

MP

Mathématiques · Informatique

2018

Sous la coordination de

William AUFORT

ENS Lyon

ancien élève de l'École Normale Supérieure (Lyon)

Benjamin MONMEGE

enseignant-chercheur à l'université

ancien élève de l'École Normale Supérieure (Paris-Saclay)

Vincent PUYHAUBERT

professeur en CPGE

ancien élève de l'École Normale Supérieure (Paris-Saclay)

Par

Céline CHEVALIER

enseignant-chercheur à l'université

Jean-Julien FLECK

professeur en CPGE

Martin GUY

ENS Lyon

Emma KERINEC

ENS Lyon

Juliette BRUN-LELOUP

professeur en CPGE

Théo LENOIR

ENS Ulm

Hugo MENET

ENS Lyon

Florian METZGER

docteur en mathématiques

Clément MIFSUD

ENS Paris-Saclay

Angèle NICLAS

ENS Lyon

Sophie RAINERO

professeur en CPGE

Cyril RAVAT

professeur en CPGE

Sommaire

		Énoncé	Corrigé
CONCOURS COMMUNS POLYTECHNIQUES			
Mathématiques 1	Estimations numériques d'intégrales. <i>polynômes, intégrales de fonctions continues sur un segment, produits scalaires</i>	17	24
Mathématiques 2	Réduction de produits tensoriels de matrices. <i>espaces euclidiens, probabilités, réduction</i>	38	44

CENTRALE-SUPÉLEC

Mathématiques 1	Aplatissement aléatoire d'un ensemble de points en grande dimension. <i>probabilités, espérance conditionnelle, produits scalaires, distances</i>	61	67
Mathématiques 2	Fonctions harmoniques et problème de Dirichlet. <i>calcul différentiel, fonctions définies par une intégrale, équations différentielles du second ordre</i>	92	96
Informatique commune	Simulation de la cinétique d'un gaz parfait. <i>algorithmique, programmation, complexité, représentation des nombres, bases de données</i>	119	127
Informatique optionnelle	Étude du jeu Ricochet Robots. <i>recherche dichotomique, tri par insertion, graphes, calcul de complexité en moyenne, programmation</i>	144	150

MINES-PONTS

Mathématiques 1	Lemme de Fekete et théorème d'Erdős-Szekeres. <i>variables aléatoires, suites numériques, dénombrement, combinatoire, développements limités</i>	165	171
Mathématiques 2	Racine carrée d'une matrice. <i>réduction, récurrence, algèbre linéaire, calcul différentiel, suites</i>	192	198
Informatique commune	Mesures de houle. <i>taille et lecture de fichiers, calculs de moyenne et d'intégrale, recherche dans une liste, tri, SQL, récursivité</i>	218	227
Informatique optionnelle	Recherche de motifs. <i>algorithmes de textes, automates</i>	239	248

POLYTECHNIQUE-ENS

Mathématiques A	Décomposition en valeurs singulières. <i>algèbre linéaire, réduction des endomorphismes, produit scalaire, topologie</i>	261	267
Mathématiques B	Étude de la borne inférieure d'une forme linéaire sur un sous-ensemble convexe de polynômes. <i>topologie, polynômes orthogonaux, résolution d'équations dans les complexes, étude de polynômes, intégration, fonctions de plusieurs variables</i>	294	298
Informatique commune	Implémentation de requêtes SQL en Python. <i>listes et dictionnaires Python, requêtes basiques en SQL</i>	318	331

FORMULAIRES

Développements limités usuels en 0	342
Développements en série entière usuels	343
Dérivées usuelles	344
Primitives usuelles	345
Trigonométrie	348

Sommaire thématique de mathématiques

2015 – 2018

CCP MP Maths 1		•	•		•		•	••	•	•	••		•	••
CCP MP Maths 2	•	•	•	••	••						•	•		•
CCP PC Maths		••	•	•	•	•		•	•		••			••
CCP PSI Maths			•	••	•	•	•		•	•	•	••		•
Centrale MP Maths 1			•	•		•					•	•		•
Centrale MP Maths 2							•	•	•		••	•	•	••
Centrale PC Maths 1		•	•	•					••		•	•	•	••
Centrale PC Maths 2	•	•	•				•	•		•	•			••
Centrale PSI Maths 1		•	•	•		•	•			•				••
Centrale PSI Maths 2		•	•				•		•	•	••	•	•	••
Mines MP Maths 1			•	•	•		•	•				•		••
Mines MP Maths 2		•	•	•	•	•	•	•			•		•	•
Mines PC Maths 1			•	•			•		•	•				••
Mines PC Maths 2		•	•				•		•	•	•	•	•	•
Mines PSI Maths 1			•	•			•		•	•				••
Mines PSI Maths 2			••	•						•	•			•
X/ENS MP Maths A		••	••	••	•	•								
X/ENS MP Maths B		•			•	•		•		••	••		•	•
X/ENS PC Maths			•	•		•				•	•		•	•
X/ENS PSI Maths		•	••		•	•				•		•	•	
	Structures algébriques et arithmétique	Polynômes	Algèbre linéaire générale	Réduction des endomorphismes	Produit scalaire et espaces euclidiens	Topologie des espaces vectoriels normés	Suites et séries numériques	Suites et séries de fonctions	Séries entières	Analyse réelle	Intégration	Équations différentielles	Fonctions de plusieurs variables	Dénombrement et probabilités

SESSION 2018

MPMA102

**CONCOURS COMMUNS
POLYTECHNIQUES****ÉPREUVE SPÉCIFIQUE - FILIÈRE MP****MATHÉMATIQUES 1****Lundi 30 avril : 14 h - 18 h**

N.B. : le candidat attachera la plus grande importance à la clarté, à la précision et à la concision de la rédaction. Si un candidat est amené à repérer ce qui peut lui sembler être une erreur d'énoncé, il le signalera sur sa copie et devra poursuivre sa composition en expliquant les raisons des initiatives qu'il a été amené à prendre.

Les calculatrices sont interdites
--

Le sujet est composé d'un problème avec quatre parties.

ESTIMATIONS NUMÉRIQUES D'INTÉGRALES

Objectifs

Le fil conducteur de ce sujet est le calcul approché d'intégrales.

La partie I est indépendante des autres parties. À travers l'exemple de l'intégrale de Gauss, on utilise des suites de fonctions et on « permute limite et intégrale ».

Les parties II et III peuvent être traitées de manière indépendante. La partie IV utilise des résultats des parties II et III.

Les parties II, III et IV traitent de l'utilisation des polynômes interpolateurs pour le calcul approché d'intégrales : on présente le principe des méthodes de quadrature, dites de Newton-Cotes, ainsi qu'un raffinement avec la méthode de quadrature de Gauss.

Le sujet comporte aussi quelques questions notées *Informatique* portant sur le programme « informatique pour tous ». Les algorithmes demandés doivent être écrits en langage Python.

Notations

- Si f est une fonction réelle bornée sur $[a, b]$ avec $a < b$, on pose :

$$\|f\|_{\infty} = \sup_{x \in [a, b]} |f(x)|.$$

- On note $\mathbb{R}_n[X]$ l'ensemble des polynômes à coefficients réels de degré inférieur ou égal à n .
On pourra confondre les expressions « polynômes » et « fonctions polynomiales ».

Partie I - « Permutation limite-intégrale » et intégrale de Gauss

On considère l'intégrale de Gauss :

$$I = \int_0^1 e^{-x^2} dx.$$

I.1 - Utilisation d'une série entière

Q1. Démontrer à l'aide d'une série entière que :

$$I = \sum_{n=0}^{+\infty} \frac{(-1)^n}{(2n+1)n!}.$$

On pose pour $n \in \mathbb{N}$:

$$s_n = \sum_{k=0}^n \frac{(-1)^k}{(2k+1)k!}.$$

Q2. Justifier que pour tout $n \in \mathbb{N}$, on a :

$$|I - s_n| \leq \frac{1}{(2n+3)(n+1)!}.$$

Q3. *Informatique* : écrire une fonction récursive **factorielle** qui prend en argument un entier naturel n et renvoie l'entier $n!$.

Q4. *Informatique* : en déduire un script, qui détermine un entier N , tel que $|I - s_N| \leq 10^{-6}$.

I.2 - Utilisation d'une autre suite de fonctions

Pour tout $n \in \mathbb{N}^*$, on définit sur $[0, +\infty[$ la fonction f_n par :

$$f_n(x) = \left(1 - \frac{x^2}{n}\right)^n.$$

Q5. Déterminer, en détaillant, la limite simple de la suite de fonctions $(f_n)_{n \in \mathbb{N}^*}$.

Q6. Soit $n \in \mathbb{N}^*$. Démontrer que $\forall x \in [0, 1]$, $|f_n(x)| \leq e^{-x^2}$.
En déduire que :

$$I = \lim_{n \rightarrow +\infty} \sum_{k=0}^n \binom{n}{k} \frac{(-1)^k}{n^k(2k+1)}.$$

Partie II - Notion de polynôme interpolateur

Soit $f : [a, b] \rightarrow \mathbb{R}$ une fonction continue. On se donne $n+1$ points $x_0, x_1, \dots, x_n$ dans $[a, b]$, deux à deux distincts.

On appelle polynôme interpolateur de f aux points x_i , un polynôme $P \in \mathbb{R}_n[X]$ qui coïncide avec f aux points x_i , c'est-à-dire tel que pour tout $i \in \llbracket 0, n \rrbracket$, $P(x_i) = f(x_i)$.

II.1 - Existence du polynôme interpolateur

Pour tout entier i de $\llbracket 0, n \rrbracket$, on définit le polynôme l_i de $\mathbb{R}_n[X]$ par :

$$l_i(X) = \prod_{\substack{k=0 \\ k \neq i}}^n \frac{X - x_k}{x_i - x_k}.$$

On pose :

$$L_n(f) = \sum_{i=0}^n f(x_i) l_i(X).$$

Q7. Démontrer que $L_n(f)$ est un polynôme interpolateur de f aux points x_i , puis démontrer l'unicité d'un tel polynôme.

Un tel polynôme est appelé polynôme interpolateur de Lagrange.

CCP Maths 1 MP 2018 — Corrigé

Ce corrigé est proposé par Clément Mifsud (professeur en CPGE) ; il a été relu par Loïc Devilliers (ENS Cachan) et Sophie Rainero (professeur en CPGE).

Cette épreuve est consacrée au calcul approché d'intégrales de fonctions continues sur un segment.

- La première partie a pour objet d'obtenir deux expressions de l'intégrale

$$I = \int_0^1 e^{-x^2} dx$$

sous forme d'une somme d'une série convergente. Cette partie comporte deux questions portant sur le programme « informatique pour tous » ; l'une calcule la factorielle, l'autre utilise la série étudiée pour calculer une approximation de I .

- La deuxième partie porte sur l'interpolation d'une fonction grâce aux polynômes interpolateurs de Lagrange et sur une majoration de l'erreur d'interpolation. Par ailleurs, deux questions portant sur le programme « informatique pour tous » s'intéressent au calcul du polynôme interpolateur de Lagrange et à la complexité de ce calcul.
- La troisième partie, assez classique, introduit un produit scalaire entre polynômes, ce qui permet de construire une famille de polynômes, dits de Legendre, qui sont orthogonaux pour ce produit scalaire. Ils sont utilisés pour développer dans la dernière partie une méthode de quadrature, c'est-à-dire de calcul approché de l'intégrale d'une fonction f , à l'aide d'une combinaison linéaire de valeurs prises par f en des points d'interpolation particuliers.
- La quatrième partie se consacre à l'analyse de cette méthode de quadrature dont les points d'interpolation associés sont les racines d'un polynôme de Legendre. L'intérêt de ce choix est d'obtenir une méthode de quadrature exacte pour les polynômes de degré $2n + 1$ avec seulement $n + 1$ points (où $n \in \mathbb{N}$).

Ce sujet est issu des mathématiques appliquées : comment fait-on calculer une intégrale à un ordinateur, en pratique ? En dehors de la première partie (indépendante du reste du sujet) et de la question 16, il n'utilise que le cours de première année, qu'il permet de bien réviser.

INDICATIONS

Partie I

- 2 Utiliser la majoration du reste d'une série alternée.
 5 Pour prouver l'inégalité demandée, commencer par démontrer que

$$\forall u > -1 \quad \ln(1+u) \leq u$$

Pour la seconde partie de la question, penser au théorème de convergence dominée.

Partie II

- 7 Montrer l'unicité du polynôme interpolateur en considérant un second polynôme interpolateur et en étudiant les racines de la différence de ces deux polynômes.
 10 Démontrer le résultat par récurrence sur $p \in \mathbb{N}^*$ et penser au théorème de Rolle.
 11 Si $x \in \sigma$, que vaut $\pi_\sigma(x)$?
 13 Utiliser les questions 10 et 11.
 14 Par hypothèse, f est de classe $\mathcal{C}^{n+1}$ sur $[a; b]$. Utiliser la question 13.
 15 On dispose d'une suite $(x_k)_{k \in \mathbb{N}}$ d'éléments deux à deux distincts de $[0; 2\pi]$ et pour chaque $n \in \mathbb{N}$, on définit le polynôme $L_n(f)$ interpolateur de f aux points $(x_k)_{k \in [0; n]}$.
 16 Lorsque $k \in \mathbb{N}$, que vaut $f^{(2k)}(0)$?

Partie III

- 18 Ne pas oublier que

$$\forall n \in \mathbb{N}^* \quad \mathbb{R}_{n-1}[X] = \text{Vect}(P_0, P_1, \dots, P_{n-1})$$

- 19 Que dire du signe de P_n sur $[-1; 1]$ si ce polynôme n'admet pas de racine dans l'intervalle $[-1; 1]$?

Partie IV

- 21 Utiliser un changement de variable.
 22 Démontrer tout d'abord le résultat pour $P = \ell_i$ avec $i \in [0; n]$.
 24 Se servir des questions 18 et 22 et de la linéarité de J .
 25 Pour la première partie de la question, poser, pour $i \in [0; n]$,

$$S_i = \prod_{\substack{k=0 \\ k \neq i}}^n (X - t_k)^2$$

Pour la seconde partie, considérer le polynôme constant égal à 1.

I. « PERMUTATION LIMITE-INTÉGRALE » ET INTÉGRALE DE GAUSS

1 D'après le développement en série entière de la fonction exponentielle, on a

$$\forall t \in \mathbb{R} \quad e^t = \sum_{n=0}^{+\infty} \frac{t^n}{n!}$$

En utilisant cette égalité en $t = -x^2$ pour $x \in \mathbb{R}$, on obtient que

$$\forall x \in \mathbb{R} \quad e^{-x^2} = \sum_{n=0}^{+\infty} (-1)^n \frac{x^{2n}}{n!}$$

On en déduit que la fonction $x \mapsto e^{-x^2}$ est développable en série entière sur $\mathbb{R}$. Par intégration de la série entière précédente sur $[0; 1]$, segment inclus dans l'intervalle ouvert de convergence, on en déduit que

$$I = \int_0^1 e^{-x^2} dx = \int_0^1 \sum_{n=0}^{+\infty} (-1)^n \frac{x^{2n}}{n!} dx = \sum_{n=0}^{+\infty} \int_0^1 (-1)^n \frac{x^{2n}}{n!} dx = \sum_{n=0}^{+\infty} \frac{(-1)^n}{n!} \int_0^1 x^{2n} dx$$

Or, pour tout entier naturel n , on a

$$\int_0^1 x^{2n} dx = \frac{1}{2n+1}$$

Finalement, on obtient

$$I = \sum_{n=0}^{+\infty} \frac{(-1)^n}{(2n+1)n!}$$

2 Définissons pour tout entier $n \in \mathbb{N}$, $a_n = 1/((2n+1)n!)$. La suite $(a_n)_{n \in \mathbb{N}}$ est décroissante de limite nulle. D'après le critère spécial des séries alternées, on sait que la série $\sum (-1)^n a_n$ converge et de plus,

$$\forall n \in \mathbb{N} \quad \left| \sum_{k=n+1}^{+\infty} (-1)^k a_k \right| \leq a_{n+1}$$

Ainsi, par définition des suites $(a_n)_{n \in \mathbb{N}}$ et $(s_n)_{n \in \mathbb{N}}$ et d'après la question 1, il vient

$$|I - s_n| = \left| \sum_{k=n+1}^{+\infty} \frac{(-1)^k}{(2k+1)k!} \right| \leq \frac{1}{(2n+3)(n+1)!}$$

3 La fonction récursive suivante, prenant en argument un entier naturel n , renvoie sa factorielle.

```
def factorielle(n):
    if n==0:
        return 1
    else:
        return n*factorielle(n-1)
```

4 D'après la question 2, il suffit de trouver un entier naturel N tel que

$$\frac{1}{(2N+3)(N+1)!} \leq 10^{-6}$$

pour obtenir que $|I - s_N| \leq 10^{-6}$. On en déduit le script suivant, grâce à la fonction définie à la question 3.

```

N = 0
while 1/((2*N+3)*factorielle(N+1))>10**(-6):
    N = N+1
print(N)

```

5 Soit $x \in \mathbb{R}_+$. Soit $n \in \mathbb{N}$ tel que $n > x^2$, alors $1 - x^2/n > 0$ et ainsi

$$f_n(x) = \left(1 - \frac{x^2}{n}\right)^n = e^{n \ln\left(1 - \frac{x^2}{n}\right)}$$

Par ailleurs,

$$\ln\left(1 - \frac{x^2}{n}\right) = -\frac{x^2}{n} + o_{n \rightarrow +\infty}\left(\frac{1}{n}\right)$$

Ainsi,

$$n \ln\left(1 - \frac{x^2}{n}\right) = -x^2 + o_{n \rightarrow +\infty}(1)$$

d'où, par continuité de la fonction exponentielle, $f_n(x) \xrightarrow{n \rightarrow +\infty} e^{-x^2}$. En conclusion,

La suite de fonctions $(f_n)_{n \in \mathbb{N}^*}$ converge simplement sur $\mathbb{R}_+$ vers la fonction $x \mapsto e^{-x^2}$.

6 Avant de démontrer l'inégalité demandée, commençons par démontrer que

$$\forall u > -1 \quad \ln(1+u) \leq u$$

La fonction $f: u \mapsto u - \ln(1+u)$ est définie et dérivable sur $] -1; +\infty[$ et, pour tout $u \in] -1; +\infty[$, sa dérivée vaut

$$f'(u) = 1 - \frac{1}{1+u} = \frac{u}{1+u}$$

Ainsi, f' est à valeurs positives sur $[0; +\infty[$ et à valeurs négatives sur $] -1; 0[$. Par conséquent, la fonction f est décroissante sur $] -1; 0[$ et croissante sur $[0; +\infty[$. Par étude des variations de f , on en déduit que pour tout $u > -1$, $f(u) \geq f(0) = 0$, c'est-à-dire,

$$\forall u > -1 \quad \ln(1+u) \leq u$$

Pour démontrer ce résultat, on peut aussi utiliser la concavité de la fonction

$$u \in] -1; +\infty[\mapsto \ln(1+u)$$

Démontrons maintenant l'inégalité de l'énoncé. Soient $x \in [0; 1]$ et $n \in \mathbb{N}^*$. Observons tout d'abord que le nombre x^2/n appartient à $[0; 1]$ et par conséquent, $f_n(x) \geq 0$. Puis distinguons deux cas.

- Si $x^2/n = 1$, alors $f_n(x) = 0$ et ainsi l'inégalité $|f_n(x)| \leq e^{-x^2}$ est évidente.
- Si $x^2/n \in [0; 1[$, alors $-x^2/n > -1$ et d'après l'inégalité précédente, on a

$$\ln\left(1 - \frac{x^2}{n}\right) \leq -\frac{x^2}{n}$$

$$\text{Puisque } n \geq 0, \quad n \ln\left(1 - \frac{x^2}{n}\right) \leq -n \times \frac{x^2}{n} = -x^2$$

Par croissance de la fonction exponentielle, on en déduit que

$$f_n(x) = e^{n \ln(1 - x^2/n)} \leq e^{-x^2}$$

CCP Maths 2 MP 2018 — Corrigé

Ce corrigé est proposé par Théo Lenoir (ENS Ulm) ; il a été relu par Céline Chevalier (enseignant-chercheur à l'université) et Benjamin Monmege (enseignant-chercheur à l'université).

L'épreuve se compose de deux exercices et d'un problème, tous indépendants.

- Le premier exercice demande de mener des calculs : dans l'espace des fonctions continues sur $[-1; 1]$ muni de son produit scalaire usuel, on cherche le projeté orthogonal de l'exponentielle sur le sous-espace des fonctions polynomiales de degré 1.
- Le second exercice porte sur les probabilités. Il consiste à vérifier que l'on peut approximer des variables binomiales de paramètres n et p par des variables de Poisson de paramètre np sous certaines hypothèses.
- Le problème a pour thème la réduction. Il est assez calculatoire et traite des produits tensoriels de matrices (ce qui revient à multiplier chaque coefficient d'une matrice par une même matrice).

Tout d'abord, on montre l'équivalence entre la diagonalisabilité sur $\mathbb{R}$ d'une matrice et le fait qu'elle annule un polynôme scindé à racines simples.

On étudie ensuite la diagonalisabilité en fonction de la matrice A de la matrice définie par blocs

$$\begin{pmatrix} 4A & 2A \\ -3A & A \end{pmatrix}$$

puis la diagonalisabilité et la trigonalisabilité en fonction de la matrice A de

$$\begin{pmatrix} 3A & -2A \\ 2A & -A \end{pmatrix}$$

Enfin, on réutilise les techniques précédentes pour trouver des plans stables par des endomorphismes, et même calculer une exponentielle de matrice.

Le sujet est abordable et contient quelques questions proches du cours. C'est un très bon entraînement technique, car il comporte de nombreux calculs. Le problème constitue une bonne révision du cours sur la réduction et une introduction aux produits tensoriels de matrices. Il est néanmoins assez répétitif.

INDICATIONS

Exercice I

- 2 On pourra remarquer que u et v sont orthogonaux.
- 3 Utiliser la formule du projeté orthogonal sur un espace de dimension finie et exploiter la base orthonormale construite précédemment.

Exercice II

- 4 Calculer la limite lorsque n tend vers $+\infty$ de

$$\frac{n}{n} \frac{n-1}{n} \dots \frac{n-k+1}{n}$$

Donner une expression de ce terme avec des factorielles. Utiliser l'expression explicite du coefficient binomial pour ensuite calculer $\lim_{n \rightarrow +\infty} P(X_n = k)$.

- 5 Décomposer X_n comme une somme de variables aléatoires.

Problème

- 7 Écrire la matrice de u dans une base adaptée.
- 8 Considérer une base adaptée à la décomposition.
- 10 Vérifier que $Q^{-1} = \begin{pmatrix} -I_n & -I_n \\ 3I_n & 2I_n \end{pmatrix}$ et calculer $Q B Q^{-1}$.
- 11 Penser au résultat de la question précédente.
- 12 Utiliser les questions 10 puis 11.
- 14 Suivre la même méthode qu'à la question 10.
- 15 On vérifiera que pour tout polynôme $P \in \mathbb{R}[X]$

$$P(F) = \begin{pmatrix} P(A) & -2 A P'(A) \\ 0 & P(A) \end{pmatrix}$$

On pourra le montrer dans un premier temps pour $P = X^k$ pour tout entier naturel k , en procédant par récurrence.

- 16 Poser T le polynôme minimal de A . Exploiter la question précédente afin d'obtenir des divisibilités.
- 17 Utiliser la question précédente.
- 18 Penser au fait qu'une matrice est trigonalisable sur $\mathbb{R}$ si et seulement si son polynôme caractéristique est scindé sur $\mathbb{R}$.
- 19 Exploiter les questions 14 et 18 pour trouver une condition nécessaire et suffisante sur A pour que ce soit le cas, et trouver A la vérifiant.
- 20 Observer que $M = \begin{pmatrix} A & 2A \\ 2A & A \end{pmatrix}$ avec $A = \begin{pmatrix} 1 & -3 \\ 2 & 2 \end{pmatrix}$.
Il faut s'inspirer des questions 9 et 10.
- 21 Observer que $M = \begin{pmatrix} 4A & 2A \\ 2A & 4A \end{pmatrix}$ avec $A = I_2$.

On pourra ensuite réemployer les techniques des questions 9 et 10.

- 22 Poser, pour tout réel t , $X(t) = (x_1(t), x_2(t), x_3(t), x_4(t))$ et vérifier que $X' = M X$. Trouver une équation différentielle simple vérifiée par la fonction $P^{-1} X$ et la résoudre.
- 23 Utiliser la question précédente.

EXERCICE I

1 Si f et g sont continues sur le segment $[-1; 1]$ et à valeurs réelles, fg est continue sur le segment $[-1; 1]$ et à valeurs réelles, donc

$$(f | g) = \int_{-1}^1 f(t)g(t) dt$$

est bien défini. Pour vérifier que la forme $(\cdot | \cdot)$ est un produit scalaire, montrons qu'elle est symétrique, bilinéaire et définie positive. Soient f, g et h continues sur le segment $[-1; 1]$ et à valeurs réelles.

- La forme $(\cdot | \cdot)$ est symétrique. En effet,

$$(f | g) = \int_{-1}^1 f(t)g(t) dt = \int_{-1}^1 g(t)f(t) dt = (g | f)$$

- La forme $(\cdot | \cdot)$ est bilinéaire. Pour la linéarité à gauche :

$$\begin{aligned} (f + \lambda g | h) &= \int_{-1}^1 (f(t) + \lambda g(t))h(t) dt \\ &= \int_{-1}^1 f(t)h(t) dt + \lambda \int_{-1}^1 g(t)h(t) dt \\ (f + \lambda g | h) &= (f | h) + \lambda(g | h) \end{aligned}$$

Pour la linéarité à droite, utilisons la symétrie et la linéarité à gauche :

$$(h | f + \lambda g) = (f + \lambda g | h) = (f | h) + \lambda(g | h) = (h | f) + \lambda(h | g)$$

- La forme $(\cdot | \cdot)$ est positive. En effet,

$$(f | f) = \int_{-1}^1 f(t)^2 dt \geq 0$$

- La forme $(\cdot | \cdot)$ est définie : en effet $f^2 \geq 0$ et on sait que l'intégrale sur un segment d'une fonction continue de signe constant est nulle si et seulement si la fonction est nulle. Ainsi, $(f | f)$ est nul si et seulement si f est nulle.

On en déduit que

La forme $(\cdot | \cdot)$ est un produit scalaire.

Plus généralement, si a et b sont des réels tels que $a < b$, la forme $(f | g)$ définie pour toutes fonctions continues f et g sur $\mathbb{R}$ et à valeurs réelles par l'intégrale de fg sur l'intervalle $[a; b]$ est un produit scalaire.

2 Par imparité, $(u | v) = \int_{-1}^1 t dt = 0$

Ainsi, une base orthonormée de F est $(u/\|u\|, v/\|v\|)$. Or,

$$\|u\|^2 = (u | u) = \int_{-1}^1 1 dt = 2 \quad \text{et} \quad \|v\|^2 = (v | v) = \int_{-1}^1 t^2 dt = \left[\frac{t^3}{3} \right]_{-1}^1 = \frac{2}{3}$$

Une base orthonormée de F est $\left(\frac{u}{\sqrt{2}}, \frac{\sqrt{3}v}{\sqrt{2}} \right)$.

3 Notons p le projeté orthogonal sur F de la fonction $\exp$. Comme on a déterminé une base orthonormée de F à la question précédente, on peut désormais calculer le projeté orthogonal de $\exp$ grâce à la formule suivante :

$$p = \left(\frac{u}{\sqrt{2}} \mid \exp \right) \frac{u}{\sqrt{2}} + \left(\frac{\sqrt{3}v}{\sqrt{2}} \mid \exp \right) \frac{\sqrt{3}v}{\sqrt{2}} = \frac{1}{2}(u \mid \exp) u + \frac{3}{2}(v \mid \exp) v$$

Calculons $(u \mid \exp) = \int_{-1}^1 e^t dt = [e^t]_{-1}^1 = e - \frac{1}{e}$

Par intégration par parties (on pose $f(t) = e^t$, $f'(t) = e^t$, $g(t) = t$, $g'(t) = 1$) :

$$(v \mid \exp) = \int_{-1}^1 t e^t dt = [t e^t]_{-1}^1 - \int_{-1}^1 1 e^t dt = e + \frac{1}{e} - e + \frac{1}{e} = \frac{2}{e}$$

De ceci, on déduit que

Le projeté orthogonal de la fonction exponentielle vaut $p = \frac{1}{2}\left(e - \frac{1}{e}\right)u + \frac{3}{e}v$.

Remarquons que

$$\begin{aligned} \inf_{(a,b) \in \mathbb{R}^2} \int_{-1}^1 (e^t - (a + bt))^2 dt &= \inf_{f \in F} (\exp - f \mid \exp - f) \\ &= (\exp - p \mid \exp - p) \end{aligned}$$

On cherche à calculer la norme de $\exp - p$. Or, par définition du projeté orthogonal, $\exp - p$ est orthogonal à F , et donc normal à p . D'après le théorème de Pythagore,

$$(\exp - p \mid \exp - p) + (p \mid p) = (\exp \mid \exp)$$

Calculons d'abord

$$(\exp \mid \exp) = \int_{-1}^1 e^{2t} dt = \left[\frac{e^{2t}}{2} \right]_{-1}^1 = \frac{e^2 - e^{-2}}{2}$$

Calculons ensuite $(p \mid p)$ en utilisant les différents calculs faits à la question 2 :

$$\begin{aligned} (p \mid p) &= \left(\frac{1}{2}\left(e - \frac{1}{e}\right)u + \frac{3}{e}v \mid \frac{1}{2}\left(e - \frac{1}{e}\right)u + \frac{3}{e}v \right) \\ &= \frac{1}{4}\left(e - \frac{1}{e}\right)^2 (u \mid u) + 2\frac{1}{2}\left(e - \frac{1}{e}\right)\frac{3}{e}(u \mid v) + \left(\frac{3}{e}\right)^2 (v \mid v) \\ &= \frac{1}{4}\left(e - \frac{1}{e}\right)^2 2 + \left(\frac{3}{e}\right)^2 \times \frac{2}{3} \\ &= \frac{e^2}{2} - 1 + \frac{e^{-2}}{2} + 6e^{-2} \\ (p \mid p) &= \frac{e^2}{2} + \frac{13e^{-2}}{2} - 1 \end{aligned}$$

On obtient alors

$$\begin{aligned} (\exp - p \mid \exp - p) &= (\exp \mid \exp) - (p \mid p) \\ &= \frac{e^2 - e^{-2}}{2} - \left(\frac{e^2}{2} + \frac{13e^{-2}}{2} - 1 \right) = 1 - \frac{7}{e^2} \end{aligned}$$

En conclusion,

$$\inf_{(a,b) \in \mathbb{R}^2} \int_{-1}^1 (e^t - (a + bt))^2 dt = 1 - \frac{7}{e^2}$$

Centrale Maths 1 MP 2018 — Corrigé

Ce corrigé est proposé par Céline Chevalier (enseignant-chercheur à l'université) ; il a été relu par Florian Metzger (docteur en mathématiques) et Vincent Puyhaubert (professeur en CPGE).

L'objectif de ce problème est la preuve d'un théorème de Johnson–Lindenstrauss concernant l'aplatissement aléatoire d'un ensemble de points en grande dimension : étant donné $v_1, \dots, v_N$ distincts dans $\mathbb{R}^d$, on cherche à envoyer linéairement ces points dans un espace de plus petite dimension sans trop modifier les distances. Plus précisément, on fixe $\varepsilon > 0$ puis on cherche $f : \mathbb{R}^d \rightarrow \mathbb{R}^k$ linéaire telle que

$$\forall (i, j) \in \llbracket 1; N \rrbracket^2 \quad (1 - \varepsilon) \|v_i - v_j\| \leq \|f(v_i) - f(v_j)\| \leq (1 + \varepsilon) \|v_i - v_j\|$$

Le théorème que l'on cherche à démontrer assure l'existence d'une telle application dès lors que $k \geq c \ln N / \varepsilon^2$ où la constante c ne dépend ni de N , ni de ε . Ce sujet combine un peu d'algèbre euclidienne et beaucoup de probabilités.

- Dans une première partie, on montre d'abord un résultat de projection sur un convexe fermé, puis une inégalité classique de Hölder pour l'espérance, qui servira fréquemment dans la suite. Le sujet définit ensuite la notion d'espérance conditionnelle et introduit les variables aléatoires à queue sous-gaussienne, qui sont les variables aléatoires réelles X telles qu'il existe deux réels strictement positifs a et b vérifiant

$$\forall t \geq 0 \quad \mathbb{P}(|X| \geq t) \leq a \exp(-bt^2)$$

La partie se conclut en prouvant l'inégalité suivante sur ces variables, qui sera elle aussi utilisée plus loin : si δ est un réel tel que $0 \leq |\delta| \leq \sqrt{a/b}$, alors

$$\forall t \geq 0 \quad \mathbb{P}(|X + \delta| \geq t) \leq a \exp(a) \exp(-bt^2/2)$$

- La deuxième partie, assez longue et technique, introduit les variables aléatoires de Rademacher (vérifiant $Z(\Omega) = \{-1; 1\}$ et $\mathbb{P}(Z = -1) = \mathbb{P}(Z = 1) = 1/2$) et se donne pour objectif de prouver l'inégalité de concentration de Talagrand, d'abord sur un cas particulier puis par récurrence : si E est un espace vectoriel euclidien muni d'une base orthonormée $(e_1, \dots, e_n)$, si $\varepsilon_1, \dots, \varepsilon_n$ sont des variables aléatoires de Rademacher indépendantes dans leur ensemble et si $X = \varepsilon_1 e_1 + \dots + \varepsilon_n e_n$, alors, pour tout convexe fermé non vide C de E ,

$$\mathbb{P}(X \in C) \cdot \mathbb{E}(\exp(d(X, C)^2/8)) \leq 1$$

- Enfin, la troisième partie commence par quelques résultats préliminaires de probabilité, dont la difficulté consiste à réutiliser de nombreuses questions des deux premières parties. Elle s'achève par une mobilisation de tous ces résultats pour démontrer le théorème de Johnson–Lindenstrauss.

Ce sujet est long et technique. Il permet de faire le point sur ses capacités à manipuler tous les outils du programme liés aux probabilités, tout en y mêlant un peu d'algèbre euclidienne et de calculs standards en analyse (inégalités). C'est également une bonne occasion d'apprendre à réutiliser des questions, parfois bien plus loin dans le problème.

INDICATIONS

Partie I

- 2 Poser $a = (u - v)/2$ et $b = (u - v')/2$ dans l'égalité de la question 1.
- 3 Montrer que l'ensemble $\{\|u - w\| \mid w \in F\}$ admet une borne inférieure et utiliser la caractérisation séquentielle puis le théorème de Bolzano–Weierstrass.
- 4 Utiliser la convexité de C ainsi que la question 2 afin de prouver l'unicité.
- 6 En suivant l'indication de l'énoncé, utiliser la question 5 pour chaque terme de la somme définissant $E(|XY|)$, puis la formule des probabilités totales.
- 8 À l'aide de la relation de Chasles, exprimer l'intégrale comme somme des intégrales sur tous les intervalles de la forme $[\sqrt{y_i}; \sqrt{y_{i+1}}]$.
- 10 Procéder par inclusion d'événements.
- 13 Justifier que le majorant de la question 12 est supérieur à 1 dans ce cas.

Partie II

- 15 Expliciter $d(X, u)$ à l'aide des coordonnées de u .
- 17 Combiner les questions 16 et 15.
- 20 Expliciter les deux éléments de $C \cap X(\Omega)$ (ils reserviront également dans la suite du problème).
- 22 Montrer que $Y_{-\varepsilon_n} - \varepsilon_n e_n$ et $Y_{\varepsilon_n} + \varepsilon_n e_n$ appartiennent à C .
- 23 Pour la première inégalité, séparer les termes sur E' et ceux sur $\mathbb{R}e_n$ puis utiliser le théorème de Pythagore. Pour la deuxième, utiliser l'inégalité triangulaire puis une inégalité de convexité.
- 24 Utiliser le vecteur appartenant à C_{-1} puis l'indépendance des variables $\varepsilon_1, \dots, \varepsilon_n$.
- 25 Démontrer le lemme suivant :

Lemme : Soit $F: \begin{cases} \mathbb{R}^{n-1} \times \{-1; 1\} \longrightarrow \mathbb{R} \\ (u, \varepsilon) \longmapsto F(u, \varepsilon) \end{cases}$ une fonction quelconque. Alors

$$E(F(X', \varepsilon_n) \mid \varepsilon_n = -1) = E(F(X', -1))$$

puis conclure à l'aide de la linéarité et la croissance de l'espérance conditionnelle.

- 26 Combiner les questions 25 et 6.
- 27 La question 26 reste vraie en échangeant les rôles de 1 et -1 . Appliquer ce résultat modifié à $\lambda = 0$.
- 28 Appliquer l'hypothèse de récurrence à C_{-1} puis la question 7. Conclure à l'aide des questions 26, 27 et 28.
- 30 On pourra dériver deux fois la fonction pour étudier son signe.
- 32 Combiner les questions 29, 31 et 21.
- 33 Utiliser l'inégalité de Markov et la question 32.

Partie III

- 35 Détailler la valeur des normes et appliquer l'inégalité de Cauchy–Schwarz.
- 36 Écrire $M \cdot u = (M - N) \cdot u + N \cdot u$, où $N \in \mathcal{M}_{k,d}(\mathbb{R})$ vérifie $d(M, C) = \|M - N\|_F$.
- 37 Utiliser la contraposée de la question 36 et la question 33.

- 38 Noter $g_1, \dots, g_n$ les valeurs prises par $g(X)$ et poser $m = g_i$ pour un entier i bien choisi.
- 39 Appliquer la question 37 à des réels particuliers dépendant de m et de t , puis la question 38.
- 40 Calculer l'espérance à l'aide de la question 8 et la majorer à l'aide de la question 39.
- 41 Développer $g(X)^2$ et détailler le calcul de l'espérance en utilisant l'indépendance des variables ε_{ij} et la valeur de la norme de u .
- 42 Il suffit de développer le carré dans l'espérance.
- 43 D'après la question 39, la variable aléatoire $g(X) - m$ vérifie les hypothèses de la partie I.D avec $a = 4$ et $b = 1/8$. Noter alors

$$g(X) - \sqrt{k} = (g(X) - m) + (m - \sqrt{k})$$

Vérifier les hypothèses sur δ à l'aide des questions 40 et 42 et conclure à l'aide des questions 12 et 13.

- 44 Détailler la probabilité et appliquer la question 43.

- 45 Poser
$$u_{ij} = \frac{v_i - v_j}{\|v_i - v_j\|} \quad \text{pour } i \neq j$$

C'est un vecteur unitaire de $\mathbb{R}^d$. Réécrire E_{ij} en fonction de ce vecteur afin de pouvoir appliquer la question 44.

- 46 Passer à l'événement contraire et exprimer le contraire d'une intersection en fonction de l'union des contraires.
- 47 Faire le lien avec les événements E_{ij} .

I. PRÉLIMINAIRES

1 Soit $(a, b) \in E^2$. La linéarité et la symétrie du produit scalaire donnent

$$\|a + b\|^2 = \langle a + b | a + b \rangle = \|a\|^2 + \|b\|^2 + 2\langle a | b \rangle$$

et
$$\|a - b\|^2 = \langle a - b | a - b \rangle = \|a\|^2 + \|b\|^2 - 2\langle a | b \rangle$$

d'où

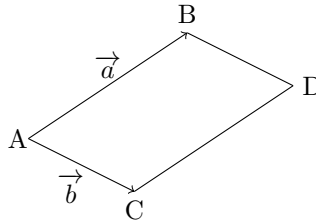
$$\boxed{\forall (a, b) \in E^2 \quad \|a + b\|^2 + \|a - b\|^2 = 2(\|a\|^2 + \|b\|^2)}$$

Ce résultat porte le nom d'égalité du parallélogramme. Il signifie que

Dans un parallélogramme, la somme des carrés des longueurs des diagonales est égale à la somme des carrés des longueurs des quatre côtés.

Par exemple, dans le schéma suivant dans $\mathbb{R}^2$, si $\vec{a} = \overrightarrow{AB}$ et $\vec{b} = \overrightarrow{AC}$, on a

$$AD^2 + BC^2 = 2(AB^2 + AC^2) = AB^2 + AC^2 + CD^2 + BD^2$$



On peut justifier qu'une norme vérifie l'identité du parallélogramme si, et seulement si, elle est euclidienne (c'est-à-dire qu'elle provient d'un produit scalaire). C'est la caractérisation de Fréchet–Von Neumann–Jordan.

2 Soit $(u, v, v') \in E^3$. Supposons $v \neq v'$ et $\|u - v\| = \|u - v'\|$. Posons

$$a = \frac{u - v}{2} \quad \text{et} \quad b = \frac{u - v'}{2}$$

de sorte que
$$a + b = u - \frac{v + v'}{2} \quad \text{et} \quad a - b = \frac{v' - v}{2}$$

Appliquons la question 1 à a et b :

$$\begin{aligned} \left\| u - \frac{v + v'}{2} \right\|^2 + \left\| \frac{v' - v}{2} \right\|^2 &= 2 \left(\left\| \frac{u - v}{2} \right\|^2 + \left\| \frac{u - v'}{2} \right\|^2 \right) \\ &= 2 \left(\frac{1}{4} \|u - v\|^2 + \frac{1}{4} \|u - v'\|^2 \right) \\ \left\| u - \frac{v + v'}{2} \right\|^2 + \left\| \frac{v' - v}{2} \right\|^2 &= \|u - v\|^2 \end{aligned}$$

puisque $\|u - v\| = \|u - v'\|$. Comme $v \neq v'$,

$$\left\| \frac{v' - v}{2} \right\|^2 > 0 \quad \text{donc} \quad \left\| u - \frac{v + v'}{2} \right\|^2 < \|u - v\|^2$$

La norme d'un vecteur étant positive, il vient, par croissance de la racine carrée,

$$\boxed{\left\| u - \frac{v + v'}{2} \right\| < \|u - v\|}$$

Centrale Maths 2 MP 2018 — Corrigé

Ce corrigé est proposé par Angèle Niclas (ENS Lyon) ; il a été relu par Guillaume Batog (professeur en CPGE) et Benjamin Monmege (enseignant-chercheur à l'université).

Ce sujet d'analyse propose d'étudier les fonctions harmoniques, qui sont très utilisées en analyse complexe, et leurs liens avec les séries entières. Il propose également une application de ces résultats à la théorie des équations aux dérivées partielles avec la résolution du problème de Dirichlet sur le disque unité de $\mathbb{R}^2$. Les deux premières parties sont largement liées, tandis que les parties trois et quatre sont indépendantes des deux premières. La cinquième utilise des résultats issus de l'ensemble du sujet et permet une synthèse du problème.

- Dans la première partie, on se familiarise avec les fonctions harmoniques et on démontre quelques propriétés indispensables pour la suite du problème.
- Dans la deuxième partie, on cherche à expliciter la forme des fonctions harmoniques dans le cas où la fonction est à variables séparables puis radiales. En particulier, la formule du laplacien est démontrée en coordonnées polaires.
- Dans la troisième partie, on démontre le principe du maximum faible avec un raisonnement par l'absurde.
- Dans la quatrième partie, on se focalise sur le lien important qui existe entre les fonctions harmoniques et les fonctions développables en série entière. Les résultats qui jalonnent cette partie permettent en particulier de démontrer le célèbre théorème de d'Alembert-Gauss.
- La dernière partie utilise les précédentes pour montrer l'existence et l'unicité d'une solution au problème de Dirichlet sur le disque unité de $\mathbb{R}^2$: étant donné une fonction f continue définie sur le cercle unité, peut-on trouver une fonction harmonique définie sur le disque unité fermé qui coïncide avec f sur le cercle ?

Ce problème nécessite des connaissances solides en calcul différentiel. On manipule de nombreuses dérivées partielles et les calculs peuvent être longs et fastidieux. La connaissance des dérivées partielles en coordonnées polaires est un avantage incontestable puisqu'elle permet la vérification des calculs effectués. Le sujet fait aussi appel à des connaissances en résolution d'équations différentielles du premier et du deuxième ordre, à de nombreuses reprises. La dernière partie nécessite également la maîtrise des théorèmes liés à la régularité des intégrales à paramètre et un certain recul sur le programme de l'année de MP.

La résolution de ce sujet est particulièrement intéressante puisqu'elle permet de démontrer des grands théorèmes d'analyse fonctionnelle à partir d'outils extraits du programme de prépa.

INDICATIONS

Partie I

- 2 Raisonner par récurrence sur l'ordre des dérivées partielles et penser à utiliser le théorème de Schwarz.
- 3 Développer $\Delta(f^2)$ et utiliser le fait qu'une somme d'entiers positifs est nulle seulement si chaque entier est nul.
- 4 Chercher une fonction polynomiale très simple, puis utiliser la question 3.

Partie II

- 5 Calculer les dérivées partielles de f en fonction de u et v et évaluer cette expression en des points où u et/ou v ne sont pas nulles.
- 8 Utiliser la règle de la chaîne.
- 9 Dériver les expressions de la question 8 et utiliser la règle de la chaîne.
- 10 Utiliser les dérivées calculées en question 9.
- 11 Supprimer la dépendance en θ dans le résultat de la question 10 puis chercher l'expression de $\partial g / \partial r$ avant d'en calculer la primitive.
- 12 Chercher une fonction radiale et utiliser la question 11 pour se ramener à la résolution d'un système.
- 13 Chercher un r tel que $u(r) \neq 0$ et obtenir une expression de v en fonction de f et u .
- 14 Utiliser la question 10 et calculer les dérivées partielles de g en fonction de u et v puis évaluer cette expression en des points où u et/ou v ne sont pas nulles.
- 16 Trouver l'expression de z' puis calculer sa primitive.
- 17 Utiliser les questions 15 et 16.
- 18 Séparer le cas $\lambda > 0$ et $\lambda < 0$ et chercher les solutions 2π -périodiques.
- 19 Chercher l'équation vérifiée par Z et la résoudre.
- 20 Faire tendre r vers 0 dans les expressions de la question 18.

Partie III

- 21 Utiliser le fait que $\overline{U}$ soit compact.
- 22 Dresser un tableau de variations de φ et chercher une contradiction.
- 24 Utiliser la question 23 sur g_ε et faire tendre ε vers 0.
- 25 Appliquer la question 24 à $f_1 - f_2$ et $f_2 - f_1$.

Partie IV

- 26 Utiliser les théorèmes de dérivation sous le signe somme pour obtenir les dérivées partielles de f .
- 27 Comparer $\partial f / \partial x$ et $\partial f / \partial y$ et déduire des relations entre les dérivées partielles de u et v .
- 31 Exprimer $\partial H / \partial x$ en fonction de g et de $\operatorname{Re}(H)$.
- 32 Remplacer f par son expression et utiliser un théorème d'inversion de somme et d'intégrale.
- 34 Utiliser la question 32 et l'inégalité triangulaire.

- 33 Conclure grâce aux deux questions précédentes.
- 36 Utiliser la question 32 et le fait que $|f|$ admette un maximum en 0 pour obtenir une intégrale d'une fonction positive égale à 0.
- 37 Appliquer les questions 28 et 34 à l'inverse du polynôme considéré, puis faire tendre r vers $+\infty$.

Partie V

- 38 Utiliser le développement en série entière de $z \mapsto 1/(1 - z)$ et appliquer un théorème d'interversion de série et d'intégrale.
- 39 Choisir $h = 1$ dans la question 38.
- 40 Remarquer que h et $\mathcal{P}$ sont 2π -périodiques.
- 42 Utiliser une suite $(z_n)_{n \in \mathbb{N}}$ qui tend vers $e^{i\varphi}$ et appliquer un théorème d'inversion limite intégrale.
- 43 Utiliser les questions 39 et 40 et séparer l'intégrale en trois parties.
- 44 Utiliser les questions 42 et 43 pour montrer que g est solution du problème, puis la question 25 pour l'unicité.

I. FONCTIONS HARMONIQUES : QUELQUES PROPRIÉTÉS

1 Soient f et g deux fonctions de $\mathcal{H}(U)$ et $\lambda \in \mathbb{R}$. Utilisons la caractérisation des sous-espaces vectoriels :

- $0 \in \mathcal{H}(U)$ car $(x \mapsto 0) \in \mathcal{C}^2(U, \mathbb{R})$ et $\Delta(x \mapsto 0) = 0$.
- Si on pose $h = f + \lambda g$, $h \in \mathcal{C}^2(U, \mathbb{R})$ et par linéarité de la dérivation,

$$\Delta h = \sum_{i=1}^n \frac{\partial^2 h}{\partial x_i^2} = \sum_{i=1}^n \left(\frac{\partial^2 f}{\partial x_i^2} + \lambda \frac{\partial^2 g}{\partial x_i^2} \right) = \sum_{i=1}^n \frac{\partial^2 f}{\partial x_i^2} + \lambda \sum_{i=1}^n \frac{\partial^2 g}{\partial x_i^2} = \Delta f + \lambda \Delta g = 0$$

On en déduit que $h \in \mathcal{H}(U)$.

Ainsi,

$$\boxed{\mathcal{H}(U) \text{ est un sous-espace vectoriel de } \mathcal{C}^2(U, \mathbb{R}).}$$

2 Montrons par récurrence que la proposition

$$\mathcal{P}(k) : \quad \forall (i_1, i_2, \dots, i_k) \in \llbracket 1; n \rrbracket^k \quad \frac{\partial^k f}{\partial x_{i_1} \cdots \partial x_{i_k}} \in \mathcal{H}(U) \gg$$

est vraie pour tout $k \in \mathbb{N}$.

- $\mathcal{P}(0)$ est vraie car $f \in \mathcal{H}(U)$.
- $\mathcal{P}(k) \implies \mathcal{P}(k+1)$: soit $k \in \mathbb{N}$ tel que $\mathcal{P}(k)$ soit vraie. Choisissons $k+1$ indices $i_1, i_2, \dots, i_{k+1}$ de $\llbracket 1; n \rrbracket$ et notons

$$g = \frac{\partial^k f}{\partial x_{i_2} \cdots \partial x_{i_{k+1}}}$$

La propriété $\mathcal{P}(k)$ assure que $g \in \mathcal{H}(U)$. On a alors

$$\frac{\partial^{k+1} f}{\partial x_{i_1} \cdots \partial x_{i_{k+1}}} = \frac{\partial g}{\partial x_{i_1}}$$

Comme $f \in \mathcal{C}^\infty(U, \mathbb{R})$, ses dérivées $(k+1)$ -ième sont de classe $\mathcal{C}^2$ et g vérifie donc les hypothèses du théorème de Schwarz généralisé. On peut ainsi calculer

$$\begin{aligned} \Delta \left(\frac{\partial^{k+1} f}{\partial x_{i_1} \cdots \partial x_{i_{k+1}}} \right) &= \sum_{i=1}^n \frac{\partial^2}{\partial x_i^2} \left(\frac{\partial g}{\partial x_{i_1}} \right) \\ &= \sum_{i=1}^n \frac{\partial}{\partial x_{i_1}} \left(\frac{\partial^2 g}{\partial x_i^2} \right) && \text{(théorème de Schwarz)} \\ &= \frac{\partial}{\partial x_{i_1}} \left(\sum_{i=1}^n \frac{\partial^2 g}{\partial x_i^2} \right) && \text{(linéarité de } \frac{\partial}{\partial x_i} \text{)} \\ \Delta \left(\frac{\partial^{k+1} f}{\partial x_{i_1} \cdots \partial x_{i_{k+1}}} \right) &= \frac{\partial}{\partial x_{i_1}} (0) = 0 && (g \in \mathcal{H}(U)) \end{aligned}$$

On en déduit que $\frac{\partial^{k+1} f}{\partial x_{i_1} \cdots \partial x_{i_{k+1}}} \in \mathcal{H}(U)$. $\mathcal{P}(k+1)$ est donc vraie.

- Conclusion :

$\boxed{\text{Toute dérivée partielle à tout ordre de } f \in \mathcal{H}(U) \cap \mathcal{C}^\infty(U, \mathbb{R}) \text{ appartient à } \mathcal{H}(U).}$

Centrale Informatique MP-PC-PSI 2018 — Corrigé

Ce corrigé est proposé par Cyril Ravat (professeur en CPGE) ; il a été relu par Jean-Julien Fleck (professeur en CPGE) et Guillaume Batog (professeur en CPGE).

Ce sujet d'informatique a pour objectif la simulation microscopique d'un gaz parfait. Le mouvement particulaire d'un tel gaz, abordé en physique en première année de prépa, permet de s'appuyer sur un modèle simple et accessible. L'étude est divisée en cinq parties de tailles sensiblement équivalentes.

- La première aborde l'initialisation de la simulation, c'est-à-dire le placement originel et aléatoire des particules dans l'espace. Si les premières questions, faciles, permettaient à tous les candidats d'engranger quelques points, les dernières sont beaucoup plus difficiles à cause d'un cahier des charges complexe ; elles donnaient l'occasion aux meilleurs candidats de se mettre en valeur. On notera une question 12 très difficile et relativement mal posée.
- On traite ensuite l'aspect physique du mouvement des particules, dans un modèle simplifié à une dimension. Les fonctions demandées sont, tout comme leur support physique, assez élémentaires.
- Dans la troisième partie, le problème de la gestion des différents événements pour un système de N particules est posé. Une solution intéressante est fournie et étudiée en détail, notamment au niveau de la complexité des algorithmes mis en jeu. Ces questions, intéressantes, demandent du temps et de l'attention pour être traitées correctement.
- La quatrième partie rassemble les résultats des questions précédentes pour établir les fonctions globales permettant la réalisation de la simulation. Elle demande d'avoir bien compris les différents modèles et outils utilisés précédemment et favorisait donc les candidats ayant pris le temps de réfléchir sur l'ensemble du sujet.
- Enfin, on exploite une base de données pour enregistrer une partie des informations liées à des simulations. Il s'agit de la partie la moins bien écrite du sujet. Les bases de données sont assez mal formées, très partielles pour les informations stockées et avec un formalisme éloigné des canons du domaine. Bien qu'il n'y ait que trois questions, elles sont répétitives.

Ce sujet est progressif et intéressant (la courte partie 5 exceptée). Le contexte utilisé est assez familier et la physique sous-jacente suffisamment simple pour ne pas poser de problème à la plupart des candidats. Les données sont représentées par des tableaux `numpy` car ils permettent l'addition et la multiplication par des scalaires. Il est donc nécessaire de savoir manipuler ces opérations. Il s'agit, pour les quatre premières parties, d'un bon entraînement accessible dès la fin de la première année. Les notions d'ingénierie numérique n'apparaissent pas, mais des questions associées à la représentation des nombres flottants sont présentes.

INDICATIONS

Partie I

- 1 Ne pas confondre la multiplication par un scalaire d'une liste Python ou d'un tableau `np.array`.
- 10 Il peut être profitable de commencer par un schéma d'une situation quelconque, sans hésiter à créer des positions proches ou éloignées les unes des autres. Puis dérouler l'algorithme en déplaçant les particules sur le schéma. Ne pas oublier de mettre en forme le retour comme une liste de tableaux `np.array`.
- 12 Le terme « histogramme » est très mal choisi. L'énoncé souhaite que l'on trace la fonction de densité de probabilité. Pour les cas $N = 1$ et $N = 5$, la réponse est intuitive. Mais pour le cas $N = 2$, il faut multiplier des probabilités conditionnelles.

Partie II

- 17 Les tableaux `np.array` permettent l'utilisation de l'addition et de la multiplication de façon naturelle. La réponse attendue est très courte.

Partie III

- 20 Il s'agit bien d'une question en une dimension, ce qui simplifie largement l'étude. Trois cas sont possibles en fonction de la valeur précédente de la vitesse.
- 21 Comme à la question précédente, l'étude est en une dimension. Les deux particules ne se choqueront que si elles sont en train de se rapprocher, c'est-à-dire si leur distance diminue en valeur absolue.
- 22 Une fois trouvé l'endroit où insérer l'événement dans le catalogue, la méthode `insert`, dont l'énoncé donne l'aide en annexe, est incontournable.
- 23 La fonction demandée utilise les trois fonctions écrites précédemment. Les fonctions `tr` et `tc` ont un comportement similaire pour leur retour, notamment lorsqu'il n'y a pas d'événement associé. Ce retour est à tester avant la fabrication de l'événement.
- 26 Il faut bien compter toutes les complexités dans le pire des cas où chaque particule est capable de rencontrer toutes les autres. Le résultat peut être exprimé sous la forme d'une somme d'entiers consécutifs qui se simplifie.
- 27 Ne surtout pas aller chercher un algorithme trop compliqué ou éloigné du cours.

Partie IV

- 29 Mettre à jour les positions pour toutes les particules. Ensuite, ne mettre à jour que les vitesses des particules ayant interagi. Toutes les fonctions nécessaires ont déjà été écrites.
- 30 L'invalidation des événements doit se faire à la main, selon un test d'appartenance long à écrire, mais simple à concevoir.
- 31 On ne sait pas à l'avance combien d'itérations seront réalisées. À chaque itération, il faut commencer par supprimer les événements invalides en fin de catalogue.
- 33 L'erreur numérique associée à la représentation des valeurs flottantes est surtout proportionnelle à la valeur représentée.

Partie V

- 34 Il s'agit d'une requête d'agrégation selon un critère à définir.
- 35 Cette question est très proche de la précédente. La jointure n'est pas nécessaire.
- 36 La requête est plus longue à écrire, mais il suffit de traduire ce que dit l'énoncé.

I. INITIALISATION

Pour la seconde fois, le concours Centrale-Supélec utilise la syntaxe des définitions de fonctions appelée « annotations ». Elle rend le code plus explicite en précisant les types des arguments et des retours des fonctions. Il est probable que cela devienne une habitude dans ce concours et c'est une bonne idée dont pourraient s'inspirer les autres.

1 La ligne 9 du code proposé crée un tableau `np.array` d'une unique valeur, générée aléatoirement entre 0 inclus et L exclu.

Il fallait bien lire la documentation de la fonction `np.random.rand`, qui ne fait pas partie des fonctions à connaître.

Attention à ne pas confondre le comportement de la multiplication sur les listes et sur les objets `np.array`. La multiplication d'une liste par un entier provoque une concaténation multiple de cette liste, comme il est rappelé dans l'annexe de l'énoncé, tandis que pour un `np.array` cela multiplie chaque élément du tableau par l'entier ou le flottant multiplicateur. Ce détail aurait pu être rappelé dans l'annexe.

2 L'argument `c` de la fonction `possible` peut être interprété à partir de l'appel à la fonction à la ligne 10. Il s'agit d'un tableau `np.array` à une dimension, contenant la position d'une nouvelle particule à placer parmi celles déjà présentes dans `res`.

3 La ligne 3 évacue les deux cas où le placement de la particule est impossible car trop près des bords de l'espace disponible. Elle conduit la fonction `possible` à renvoyer `False` s'il faut générer une nouvelle position.

4 Les lignes 4 et 5 testent pour chaque particule déjà présente si la nouvelle particule à insérer est trop proche. Elles ont la même conclusion que la ligne 3.

5 La fonction `possible` teste la possibilité de placement de la nouvelle particule. Elle renvoie `True` si c'est possible et `False` s'il faut générer une nouvelle position.

6 Le rejet réalisé à la ligne 3 peut être évité en générant une valeur comprise entre R et $L - R$, en remplaçant la ligne 9 par

```
p = R + (L-2*R) * np.random.rand(1)
```

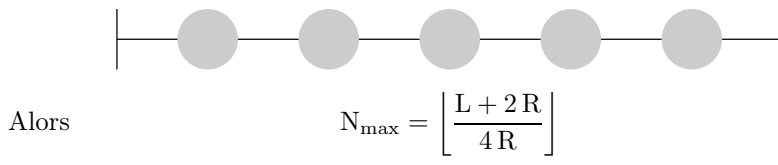
7 Dans la configuration proposée, chaque particule est espacée de 0,5 des deux particules adjacentes ou du bord. Il n'y a donc pas de place pour positionner une quatrième particule, et la boucle `while` devient une « boucle infinie ». Ainsi,

La suite de l'appel à `placement1D` ne termine pas.

8 Dans le cas où $N \ll N_{\max}$, on peut supposer que presque aucun échec de placement de particule ne survient. Le contenu de la boucle `while` est alors répété N fois et contient des instructions de complexité constante ainsi qu'un appel à la fonction `possible`. Cette fonction est de complexité linéaire en le nombre d'éléments déjà dans `res`. Puisque $O(1 + 2 + 3 + \dots + N) = O(N^2)$,

La complexité de la fonction `placement` est quadratique.

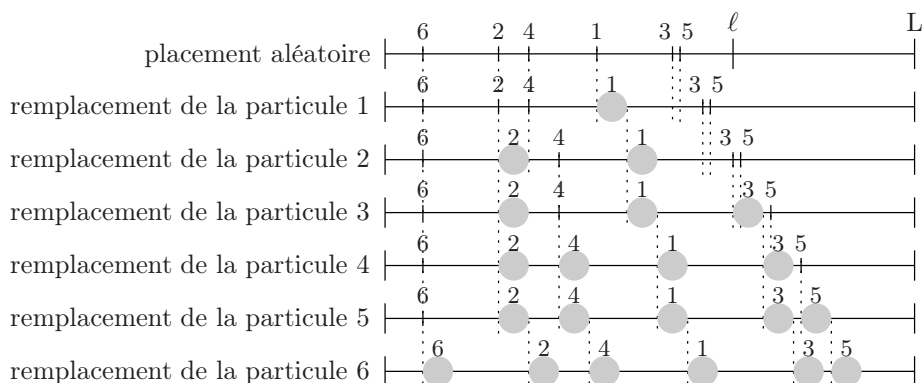
La somme des diamètres des particules à placer à la question 7 est pourtant inférieure à la longueur totale du segment. Cela signifie que le nombre $N_{\max}$ est certainement inférieur au rapport $L/(2R)$. Si l'on souhaite éviter la boucle infinie, il ne faut pas arriver pour autant à la situation où $N - 1$ particules sont écartées l'une de l'autre et du bord d'une distance légèrement inférieure à $2R$. À la limite de ce raisonnement, la configuration est



9 Pour recommencer à zéro le placement des particules dès qu'une nouvelle position est impossible, il suffit d'ajouter ce comportement lorsque `possible(p)` vaut `False`.

```
res = []
while len(res) < N:
    p = L * np.random.rand(1)
    if possible(p): res.append(p)
    else: res = []
return res
```

10 L'énoncé demande une fonction en trois étapes. La troisième est une étape de déplacements successifs des particules, de l'espace $[0; \ell]$ vers l'espace $[0; L]$. Par exemple, ces déplacements peuvent être représentés par le schéma suivant :



Le placement rapide de N particules peut donc se faire ainsi :

```
def placement1Drapide(N, R, L):
    # Étape 1 : calcul de l'espace libre final
    l = L - 2 * R * N
    # Étape 2 : placement aléatoire des N particules sur [0; l]
    positions = l * np.random.rand(N)
    # Étape 3 : déplacement des particules
    for i in range(N):
        for j in range(N): # Déplacement des particules à droite
            if positions[j] > positions[i]:
                positions[j] = positions[j] + 2 * R
        # Transformation en particule réelle
        positions[i] = positions[i] + R
```

Centrale Informatique optionnelle MP 2018

Corrigé

Ce corrigé est proposé par Hugo Menet (ENS Lyon) ; il a été relu par Martin Guy (ENS Lyon) et Guillaume Batog (professeur en CPGE).

L'épreuve propose de travailler sur le déplacement de robots placés sur une grille parsemée d'obstacles via la résolution d'un jeu de société intitulé « Ricochet Robot ». À cette occasion, le sujet fait appel à différentes facettes de l'algorithmique. On demande d'abord de concevoir quelques fonctions (en Caml) manipulant des tableaux et des listes avant d'introduire une nouvelle structure de données, les tables de hachage. La dernière partie est consacrée à l'étude de graphes, notamment au parcours en largeur. Le sujet comporte également quelques calculs de complexité (dont l'un fait intervenir des probabilités) et des études du tri par insertion et de l'algorithme de parcours en largeur d'un graphe.

- La première partie est consacrée à la construction de fonctions manipulant des tableaux qui modélisent les déplacements des robots sur la grille.
- La deuxième partie s'intéresse aux listes. On implémente le tri par insertion et des tests d'appartenance à des listes de couples. Cette partie assez proche du cours permet de vérifier que l'on a bien compris la manipulation de listes en Caml.
- La troisième partie présente les tables de hachage, une structure de données implémentant un « dictionnaire » qui est souvent utilisée en pratique : là où une liste indexe des valeurs par des entiers $0, 1, 2, \dots, n - 1$, un dictionnaire associe des valeurs à des clés quelconques, par exemple des chaînes de caractères. On travaille d'abord sur des tables de largeur fixée avant de se tourner vers des tables dynamiques. La partie se termine par deux calculs de complexité en moyenne.
- Pour la dernière partie, on introduit une modélisation du problème par un graphe. On revient à cette occasion aux fonctions écrites dans la première partie pour déterminer les voisins d'un sommet dans ce modèle. Pour finir, le sujet propose une étude théorique de l'algorithme de parcours en largeur, avec des questions sur la terminaison, la correction et la complexité.

Le sujet est très bien écrit et propose de travailler sur de nombreux aspects du programme d'informatique optionnelle (tri, graphe, calcul de complexité, preuve d'algorithme) et propose de nombreuses fonctions à programmer manipulant listes et tableaux, ce qui en fait un exercice très formateur.

Dans le corrigé, les codes sont écrits en OCaml plutôt qu'en Caml Light, comme le demandait pourtant l'énoncé, car c'est le langage qui deviendra obligatoire aux concours à partir de la session 2019.

INDICATIONS

Partie I

- 1 Commencer par comparer **a** à l'élément placé au milieu du tableau et déterminer, à partir de cette comparaison, dans quelle moitié du tableau il faut continuer la recherche.
- 2 Utiliser une seule fois la fonction **dichotomie** pour trouver les déplacements dans deux directions opposées.
- 4 Écrire une condition pour que le robot en position (c, d) bloque l'autre robot dans son déplacement de (a, b) vers (a', b') .
- 5 Écrire une fonction récursive qui parcourt la liste **q** pour mettre à jour la copie de la matrice de déplacements robot par robot.
- 6 Dénombrer les suites de longueur k de couples (R, d) où R a 4 valeurs possibles, l'indice d'un des 4 robots, et d a 4 valeurs possibles, ouest, est, nord, sud.

Partie II

- 7 Écrire une fonction récursive pour parcourir la liste **q**.
- 8 Partir de la liste vide et la garder triée en ajoutant un à un les éléments de **q**.
- 9 Un meilleur cas est donné par une liste déjà triée, un pire cas par une liste triée dans l'ordre décroissant.
- 10 On cherche un élément de la forme (x, y) avec y quelconque dans une liste **q** de couples.
- 11 Utiliser la même structure de fonction récursive que pour la question 10.

Partie III

- 12 Remarquer que si $P = \sum_{i=0}^p \lambda_i X^{2i}$ alors $P = \lambda_0 + X^2 Q$ où $Q = \sum_{i=0}^{p-1} \lambda_{i+1} X^{2i}$.
- 14 Utiliser la fonction **mem1** de la question 10.
- 15 Adapter la fonction de la question 14 en utilisant cette fois la fonction **assoc** de la question 11.
- 16 Utiliser la fonction **mem1** de la question 10 pour séparer les deux cas.
- 17 Construire une fonction **del1** sur le modèle de la fonction **mem1** de la question 10 qui supprime un élément. Puis adapter la fonction de la question 16.
- 18 L'hypothèse de hachage uniforme simple implique que la clé k est choisie de sorte que toutes les alvéoles sont équiprobables pour son haché $h_w(k)$. Exprimer alors le coût de la recherche de cette clé dans cette alvéole en fonction de la longueur de celle-ci.
- 19 Les clés de la table sont également choisies de manière aléatoire dans cette question. Utiliser la formule des probabilités totales pour en déduire la probabilité qu'une clé choisie uniformément parmi les clés de la table soit hachée dans une alvéole fixée.
- 20 Adapter la réponse de la question 13.
- 21 Utiliser une fonction auxiliaire pour parcourir une liste afin de parcourir chacune des listes de la table **donnees** initiale. Pour chaque élément rencontré, calculer son haché avec la nouvelle fonction de hachage et le placer dans la liste correspondante dans une nouvelle table de données préalablement construite. Pour la complexité, remarquer qu'on rencontre une seule fois chacun des n éléments de la table.

- 22 Commencer par ajouter l'élément comme en question 16 en utilisant à nouveau la fonction `mem1` de la question 10 puis utiliser la fonction `rearrange` si besoin.

Partie IV

- 24 Utiliser la fonction `deplacements_robots` de la question 5. On pourra également utiliser la fonction `insertion` de la question 7. Il pourra être utile de construire une fonction `suppr` pour supprimer un élément d'une liste.
- 25 Montrer qu'un sommet du graphe ne peut pas être ajouté deux fois dans la file `F`.
- 26 Pour un sommet s quelconque accessible depuis s_0 , montrer par récurrence que tous les sommets d'un chemin de s_0 à s sont ajoutés à `F` pendant l'exécution de l'algorithme.
- 27 Remarquer qu'un chemin de s_0 à s est un chemin de s_0 à π_s auquel on ajoute le parcours de l'arête entre π_s et s .
- 28 Considérer le lemme suivant : « Il existe un nombre q_n de tours de boucle **tant que** tel que la file `F` contient exactement l'ensemble des sommets à distance n de s_0 ». L'adapter, par exemple, en s'intéressant aux sommets qui n'ont pas encore été enfilés dans `F`, afin de pouvoir le démontrer par récurrence.
- 29 Raisonner comme en question 21 et remarquer que

$$\sum_{s \in S} |V(s)| = 2|A|$$

où $V(s)$ est l'ensemble des voisins du sommet s .

1 Appliquons le principe de la dichotomie pour rechercher un élément **a** dans un tableau trié **t**. Après avoir comparé **a** avec l'élément **b** situé au milieu du tableau **t**, la recherche continue de manière récursive dans la première ou la seconde moitié du tableau, selon que **a** est inférieur ou supérieur à **b**.

Utilisons pour ce faire une fonction récursive `recherche_sous_tableau` de type `int -> int array -> int -> int -> int`. Elle prend en argument l'élément **a**, le tableau **t** et deux indices pour marquer le début et la fin d'un sous-tableau qui vérifie l'invariant

$$t.(debut) \leq a < t.(fin)$$

Elle renvoie l'unique indice **i** du sous-tableau tel que

$$t.(i) \leq a < t.(i + 1)$$

Cette définition n'est valable que pour un tableau de taille au moins deux.

Comme on suppose le tableau de taille au moins deux, le cas de base de la fonction récursive correspond au cas où le sous-tableau contient deux éléments, l'invariant donne donc l'élément à renvoyer.

Pour un sous-tableau de longueur strictement supérieure à deux, une comparaison entre **a** et l'élément **b** au milieu du sous-tableau permet de conserver soit le sous-tableau à gauche, dans le cas où $a < b$, soit le sous-tableau à droite. Pour conserver l'invariant, on garde l'élément milieu dans chacun des sous-tableaux. Cela permet également d'éviter de se retrouver à traiter un sous-tableau de taille 1 pour lequel la fonction n'est pas définie (ce qui arriverait en coupant un tableau de taille trois en deux sous-tableaux disjoints). De plus initialement on a bien

$$t.(0) \leq a < t.(n - 1)$$

où **n** est la taille du tableau **t** puisqu'on suppose que l'élément **a** en entrée est strictement inférieur au plus grand élément de **t**.

La quantité `fin - debut` décroît bien strictement à chaque appel récursif si le cas terminal est celui d'un tableau de taille deux, et ce même si on garde la quantité `milieu` et pas `milieu - 1` comme on le fait habituellement pour une recherche dichotomique.

```
let dichotomie a t =
  let rec recherche_sous_tableau a t debut fin =
    let milieu = (debut + fin)/2 in (* t.(debut) <= a < t.(fin) *)
    if (debut + 1) = fin then debut
    else
      if a < t.(milieu) then
        recherche_sous_tableau a t debut milieu
      else
        recherche_sous_tableau a t milieu fin
  in recherche_sous_tableau a t 0 (Array.length(t)-1);;
```

On remarque que la fonction a bien une complexité logarithmique en la taille du tableau comme demandé.

Du fait du changement de langage au programme, les fonctions sur les `vect` de Caml Light sont remplacées par des fonctions sur les `Array` de OCaml.

Mines Maths 1 MP 2018 — Corrigé

Ce corrigé est proposé par Sophie Rainero (professeur en CPGE) ; il a été relu par Juliette Brun Leloup (professeur en CPGE) et Florian Metzger (docteur en mathématiques).

Ce sujet traite :

- du lemme de sous-additivité de Fekete, qui affirme que si $(u_n)_{n \in \mathbb{N}^*}$ est une suite réelle sous-additive, c'est-à-dire telle que $u_{i+j} \leq u_i + u_j$ pour tout $(i, j) \in \mathbb{N}^{*2}$, alors la suite $(u_n/n)_{n \in \mathbb{N}^*}$ converge ;
- du théorème d'Erdős-Szekeres : dans une liste de $pq + 1$ nombres réels distincts, il y a au moins une liste extraite croissante de longueur $p + 1$ ou une liste extraite décroissante de longueur $q + 1$.

Il en donne ensuite des applications probabilistes. Il se compose de cinq parties.

- Dans la partie A, on établit deux résultats préliminaires indépendants : une majoration de l'espérance d'une variable aléatoire à support dans $\llbracket 1 ; n \rrbracket$ et une minoration de $n!$, pour $n \in \mathbb{N}^*$.
- Dans la partie B, on définit les limites inférieure et supérieure d'une suite bornée et on démontre le lemme de sous-additivité de Fekete. Il s'agit d'un exercice classique.
- La partie C donne une application probabiliste de ce lemme. On y considère une suite $(X_i)_{i \in \mathbb{N}^*}$ de variables aléatoires mutuellement indépendantes et de même loi. On définit, pour tout $n \in \mathbb{N}^*$,

$$Y_n = \frac{1}{n} \sum_{i=1}^n X_i$$

et on démontre que, pour tout réel x , la suite $(P(Y_n \geq x)^{1/n})_{n \in \mathbb{N}^*}$ converge.

- La partie D établit le théorème d'Erdős-Szekeres.
- Enfin, on étudie dans la partie E le comportement asymptotique d'une suite aléatoire définie à l'aide d'une variable aléatoire à valeurs dans l'ensemble S_n des permutations de $\llbracket 1 ; n \rrbracket$.

Les parties A, B et D sont autonomes. La partie E est la plus ardue.

Ce sujet est abordable dès la fin de la MPSI puisqu'il porte sur les suites en analyse et les variables aléatoires à support fini en probabilités, à l'exception de la partie C qui s'intéresse à des suites de variables aléatoires discrètes dont l'étude n'est effectuée qu'en seconde année. C'est un joli sujet, dans lequel on démontre des résultats intéressants et originaux, en mobilisant finalement assez peu de connaissances. Notons cependant que les questions 16, 17 et 20 requièrent une bonne maîtrise technique.

INDICATIONS

Partie A

- 1 Découper en deux la somme définissant $E(X)$.

Partie B

- 3 Se souvenir que la borne supérieure est le plus petit des majorants.
- 4 Utiliser la question 3 et considérer une autre suite v décroissante et plus grande que u pour montrer que $\overline{u}$ est la plus petite.
- 6 Procéder par double implication. Utiliser le théorème d'encadrement dans un sens et la définition de la limite dans l'autre.
- 7 Remarquer puis démontrer par récurrence que $u_{ab} \leq b u_a$ pour tout $(a, b) \in \mathbb{N}^{*2}$.
- 8 Appliquer la question 7 avec $n = 1$, puis utiliser les résultats des questions 5, 6 et 7.
- 9 Prouver à l'aide des questions 8 et 5 que la limite inférieure de la suite $(u_n/n)_{n \in \mathbb{N}^*}$ est supérieure ou égale à sa limite supérieure. Conclure à l'aide du résultat de la question 6.

Partie C

- 10 Comparer les événements $\{Y_n < x\}$ et $\{X_1 < x\} \cap \dots \cap \{X_n < x\}$. Utiliser la croissance de la probabilité et les propriétés de la suite $(X_i)_{i \in \mathbb{N}^*}$. La seconde partie de la question se traite de manière similaire.
- 11 Utiliser la croissance de la probabilité et les propriétés de la suite $(X_i)_{i \in \mathbb{N}^*}$. Remarquer que les variables aléatoires Y_n et

$$\frac{1}{n} \sum_{k=m+1}^{m+n} X_k$$

ont la même loi.

- 12 Distinguer deux cas selon la valeur de $P(X_1 < x)$ en se servant de la question 10. Lorsque $P(X_1 < x) < 1$, introduire la suite de terme général $-\ln(P(Y_n \geq x))$ et se servir des questions 9 et 11 pour conclure.

Partie D

- 13 Dans l'hérédité, observer que si un jeton j est dans la dernière pile, sa valeur est nécessairement plus petite que celle d'un jeton de l'avant-dernière pile, celui qui était au sommet de cette pile au moment où on a posé le jeton j .
- 14 Distinguer deux cas selon que le nombre de piles obtenues à la fin du processus est supérieur ou égal à $q + 1$ ou pas. Dans le premier cas, utiliser la question 13 et dans le second se demander combien de jetons peut contenir chaque pile.

Partie E

- 15 Démontrer que les variables aléatoires A_1 et A_2 ne sont pas indépendantes en considérant par exemple l'événement $\{A_1 = 1, A_2 = 1\}$.

- 16 Attention, la liste s doit être strictement croissante, il y a une petite erreur dans l'énoncé. En notant E l'ensemble des permutations σ de $\llbracket 1; n \rrbracket$ telles que

$$\sigma(s_1) < \cdots < \sigma(s_k)$$

démontrer que
$$P(A^s) = P(B \in E) = \frac{\text{Card } E}{n!}$$

et dénombrer les éléments de E .

- 17 Introduire la variable aléatoire $B' = \varphi \circ B$ où $\varphi: \sigma \mapsto (\sigma(n), \dots, \sigma(1))$ et justifier que B' est de même loi que B . Définir les variables aléatoires C'_n et D'_n associées à B' comme C_n et D_n le sont à B .

Pour la minoration de l'espérance, écrire que $2E(C_n) = E(C_n) + E(D_n)$ et appliquer la question 14 avec $p = q = \max\{k \in \mathbb{N}^* \mid 1 + k^2 \leq n\}$.

- 18 Utiliser la question 16 et dénombrer les listes strictement croissantes, de longueur k , d'éléments de $\llbracket 1; n \rrbracket$.
- 19 Vérifier que $k = -\lfloor -\alpha e \sqrt{n} \rfloor$ convient. Remarquer que C_n est à valeurs entières pour majorer la probabilité considérée à l'aide des questions 18 et 2.
- 20 Observer que la majoration de la question 1 est également valable pour $m \geq n+1$. Utiliser les questions 19 et 1 avec $\alpha = 1 + n^{-1/4}$ et l'entier $k \in \mathbb{N}^*$ qui est associé à α dans la question 19.

Faire un développement asymptotique pour justifier que la suite $(\varepsilon_n)_{n \in \mathbb{N}^*}$ tend vers 0 et conclure à l'aide des questions 3, 5 et 6.

1 Soit X une variable aléatoire à valeurs dans $\llbracket 1; n \rrbracket$. Comme elle est à support fini, son espérance est bien définie, en tant que somme finie. Soit $m \in \llbracket 1; n \rrbracket$; calculons $E(X)$:

$$E(X) = \sum_{k=1}^n k P(X = k) = \sum_{k=1}^{m-1} k P(X = k) + \sum_{k=m}^n k P(X = k)$$

$$\text{Or, } \sum_{k=1}^{m-1} k P(X = k) \leq (m-1) \sum_{k=1}^{m-1} P(X = k) \leq (m-1) \sum_{k=1}^n P(X = k) = m-1$$

par positivité d'une probabilité et parce que

$$\sum_{k=1}^n P(X = k) = 1 \quad (\text{puisque } X(\Omega) = \llbracket 1; n \rrbracket)$$

$$\text{De plus, } \sum_{k=m}^n k P(X = k) \leq n \sum_{k=m}^n P(X = k) = n P(X \geq m)$$

puisque $X(\Omega) = \llbracket 1; n \rrbracket$. Finalement, par somme d'inégalités,

$$\boxed{\forall m \in \llbracket 1; n \rrbracket \quad E(X) \leq m-1 + n P(X \geq m)}$$

2 La fonction définie sur $[0; +\infty[$ par $x \mapsto \ln(x)$ est continue, croissante et positive. Soit $n \in \mathbb{N}^*$. Pour tout $k \in \mathbb{N}^*$, $\ln$ est donc majorée par $\ln(k+1)$ sur le segment $[k; k+1]$. Ainsi, par croissance de l'intégrale, pour tout $k \in \mathbb{N}^*$,

$$\int_k^{k+1} \ln(t) dt \leq \int_k^{k+1} \ln(k+1) dt = (k+1-k) \ln(k+1) = \ln(k+1)$$

Sommons ces inégalités pour k allant de 1 à $n-1$, il vient

$$\sum_{k=1}^{n-1} \int_k^{k+1} \ln(t) dt \leq \sum_{k=1}^{n-1} \ln(k+1) = \sum_{j=2}^n \ln(j) = \sum_{j=1}^n \ln(j)$$

puisque $\ln(1) = 0$. Ainsi, en vertu de la relation de Chasles,

$$\int_1^n \ln(t) dt = \sum_{k=1}^{n-1} \int_k^{k+1} \ln(t) dt \leq \sum_{j=1}^n \ln(j)$$

Il reste à calculer l'intégrale :

$$\int_1^n \ln(t) dt = [t \ln(t) - t]_1^n = n \ln(n) - n - \ln(1) + 1 = n \ln(n) - n + 1$$

En conclusion,

$$\boxed{\forall n \in \mathbb{N}^* \quad n \ln(n) - n + 1 \leq \sum_{k=1}^n \ln(k)}$$

Rappelons brièvement comment retrouver une primitive de la fonction continue $\ln$ sur l'intervalle $]0; +\infty[$. Soit $x > 0$. Alors

$$\int_1^x \ln(t) dt = \int_1^x u(t) v'(t) dt$$

en posant $u: t \mapsto \ln(t)$ et $v: t \mapsto t$. Les fonctions u et v étant de classe $\mathcal{C}^1$ sur le segment d'intégration, il vient par intégration par parties

$$\int_1^x \ln(t) dt = [u(t)v(t)]_1^x - \int_1^x u'(t)v(t) dt = [t \ln(t)]_1^x - \int_1^x 1 dt$$

Mines Maths 2 MP 2018 — Corrigé

Ce corrigé est proposé par Florian Metzger (docteur en mathématiques) ; il a été relu par Thierry Limoges (professeur en CPGE) et Sophie Rainero (professeur en CPGE).

Ce sujet porte sur l'étude de plusieurs aspects des racines carrées d'une matrice complexe donnée. Il est composé de cinq parties non indépendantes.

- La première partie est l'occasion d'étudier l'existence d'une racine carrée pour quelques matrices données. On prouve également l'existence et l'unicité d'une racine carrée pour toute matrice symétrique réelle à spectre dans $\mathbb{R}_+$. Les questions sont très classiques.
- La deuxième partie a pour but de montrer l'existence d'une racine carrée de toute matrice complexe inversible en résolvant le système linéaire associé après trigonalisation de la matrice.
- La troisième partie est consacrée à la méthode de Newton pour le calcul effectif d'une racine carrée. On construit une suite de matrices analogue à celle utilisée dans le cas des suites réelles pour résoudre l'équation $f(x) = 0$. On montre la convergence de la méthode avec des outils d'algèbre linéaire, dont certains font appel à des résultats sortis du programme il y a peu (norme subordonnée). Certaines questions nécessitent une solide maîtrise du programme pour les résoudre.
- Dans la quatrième partie est décrite une formulation récurrente équivalente à la méthode de Newton qui permet de montrer la convergence de la méthode pour le calcul de la racine carrée. Beaucoup de raisonnements par récurrence sont employés.
- Enfin, la dernière partie permet d'obtenir une condition suffisante sur le conditionnement d'une matrice afin que la méthode de Newton soit stable, c'est-à-dire qu'elle converge encore vers la racine si on bouge un peu le premier terme de la suite utilisée pour le calcul des termes par récurrence.

Le sujet aborde plusieurs notions fondamentales du programme de prépa : algèbre linéaire, réduction, espaces vectoriels normés, calcul différentiel. Mathématiquement intéressant, de difficulté inégale (parfois soutenue mais jamais excessive), il constitue un bon entraînement aux concours.

INDICATIONS

Partie A

- 1 Considérer pour tout réel $a \neq 0$, la symétrie par rapport à la droite d'équation $y = ax$ dans $\mathbb{R}^2$. Attention : on demande quelles sont les matrices racines de A qui sont des polynômes en A. On ne cherche pas les polynômes parmi l'infinité de racines trouvée précédemment. Cette remarque est valable pour la deuxième question.
- 2 Pour l'existence de racines carrées, on pourra expliciter des matrices triangulaires supérieures strictes.
- 3 Pour l'existence utiliser la réduction des matrices réelles symétriques. Pour l'unicité montrer l'égalité des spectres puis $\text{Ker}(A - \lambda I_n) = \text{Ker}(R - \sqrt{\lambda} I_n)$ pour toute racine carrée R de A vérifiant les conditions requises.

Partie B

- 4 Exprimer sous la forme d'un système l'égalité des coefficients de U^2 et T en utilisant le fait que les matrices sont triangulaires. Pour la résolution du système, remarquer que la condition à vérifier permet de construire les coefficients par récurrence double sur les indices i et j . Noter en outre que $u_{i,i} + u_{j,j} = 0$ implique $t_{i,i} = t_{j,j}$ puis considérer les éléments diagonaux distincts de T.
- 5 La condition sur le spectre de A assure que les racines carrées de ses valeurs propres sont de partie imaginaire non nulle. Se servir alors de la construction faite à la question 4.

Partie C

- 6 Utiliser la formule sommatoire du coefficient $(AB)_{i,j}$.
- 7 Raisonner par double implication. Pour le sens direct utiliser un vecteur propre de B. Pour l'autre, justifier que m_A et m_B sont premiers entre eux et écrire une relation de Bezout.
- 8 Pour $\lambda \in \text{sp } A \cap \text{sp } B$, montrer que $\lambda \in \text{sp } ({}^t B)$ et considérer deux vecteurs propres pour A et ${}^t B$.
- 9 Remarquer que $H \in \text{Ker } dF_X$ s'écrit $XH = H(-X)$ et mettre cela en perspective avec les questions 7 et 8.
- 10 Montrer plus généralement que dans E espace vectoriel normé de dimension finie, l'ensemble $GL(E)$ est un ouvert en explicitant l'inverse de $u + h$ avec la série de terme général $(-1)^n (h u^{-1})^n$ pour h assez petit. On pourra d'abord montrer que

$$u \longmapsto \sup_{\|x\| \leq 1} \|u(x)\|$$

définit une norme sur $\mathcal{L}(E)$ qui vérifie la même propriété sur l'espace vectoriel $\mathcal{L}(\mathcal{M}_n(\mathbb{C}))$ que celle démontrée à la question 6.

- 11 Expliciter dF_{X^*+H} et son inverse.
- 12 Utiliser la question 11 avec $X = X^* + H$ ainsi que la norme de la question 10.
- 13 Remarquer que $X_{k+1} = G(X_k)$ et en déduire le résultat par récurrence sur k . L'énoncé donne une propriété sur ρ que l'on n'est pas en mesure de prouver : établir que $\|X_k - X^*\| \leq (\rho C)^{2^k} / C$ pour tout $k \geq 0$ avec $0 < \rho < \text{Min}(r, 1/C)$.

Partie D

- 14 Raisonner par récurrence pour les deux implications. Pour la réciproque, montrer que dF_{X_k} est inversible en remarquant que $dF_{X_k}(Y) = -F(X_k)$ est une équation affine n'admettant qu'une seule solution.
- 15 Raisonner par récurrence et prouver que $G_k = H_k$.
- 16 Utiliser la relation (II) pour prouver les propriétés demandées par récurrence sur k .
- 17 Remarquer que $\lambda_{0,\ell} = \mu$ pour initialiser une récurrence sur k et faire appel à la question 16.
- 18 La question 17 entraîne que $\lambda_{k,\ell} \xrightarrow[k \rightarrow +\infty]{} \sqrt{\lambda_\ell}$ pour tout ℓ . Utiliser ensuite une diagonalisation de V_k et passer à la limite.

Partie E

- 19 Calculer $(V_0 + \Delta)(V_0^{-1} - V_0^{-1}\Delta V_0^{-1})$ puis montrer que $(\Delta V_0^{-1})^2 = 0$.
- 20 Utiliser la question 19 pour prouver le résultat par récurrence sur k .
- 21 La question 20 fournit un critère de convergence portant sur η . En déduire une condition suffisante sur le conditionnement.

1 Soient $a \in \mathbb{R}^*$ et s_a la symétrie axiale par rapport à la droite d'équation $y = ax$ dans $\mathbb{R}^2$. Alors, d'après le cours, s_a est une application linéaire involutive : $s_a^2 = \text{Id}_{\mathbb{R}^2}$. Soit S_a la matrice de s_a dans la base canonique. On a alors $S_a^2 = I_2$. Comme $s_a \neq s_b$ pour tous réels non nuls $a \neq b$ et que $\mathbb{R}$ est de cardinal infini et inclus dans $\mathbb{C}$, il en résulte que

La matrice $A = I_2$ admet une infinité de racines carrées dans $\mathcal{M}_2(\mathbb{C})$.

La matrice de la symétrie par rapport à la droite d'équation $y = (\tan \alpha)x$ est

$$M_\alpha = \begin{pmatrix} \cos(2\alpha) & \sin(2\alpha) \\ \sin(2\alpha) & -\cos(2\alpha) \end{pmatrix}$$

En effet on vérifie que

$$M_\alpha^t (\cos \alpha, \sin \alpha) = {}^t (\cos \alpha, \sin \alpha)$$

ainsi que

$$M_\alpha^t (-\sin \alpha, \cos \alpha) = -{}^t (-\sin \alpha, \cos \alpha)$$

Ce qui caractérise ladite symétrie.

On calcule aisément avec $A = I_2$ que $A^k = I_2$ pour tout entier $k \in \mathbb{N}$. Soit $P \in \mathbb{C}[X]$ écrit sous forme développée $P = \sum_{k=0}^n a_k X^k$. Il vient donc

$$P(A) = \sum_{k=0}^n a_k I_2 = \left(\sum_{k=0}^n a_k \right) I_2 = P(1) I_2$$

Il en découle que

$$\begin{aligned} P(A)^2 = A &\iff P(1)^2 I_2 = I_2 \\ &\iff P(1)^2 = 1 \\ &\iff P(1) = \pm 1 \\ P(A)^2 = A &\iff P(A) = \pm I_2 \end{aligned}$$

Les seules racines carrées de I_2 qui sont des polynômes en I_2 sont I_2 et $-I_2$.

2 Posons pour tout $a \in \mathbb{C}$

$$J_a = \begin{pmatrix} 0 & 1 & a \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$$

Il vient alors $J_a^2 = A$ pour tout $a \in \mathbb{C}$. Comme $J_a \neq J_b$ pour tous nombres complexes $a \neq b$, on obtient

La matrice $\begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$ admet une infinité de racines carrées dans $\mathcal{M}_3(\mathbb{C})$.

On calcule aisément $A^2 = 0$ puis $A^k = 0$ pour tout entier $k \geq 2$. Soit alors P un polynôme de $\mathbb{C}[X]$ écrit sous forme développée $P = \sum_{k=0}^n a_k X^k$. Il vient donc

$$P(A) = \sum_{k=0}^n a_k A^k = a_0 I_3 + a_1 A$$

Comme I_2 et A commutent, il en découle avec la formule du binôme de Newton

$$P(A)^2 = a_0^2 I_2 + 2a_0 a_1 A = \begin{pmatrix} a_0^2 & 0 & 2a_0 a_1 \\ 0 & a_0^2 & 0 \\ 0 & 0 & a_0^2 \end{pmatrix}$$

Mines Informatique MP-PC-PSI 2018 — Corrigé

Ce corrigé est proposé par Jean-Julien Fleck (professeur en CPGE) ; il a été relu par Virgile Andreani (ENS Ulm) et Guillaume Batog (professeur en CPGE).

Le sujet s'inspire d'un problème physique réel (mesures de houle en divers points du globe) et se découpe en cinq grandes parties qui visent chacune à balayer une partie du programme d'informatique commune.

- La première partie s'intéresse au stockage des données récoltées par les instruments de mesure. Elle permet de vérifier que les candidats ont des notions concernant la place occupée en mémoire par un caractère et sur la lecture d'un fichier de données.
- La deuxième partie permet de rentrer plus avant dans le traitement des données en demandant d'implémenter quelques algorithmes simples qui sont au programme de prépa : calcul d'une moyenne, intégration à partir d'une liste de valeurs, recherche d'une particularité dans une liste, calcul de propriétés à partir des spécifications de l'énoncé, etc.
- La troisième partie, après l'implémentation d'une recherche de maximum sur une liste de listes, s'intéresse plus particulièrement au programme de deuxième année avec une petite incursion dans les algorithmes de tri, demandant par exemple de compléter une implémentation préremplie du tri rapide, puis d'écrire entièrement une implémentation d'un tri par insertion. Une analyse de programme est aussi demandée pour y détecter un calcul inefficace et demander de l'améliorer.
- La quatrième partie concerne les bases de données. Il s'agit de survoler rapidement toute une page de présentation de la base de données considérée pour pouvoir écrire les trois requêtes SQL demandées. Là encore, il ne s'agit pas de comprendre *in extenso* tous les tenants et les aboutissants de la base, mais d'identifier les points clés afin de répondre rapidement à la question posée.
- La cinquième partie présente et demande d'implémenter l'algorithme de Cooley-Tukey de transformée de Fourier discrète en imposant une approche récursive (au programme de seconde année). C'est néanmoins tout à fait abordable en première année.

Le sujet est globalement équilibré. Il comporte de nombreuses questions simples de programmation et explore l'ensemble du programme des deux années (avec notamment des questions sur les tris et une implémentation récursive pour le programme de seconde année) mais peut être abordé en grande partie en première année.

Comme chaque année, l'épreuve des Mines nécessite d'être efficace vu sa durée limitée (seulement 1h30). Les questions ne sont pas particulièrement difficiles et il faut savoir aller à l'essentiel pour pouvoir toutes les traiter, quitte à en laisser certaines de côté si elles semblent demander trop de temps de réflexion.

INDICATIONS

Partie I

- 4 L'ouverture d'un descripteur de fichier se fait à l'aide de la commande `open`. La lecture des lignes dans un tableau se fait à l'aide de `readlines`. Ne pas oublier la conversion en flottant à l'aide de la fonction `float` qui gère naturellement les signes et les caractères de fin de ligne.

Partie II

- 7 La méthode des trapèzes sur un découpage $(x_i)_{i \in \llbracket 0; n \rrbracket}$ en n intervalles revient à approximer

$$\int_{x_0}^{x_n} f(x) \, dx \approx \sum_{i=0}^{n-1} \left(\frac{f(x_i) + f(x_{i+1})}{2} \right) \times (x_{i+1} - x_i)$$

La moyenne d'une fonction η sur un intervalle de temps t_{tot} se définit comme

$$\langle \eta(t) \rangle = \frac{1}{t_{\text{tot}}} \int_0^{t_{\text{tot}}} \eta(t) \, dt$$

- 8 Penser à calculer la moyenne en dehors de la boucle pour éviter de refaire de nombreuses fois le même calcul.
- 9 Le calcul de la moyenne est déjà de complexité linéaire : il faut que celle-ci soit fournie en argument de la fonction si on veut espérer atteindre une complexité en $O(1)$ dans le meilleur des cas.
- 10 Attention, la définition des indices qu'il faut stocker dans la liste des successeurs n'est plus celle des questions 8 et 9 : il y a un décalage d'une position supplémentaire.
- 11 Penser à utiliser la fonction précédente pour calculer la liste des successeurs et utiliser le *slicing* pour découper en tranches correspondant aux vagues.
- 12 Utiliser de même la fonction définie à la question précédente pour obtenir les différentes vagues. Attention au fait que la hauteur maximale se calcule à cheval sur deux vagues (maximum de l'une auquel on soustrait le minimum de la suivante). Attention aussi au fait qu'il faut calculer le maximum précédant le premier PND.

Partie III

- 13 La structure particulière de la liste passée en argument impose d'implémenter la recherche de maximum à la main.
- 14 Ne pas essayer de comprendre ce que fait le code en détail, il suffit de trouver « par homogénéité » la forme attendue pour le pivot.
- 16 Penser à la manière dont on trie ses cartes en propageant la place libre jusqu'à atteindre le bon emplacement.
- 17 Recalculer de nombreuses fois la même chose ne sert à rien : penser à calculer la moyenne une unique fois avant de rentrer dans la boucle.

Partie IV

- 19 La deuxième requête peut se faire en comptant le nombre d'apparition d'une bouée dans la table **Tempete**, ledit nombre devant être égal à 0 si la bouée n'a pas connu de tempête. On peut aussi utiliser une différence ensembliste à l'aide de **EXCEPT** ou **MINUS**.

La troisième requête se fait assez simplement avec une jointure et une unique fonction d'agrégation.

Partie V

- 20 On peut estimer la complexité en procédant d'une manière proche du tri fusion : on a $k = \log_2(N)$ étages de divisions par 2 avec une complexité linéaire à chaque étage.

I. STOCKAGE INTERNE DES DONNÉES

1 Hormis la première ligne qu'on demande d'ignorer, chaque ligne est constituée de 8 caractères (5 chiffres, un signe, un point et le caractère de fin de ligne comme le signale l'énoncé) donc occupe 8 octets. 20 minutes correspondent à 1 200 s. En outre, comme il y a deux mesures par seconde, cela correspond à 2 400 mesures au total, soit une taille de $8 \times 2\,400 = 19\,200$ octets = 19,2 ko.

2 La campagne de mesure fait état d'un fichier récolté toutes les demi-heures pendant 15 jours. Il y a donc $15 \times 24 \times 2 = 720$ fois les informations précédentes collectées, ce qui représente donc une taille totale de 13 824 ko, soit 13,8 Mo.

Une carte mémoire de 1 Go est largement suffisante.

3 Ôter un chiffre revient à passer chaque ligne de 8 à seulement 7 octets, d'où une réduction de $1/8 \approx 12\%$ de la taille totale du fichier.

4 La lecture d'un fichier texte peut se faire de la manière suivante

```
with open('donnees.txt') as f: # Ouverture du descripteur de fichier
    L = f.readlines()          # Lecture effective du fichier
    liste_niveaux = []         # Définition du conteneur
    for i in range(1,len(L)): # On saute la première ligne de texte
        liste_niveaux.append(float(L[i])) # et on convertit en flottants
```

Bien sûr, on peut aussi demander à Numpy de faire tout le travail à notre place

```
import numpy as np # Importation de la bibliothèque
# On saute la première ligne et on convertit au vol en liste car np.loadtxt
# renvoie normalement un np.array
liste_niveaux = list(np.loadtxt('donnees.txt',skiprows=1))
```

La lecture de données numériques dans un fichier est tellement utile en pratique (par exemple en TIPE) qu'il faut que ce soit maîtrisé par les candidats. Ici, l'usage de la fonction `float` n'est pas rappelé, mais on peut partir du principe qu'elle a été conçue de manière à ce que tous les nombres représentés de manière « usuelle » puissent se convertir facilement depuis une représentation en chaîne de caractères. En particulier, le `+` n'est pas gênant en début de nombre, ni le caractère de fin de ligne (ou tout autre espace qui pourrait être présent tout à droite ou tout à gauche de la chaîne de caractères). En revanche, `float` n'est pas capable d'effectuer des opérations de calcul comme « `float("2+2")` », il ne faut tout de même pas trop lui en demander.

La construction avec `with` permet de s'assurer que le descripteur de fichier sera fermé quoi qu'il arrive et que le fichier ne soit pas corrompu (voir par exemple <https://tinyurl.com/with-open-python>), mais la structure suivante sera acceptable de la même manière.

```
f = open('donnees.txt') # Ouverture du descripteur de fichier
L = f.readlines()       # Lecture effective du fichier
liste_niveaux = []      # Définition du conteneur
for i in range(1,len(L)): # On saute la première ligne de texte
    liste_niveaux.append(float(L[i])) # et on convertit en flottants
f.close()               # Fermeture du descripteur de fichier
```

Mines Informatique optionnelle MP 2018 — Corrigé

Ce corrigé est proposé par Martin Guy (ENS Lyon) ; il a été relu par Hugo Menet (ENS Lyon) et Benjamin Monmege (enseignant-chercheur à l'université).

L'épreuve est divisée en quatre parties portant sur la recherche d'un motif s dans une chaîne de caractères t .

- Le sujet commence par une introduction générale au problème et définit toutes les notions nécessaires. Après avoir prouvé des propriétés basiques sur les motifs et implémenté des fonctions Caml les manipulant (sous forme de liste), on étudie un premier algorithme naïf de recherche de motif. Cette première partie permet de vérifier que l'on maîtrise les bases de Caml et des études de complexité.
- La deuxième partie, indépendante de la première, présente les automates finis déterministes à repli (AFDR). Les premières questions permettent de s'approprier la notion, qui se différencie subtilement des automates classiques au programme. Il s'ensuit des implémentations en Caml, accompagnées de calculs de complexité. C'est le moment de vérifier sa maîtrise des types en Caml.
- La troisième partie, la plus technique et la plus conséquente, introduit les automates de Knuth-Morris-Pratt, basés sur l'algorithme éponyme. On prouve différentes propriétés utiles pour implémenter un algorithme de recherche de motif plus efficace. Cette partie permet de vérifier les acquis en manipulation de mots et en programmation.
- Enfin, on étend la recherche à un ensemble de motifs dans un dictionnaire S . On utilise des AFDR reconnaissant plusieurs motifs, et on termine par une nouvelle approche de reconnaissance pour un ensemble de motifs. La dernière question, ouverte, est intéressante.

Ce sujet plutôt long propose un bon équilibre entre implémentation en Caml et preuves sur les automates, ce qui en fait un très bon exercice technique.

Dans le corrigé, les codes sont écrits en OCaml plutôt qu'en Caml Light, comme le demandait pourtant l'énoncé, car c'est le langage qui deviendra obligatoire aux concours à partir de la session 2019.

INDICATIONS

Partie I

- 1 Trouver un contre-exemple où s apparaît deux fois dans t de sorte que les deux occurrences se superposent dans t .
- 2 Sans se soucier des lettres dans s et dans t , quelles sont les positions où l'on peut trouver le motif s dans t ?
- 4 Parcourir en même temps les deux listes $\mathbf{s}$ et $\mathbf{t}$ et comparer lettre par lettre. Pour la complexité, séparer selon que k est plus petit ou plus grand que n .
- 5 Tester chaque position possible à l'aide de la fonction `prefixe`. Il y a une erreur dans l'énoncé : les arguments en entrée sont de type `int list` et non `string`.

Partie II

- 7 Utiliser le fait que pour $q \in \mathcal{Q}_A$, on a $\rho(q) < q$ et qu'une suite d'entiers strictement décroissante bornée devient stationnaire.
- 8 Commencer par redessiner les états et les transitions existantes. Ensuite, compléter l'automate avec les transitions manquantes en utilisant la caractérisation des états de repli $\rho^j(q_{i-1})$ avec j le plus petit entier tel que $\delta(\rho^j(q_{i-1}), u_i)$ est défini.
- 10 Itérer sur chaque ligne pour copier la matrice de transitions.
- 11 Pour tout état q et tout symbole α où $\delta(q, \alpha)$ n'est pas défini, utiliser la transition depuis l'état de repli $\rho(q)$.
- 13 Utiliser une fonction auxiliaire récursive pour parcourir le motif u , en faisant attention à l'ordre de reconstruction de la liste d'indices.

Partie III

- 14 Placer les états puis les transitions, et enfin calculer la fonction de repli ρ en trouvant parmi les préfixes de s le plus grand suffixe possible étant aussi un préfixe. On pourra faire un tableau où, pour chaque préfixe de s , on affiche le plus grand suffixe possible.
- 16 Cette question comporte une erreur pour $i = 1$. Considérer $i \geq 2$. Remarquer que le plus long suffixe $u_1 \dots u_j$ de $u_1 \dots u_i$ peut être vu comme le plus long suffixe de $u_1 \dots u_{i-1}$ auquel on rajoute le symbole u_i . Exhiber alors une récurrence.
- 17 Séparer le code en trois morceaux : définir les structures de données et initialiser avec les bonnes valeurs ; calculer toutes les transitions ; puis calculer la fonction de repli à l'aide de la caractérisation de la question 16.
- 18 Remarquer que pour $j \leq i$, $\rho^j(i) \leq i - j$ et se servir du fait que $\delta(i - 1, u_i) = i$. Pour la seconde partie de la question, faire apparaître une somme télescopique.
- 19 Calculer séparément la complexité de chaque morceau de code et utiliser le résultat de la question 18.

Partie IV

- 23 Il suffit d'appeler `recherche_kmp` sur chaque élément du dictionnaire et de rassembler les résultats dans une seule et même liste.
- 24 Plutôt que de traiter chaque motif indépendamment comme en question 23, créer un automate à l'image de la question 22 pour ensuite y trouver les occurrences.

I. RECHERCHE NAÏVE D'UN MOTIF

1 Soient s et t deux chaînes de caractères sur l'alphabet $\Sigma = \{a, b\}$. En prenant $s = abba$ et $t = abbabba$, on trouve des occurrences de s dans t aux positions $y = 4$ et $y' = 7$. La chaîne s est de longueur $k = 4$, ce qui donne $y' - k + 1 = 4$. Ainsi,

Il est possible d'avoir deux occurrences y et y' de s dans t avec $y < y'$ et $y \geq y' - k + 1$.

2 Soient $s = \alpha_1 \dots \alpha_k$ et $t = \beta_1 \dots \beta_n$ deux chaînes de caractères sur un alphabet Σ . Par définition, une occurrence indique la position du dernier caractère du motif s dans la chaîne t . Ce motif s étant de longueur k , la première occurrence de s ne peut se trouver avant la position k . On peut trouver d'autres occurrences aux positions $k + 1$, $k + 2 \dots$ et ce jusqu'à la position n . Par conséquent, on peut en trouver au maximum $n - k + 1$. Ainsi,

Le nombre maximal d'occurrences d'un motif de longueur k dans une chaîne de longueur $n \geq k$ est $n - k + 1$.

Cette borne est atteinte dans le cas où $\alpha_i = a$ pour tout $i \leq k$ et $\beta_j = a$ pour tout $j \leq n$ (on trouve alors une occurrence pour chaque position possible). Autrement dit,

La borne est atteinte si s et t ne sont constitués que du même symbole.

3 On calcule récursivement la taille de la liste en comptant 1 à chaque élément vu.

```
let rec longueur l = match l with
| [] -> 0
| t::q -> 1 + longueur q;;
```

Le nombre $C(n)$ d'appels récursifs lors de l'appel de `longueur` avec une liste de longueur $n \in \mathbb{N}$ vérifie la définition par récurrence :

$$\begin{cases} C(0) = 0 \\ \forall n \in \mathbb{N}^*, C(n) = 1 + C(n - 1) \end{cases}$$

On a donc $C(n) = n$, pour tout $n \in \mathbb{N}$. À chaque appel de `longueur`, on exécute un nombre constant d'opérations élémentaires. Ainsi,

La fonction `longueur` a une complexité en $O(n)$.

4 On parcourt récursivement les deux listes représentant les chaînes de caractères s et t simultanément. On arrête le parcours dès que deux lettres de s et t sont différentes. On conclut également négativement dans le cas où le motif s est plus long que la chaîne t , auquel cas s ne peut pas être un préfixe.

```
let rec prefixe s t = match s,t with
| [], _ -> true
| si::q, [] -> false
| si::q, ti::v -> (si = ti) && (prefixe q v);;
```

Séparons l'étude de la complexité de `prefixe s t` en deux cas, en notant k et n les longueurs respectives de s et t . Si $k > n$, on parcourt chaque symbole de t , ce qui fait n opérations. Si $k \leq n$, on fait au plus k opérations. Dans tous les cas,

La fonction `prefixe` a une complexité en $O(\min(k, n))$.

5 La recherche la plus directe est de tester, pour chaque position possible i dans t , si s est un préfixe de $t_i \dots t_n$. Si tel est le cas, on a trouvé une occurrence de s dans t à la position $i + k - 1$. Ainsi, la première occurrence testée est celle en position k .

Pour cela, on implémente une fonction auxiliaire récursive **parcourt**, de type `int list -> int list -> int list`, permettant de parcourir le texte t , tout en maintenant la position i de la première lettre. Initialement, t est le texte complet et i vaut k . Pour chaque position dans t , si s est un préfixe, on garde la position, et dans tous les cas on appelle récursivement la fonction en incrémentant i . La liste des occurrences est ainsi construite récursivement, dans l'ordre croissant.

```
let recherche_naive s t =
  let k = longueur s in
  let rec parcourt t i = match t with
    | [] -> []
    | _::q -> if prefixe s t then
      i::(parcourt q (i+1))
    else
      parcourt q (i+1)
  in parcourt t k;;
```

L'énoncé indique que **recherche_naive** est de type `string -> string -> int list`, mais il doit s'agir d'une erreur étant donné qu'il est aussi indiqué que le type `string` ne sera jamais utilisé.

6 Pour chaque position possible d'un motif dans t , c'est-à-dire $\{k, \dots, n\}$, on appelle la fonction **prefixe**, ce qui fait $n - k + 1$ appels d'une complexité $O(\text{Min}(k, n))$. On effectue de plus un appel à **longueur** de complexité $O(k)$. Ainsi, en supposant $k \leq n$,

La complexité de **recherche_naive** est en $O((n - k + 1) \times k)$.

II. AUTOMATES FINIS DÉTERMINISTES À REPLI

7 Pour un état $q \in \mathcal{Q}_A$, la suite des $(\rho^j(q))_{j \in \mathbb{N}}$ est une suite d'entiers naturels strictement décroissante tant que $\rho^j(q) \neq 0$ (car $\rho(p) < p$ pour $p \neq 0$). Elle converge donc vers l'état $q = 0$, où elle devient stationnaire car $\rho(0) = 0$ par définition. Ainsi, il existe un rang j à partir duquel $\rho^i(q) = 0$ pour tout $i \geq j$. De plus, par définition des AFDR, on impose que pour tout $\alpha \in \Sigma$, $\delta(0, \alpha)$ soit défini. On a donc que

Pour tout $q \in \mathcal{Q}_A$, pour tout $\alpha \in \Sigma$, il existe $j \geq 0$ tel que $\delta(\rho^j(q), \alpha)$ est défini.

8 L'AFDC à construire doit avoir le même nombre d'états et reconnaître le même langage. On commence donc par redessiner les mêmes états en ajoutant les transitions déjà existantes. Enfin, il reste à compléter l'automate avec les transitions manquantes, en posant pour tout état q , $\delta(q, \alpha) = \delta(\rho^j(q), \alpha)$ avec j le plus petit entier tel que la transition soit définie.

X/ENS Maths A MP 2018 — Corrigé

Ce corrigé est proposé par Florian Metzger (docteur en mathématiques) ; il a été relu par Sélim Cornet (ENS Paris-Saclay) et Benoit Chevalier (ENS Ulm).

Ce sujet traite de la décomposition d'une matrice en valeurs singulières, ce qui est une généralisation du théorème spectral à des matrices non carrées : toute matrice $A \in \mathcal{M}_{n,p}(\mathbb{R})$ de rang k s'écrit $A = U \operatorname{diag}(\sqrt{\lambda_1}, \dots, \sqrt{\lambda_k}) {}^tV$, avec $U \in \mathcal{M}_{n,k}(\mathbb{R})$, $V \in \mathcal{M}_{p,k}(\mathbb{R})$ vérifiant ${}^tUU = I_k$ et ${}^tVV = I_k$. Les réels λ_i sont les valeurs propres strictement positives de $A {}^tA$. Les démonstrations de ce résultat et de quelques applications qui en découlent font l'objet du sujet.

- Les trois questions préliminaires permettent d'établir des inégalités, qui seront nécessaires ultérieurement, sur la norme F (pour Froebenius) associée au produit scalaire canonique sur $\mathcal{M}_{n,p}(\mathbb{R})$ défini par $\langle A, B \rangle = \operatorname{Tr}({}^tAB)$.
- La première partie est l'occasion de prouver l'existence d'une décomposition en valeurs singulières. Elle ne comporte pas de difficulté particulière si l'on est à l'aise avec l'algèbre linéaire.
- La deuxième partie permet de montrer que la meilleure approximation (au sens de la norme F) d'une matrice A de rang k par une matrice de rang l s'obtient par une sorte de « troncature à l'ordre l » des matrices obtenues dans la décomposition en valeurs singulières. La dernière question de cette partie demande d'excellentes capacités d'initiative, de synthèse et de persévérance dans les calculs !
- La troisième partie montre des résultats sur des matrices de projection attachées aux matrices obtenues dans la décomposition en valeurs singulières, ainsi que des résultats topologiques utiles dans les espaces de matrices.
- Enfin, dans la dernière partie, on explicite l'espace tangent à $\mathcal{M}_{n,p}^k(\mathbb{R})$ en une matrice A de rang k en fonction des matrices U, Σ, V intervenant dans sa décomposition en valeurs singulières, à savoir : si $A = U\Sigma {}^tV$, alors l'espace tangent est

$$T_A = \{ \overline{U}\Sigma {}^tV + U\overline{\Sigma} {}^tV + U\Sigma {}^t\overline{V} \mid (\overline{U}, \overline{\Sigma}, \overline{V}) \in \mathcal{E} \text{ et } {}^t\overline{U}U = {}^t\overline{V}V = 0 \}$$

avec

$$\mathcal{E} = \mathcal{M}_{n,k}(\mathbb{R}) \times \mathcal{M}_k(\mathbb{R}) \times \mathcal{M}_{p,k}(\mathbb{R})$$

Si une inclusion est rapidement obtenue, il faut en revanche beaucoup plus travailler pour montrer que les vecteurs tangents sont de cette forme. Certaines questions demandent un engagement dans le sujet et une compréhension fine des résultats démontrés et des techniques employées.

En conclusion, les questions préliminaires ainsi que les trois premières parties (à l'exception de la dernière question de la deuxième) sont abordables pour des candidats solidement préparés. Le sujet est très fourni pour le temps imparti et la dernière partie est ardue. Il établit cependant des résultats intéressants et permet de bien s'entraîner sur l'algèbre linéaire au programme de prépa.

INDICATIONS

Preliminaires

- 1 Écrire le produit scalaire $\langle A, B \rangle_F$ avec la définition et développer les coefficients du produit matriciel AB .
- 2 Mettre à profit l'inégalité de Cauchy-Schwarz dans $\mathbb{R}^p$ muni du produit scalaire canonique.
- 3 On pourra commencer par utiliser l'inégalité de Cauchy-Schwarz pour $\langle \cdot, \cdot \rangle_F$ puis montrer que $\|{}^tMM\|_F \leq \|M\|_F^2$ pour toute matrice M .

Première partie

- 5.a Utiliser le fait que $\|AX\|_2^2 = {}^tX^tAAX$ pour X vecteur propre de A . Pour l'égalité des images raisonner par inclusion et dimension.
- 5.b Il est intéressant d'utiliser la relation $\|v\|_2^2 = {}^tvv$ pour v vecteur colonne.
- 6.a La matrice S est symétrique réelle à valeurs propres positives. On pourra ordonner ses valeurs propres par ordre décroissant. Utiliser le théorème spectral pour construire U et vérifier que cette matrice convient en raisonnant sur des produits matriciels par blocs.
- 6.b L'inclusion $\text{Im } S \subset \text{Im } U$ fournit un début de réponse. Prouver ensuite que $\text{rg } U = k$. Utiliser la caractérisation d'un projecteur comme idempotent puis montrer que son image et son noyau sont orthogonaux.
- 6.c Effectuer les produits matriciels en utilisant les définitions des matrices ainsi que ${}^tUU = I_k$. Montrer que $U^tUA = A$.
- 7 Se servir de la relation $U^tUA = A$ dans le calcul du produit matriciel $U\Sigma^tV$.

Deuxième partie

- 8.a Montrer que $A - A\tilde{V}{}^t\tilde{V}$ et $A\tilde{V}{}^t\tilde{V}$ sont orthogonaux pour $\langle \cdot, \cdot \rangle_F$.
- 8.b Exprimer $\|A\tilde{V}{}^t\tilde{V}\|_F^2$ à l'aide d'une trace faisant intervenir $V, \tilde{V}$ et Λ . Développer ensuite cette trace à l'aide d'une formule sommatoire. Il pourra être judicieux d'explicitier les coefficients $({}^t\tilde{V}V)_{m,h}$.
- 9.a Montrer que les colonnes de $\tilde{V}$ et respectivement V sont des familles orthonormées de vecteurs de $\mathbb{R}^p$ qui peuvent être complétées en bases orthonormées. Prouver que $\sum_{1 \leq j \leq l} b_j - \sum_{l+1 \leq i \leq k} a_i \geq 0$ en développant les sommes.
- 9.b Utiliser la relation de la question 8.b ainsi que la majoration de la question 9.a avec l'inégalité $\lambda_l > \lambda_{l+1}$.
- 9.c Difficile question de synthèse! Appliquer le résultat de la question 9.b à $\tilde{V}$, matrice constituée des vecteurs colonnes d'une base orthonormée de $(\text{Ker } M)^\perp$, qui est de dimension l . Remarquer que ${}^t\tilde{V}\tilde{V} = I_l$ et qu'alors $M\tilde{V}{}^t\tilde{V} = M$. Justifier que $A - A\tilde{V}{}^t\tilde{V}$ et $A\tilde{V}{}^t\tilde{V} - M$ sont orthogonaux pour $\langle \cdot, \cdot \rangle_F$ et utiliser le théorème de Pythagore ainsi que le résultat de la question 9.b pour obtenir la minoration et l'unicité du point d'atteinte de l'égalité. Il restera enfin à voir que la matrice proposée convient à l'aide d'un produit par blocs.

Troisième partie

- 10.a Montrer que $\text{Ker } M_{V,W}$ est réduit au vecteur nul ou remarquer qu'un produit par blocs permet de déterminer explicitement l'inverse en fonction de V et W .
- 10.b Commencer par prouver l'indication fournie par l'énoncé et l'utiliser pour montrer que l'intersection des sous-espaces est réduite au vecteur nul. Montrer ensuite que la somme de leurs dimensions est égale à p .
- 10.c Montrer que $P_{V,W}X = X$ si $X \in \text{Im } V$ et $P_{V,W}X = 0$ si $X \in (\text{Im } W)^\perp$.
On pourra calculer $M_{V,W} \begin{pmatrix} 0_{k,1} \\ X \end{pmatrix}$ pour $X \in (\text{Im } W)^\perp$ ainsi que $M_{V,W} \begin{pmatrix} I_p \\ 0_{k,p} \end{pmatrix}$ en notant ainsi les vecteurs colonne définis par blocs.
- 11 Utiliser une formule du déterminant qui montre son caractère polynomial en les coefficients de la matrice ainsi que la formule de l'inverse qui fait appel à la comatrice.
- 12 Remarquer que l'application $f : W \mapsto {}^tWV$ est continue et que $f(V) = I_k$.

Quatrième partie

- 13.a Utiliser la caractérisation du rang par les déterminants extraits pour montrer que $\text{rg}(U + t\bar{U}) = k$ pour t au voisinage de 0, idem pour Σ et V .
- 13.b Prouver l'inégalité $\text{rg}(ABC) \leq \min(\text{rg } A, \text{rg } B, \text{rg } C)$ avec la question 13.a.
- 13.c L'application $t \mapsto \gamma(t)$ est polynomiale de degré 3 en t .
- 14.a Montrer que T_A est l'image d'une application linéaire ψ et déterminer $\text{Ker } \psi$.
- 14.b Montrer que $N_A \perp T_A$ puis expliciter les matrices de N_A et mettre cet espace en bijection avec un espace de matrices. Pour ce faire on pourra écrire que les matrices U, V de rang k sont équivalentes à la matrice diagonale par blocs avec un bloc I_k et des zéros partout ailleurs.
- 15.a Prouver et utiliser le fait que $(\text{Im } M)^\perp = \text{Ker } {}^tM$ pour toute matrice M .
- 15.b Noter que A vérifie (C). L'inclusion $\text{Im}(\tilde{A}V {}^tV) \subset \text{Im } \tilde{A}$ assure qu'il suffit de prouver que $\text{rg}(\tilde{A}V {}^tV) \geq k$ si $\tilde{A}$ est ε -proche de A et de même avec ${}^t\tilde{A}U {}^tU$. S'inspirer alors de 13.b : le rang ne peut qu'augmenter localement.
- 16.a Établir par double inclusion que $\text{Ker } \phi = N_A$.
- 16.b Faire appel aux résultats des questions 14.b et 16.a.
- 16.c Attention : ici on ne suppose pas que tWV est inversible donc le cadre de la troisième partie n'est pas applicable. Justifier que $(\text{Im } W)^\perp = \text{Ker } \tilde{A}$. Prouver ensuite que $\text{Im } V \cap \text{Ker } \tilde{A} = \{0\}$ en utilisant le fait que c'est le noyau de $\tilde{A}|_{\text{Im } V}$. Enfin utiliser cette somme directe pour obtenir l'égalité des matrices.
- 17 Attention : ici la restriction n'est pas faite sur un sous-espace vectoriel, il faut donc comprendre l'injectivité au sens d'une application quelconque et montrer $\pi_A(B) = \pi_A(B') \implies B = B'$. Mettre à profit le ε construit à la question 15.b.
- 18.a Utiliser que $\mathcal{M}_{n,p}(\mathbb{R})$ est somme directe de N_A et T_A , lesquels sont orthogonaux, pour montrer que l'application donnée vérifie la définition du projecteur orthogonal attendu.
- 18.b Prouver que $AB \in N_A^\perp$ pour toute matrice $B \in \mathcal{M}_p(\mathbb{R})$.
- 18.c On pourra justifier que $P_{V,V} = V {}^tV$ et remarquer que le membre de droite est de la forme $\rho_A(A')$ avec A' à expliciter.

- 18.d Se servir des résultats des questions 3 et 18.c. Calculer les normes de $I_n - U^t U$, $V^t V$ ainsi que $I_p - V^t V$.
- 19 Remarquer qu'il suffit d'établir que $\mathcal{T}$ l'ensemble des vecteurs tangents est dans le noyau de ρ_A . Pour ce faire, fixer $T \in \mathcal{T}$ et un arc c dérivable en 0 vérifiant $c(0) = A$ et $c'(0) = T$. Montrer et exploiter ensuite que

$$\rho_A(T) = \lim_{t \rightarrow 0} \frac{1}{t} \rho_A(c(t))$$

à l'aide des questions 18.d et 15.b. Prouver que $P_{V, W_0} = P_{V, V}$ où l'on définit $W_s = {}^t c(s) U^t U$ et utiliser le résultat de la question 12 en établissant l'égalité $P_{V, W_s} = P_{V, {}^t c(s) U \Sigma^{-1}}$ et remarquant que ${}^t c(s) U \Sigma^{-1} \in \mathcal{V}$ pour s suffisamment petit.

X Maths B MP 2018 — Corrigé

Ce corrigé est proposé par Juliette Brun Leloup (professeur en CPGE) ; il a été relu par Sophie Rainero (professeur en CPGE) et Tristan Poullaouec (professeur en CPGE).

L'objet du problème consiste à étudier la borne inférieure de la forme linéaire

$$L: \begin{cases} A_N \longrightarrow \mathbb{R} \\ P \longmapsto \int_{-1}^1 P(x) \, dx \end{cases}$$

définie sur le sous-ensemble A_N convexe de $\mathbb{R}_N[X]$, constitué des polynômes P à valeurs positives sur $[-1; 1]$ tels que $P(1) = P(-1) = 1$.

- Les questions préliminaires sont consacrées à l'étude de l'existence de cette borne inférieure et à montrer qu'elle est atteinte en au moins un polynôme dont la fonction polynomiale est paire. Il s'agit de questions essentiellement topologiques dont les résultats sont donnés dans l'énoncé.
- Dans la première partie, très classique, on étudie une famille de polynômes orthogonaux pour le produit scalaire sur $\mathbb{R}[X]$ défini par

$$\langle P \mid Q \rangle = \int_{-1}^1 P(x)Q(x) \, dx$$

Le but est d'établir certaines propriétés de cette famille et de calculer les normes de ces polynômes pour pouvoir réutiliser certains résultats dans la troisième et dernière partie. Elle n'a pas de lien avec les questions préliminaires et à elle seule, elle peut constituer un entraînement intéressant en algèbre bilinéaire.

- La deuxième partie s'intéresse à la localisation des racines d'un polynôme pair de A_N qui minimise la forme linéaire L . On montre que toutes les racines de ce polynôme, noté R_N , sont dans l'intervalle $] -1; 1 [$. Les questions 7, 8 et 9 sont abordables, traitant de décomposition de polynômes en produits de facteurs irréductibles et utilisant des résultats des préliminaires. Les questions 10 et 11 sont en revanche plus ardues, avec l'étude de lieux géométriques, ce qui n'est plus tout à fait dans l'esprit du programme actuel.
- Enfin, la dernière partie fait la synthèse des deux précédentes pour déterminer explicitement l'expression du polynôme R_N en fonction des polynômes introduits dans la première partie. Cette partie est tout à fait abordable et on peut la traiter sans problème à condition d'admettre les résultats précédents et d'avoir trouvé l'expression de la norme des polynômes de la famille introduite à la première partie.

Mis à part la deuxième partie qui, sur certains points, est un peu technique, l'ensemble du problème constitue un entraînement intéressant pour la préparation aux écrits. Les questions sont parfois difficiles mais elles prennent leur sens une fois que l'on est bien rentré dans le sujet.

INDICATIONS

Questions préliminaires

- 1.c Utiliser la caractérisation séquentielle pour montrer que A_N est fermé. Considérer l'équivalence de la norme $\|\cdot\|_1$ et de la norme infinie sur l'intervalle $[-1; 1]$ pour montrer que la limite de la suite considérée est bien dans A_N .
- 2.a Remarquer que L est continue et que la borne inférieure est atteinte sur le compact $B_{f, \|\cdot\|_1}(0, 2)$.
- 2.b Montrer que, comme $\mathbb{R}_N[X]$ est de dimension finie, B_N est un fermé borné.
- 2.c Considérer le polynôme $(P(X) + P(-X))/2$ où P appartient à B_N .

Première Partie

- 3.c Écrire $(X^2 - 1)^j = (X - 1)^j(X + 1)^j$ et utiliser la formule de Leibniz pour développer le polynôme P_j .
- 4 Considérer $j < k$ et faire j intégrations par parties successives en dérivant successivement les polynômes issus de P_j .
- 5.a Reprendre la succession d'intégrations par parties de la question 4 avec $j = k$.
- 5.b Dans le calcul de $I_{j-1} - I_j$, faire une intégration par parties en considérant une primitive de la fonction $x \mapsto x(1 - x^2)^{j-1}$ et en dérivant la fonction $x \mapsto x$.

Deuxième Partie

- 7 Utiliser la décomposition en produit de facteurs irréductibles dans $\mathbb{C}[X]$ de R_N puis la parité de R_N , ses coefficients réels et enfin ses valeurs en 1 et en -1 .
- 10.a Résoudre directement l'équation en considérant l'écriture algébrique de z .
- 10.b En remplaçant y par sa valeur obtenue à la question 10.a se ramener à une inégalité sur w qu'on démontre à l'aide de l'inégalité triangulaire sur la norme des complexes. La stricte égalité est obtenue en montrant que le cas d'égalité est absurde.
- 11 Faire la synthèse des questions 8, 9 et 10.
 - Supposer $s \geq 1$ et considérer le polynôme S_N . Montrer qu'il est égal à R_N et aboutir à une absurdité.
 - Raisonner de même pour montrer que pour tout $1 \leq j \leq r$, $c_j \in [-1; 1]$.
 - Pour le cas $t \geq 1$, considérer le polynôme

$$V_N(X) = \prod_{j=1}^r \frac{X^2 - c_j^2}{1 - c_j^2} \prod_{\ell=1}^t \left[\frac{X^2 - y_\ell^2}{1 - y_\ell^2} \right]^2$$

et montrer que V_N appartient à A_N et vérifie $L(V_N) < L(R_N)$ puis conclure.

Troisième Partie

- 12 Raisonner par l'absurde en supposant que $\deg(R_N) = 2p$ avec $p < n$ puis considérer le polynôme $X^2 R_N$.
- 13 Reprendre la décomposition en produit de facteurs irréductibles dans $\mathbb{C}[X]$ du polynôme R_N en distinguant 0 parmi les racines. Utiliser la positivité de la fonction polynomiale associée à R_N sur $[-1; 1]$ pour conclure.
- 16 Montrer que la parité de n est la même que celle de U_N .
- 17 Utiliser l'expression de U_N de la question 14.c et les résultats de la question 16.

QUESTIONS PRÉLIMINAIRES

1.a Soient $(P, Q) \in A_N^2$ et $\lambda \in [0; 1]$. On pose $R = \lambda P + (1 - \lambda)Q$. On a déjà $R \in \mathbb{R}_N[X]$. De plus, $R(1) = \lambda P(1) + (1 - \lambda)Q(1) = \lambda + 1 - \lambda = 1$. De même, on a également $R(-1) = 1$.

Enfin, pour tout $x \in [-1; 1]$, comme $P(x) \geq 0$, $Q(x) \geq 0$ et $\lambda \in [0; 1]$, il vient $R(x) \geq 0$, d'où $R \in A_N$. On en déduit que

$$\boxed{A_N \text{ est une partie convexe de } \mathbb{R}_N[X].}$$

1.b Soit $P \in \mathbb{R}_N[X]$. La fonction $x \mapsto |P(x)|$ est continue sur le segment $[-1; 1]$. Par conséquent, l'intégrale

$$\int_{-1}^1 |P(x)| \, dx$$

est bien définie. De plus, on vérifie les propriétés suivantes :

- Soient $\lambda \in \mathbb{R}$ et $P \in \mathbb{R}_N[X]$. Par linéarité de l'intégrale,

$$\|\lambda P\|_1 = \int_{-1}^1 |\lambda| |P(x)| \, dx = |\lambda| \int_{-1}^1 |P(x)| \, dx = |\lambda| \|P\|_1$$

- Soit $P \in \mathbb{R}_N[X]$. Si $\|P\|_1 = 0$, alors

$$\int_{-1}^1 |P(x)| \, dx = 0$$

La fonction $x \mapsto |P(x)|$ est ainsi continue, positive et d'intégrale nulle sur le segment $[-1; 1]$. On en déduit qu'elle est nulle sur $[-1; 1]$. Il s'ensuit que le polynôme P , nul sur $[-1; 1]$, a une infinité de racines, d'où $P = 0_{\mathbb{R}_N[X]}$.

- Soit $(P, Q) \in \mathbb{R}_N[X]^2$. La valeur absolue $|\cdot|$ étant une norme sur $\mathbb{R}$, on a

$$\forall x \in [-1; 1] \quad |P(x) + Q(x)| \leq |P(x)| + |Q(x)|$$

Par croissance et linéarité de l'intégrale, on en déduit

$$\int_{-1}^1 |(P + Q)(x)| \, dx \leq \int_{-1}^1 |P(x)| \, dx + \int_{-1}^1 |Q(x)| \, dx$$

Par conséquent, $\|P + Q\|_1 \leq \|P\|_1 + \|Q\|_1$

On peut conclure que $\boxed{\|\cdot\| \text{ définit une norme sur } \mathbb{R}_N[X].}$

1.c Utilisons la caractérisation séquentielle pour montrer que A_N est un fermé de $(\mathbb{R}_N[X], \|\cdot\|)$. Soient $(P_r)_{r \in \mathbb{N}}$ une suite de polynômes de A_N et $P \in \mathbb{R}_N[X]$ tels que

$$\|P_r - P\|_1 \xrightarrow{r \rightarrow +\infty} 0$$

L'application

$$\|\cdot\|_{\infty, [-1; 1]} : Q \mapsto \sup_{[-1; 1]} |Q(x)|$$

définit une norme sur $\mathbb{R}_N[X]$. En effet, pour $Q \in \mathbb{R}_N[X]$, la fonction $x \mapsto |Q(x)|$ est continue sur le segment $[-1; 1]$. Elle est donc bornée et atteint ses bornes. Par suite, la quantité $\|Q\|_{\infty, [-1; 1]}$ est bien définie et en outre il existe $x_0 \in [-1; 1]$ tel que $|Q|(x_0) = \|Q\|_{\infty, [-1; 1]}$.

- Si $\|Q\|_{\infty,[-1;1]} = 0$, alors pour tout $x \in [-1;1]$, $Q(x) = 0$. Le polynôme Q ayant une infinité de racines sur $[-1;1]$, c'est le polynôme nul.
- Soit $\lambda \in \mathbb{R}$. Pour tout $x \in [-1;1]$, $|\lambda Q|(x) \leq |\lambda| \|Q\|_{\infty,[-1;1]}$. Dès lors,

$$\|\lambda Q\|_{\infty,[-1;1]} \leq |\lambda| \|Q\|_{\infty,[-1;1]}$$

$$\text{En outre, } |\lambda| \|Q\|_{\infty,[-1;1]} = |\lambda| |Q(x_0)| \leq \|\lambda Q\|_{\infty,[-1;1]}$$

$$\text{Finalement, on a } \|\lambda Q\|_{\infty,[-1;1]} = |\lambda| \|Q\|_{\infty,[-1;1]}$$

- Soit $R \in \mathbb{R}_N[X]$. Pour tout $x \in [-1;1]$,

$$|Q(x) + R(x)| \leq |Q(x)| + |R(x)| \leq \|Q\|_{\infty,[-1;1]} + \|R\|_{\infty,[-1;1]}$$

$$\text{On a donc } \|Q + R\|_{\infty,[-1;1]} \leq \|Q\|_{\infty,[-1;1]} + \|R\|_{\infty,[-1;1]}.$$

L'espace $\mathbb{R}_N[X]$ étant de dimension finie, les normes $\|\cdot\|_1$ et $\|\cdot\|_{\infty,[-1;1]}$ sont équivalentes. De plus,

$$\forall r \in \mathbb{N} \quad 0 \leq |P_r(1) - P(1)| \leq \|P_r - P\|_{\infty,[-1;1]}$$

Par théorème d'encadrement, on en déduit que $\lim_{r \rightarrow +\infty} P_r(1) = P(1)$, d'où $P(1) = 1$ car pour tout $r \in \mathbb{N}$, $P_r(1) = 1$.

On déduit de la même manière que $P(-1) = 1$ et que pour tout $x \in [-1;1]$,

$$\lim_{r \rightarrow +\infty} P_r(x) = P(x)$$

Or, pour tout $(x, r) \in [-1;1] \times \mathbb{N}$, $P_r(x) \geq 0$, ainsi $P(x) \geq 0$. On peut donc conclure que $P \in A_N$. Dès lors,

L'ensemble A_N est fermé dans l'espace vectoriel normé $(\mathbb{R}_N[X], \|\cdot\|_1)$.

2.a La partie A_N est non vide puisque le polynôme constant égal à 1 en est un élément. En outre, pour tout $P \in A_N$, on a :

$$\forall x \in [-1;1] \quad P(x) \geq 0$$

d'où $L(P) \geq 0$. On en déduit que l'ensemble $\{L(P) \mid P \in A_N\}$ est un ensemble non vide minoré par 0. Par théorème de la borne inférieure, l'ensemble admet une borne inférieure a_N .

On considère ensuite une suite $(P_n)_{n \in \mathbb{N}}$ de polynômes de A_N telle que $(L(P_n))_{n \in \mathbb{N}}$ converge vers a_N .

Pour rappel, on construit cette suite par exemple de la manière suivante à l'aide de la définition de la borne inférieure. Soit $n \in \mathbb{N}$:

$$\frac{1}{n+1} > 0 \quad \text{et} \quad a_N + \frac{1}{n+1} > a_N$$

Par définition de la borne inférieure, il existe un polynôme $P_n \in A_N$ tel que $L(P_n) \leq a_N + 1/(n+1)$. La suite $(P_n)_{n \in \mathbb{N}}$ étant ainsi construite, on remarque que pour tout $n \in \mathbb{N}$,

$$0 \leq L(P_n) - a_N \leq \frac{1}{n+1}$$

Par théorème d'encadrement, on en déduit que la suite $(L(P_n))_{n \in \mathbb{N}}$ converge vers a_N .

X/ENS Informatique B MP-PC-PSI 2018 — Corrigé

Ce corrigé est proposé par Emma Kerinec (ENS Lyon) ; il a été relu par Cyril Ravat (professeur en CPGE) et Vincent Puyhaubert (professeur en CPGE).

Ce sujet regroupe une collection de requêtes SQL élémentaires et leur traduction en algèbre relationnelle, qu'il faut implémenter en Python. Aucune connaissance théorique spécifique n'est requise ; seule la maîtrise primaire de la syntaxe SQL est nécessaire. Le sujet est idéal pour s'entraîner sur des requêtes SQL simples, ainsi que sur les principes basiques des algorithmes et leur implémentation en Python. La première partie doit être abordée avant les deux suivantes, lesquelles peuvent en revanche être traitées indépendamment l'une de l'autre.

- La première partie introduit l'utilisation des listes et des tables afin de définir des fonctions servant de briques de base pour les parties suivantes, notamment la jointure de tables et la sélection d'éléments selon différents critères.
- La deuxième partie met en évidence le lien entre ces fonctions et les principales requêtes SQL ; il faut alors se servir de la partie précédente pour implémenter quelques requêtes SQL simples.
- La dernière partie propose d'améliorer les fonctions définies dans la première partie en exploitant des propriétés de tri concernant les tables ou grâce à l'introduction de dictionnaires.

Si ce sujet est d'une difficulté moyenne, il a sans doute déconcerté les candidats ayant fait l'impasse sur le langage SQL. Il introduit efficacement les concepts utilisés, notamment la structure hors-programme de dictionnaire, ainsi que tous les outils associés auxquels doivent se restreindre les candidats. Malgré l'évocation de tris dans la dernière partie, ce sujet est abordable en fin de première année.

INDICATIONS

Partie I

- I.5 Observer que les enregistrements d'une table sont indépendants.
- I.6 Parcourir les deux tables et utiliser la concaténation.
- I.7 Parcourir les deux tables et comparer les bons attributs, ne pas oublier de supprimer l'attribut associé au second indice.
- I.9 Comparer chaque enregistrement de la table à tous ceux déjà retenus dans le résultat.

Partie II

- II.1 Utiliser la fonction **SelectionConstante** définie dans la partie précédente.
- II.2 L'opération demandée est le produit cartésien de **Trajet** et **Véhicule**.
- II.3 On peut enchaîner les fonctions élémentaires.
- II.4 Attention aux indices des colonnes après jointure.
- II.5 Il vaut mieux faire une double jointure suivie d'une unique sélection.
- II.6 Penser à réutiliser le résultat précédent, sous forme de jointure.

Partie III

- III.1 Parcourir la table en comparant les enregistrements consécutifs.
- III.2 Quel est l'algorithme classique de recherche sur les listes triées ?
- III.3 Parcourir en parallèle les deux tables et les synchroniser.

I.1 On parcourt tous les enregistrements de `table`, et on ajoute dans le résultat ceux dont la valeur pour l'indice `indice` vaut `constante`.

```
def SelectionConstante(table, indice, constante):
    resultat = []
    for enr in table:
        if enr[indice] == constante:
            resultat.append(enr)
    return resultat
```

I.2 Dans la fonction précédente, il n'y a pas d'opération coûteuse en dehors de la boucle `for`. Celle-ci s'exécute `len(table)` fois et on effectue deux opérations élémentaires au plus à chaque étape. Ainsi,

La complexité de la fonction `SelectionConstante` est en $O(\text{len}(\text{table}))$.

I.3 Parcourons tous les enregistrements de la table `table` et ajoutons dans le résultat ceux dont la valeur pour l'indice `indice1` est égale à celle pour l'indice `indice2`.

```
def SelectionEgalite(table, indice1, indice2):
    resultat = []
    for enr in table:
        if enr[indice1] == enr[indice2]:
            resultat.append(enr)
    return resultat
```

I.4 Il suffit de parcourir la liste `listeIndices` en créant un nouvel enregistrement auquel on ajoute les valeurs de `enregistrement` pour les indices souhaités, donnés dans `listeIndices`.

```
def ProjectionEnregistrement(enregistrement, listeIndices):
    resultat = []
    for indice in listeIndices:
        resultat.append(enregistrement[indice])
    return resultat
```

I.5 Il suffit d'appliquer la fonction précédente à tous les éléments de `table`, car ceux-ci sont indépendants, et de rassembler les nouveaux enregistrements dans une nouvelle table que l'on renvoie en fin de programme.

```
def ProjectionTable(table, listeIndices):
    resultat = []
    for enr in table:
        proj = ProjectionEnregistrement(enr, listeIndices)
        resultat.append(proj)
    return resultat
```

I.6 L'utilisation de deux boucles `for` permet de créer toutes les concaténations d'un élément de `table1` et d'un élément de `table2`, qui sont successivement ajoutés à une nouvelle table.

```
def ProduitCartesien(table1, table2):
    resultat = []
    for enr1 in table1:
        for enr2 in table2:
            resultat.append(enr1 + enr2)
    return resultat
```

I.7 Comme dans la question précédente, on parcourt les couples d'enregistrements à l'aide de deux boucles. Si les attributs de ces deux enregistrements coïncident, on concatène au premier enregistrement une copie du second dans laquelle seul l'attribut d'indice i_2 n'a pas été recopié. Le résultat de cette concaténation est ensuite ajouté dans la nouvelle table.

```
def JointEnregistrements(enr1, enr2, indice2):
    return (enr1 + enr2[0:indice2] + enr2[indice2 + 1:])
```

| L'utilisation d'une fonction annexe n'est pas obligatoire.

```
def Jointure(table1, table2, indice1, indice2):
    resultat = []
    for enr1 in table1:
        for enr2 in table2:
            if enr1[indice1] == enr2[indice2]:
                jointure = JointEnregistrements(enr1, enr2, indice2)
                resultat.append(jointure)
    return resultat
```

| On peut aussi utiliser une boucle for pour la jointure :

```
def JointEnregistrements(enr1, enr2, indice2):
    resultat = enr1[:]
    for i in range(len(enr2)):
        if i != indice2:
            resultat.append(enr2[i])
    return resultat
```

| L'astuce `enr1[:]` sert à créer une copie de `enr1` ; il ne faudrait pas que les modifications apportées à `resultat` modifient également l'argument `enr1`.

I.8 La fonction `Jointure` fait intervenir deux boucles. La première boucle `for` s'exécute `len(table1)` fois. Pour chaque étape, une nouvelle boucle `for` s'exécute `len(table2)` fois, en effectuant une lecture de liste et une copie d'un enregistrement de `table2`. Finalement,

La complexité de la fonction `Jointure` est en $O(\text{len}(\text{table1}) \cdot \text{len}(\text{table2}) \cdot \text{len}(\text{table2}[0]))$.

I.9 La nouvelle table sans doublons se construit au fur et à mesure. Initialisée au premier enregistrement `table[0]`, on ajoute ensuite successivement les autres enregistrements, en vérifiant toutefois à chaque fois qu'il n'y a pas déjà un élément identique dans la nouvelle table. Cette vérification se fait de manière naïve, en comparant la nouvelle valeur à insérer à toutes les autres déjà sélectionnées.