogonal donne $E_\lambda(t) = W_1 \oplus F_2$, avec somme orthogonale et $\dim F_2 = \dim E_\lambda(t) - 2$.

Si $F_2 \neq \{0_E\}$, choisissons $e_2 \in F_2$ non nul. La preuve de 5a montre que $e_2, g(e_2)$ sont non nuls, orthogonaux et libres dans F_2 . Leur plan W_2 est stable, et son orthogonal dans F_2 est stable par la même démonstration.

On poursuit cette construction. À chaque étape, la dimension restante diminue de 2. L'espace restant ne peut jamais avoir dimension 1 : s'il contenait un vecteur non nul e , il contiendrait aussi $g(e)$ et, par 5a, les deux vecteurs y formeraient une famille libre de cardinal 2, ce qui contredirait cette dimension.

La dimension initiale est finie ; le procédé termine donc avec un espace restant nul. Il existe $p \geq 1$ tel que $\dim E_\lambda(t) = 2p$, et les plans successifs sont deux à deux orthogonaux. La famille $(e_1, g(e_1), \dots, e_p, g(e_p))$ est ainsi orthogonale, libre et de cardinal $\dim E_\lambda(t)$: c'est la base orthogonale demandée.

6a) Comme $g^2(e_k) = t(e_k) = \lambda e_k$, l'antisymétrie fournit :

$$\|g(e_k)\|^2 = \langle g(e_k), g(e_k) \rangle = -\langle e_k, g^2(e_k) \rangle = -\lambda \|e_k\|^2.$$

Puisque $\lambda < 0$ et $e_k \neq 0_E$, les deux normes sont strictement positives et

$$\|g(e_k)\| = \sqrt{-\lambda} \|e_k\|.$$

6b) Les divisions définissant e'_k et e''_k sont légitimes d'après 6a. Par linéarité et en utilisant la relation entre les normes :

$$g(e'_k) = \frac{g(e_k)}{\|e_k\|} = \frac{\|g(e_k)\|}{\|e_k\|} e''_k = \sqrt{-\lambda} e''_k.$$

D'autre part :

$$g(e''_k) = \frac{g^2(e_k)}{\|g(e_k)\|} = \frac{\lambda \|e_k\|}{\|g(e_k)\|} e'_k = \frac{\lambda}{\sqrt{-\lambda}} e'_k = -\sqrt{-\lambda} e'_k.$$

La normalisation conserve l'orthogonalité et donne des normes égales à 1. On obtient donc une base orthonormale de $E_\lambda(t)$ dans laquelle chaque paire engendre un plan stable, avec les deux relations annoncées.

7a) L'espace $F = \text{Im}(f)$ est euclidien, et t y est symétrique. Par le théorème spectral, si $\lambda_1, \dots, \lambda_m$ sont ses valeurs propres distinctes, on a la décomposition orthogonale :

$$F = E_{\lambda_1}(t) \oplus \dots \oplus E_{\lambda_m}(t).$$

D'après 5b, chaque espace propre a une dimension paire, notée $2p_j$. Par conséquent :

$$\text{rg}(f) = \dim F = \sum_{j=1}^m 2p_j = 2 \sum_{j=1}^m p_j.$$

Le rang de f est donc pair. Il est non nul puisque $f \neq 0$, donc il est au moins égal à 2.

7b) Dans chaque $E_{\lambda_j}(t)$, la question 6b fournit une base orthonormale formée de paires (e'_k, e''_k) . Pour chaque paire, posons $a = \sqrt{-\lambda_j} > 0$. Les relations $f(e'_k) = ae''_k$ et $f(e''_k) = -ae'_k$ sont valides car $f = g$ sur F .

Les colonnes de la matrice dans cette paire donnent donc le bloc $\begin{pmatrix} 0 & -a \\ a & 0 \end{pmatrix}$. Les espaces propres distincts de t sont orthogonaux, si bien que la réunion de toutes les paires est une base orthonormale de F . Il y a exactement $r = \frac{1}{2} \text{rg}(f)$ paires.

Par la question 2, $\text{Ker}(f) = F^\perp$. Choisissons une base orthonormale du noyau, éventuellement vide si le noyau est nul, et plaçons-la après les paires précédentes. La famille obtenue est une base orthonormale $\mathcal{B}$ de E . Sur les vecteurs du noyau, f vaut 0_E ; sur chaque paire, son action est celle du bloc correspondant.

Ainsi, en notant $d = \dim \text{Ker}(f)$ et $B_j = \begin{pmatrix} 0 & -a_j \\ a_j & 0 \end{pmatrix}$:

$$M = \text{diag}(B_1, \dots, B_r, 0_d) \text{ si } d > 0, \text{ et } M = \text{diag}(B_1, \dots, B_r) \text{ si } d = 0.$$

Ici diag désigne la matrice ayant les blocs indiqués sur sa diagonale et des coefficients nuls ailleurs. Tous les a_j sont strictement positifs. Ils ne sont pas nécessairement distincts, car un même espace propre de t peut fournir plusieurs paires.

Problème

Partie 1 : calcul d'intégrales utiles pour la suite

- 1) Les fonctions intégrées sont polynomiales, donc continues sur le segment $[0, 1]$; les intégrales existent. Pour $p, q \in \mathbb{N}$, leurs primitives donnent :

$$I(p, 0) = \left[\frac{x^{p+1}}{p+1} \right]_0^1 = \frac{1}{p+1},$$

$$I(0, q) = \left[-\frac{(1-x)^{q+1}}{q+1} \right]_0^1 = \frac{1}{q+1}.$$

En particulier, $I(0, 0) = 1$; les dénominateurs sont toujours non nuls.

- 2) Pour $p \in \mathbb{N}$ et $q \in \mathbb{N}^*$, on effectue une intégration par parties sur $[0, 1]$, avec les fonctions de classe C^1 $u(x) = (1-x)^q$ et $v(x) = x^{p+1}/(p+1)$. On a $u'(x) = -q(1-x)^{q-1}$ et $v'(x) = x^p$.

Ainsi :

$$I(p, q) = \left[\frac{x^{p+1}(1-x)^q}{p+1} \right]_0^1 + \frac{q}{p+1} \int_0^1 x^{p+1}(1-x)^{q-1} dx.$$

Le terme de bord vaut 0 : à gauche $x^{p+1} = 0$, et à droite $(1-x)^q = 0$ puisque $q \geq 1$. Par conséquent :

$$I(p, q) = \frac{q}{p+1} I(p+1, q-1).$$

- 3) On raisonne par récurrence sur q , en conservant le quantificateur « pour tout $p \in \mathbb{N}$ » dans l'hypothèse.

Initialisation : pour $q = 0$ et tout $p \in \mathbb{N}$, $\frac{p!0!}{(p+0)!} I(p+0, 0) = I(p, 0)$. Donc H_0 est vraie.

Hérédité : supposons H_q vraie pour un entier $q \geq 0$. Soit $p \in \mathbb{N}$ quelconque. La relation de la question 2, appliquée à $(p, q + 1)$, puis H_q appliquée à l'entier $p + 1$, donnent :

$$I(p, q + 1) = \frac{q + 1}{p + 1} I(p + 1, q) = \frac{q + 1}{p + 1} \frac{(p + 1)! q!}{(p + q + 1)!} I(p + q + 1, 0).$$

Puisque $(p + 1)! = (p + 1)p!$ et $(q + 1)q! = (q + 1)!$, cela devient :

$$I(p, q + 1) = \frac{p!(q + 1)!}{(p + q + 1)!} I(p + q + 1, 0).$$

Cette égalité vaut pour tout p , donc H_{q+1} est vraie. La récurrence établit H_q pour tout $q \in \mathbb{N}$.

4) On combine la question 3 avec $I(p + q, 0) = 1/(p + q + 1)$ obtenu en question 1 :

$$I(p, q) = \frac{p!q!}{(p + q)!} \frac{1}{p + q + 1} = \frac{p!q!}{(p + q + 1)!}.$$

En prenant $p = q = n$, on obtient, pour tout $n \in \mathbb{N}$:

$$I(n, n) = \frac{(n!)^2}{(2n + 1)!}.$$

Ces formules incluent les cas $p = 0$, $q = 0$ et $n = 0$, avec $0! = 1$.

Partie 2 : étude d'une suite de variables aléatoires

5) Pour $n \in \mathbb{N}$, la fonction b_n est positive sur $\mathbb{R}$. Sur $[0, 1]$, elle est polynomiale ; à l'extérieur, elle est nulle. Elle est donc continue sauf éventuellement aux points 0 et 1, ce qui suffit pour une densité au sens du programme. Pour $n = 0$, la convention explicitée dans le sujet donne $b_0(x) = 1$ sur $[0, 1]$.

En posant $c_n = (2n + 1)!/(n!)^2$, la question 4 donne :

$$\int_{-\infty}^{+\infty} b_n(x) dx = c_n \int_0^1 x^n (1 - x)^n dx = c_n I(n, n) = 1.$$

Cette intégrale existe car la fonction est bornée et nulle hors d'un segment. Ainsi b_n est bien une densité de probabilité.

6) Pour $n = 0$, on a $b_0(x) = 1$ si $x \in [0, 1]$, et $b_0(x) = 0$ sinon. C'est une densité de la loi uniforme sur $[0, 1]$. Donc $X_0 \sim \mathcal{U}([0, 1])$.

7a) Méthode 1 — Utiliser les intégrales de la première partie.

La loi de X_n est portée par $[0, 1]$, donc $0 \leq X_n \leq 1$ presque sûrement. Par domination par la constante 1, son espérance existe. Par transfert et la question 4 :

$$E(X_n) = c_n \int_0^1 x^{n+1} (1-x)^n dx = c_n I(n+1, n) = \frac{(2n+1)! (n+1)! n!}{(n!)^2 (2n+2)!}.$$

On simplifie $(n+1)!/n! = n+1$ et $(2n+2)!/(2n+1)! = 2n+2$. Ainsi

$$E(X_n) = (n+1)/(2n+2) = 1/2.$$

Méthode 2 — Exploiter la symétrie de la densité.

L'existence de l'espérance est assurée par la même domination. On a $b_n(1-x) = b_n(x)$ pour $x \in [0, 1]$. Le changement de variable affine $y = 1-x$ donne :

$$E(X_n) = \int_0^1 x b_n(x) dx = \int_0^1 (1-y) b_n(1-y) dy = \int_0^1 b_n(y) dy - \int_0^1 y b_n(y) dy = 1 - E(X_n).$$

Donc $E(X_n) = 1/2$. La première méthode est imposée par la question ; la seconde permet d'identifier la moyenne sans calcul de factorielles, grâce à la symétrie autour de $1/2$.

7b) La domination $0 \leq X_n^2 \leq 1$ assure l'existence du moment d'ordre 2, puis de la variance. Par transfert et la première partie :

$$E(X_n^2) = c_n I(n+2, n) = \frac{(2n+1)! (n+2)! n!}{(n!)^2 (2n+3)!} = \frac{(n+2)(n+1)}{(2n+3)(2n+2)} = \frac{n+2}{2(2n+3)}.$$

En appliquant Koenig-Huygens et $E(X_n) = 1/2$:

$$V(X_n) = \frac{n+2}{2(2n+3)} - \frac{1}{4} = \frac{2(n+2) - (2n+3)}{4(2n+3)} = \frac{1}{4(2n+3)}.$$

Cette expression est strictement positive pour tout $n \geq 0$. Au rang 0, elle donne $1/12$, ce qui concorde avec la variance d'une uniforme sur $[0, 1]$.

7c) Soit $\varepsilon > 0$. L'inégalité de Bienaymé-Tchebychev s'applique puisque la variance de X_n existe :

$$P\left(\left|X_n - \frac{1}{2}\right| \geq \varepsilon\right) \leq \frac{V(X_n)}{\varepsilon^2} = \frac{1}{4(2n+3)\varepsilon^2} \rightarrow 0.$$

Ainsi (X_n) converge en probabilité vers la variable certaine égale à $1/2$. Aucune hypothèse d'indépendance entre les X_n n'est requise pour cette conclusion.

Partie 3 : simulation informatique de X_n

8) La fonction de répartition de la loi uniforme sur $[0, 1]$ est :

$$F_U(x) = \begin{cases} 0 & \text{si } x < 0, \\ x & \text{si } 0 \leq x \leq 1, \\ 1 & \text{si } x > 1. \end{cases}$$

En particulier, $F_U(0) = 0$ et $F_U(1) = 1$: une variable à densité n'a pas de masse ponctuelle aux extrémités.

9a) L'instant d'arrivée du dernier des $2n + 1$ individus est le maximum de leurs instants d'arrivée :

$$V_{2n+1} = \max(U_1, \dots, U_{2n+1}).$$

Cette identité reste valable même si plusieurs instants coïncident.

9b) Pour tout $x \in \mathbb{R}$, on a $\{V_{2n+1} \leq x\} = \bigcap_{j=1}^{2n+1} \{U_j \leq x\}$. L'indépendance mutuelle des U_j entraîne celle de ces événements, donc :

$$G_{2n+1}(x) = \prod_{j=1}^{2n+1} P(U_j \leq x) = F_U(x)^{2n+1}.$$

En utilisant la question 8 :

$$G_{2n+1}(x) = \begin{cases} 0 & \text{si } x < 0, \\ x^{2n+1} & \text{si } 0 \leq x \leq 1, \\ 1 & \text{si } x > 1. \end{cases}$$

10a) L'instant d'arrivée du premier individu est le minimum de tous les instants d'arrivée :

$$V_1 = \min(U_1, \dots, U_{2n+1}).$$

10b) On passe par la probabilité de l'événement complémentaire, car l'événement portant sur le minimum s'écrit alors comme une intersection :

$$\{V_1 > x\} = \bigcap_{j=1}^{2n+1} \{U_j > x\}.$$

Par indépendance mutuelle :

$$P(V_1 > x) = (1 - F_U(x))^{2n+1} = \begin{cases} 1 & \text{si } x < 0, \\ (1 - x)^{2n+1} & \text{si } 0 \leq x \leq 1, \\ 0 & \text{si } x > 1. \end{cases}$$

Comme $G_1(x) = 1 - P(V_1 > x)$, il vient :

$$G_1(x) = \begin{cases} 0 & \text{si } x < 0, \\ 1 - (1 - x)^{2n+1} & \text{si } 0 \leq x \leq 1, \\ 1 & \text{si } x > 1. \end{cases}$$

Pour $n = 0$, il n'y a qu'un individu, et les formules de 9b et 10b redonnent toutes deux F_U .

11) Question adaptée au programme actuel. Code adapté en Python. On génère un tableau de $2n + 1$ réalisations indépendantes de la loi uniforme sur $[0, 1]$, puis on extrait son minimum et son maximum. Ils proviennent du même tableau afin de simuler les deux instants d'une même expérience.

Python

```
import numpy as np

n = int(input("Entrer un entier naturel n : "))
U = np.random.random(2*n + 1)
V1 = np.min(U)
Vdernier = np.max(U)
print("V_1 =", V1)
print("V_(2n+1) =", Vdernier)
```

La fonction `np.random.random(2*n + 1)` renvoie un tableau de longueur $2n + 1$ dont les éléments simulent des uniformes indépendantes sur $[0, 1]$. La borne supérieure n'est pas produite par le générateur numérique, ce qui ne change pas le modèle continu, car la

probabilité d'obtenir exactement 1 est nulle. Les commandes de saisie ont été précisées dans le sujet.

Le script a été exécuté pour $n = 0$, $n = 3$ et $n = 20$: les sorties vérifient $0 \leq V_1 \leq V_{2n+1} \leq 1$, et pour $n = 0$ les deux sorties sont égales. Ce contrôle d'exécution complète la justification du modèle ; il ne constitue pas une preuve de loi par simulation.

12a) Fixons d'abord $x \in]0, 1[$. Pour chaque $j \in \{1, \dots, 2n + 1\}$, définissons B_j comme la variable valant 1 si $U_j \leq x$ et 0 sinon. Alors B_j suit la loi de Bernoulli de paramètre x , puisque $P(U_j \leq x) = x$.

Chaque B_j étant une fonction de U_j , le lemme des coalitions assure l'indépendance mutuelle de ces variables. Leur somme

$$N_x = \sum_{j=1}^{2n+1} B_j$$

compte les individus arrivés au plus tard à l'instant x ; elle suit la loi binomiale de paramètres $2n + 1$ et x .

Le $(n + 1)$ -ième instant d'arrivée est au plus x exactement lorsque au moins $n + 1$ individus sont arrivés au plus tard à cet instant. Ainsi $\{V_{n+1} \leq x\} = \{N_x \geq n + 1\}$. On en déduit :

$$G_{n+1}(x) = \sum_{k=n+1}^{2n+1} P(N_x = k) = \sum_{k=n+1}^{2n+1} \binom{2n+1}{k} x^k (1-x)^{2n+1-k}.$$

Pour $x = 0$, aucun individu n'arrive à un instant négatif ou nul, presque sûrement, donc $G_{n+1}(0) = 0$. Pour $x = 1$, tous sont arrivés, presque sûrement, donc $G_{n+1}(1) = 1$. Le polynôme du membre de droite prend ces mêmes valeurs, les puissances d'exposant nul étant les fonctions constantes égales à 1. La formule est donc valable sur tout $[0, 1]$.

12b) On dérive la somme finie de 12a et on utilise les identités binomiales pour faire

apparaître une annulation télescopique. Posons $m = 2n + 1$. Pour $x \in]0, 1[$:

$$G'_{n+1}(x) = \sum_{k=n+1}^m k \binom{m}{k} x^{k-1} (1-x)^{m-k} - \sum_{k=n+1}^{m-1} (m-k) \binom{m}{k} x^k (1-x)^{m-k-1}.$$

Le second terme de la dérivée est nul pour $k = m$, d'où sa borne supérieure $m - 1$. Une somme dont la borne inférieure dépasse la borne supérieure est ici égale à 0.

Les formules des coefficients binomiaux donnent $k \binom{m}{k} = m \binom{m-1}{k-1}$ pour $1 \leq k \leq m$, et $(m-k) \binom{m}{k} = m \binom{m-1}{k}$ pour $0 \leq k \leq m-1$. Dans la première somme, on pose $j = k - 1$. On obtient :

$$G'_{n+1}(x) = m \sum_{j=n}^{m-1} \binom{m-1}{j} x^j (1-x)^{m-1-j} - m \sum_{j=n+1}^{m-1} \binom{m-1}{j} x^j (1-x)^{m-1-j}.$$

Tous les termes de rang $j \geq n + 1$ se compensent, et seul le rang $j = n$ demeure. Comme $m - 1 = 2n$:

$$G'_{n+1}(x) = (2n + 1) \binom{2n}{n} x^n (1-x)^n = \frac{(2n + 1)!}{(n!)^2} x^n (1-x)^n.$$

En dehors de $[0, 1]$, G_{n+1} est constante, égale à 0 à gauche et à 1 à droite. Sur $[0, 1]$, elle est polynomiale et ses valeurs aux extrémités concordent avec ces constantes : elle est donc continue sur $\mathbb{R}$ et dérivable sauf éventuellement en 0 et 1. Une densité de V_{n+1} est donc :

$$g_{n+1}(x) = \begin{cases} \frac{(2n + 1)!}{(n!)^2} x^n (1-x)^n & \text{si } 0 \leq x \leq 1, \\ 0 & \text{sinon.} \end{cases}$$

Aux extrémités, on peut choisir les valeurs ci-dessus, même si la dérivée de la fonction de répartition n'y existe pas. Pour $n = 0$, la formule donne la densité uniforme ; dans le calcul précédent, la seconde somme est vide et la première se réduit au rang 0.

Ainsi $g_{n+1} = b_n$. Deux variables admettant la même densité ont la même loi, donc V_{n+1} et X_n sont de même loi. Il s'agit d'une égalité de lois, qui n'exige pas que ces variables soient égales sur l'espace probabilisé.

12c) Question adaptée au programme actuel. Code adapté en Python. La commande `np.median` calcule la médiane du tableau. En ordonnant les sept valeurs de U , on obtient 1, 2, 5, 8, 9, 13, 23. Avec sept éléments, la médiane est le quatrième élément, soit 8.

Python

```
import numpy as np

U = np.array([8, 2, 9, 13, 23, 1, 5])
V = np.median(U)
print("V =", V)
```

Ce code a été exécuté : il affiche $V = 8.0$. La valeur numérique de la médiane est donc 8 ; son affichage avec $.0$ vient du type numérique renvoyé par NumPy.

12d) Question adaptée au programme actuel. Code adapté en Python. Pour un nombre impair $2n + 1$ de valeurs, la médiane est la $(n + 1)$ -ième valeur dans l'ordre croissant. D'après 12b, la médiane de $2n + 1$ uniformes indépendantes a la loi de X_n .

Python

```
import numpy as np

n = int(input("Entrer un entier naturel n : "))
U = np.random.random(2*n + 1)
X = np.median(U)
print("X_n =", X)
```

Le tableau est non vide pour tout $n \geq 0$, et sa taille est toujours impaire. Au rang 0, il ne contient qu'une uniforme : la médiane est cette valeur elle-même, conformément à la loi de X_0 .

Le script a été exécuté pour $n = 0$, $n = 3$ et $n = 20$, et ses sorties sont dans $[0, 1]$. Un contrôle numérique complémentaire sur de nombreux tirages retrouve des moyennes proches de $1/2$ et des variances proches de $1/[4(2n + 3)]$. La preuve de la loi simulée reste celle des questions 12a et 12b.

Relevé des adaptations

- Exercice 1, 1c : remplacement du mot «égalité» par «inégalité», afin de mettre la consigne en accord avec la formule imprimée et avec la déduction demandée en 1d.
- Exercice 1, 2c et 4b : code adapté en Python, avec conservation des blancs, des initialisations, du franchissement strict du seuil et de la valeur de sortie.
- Problème, 5 et 6 : explicitation de la convention polynomiale $x^0(1 - x)^0 = 1$ sur tout $[0, 1]$, y compris aux extrémités, afin de définir sans ambiguïté b_0 ; les lois et les calculs restent identiques.
- Problème, 11 : demande de script adaptée en Python et ajout des indications sur `input` et `int`, nécessaires pour la saisie utilisateur.
- Problème, 12c et 12d : code et demandes de simulation adaptés en Python; la commande de médiane et les tailles impaires des échantillons sont conservées.