PO HS 1 : Préparation aux oraux avec Python

Correction

Oral Centrale-Supelec

L'épreuve de mathématique 2 est un oral de 30 minutes (25 en PSI) qui succède à une préparation d'environ 30 minutes également. Le sujet est constitué d'un seul exercice comportant plusieurs questions de difficultés progressives et faisant appel, pour certaines, à l'usage de l'outil informatique (un ordinateur est mis à disposition). Lors de la préparation, le candidat dispose d'un ordinateur sur lequel sont installés les logiciels Pyzo, Spyder et Scilab, ainsi que des documents d'aide fournis à tous les candidats présentant les fonctions des bibliothèques qui pourront être utiles sans pour autant être exigible.

Il est à noter qu'il s'agit avant tout d'une épreuve de mathématiques et non d'informatique. L'outil informatique n'est présent que pour conjecturer ou illustrer des résultats. La maitrise de cet outil est évidemment prise en compte dans l'évaluation globale des candidats mais dans une part moindre que celle des compétences mathématiques. Néanmoins, un candidat ne faisant pas le moindre effort pour traiter les questions de programmation sera fortement pénalisé.

Le rapport de jury : `https://www.concours-centrale-supelec.fr/CentraleSupelec` ; il y a aussi des exemples d'exercices d'oraux.

Extrait des rapports (MP,PC,PSI) 2022 (et avant) :

- Il est recommandé aux futurs candidats d'être plus vigilants aux messages d'erreur renvoyés par le logiciel lors de l'exécution d'un script : ils peuvent permettre de corriger de nombreuses fautes de syntaxe ou de mieux comprendre l'utilisation des fonctions proposés dans l'aide Python.
- Les candidats doivent s'efforcer d'écrire des programmes dans lesquels leurs notations sont aussi conformes que possible à celles du sujet. Court-circuiter les questions informatiques n'est pas une stratégie viable : celles-ci visent à établir des conjectures et sans celles-ci, un candidat se retrouve rapidement bloqué. L'autonomie est en hausse et de plus en plus de candidats parviennent à écrire un code fonctionnel durant la préparation et à l'exploiter. Pour gagner encore en aisance, on recense les points suivants : savoir commenter rapidement une partie du code, faire un usage adapté du `print` (pertinent dans l'éditeur mais pas dans la console, exception faite des polynômes), ne pas se bloquer devant un `[warning]` produit par python, oser poser `X = Polynomial([0, 1])` pour manipuler les polynômes selon l'usage courant, savoir simuler des lois classiques comme une loi uniforme et en particulier une loi uniforme sur $\{-1, 1\}$.
- Quelques candidats ne sont pas au courant des différentes fonctions Python mises à leur disposition dans les documents d'aide. On voit par exemple des candidats reprogrammer la méthode des rectangles pour le calcul approché d'intégrales ou la recherche de solutions d'équations par dichotomie.
- Si un sujet contient des fonctions ou des exemples à tester en Python, les candidats doivent les saisir durant la préparation pour que le temps de l'exposé soit majoritairement consacré aux mathématiques.
- Certains candidats ne testent pas leurs fonctions au fur et à mesure de la préparation. Il en résulte souvent une phase de débogage en début de l'oral qui pourrait être facilement évitée et qui fait perdre du temps pour les questions réellement intéressantes.
- Pendant la préparation, il est fortement conseillé de trouver un équilibre entre le temps passé sur machine et le temps passé sur des questions théoriques, quitte à sauter certaines questions. La préparation durant 30 minutes, 15 minutes sur machine est suffisant, dès lors que les premiers codes sont testés et produisent des résultats exploitables. Les calculs étant particulièrement chronophages et difficiles à mener en direct au tableau, il est vivement conseillé de s'y frotter en préparation.
- D'une manière générale, les candidats doivent avoir une idée de la complexité de leurs calculs et ne pas attendre de longues minutes qu'une boucle interminable donne un résultat hypothétique. Typiquement, rappelons qu'une implémentation récursive naïve d'une relation de récurrence double aboutit à une complexité exponentielle.
- Quand on demande une valeur numérique avec une certaine précision, il faut être capable de justifier que le résultat proposé respecte cette précision. C'est notamment le cas si on emploie une méthode de dichotomie ou si on essaie de donner une estimation de la somme d'une série numérique (ce qui implique alors de majorer un reste).
- Des fonctions très classiques comme par exemple le calcul des coefficients binomiaux doivent être programmées efficacement par les candidats, sans repasser par l'utilisation d'une fonction factorielle.

De plus dans le rapport de MP il y a : Quelques algorithmes qui reviennent souvent mériteraient d'être plus spécifiquement préparés :

- Savoir calculer le *pgcd* de deux nombres entiers a et b, autrement qu'en testant tous les nombres de 1 à $\max(a, b)$.
- Savoir programmer un test de primalité élémentaire.
- Savoir estimer une probabilité ou une espérance.

— Savoir quoi tracer pour évaluer un paramètre (par exemple, pour conjecturer la valeur de α quand $u_n \sim \frac{C}{n^\alpha}$, tracer la suite (u_n) et des courbes du type $n \mapsto \frac{1}{n^\alpha}$ pour un grand nombre de valeurs de α n'est pas très pertinent).

Centrale 2

Exercice 1 *(CENTRALE 2).*

On note χ_n le polynôme caractéristique de $A = \begin{pmatrix} 0 & 0 & \dots & 0 & 1/n \\ 1 & \ddots & \ddots & \vdots & \vdots \\ 0 & \ddots & \ddots & 0 & \vdots \\ \vdots & \ddots & \ddots & 0 & \vdots \\ 0 & \dots & 0 & 1 & 1/n \end{pmatrix} \in \mathcal{M}_n(\mathbb{R})$.

1° Écrire une fonction Python prenant pour paramètre `n` et renvoyant les coefficients de χ_n sous forme d'un tableau numpy.

2° Afficher les coefficients de χ_n pour $n \in [\![2,8]\!]$ et conjecturer la valeur de χ_n. Démontrer ce résultat.

3° Afficher les modules des racines de χ_n pour $n \in [\![2,8]\!]$. Que peut-on conjecturer ? Démontrer ce résultat.

Correction :

1°
```
np.set_printoptions(precision=3) #Limiter le nombre de décimales
def khi(n):
    A=np.zeros((n,n))
    for j in range(n-1):
        A[j+1,j]=1
    for i in range(n):
        A[i,n-1]=1/n
    return np.poly(A) # coeffs du polynome caractéristique par degré décroissant cf Fiche Matrice de Ce
```

2°
```
for n in range(2,9):
    print(khi(n))
```

Cela permet de conjecturer que $\chi_n = X^n - \frac{1}{n}\sum_{k=0}^{n-1} X^k$. Mais bon vous avez sans doute tout de suite identifié A comme étant une matrice compagnon. Le résultat se démontre bien (sinon cf DS 7)

3° Ici on fera attention : `np.poly` donne les coeffs du polynômes caractéristique par ordre décroissant, tandis que le module `numpy.polynomial` a plutôt tendance a avoir les coeffs dans l'ordre croissant.

```
def modules(n):
    P=khi(n)
    P=Polynomial([P[n-k] for k in range(n+1)]) # Confère fiche Polynôme de Centrale
    return np.abs([r for r in P.roots()])
for n in range(2,9):
    print(modules(n))
```

Il semble qu'il y ait une vp de module 1 et que toutes les autres soient de module strictement inférieur.
On a 1 racine simple de χ_n (car $\chi_n(1) = 1^n - \frac{1}{n}\sum_{k=0}^{n-1} 1^k = 0$ et $\chi_n'(1) = n1^{n-1} - \frac{1}{n}\sum_{k=1}^{n-1} k1^{k-1} = n - \frac{n-1}{2} \neq 0$).
Posons $B = A^\top$ (ainsi B est stochastique, ie la somme des coeffs de chaque ligne vaut 1). Ainsi en notant N_∞ la norme infinie de $\mathcal{M}_{n,1}$ on a pour tout vecteur colonne X que $N_\infty(BX) \leq N_\infty(X)$ (peut se faire à la main, si $X = (x_1 \dots x_n)^\top$, alors $BX = (x_2 \dots x_{n-1}\alpha_n)^\top$ où $\alpha_n = \frac{1}{n}\sum_{k=1}^n x_k$ et $|\alpha_n| \leq N_\infty(X)$), ainsi si X est vp de vp λ on en déduit que $|\lambda| \leq 1$.
Reste a montrer que 1 est la seule vp de module 1, on suppose qu'il existe une valeur propre $\lambda \neq 1$ de module 1. Comme λ racine de χ_n, on a $\lambda^n = \frac{1}{n}\sum_{k=0}^n \lambda^k$, ie $n\lambda^n = \frac{1-\lambda^n}{1-\lambda}$, ainsi $\lambda^n(n(1-\lambda)+1) = 1$, en passant en module on en déduit que $|n+1-n\lambda| = 1$, ainsi $n\lambda$ est en même temps sur le cercle de centre $((n+1),0)$ et de rayon 1 et sur le cercle de centre 0 et de rayon n (car $|n\lambda| = n$), ainsi (dessinez ces deux cercles !) $n\lambda = n$ et donc $\lambda = 1$, ce qui contredit $\lambda = 1$.

Exercice 2 *(CENTRALE PC 2 2016).* *RMS 1085*

Soit $(P_{i,j})_{(i,j)\in(\mathbb{N}^\star)^2}$ une famille de polynômes telle que $P_{1,j} = P_{i,1} = 1$ pour tout $i,j \in \mathbb{N}^\star$ et $P_{i,j} = P_{i-1,j} + P_{i,j-1} + XP_{i-1,j-1}$ pour $i,j \geq 2$.

1° Écrire une fonction `poly(i,j)` retournant $P_{i,j}$.

2° Calculer $P_{7,k}$ et $P_{k,7}$ pour $k \in [\![1,4]\!]$. Observer et faire une conjecture sur le degré de $P_{i,j}$.

3° Démontrer la conjecture.

Correction :

1°
```
from numpy.polynomial import Polynomial

def poly(i,j):
    if i==1 or j==1:
        return Polynomial([1])
    else:
        return poly(i-1,j)+poly(i,j-1)+Polynomial([0,1])*poly(i-1,j-1)
```

2°
```
for k in range(1,5):
    print(k)
    print(poly(k,7))
    print(poly(7,k))
```

Il semblerait que $P_{i,j}$ soit de dégré $\min(i,j)-1$, on peut aussi remarquer (ce qui semble assez claire avec la def) que $P_{i,j} = P_{j,i}$ et qu'il soit à coefficient positif.

3° Ici il faut bien poser l'hypothèse de récurrence pour éviter trop de lourdeur, il est de plus intéressant de mettre la positivité des coefficients. Pour tout $n \geq 1$ on note $\mathcal{P}(n)$: « pour tout $(i,j) \in (\mathbb{N}^\star)^2$ tel que $i+j \leq n$, on a $P_{i,j}$ à coefficients positifs et $\deg(P_{i,j}) = \min(i,j)-1$ » .

Initialisation : pour $n=2$, l'unique $(i,j) \in (\mathbb{N}^\star)^2$ tel que $i+j \leq n$ est le couple $(1,1)$ et on a $P_{1,1}=1$ qui est bien à coefficient positif et de degré $0 = \min(1,1)-1$.

Hérédité : Supposons $\mathcal{P}(n)$ pour $n \geq 2$. Soit $(i,j) \in (\mathbb{N}^\star)^2$ tel que $i+j \leq n+1$.

Si $i=1$ ou $j=1$, on a $P_{i,j}=1$ qui est bien à coefficient positif et de degré $0 = \min(i,j)-1$.

Sinon, $P_{i,j} = P_{i-1,j} + P_{i,j-1} + XP_{i-1,j-1}$, comme $i-1+j \leq n$, $i+j-1 \leq n$ et $i-1+j-1 \leq n$, on peut donc appliquer l'hypothèse de récurrence aux trois polynômes du membre de droite. Ils sont tous les trois à coefficients positifs, ainsi $P_{i,j}$ l'est aussi, mais cela implique que (Attention le degré d'une somme est plus petit que la somme des degrés, ici comme les coefficients dominants sont positifs ils ne peuvent pas se simplifier) :) $\deg(P_{i,j}) = \max(\deg(P_{i-1,j}), \deg(P_{i,j-1}), \deg(XP_{i-1,j-1})$. Or $\deg(P_{i-1,j}) = \min(i-1,j)-1 \leq \min(i,j)-1$, de même $\deg(P_{i,j-1}) \leq \min(i,j)-1)$ et $\deg(XP_{i-1,j-1}) = 1+\deg(P_{i-1,j-1}) = 1+\min(i-1,j-1)-1 = \min(i,j)-1$. Ainsi $\deg(P_{i,j}) = \min(i,j)-1$, ce qui termine l'hérédité et la récurrence

Exercice 3 *(CENTRALE PC 2 2017).* *BEOS planche 3111*

On considère l'équation différentielle $(1+t^2)y''(t) + ty'(t) - y(t) = 0$.

1° Tracer les solutions f et g soumises aux conditions initiales $(f(0), f'(0)) = (0,1)$ et $(g(0), g'(0)) = (1,0)$. L'une d'elles vous semble-t-elle évidente ?

2° Chercher l'autre solution sous la forme d'une somme de série entière.

3° Pour tout t dans $]-1,1[$, prouver l'égalité $g(t) = \sqrt{1+t^2}$.

Correction :

1. L'équation différentielle s'écrit : $y''(t) = \dfrac{1}{1+t^2}y(t) - \dfrac{t}{1+t^2}y'(t)$.

On pose $X(t) = \begin{pmatrix} y(t) \\ y'(t) \end{pmatrix}$, ainsi, $X'(t) = \begin{pmatrix} y'(t) \\ y''(t) \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ \dfrac{1}{1+t^2} & -\dfrac{t}{1+t^2} \end{pmatrix} \begin{pmatrix} y(t) \\ y'(t) \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ \dfrac{1}{1+t^2} & -\dfrac{t}{1+t^2} \end{pmatrix} X(t)$.

On se ramène donc à trouver les solutions approchées d'un système différentiel. Le code python s'écrit pour $y(0)=0$, $y'(0)=1$:

```
def f(x,t):
    return np.array([x[1],x[0]/(1+t**2)-t*x[1]/(1+t**2)])
T = np.arange(0,5,0.1)
X = integr.odeint(f,np.array([0,1]),T)
plt.plot(T, X[:,0])
plt.show()
```

La solution de l'équation différentielle vérifiant $f(0)=0$ et $f'(0)=1$ semble être $f : t \mapsto t$. Un calcul évident permet de le vérifier.

2. On cherche l'autre solution sous la forme $g(t)\sum_{n=0}^{+\infty} a_n t^n$ avec un rayon de convergence R non nul.

On obtient $g(0)=1=a_0$ et $g'(0)=0=a_1$.
g est de classe $\mathcal{C}^\infty$ sur $]-R,R[$ en tant que somme d'une série entière sur son intervalle ouvert de convergence, et $g'(t)=\sum_{n=1}^{+\infty} n a_n t^{n-1}$ et, $g''(t)=\sum_{n=2}^{+\infty} n(n-1)a_n t^{n-2}$.

On injecte dans l'équation différentielle et on obtient : $(1+t^2)y''(t)+ty'(t)-y(t)=0 \iff \sum_{n=2}^{+\infty} n(n-1)a_n t^{n-2}+\sum_{n=2}^{+\infty} n(n-1)a_n t^n+\sum_{n=1}^{+\infty} n a_n t^n-\sum_{n=0}^{+\infty} a_n t^n=0 \iff \sum_{n=2}^{+\infty} n(n-1)a_n t^{n-2}+\sum_{n=0}^{+\infty} n(n-1)a_n t^n+\sum_{n=0}^{+\infty} n a_n t^n-\sum_{n=0}^{+\infty} a_n t^n=0 \underset{\text{reindexation}}{\iff} \sum_{n=0}^{+\infty}(n+2)(n+1)a_{n+2}t^n+\sum_{n=0}^{+\infty} n(n-1)a_n t^n+\sum_{n=0}^{+\infty} n a_n t^n-\sum_{n=0}^{+\infty} a_n t^n=0.$
Donc, par unicité du développement en série entière, on obtient : $\forall n \in \mathcal{N}$, $(n+2)(n+1)a_{n+2}+(n^2-1)a_n=0 \iff a_{n+2}=-\dfrac{n-1}{n+2}a_n$.
Étant donné que $a_1=0$, on obtient pour tout $n\in\mathbb{N}$, $a_{2n+1}=0$.
Et, $a_{2n}=-\dfrac{2n-3}{2n}a_{2n-2}=+\dfrac{(2n-3)(2n-5)}{2n(2n-2)}a_{2n-4}=\ldots=(-1)^n\dfrac{(2n-3)(2n-5)...3.1}{2n(2n-2)...4.2}a_0=(-1)^n\dfrac{(2n-3)(2n-5)...3.1}{2n(2n-2)...4.2}$ puisque $a_0=1$.
Ainsi, $g(t)=1+\sum_{n=1}^{+\infty}(-1)^n\dfrac{(2n-3)(2n-5)...3.1}{2n(2n-2)...4.2}t^{2n}$.
On remarque que $\dfrac{a_{2n}}{a_{2n-2}} \underset{n\to+\infty}{\to} 1$, le rayon de convergence de cette série entière est donc $R=\dfrac{1}{\sqrt{1}}=1$.

3. Ainsi, pour tout $t\in]-1,1[$, $g(t)=1+\sum_{n=1}^{+\infty}(-1)^n\dfrac{(2n-3)(2n-5)...3.1}{2n(2n-2)...4.2}t^{2n}=1+\sum_{n=1}^{+\infty}(-1)^n\dfrac{(2n-3)(2n-5)...3.1}{2^n n(n-1)...2.1}t^{2n}=1+\sum_{n=1}^{+\infty}(-1)^n\dfrac{(2n-3)(2n-5)...3.1}{2^n n!}t^{2n}=(1+t^2)^{1/2}=\sqrt{1+t^2}$

Exercice 4 *(CENTRALE PSI 2 2016).* *odlt23 planche 170*

Une particule se déplace sur une surface comportant 4 positions possibles : A_0 et A_3 qui sont des puits, A_1 et A_2 qui sont des positions intermédiaires. À chaque étape :
- si la particule est dans un puits, elle y reste avec une probabilité de 1
- si elle est en A_1 , elle va en A_0 avec une probabilité p, en A_2 avec une probabilité $1-p$;
- si elle est en A_2 , elle va en A_1 avec une probabilité p, en A_3 avec une probabilité $1-p$.

On note X_n la variable aléatoire donnant la position de la particule à l'étape n : $X_n(\Omega)=[\![0,3]\!]$.

1° Écrire une fonction Python `suivant(x,p)` qui simule un saut sachant qu'on est à la position `x` puis une fonction `dplct(n,x0,p)` qui simule `n` sauts sachant qu'on part de la position `x0`.

2° Tracer l'histogramme des positions obtenues après n sauts durant N essai.

3° Soit $U_n=\begin{pmatrix}\mathbb{P}(X_n=0)\\ \vdots \\ \mathbb{P}(X_n=3)\end{pmatrix}$. Déterminer $A\in\mathcal{M}_4(\mathbb{R})$ telle que $U_{n+1}=AU_n$.

4° Montrer que A est diagonalisable si et seulement si $p\in]0,1[$.

5° Pour $p=\frac{1}{2}$, diagonaliser A avec Python et calculer $\lim\limits_{n\to+\infty} U_n$. Comparer avec les résultats précédents.

Correction :

1°
```
def suivant(x,p):
    if x in [0,3]:
        return x
```

```
    if rd.random()<p:
       return x-1
    else:
       return x+1
def dplct(n,x0,p):
    x=x0
    for k in range(n):
       x=suivant(x,p)
    return x
```

2° L'énoncé n'est pas très précis (partant de la même position initiale ? en choisissant la position initiale de manière uniforme dans $[\![0,3]\!]$? ...),

```
def essais(N,n,p):
    X0=rd.randint(0,4,N) # Choix de N positions initiales
    X=[0,1,2,3]
    Y=[0,0,0,0] # La où on stockera les résultats
    for k in range(N):
       Y[dplct(n,X0[k],p)]+=1
    plt.bar(X,Y)
    plt.xticks(X)
    plt.show()
```

3° On trouve (ne pas oublier de bien justifier le premier : SCE et FPT) : $A = \begin{pmatrix} 1 & p & 0 & 0 \\ 0 & 0 & p & 0 \\ 0 & 1-p & 0 & 0 \\ 0 & 0 & 1-p & 1 \end{pmatrix}$.

4° On a $\chi_A(X) = (X-1)^2(X^2 - p(1-p))$, de plus E_1 est de dimension 2 (le premier et dernier vecteur de la base canonique sont dans E_1).
Si $p \in]01[$ les deux autres valeurs propres sont $\pm\sqrt{p(1-p)}$, qui sont donc simples, et comme $2+1+1=4$ la matrice est diagonalisable.
Si $p = 0$ ou 1, $\chi_A(X) = (X-1)^2X^2$, or A est de rang 3 dans ce cas, ie (théorème du rang) E_0 de dimension 1, ainsi A n'est pas diagonalisable.
On a bien montré que A est diagonalisable ssi $p \in]0,1[$.

5° Pour $p = \frac{1}{2}$ en utilisant `alg.matrix_power` (et surtout pas `**`) on peut conjecturer que A^n converge vers $\begin{pmatrix} 1 & 2/3 & 1/3 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1/3 & 2/3 & 1 \end{pmatrix}$, il ne reste plus qu'a multiplier par U_0 pour avoir U_n. Ainsi si le choix du point de départ uniforme on trouve qu'on tend vers $\begin{pmatrix} 1/2 \\ 0 \\ 0 \\ 1/2 \end{pmatrix}$, chose que l'on peut vérifier avec la simulation.

Remarque : Pour le calcul à la main, de A diagonalisable on écrit $A = PDP^{-1}$, puis on montre (les vp sont 1, 1/4 et 1/4) que (D^n) converge ainsi (A^n) converge et donc (U_n) converge vers un certain U.
Ici on ne peut pas utiliser l'astuce qui consiste à remarquer que U est vecteur propre de valeur propre 1 (et que la somme de ses coeffs vaut 1), en effet E_1 est de dimension 2...

Exercice 5 *(CENTRALE PSI 2 2016).* *odlt24 planche 166*

1° Justifier que $\langle\,|\,\rangle : (P,Q) \mapsto \langle P|Q\rangle : \displaystyle\int_{-1}^{1} P(t)Q(t)\mathsf{d}t$ est un produit scalaire sur $\mathbb{R}[X]$, et que $N_\infty : P \mapsto \max_{t\in[-1,1]} |P(t)|$ est une norme.

2° On note $F = \mathbb{R}_5[X]$, déterminer une base orthonormée $(E_0,\dots,E_5)$ de F en appliquant le procédé d'orthonormalisation de Gram-Schmidt à la base canonique.

3° Tracer (sur $[-1,1]$) les courbes représentatives de $E_0,\dots,E_5$.

4° Trouver $N_\infty(E_i)$, $i \in [\![0,5]\!]$, ainsi que les valeurs de t par laquelle elle est atteinte.
Conjecturer la valeur de $(N_\infty(E_i))^2$, $i \in [\![0,5]\!]$. On admet pour la suite ces résultats mathématiquement établis.

5° Montrer que si $P \in F$ est tel que $\|P\| = 1$, alors $N_\infty(P) \leq 3\sqrt{2}$. Quand a-t-on égalité ?

6° Trouver a et b optimaux tels que pour tout $P \in F$, $a\|P\| \leq N_\infty(P) \leq b\|P\|$. Donner des exemples pour lesquels il y a égalité, à gauche ou à droite.

Correction :

1° $\langle\cdot|\cdot\rangle$ est clairement bilinéaire, symétrique et positif, montrons qu'il est défini : soit $P \in \mathbb{R}_[X]$ tel que $\langle P|P\rangle = 0$, on a donc l'intégrale d'une fonction continue et positive qui est nulle, ainsi P^2 est nul sur $[-1,1]$, ainsi P a une infinité de racines, c'est donc le polynôme nul ; ce qui termine de montrer que $\langle\cdot|\cdot\rangle$ est un produit scalaire sur $\mathbb{R}[X]$.
N_∞ est clairement positif et on a l'homogénéité, pour la séparation : soit $P \in \mathbb{R}[X]$ tel que $N_\infty(P) = 0$, ainsi P est nul sur $[-1,1]$ donc sur $\mathbb{R}$ puisque c'est un polynôme, pour l'inégalité triangulaire : Soit $(P,Q) \in \mathbb{R}[X]^2$ pour tout $t \in [-1,1]$ on a $|P(t)| \leq N_\infty$ par définition de la borne sup, ainsi en utilisant l'inégalité triangulaire pour la valeur absolue dans $\mathbb{R}$: $|P(t)+Q(t)| \leq |P(t)|+|Q(t)| \leq N_\infty(P)+N_\infty(Q)$ il ne reste plus qu'à prendre le sup sur $t \in [-1,1]$ pour conclure que $N_\infty(P+Q) \leq N_\infty(P)+N_\infty(Q)$. On a donc bien montré que N_∞ est une norme sur $\mathbb{R}[X]$.

2° On rappel la formule du procédé de Gram-Schmidt : $E_i = \dfrac{\tilde{E}_i}{\|\tilde{E}_i\|}$ où $\tilde{E}_i = X^i - \displaystyle\sum_{j=0}^{i-1} \langle X^i|E_i\rangle E_i$

```
def pscal(P,Q):
   phi = lambda t : P(t)*Q(t)
   return integr.quad(phi,-1,1)[0]
def GramSchmidt(n):
   B=[Polynomial([0]*i+[1]) for i in range(n+1)] # Base canonique
   C=[]
   for i in range(n+1):
      P=B[i]
      for j in range(i):
         P=P-pscal(P,C[j])*C[j]
      P=P/np.sqrt(pscal(P,P))
      C.append(P)
   return C
C=GramSchmidt(5)
print([list(P) for P in C])
```

On remarque qu'on peut conjecturer que E_i est de même parité que i.
Pour un affichage plus agréable :

```
def polytostr(P):
    cP=list(P)
    res="%.2f" % cP[0] # Pour n'avoir que deux décimales
    for k in range(1,len(cP)):
        if cP[k]>=0:
            res+="+"
        res+="%.2f" % cP[k] +"X^"+str(k)
    return res
for k in range(len(C)):
    print("E_"+str(k)+"="+polytostr(C[k]))
```

3°
```
def trace(n):
   X=np.linspace(-1,1,256)
   C=GramSchmidt(n)
   for i in range(n+1):
      P=C[i]
      Y=[P(x) for x in X]
      plt.plot(X,Y,label='P_'+str(i))
   plt.show()
trace(5)
```

4° Un programme pour vérifier, même s'il ne permet pas de conclure plus que la lecture graphique de la question précédente.

```
def ninf(P):
   X=np.linspace(-1,1,1024)
   M,t=0,-1
   for x in X:
      if abs(P(x))>M:
         M,t=abs(P(x)),x
   return M,t
```

```
#print([ninf(P) for P in C]) # Affichage plus rapide mais moins agréable
for k in range(len(C)):
    M,t=ninf(C[k])
    print("N_inf(E_"+str(k)+")=E_"+str(k)+"("+str(t)+")="+str(M))
    print("sqrt("+str(k)+"+1/2)="+str(np.sqrt(k+.5)))
```

On conjecture que la norme infinie de E_i n'est atteinte qu'en ± 1 (visible avec la question précédente) et qu'elle vaut $\sqrt{i+\frac{1}{2}}$. On admet ces résultats comme indiqué dans le sujet.

5° Soit $P \in F$ tel que $\|P\| = 1$, on décompose dans la BON qu'on vient de construire : $P = \sum_{i=0}^{5} \alpha_i E_i$ où $\alpha_i = \langle P|E_i\rangle$, de plus $\|P\| = 1$ implique $\sum_{i=0}^{5} \alpha_i^2 = 1$.

En appliquant l'inégalité triangulaire : $N_\infty P \leq \sum_{i=0}^{5} |\alpha_i| N_\infty(E_i)$, pour continuer il faut penser à l'inégalité de Cauchy-Schwarz (dans $\mathbb{R}^6$ muni du produit scalaire usuel) appliqué à $(|a_0|, \dots, |a_5|)$ et $(N_\infty(E_0), \dots, N_\infty(E_0))$. Ainsi on a $N_\infty \leq \sqrt{\sum_{i=0}^{5} |\alpha_i|^2}\sqrt{\sum_{i=0}^{5} N_\infty(E_i)^2} = 1\sqrt{\sum_{i=0}^{5} i+\frac{1}{2}} = \sqrt{18} = 3\sqrt{2}$.

Pour le cas d'égalité il faut égalité dans l'inégalité de Cauchy-Schwarz : $(|a_0|, \dots, |a_5|)$ et $(N_\infty(E_0), \dots, N_\infty(E_0))$ doivent être liés, comme le premier vecteur de $\mathbb{R}^6$ est unitaire on doit avoir $(|a_0|, \dots, |a_5|) = \frac{1}{3\sqrt{2}}(N_\infty(E_0), \dots, N_\infty(E_0)) = \frac{1}{3\sqrt{2}}(\sqrt{\frac{1}{2}}, \sqrt{\frac{3}{2}}, \sqrt{\frac{5}{2}}, \sqrt{\frac{7}{2}}, \sqrt{\frac{9}{2}}, \sqrt{\frac{11}{2}}) = \frac{1}{6}(1, \sqrt{3}, \sqrt{5}, \sqrt{7}, \sqrt{9}, \sqrt{11})$. Maintenant pour le cas d'égalité de $N_\infty(P) \leq \sum_{i=0}^{5} |\alpha_i| N_\infty(E_i)$, comme la norme infinie de E_i n'est atteinte qu'en 1 ou -1 pour être dans le cas d'égalité il faut être dans le cas d'égalité en 1 ou en -1, de plus comme on a utilisé l'inégalité triangulaire dans $\mathbb{R}$ on doit avoir les $\alpha_i E_i(1)$ (ou $\alpha_i E_i(-1)$) de même signe ce qui fait 4 cas : $(a_0, \dots, a_5) = \frac{\pm 1}{6}(1, \sqrt{3}, \sqrt{5}, \sqrt{7}, \sqrt{9}, \sqrt{11})$ et $(a_0, \dots, a_5) = \frac{\pm 1}{6}(1, -\sqrt{3}, \sqrt{5}, -\sqrt{7}, \sqrt{9}, -\sqrt{11})$

6° On cherche a et b optimaux tels que pour tout $P \in F$ tel que $\|P\| = 1$ on ait $a \leq N_\infty(P) \leq b$ (c'est équivalent à la question posée), on a déjà déterminé que $b = 3\sqrt{2}$ est optimal avec des cas d'égalité à la question précédente.

Or $\|P\|^2 = \int_{-1}^{1} P(t)Q(t)\mathrm{d}t \leq 2N_\infty^2(P)$, de plus on a égalité pour P constant, ainsi la valeur optimal de a est $\frac{1}{\sqrt{2}}$, et on a le cas d'égalité pour les polynômes constants (on a montré qu'un sens, pour l'autre : soit $P \in F$ tel que $\|P\|^2 = 2N_\infty(P)$, ainsi $\langle P|P\rangle = \int_{-1}^{1} N_\infty(P)\mathrm{d}t$ et donc $\int_{-1}^{1} N_\infty(P)^2 - P(t)^2\mathrm{d}t$, on a une fonction positive et continue d'intégrale nulle sur $[-1, 1]$ donc la fonction est nulle sur $[-1, 1]$ donc sur $\mathbb{R}$ puisque c'est un polynôme, ainsi P est constant).

Exercice 6 *(CENTRALE PSI 2 2021).* *Maxence SIMON*

Une permutation d'un ensemble E est une bijection de E dans E. Pour $E = [\![1, n-1]\!]$, on peut représenter une permutation σ de E avec le n-uplet $(\sigma(0), \dots, \sigma(n-1))$. On utilisera la fonction `permutations` du module `itertools` qui permet de parcourir l'ensemble des permutations d'un ensemble, par exemple pour l'ensemble des permutations de $[\![0, n-1]\!]$ pour un n préalablement défini :

```
from itertools import permutations

n=3
for s in permutations(range(n)):
    print(s)
```

Soit σ une permutation de $[\![0, n-1]\!]$, et $p \in [\![0, n-1]\!]$, on dit que p est une montée de σ si $\sigma(p) < \sigma(p+1)$.
Soit $A_{n,m}$ le nombre de permutations de $[\![0, n-1]\!]$ à m montées. On a $A_{1,0} = 1$.

1° Déterminer $A_{n,0}$, $A_{n,n-1}$ et $A_{n,m}$ pour $m \geq n$.

2° Écrire une fonction prenant en argument une permutation `L` et renvoyant son nombre de montées.

3° Écrire une fonction prenant en argument un entier `n` et renvoyant $[A_{n,0}, A_{n,1}, \dots, A_{n,n-1}]$

4° Montrer que $A_{n+1,m} = (n+1-m)A_{n,m-1} + (m+1)A_{n,m}$.

5° $A_n = \sum_{m=0}^{n-1} A_{n,m} X^m$.

Montrer que $A_{n+1} = X(1-X)A_n' + (1+nX)A_n$.

Correction :

1° Si la permutation σ n'a pas de montée alors pour tout p on a $\sigma(p) \geq \sigma(p+1)$, ainsi σ est strictement décroissante (strictement car bijective), ainsi $\sigma(0) > \sigma(1) > \ldots > \sigma(n-1)$, ainsi σ correspond à $(n-1, \ldots, 0)$ ie $\sigma = n-1-\mathrm{Id}$, ainsi $A_{n,0} = 1$.
De même si σ possède $n-1$ montées alors tout le monde est une montée, ie $\sigma(0) < \sigma(1) < \ldots < \sigma(n-1)$, ainsi σ correspond à $(0, \ldots, n-1)$, ie $\sigma = \mathrm{Id}$, ainsi $A_{n,n-1} = 1$.
Une permutation ne peux pas avoir n montées ou plus, ainsi $A_{n,m} = 0$ pour $m \geq n$.

2°
```
def nbrmontees(L):
    n=len(L)
    res=0
    for p in range(n-1):
        if L[p]<L[p+1]:
            res+=1
    return res

test=[(3,2,1,0),(2,1,3,0),(0,1,2,3)]
res=[nbrmontees(L) for L in test]
print(res)
```

3°
```
def Anm(n):
    res=[0 for k in range(n)]
    for s in permutations(range(n)):
        m=nbrmontees(s)
        res[m]+=1
    return res

for n in range(1,6):
    test=Anm(n)
    print(test,sum(test)) #sum(test)=n!
```

4° Quand on a une permutation σ de $[\![0,n]\!]$ représentée par $(a_0, \ldots, a_n)$ si on enlève l'élément n de la liste on obtient une permutation de $[\![0,n-1]\!]$, s'il y avait m montées alors la permutation obtenue possède nécessairement $m-1$ ou m montées , en effet si le n était en position k (différent de 0 et n, en 0 cela ne change rien et en n on perd une montée) alors $a_{k-1} < m > a_{k+1}$, en enlevant m on peut conserver la montée qu'on avait (si $a_{k-1} < a_{k+1}$) ou la perde (sinon).
Si on a une séquence $(a_0, \ldots, a_{n-1})$ d'éléments distincts de $[\![0, n-1]\!]$ alors on peut rajouter n dans cette séquence (il y a $n+1$ possibilités) et toutes les séquences sont obtenus de cette manière.
Si la séquence initiale possède k montées alors on peut la voire comme $k+1$ séquences décroissantes misent à la suite (elles sont séparées par une montée), si on rajoute n dans cette séquence il y a deux possibilités :
— on le met au début d'une séquence décroissante (il y a $k+1$ possibilités), alors il y a toujours autant de séquences décroissantes donc de montée
— on le met ailleurs (il y a $n-k$ possibilités), c'est nécessairement au milieu d'une séquence décroissante, ce qui fait que la séquence obtenue aura une montée de plus.
Pour que la séquence obtenue ait m montées il faut donc que :
— soit la séquence initiale en avait m (donc $A_{n,m}$ possibilités) et qu'on a pas créé de nouvelle montée en rajoutant n, il y a donc $m+1$ possibilités pour placer n, ce qui fait donc qu'il y a $(m+1)A_{n,m}$ séquences à m montées qui en garde m quand on enlève n.
— soit la séquence initiale en avait $m-1$ (donc $A_{n,m-1}$ possibilités) et qu'on a créé une nouvelle montée en rajoutant n, il y a donc $n-(m-1)$ possibilités pour placer n, ce qui fait qu'il y a $(n+1-m)A_{n,m-1}$ séquences à m montées qui ont $m-1$ montées quand on enlève n.
Ce qui montre la formule : $A_{n,m} = (n+1-m)A_{n,m-1} + (m+1)A_{n,m}$.

5° On a $A_{n+1} = \sum_{m=0}^{n} A_{n+1,m} X^m = 1 + X^n + \sum_{m=1}^{n-1} \left((n+1-m)A_{n,m-1} + (m+1)A_{n,m}\right) X^m$.

On a $XA_n' = \sum_{m=1}^{n-1} mA_{n,m}X^m$ et $X^2A_n' = \sum_{m=2}^{n} (m-1)A_{n,m-1}X^m = \sum_{m=1}^{n} (m-1)A_{n,m-1}X^m$.

Ainsi $A_{n+1} - X(1-X)A_n' = 1 + (1 + (n-1)A_{n,n-1})X^n + \sum_{m=1}^{n-1} \left(nA_{n,m-1} + A_{n,m}\right) X^m = 1 + nX^n +$

$nX\sum_{m=1}^{n-1} A_{n,m-1}X^{m-1}+\sum_{m=1}^{n-1} A_{n,m}X^m = 1+nX^n+nX\sum_{m=0}^{n-2} A_{n,m}X^m+A_n-1 = A_n+nX^n+nX(A_n-X^{n-1}) = (1+nX)A_n$.
On a bien montré $A_{n+1} = X(1-X)A_n' + (1+nX)A_n$.

Exercice 7 *(Centrale mp 2 2019).* *BEOS planche 5595,5618*

On considère p nombre premier impair, m entier impair, l'anneau $\mathbb{Z}/p\mathbb{Z}$ et $(\mathbb{Z}/p\mathbb{Z})^\times$ le groupe des inversibles de $\mathbb{Z}/p\mathbb{Z}$.

1° Soit G un groupe de cardinal $n \in \mathbb{N}^\star$. Montrer que G est cyclique si et seulement s'il existe un élément de G d'ordre n.

2° Programmer en `Python` une fonction qui renvoie le pgcd de 2 entiers.

3° Créer une fonction qui renvoie dans une liste les générateurs de $(\mathbb{Z}/n\mathbb{Z})^\times$. Afficher les résultats pour n allant de 2 à 30.
Pour $n = 25$ on doit obtenir $\{2, 3, 8, 12, 13, 17, 22, 23\}$.

4° Calculer le cardinal de $(\mathbb{Z}/p^m\mathbb{Z})^\times$.

5° Soit $k \in \mathbb{N}$. Montrer que : $(p+1)^{p^k} \equiv 1 + p^{k+1}\ [p^{k+2}]$.
En déduire que $(\mathbb{Z}/p^m\mathbb{Z})^\times$ possède un élément d'ordre p^{m-1}.

6° Soit $k \in \mathbb{Z}$. On note $\tilde{k}$ la classe d'équivalence de k dans $\mathbb{Z}/p\mathbb{Z}$ et $\bar{k}$ celle dans $\mathbb{Z}/p^m\mathbb{Z}$. On pose π l'application de $(\mathbb{Z}/p^m\mathbb{Z})^\times$ dans $(\mathbb{Z}/p\mathbb{Z})^\times$ qui à $\bar{k}$ associe $\tilde{k}$.
Montrer que π est bien définie et que c'est un morphisme surjectif de groupe.

7° On admet que $(\mathbb{Z}/p\mathbb{Z})^\times$ est cyclique. Montrer qu'il existe un élément d'ordre $p-1$ dans $(\mathbb{Z}/p^m\mathbb{Z})^\times$.

8° En déduire que $(\mathbb{Z}/p^m\mathbb{Z})^\times$ est cyclique.

Correction : Indications de l'examinateur : 4) Calculer le cardinal du complémentaire. 5) Procéder par récurrence

1° Procédons par double implication.
Sens réciproque : on suppose qu'il existe un élément $s \in G$ d'ordre n, alors pour tout $k \in [\![1, n]\!]$, on a $s^k \in G$, or n est le plus petit entier non nul tel que $s^n = e$, ainsi les s^k sont deux à deux distincts (en effet si $s^k = s^{k'}$, alors $s^{k-k'} = e$, ainsi, par minimalité de n, on a $k - k' = 0$ ou $k - k' = n$, comme $k' > 0$ et $k \leq n$ on est dans le cas $k - k' = 0$, ie $k = k'$), ainsi ils recouvrent tous les éléments de G, ce qui montre que G est engendré par s.
Sens direct : On suppose que G est cyclique, ainsi il existe $s \in G$ qui engendre G, notons ℓ l'ordre de s, ainsi $G = \{s^k,\ k \in [\![1, \ell]\!]\}$ (double inclusion immédiate, pour l'inclusion directe un élément g de G s'écrit s^k pour $k \in \mathbb{Z}$, si k' est congru à k modulo ℓ alors $s^k = s^{k'}$, ainsi on peut choisir k entre 1 et ℓ), comme G est de cardinal n, on a bien $\ell = n$.

2°
```
def pgcd(a,b):
    if b==0:
        return a
    return pgcd(b,a%b)
```

3° Une première fonction pour avoir les éléments de $(\mathbb{Z}/n\mathbb{Z})^\times$, une seconde pour l'ordre d'un de ses éléments et enfin on utilise la première question pour avoir les générateurs. On remarquera que pour $n \in \{8, 12, 15, 16, 20, 21, 24, 28, 30\}$, $(\mathbb{Z}/n\mathbb{Z})^\times$ n'est pas cyclique.

```
def inversible(n): # Éléments de (Z/nZ)^*
    L=[]
    for k in range(n):
        if (pgcd(k,n)==1):
            L.append(k)
    return L

def ordre(k,n): # Ordre de k dans (Z/nZ)^*
    assert pgcd(k,n)==1
    p=k%n
    for i in range(n):
        if (p==1):
            return i+1
        p=(p*k)%n
```

```
def generateur(n): # Générateurs de (Z/nZ)^*
    G=inversible(n)
    N=len(G)
    gen=[]
    for k in G:
        if ordre(k,n)==N:
            gen.append(k)
    return gen

for n in range(2,31):
    print("générateurs de (Z/"+str(n)+"Z)^* : ",generateur(n))
```

4° Les éléments de $(\mathbb{Z}/p^m\mathbb{Z})^\times$ sont les éléments k de $\mathbb{Z}/p^m\mathbb{Z}$ tels que $k\wedge p^m=1$, ainsi les éléments qui ne sont pas dans $(\mathbb{Z}/p^m\mathbb{Z})^\times$ sont exactement les éléments qui ne sont pas premiers avec p^m, c'est à dir tous les multiples de p, ie $p, 2p, 3p, \dots, p^{m-1}p$, il y en a donc p^{m-1}, ainsi le cardinal de $(\mathbb{Z}/p^m\mathbb{Z})^\times$ est $p^m-p^{m-1}=(p-1)p^{m-1}$, on peut vérifier rapidement avec python :

```
p,m=7,4
print(len(inversible(p**m)))
print(p**m-p**(m-1))
```

5° Montrons, par récurrence sur k, que $(p+1)^{p^k}\equiv 1+p^{k+1}\,[p^{k+2}]$.
Initialisation : Pour $k=0$, on doit montrer que $p+1\equiv 1+p\,[p^2]$, ce qui est bien vérifié.
Hérédité : Supposons le résultat au rang k, ainsi il existe $a\in\mathbb{Z}$ tel que $(p+1)^{p^k}=1+p^{k+1}+ap^{k+2}$, ainsi $(p+1)^{p^{k+1}}=(1+p^{k+1}+ap^{k+2})^p=\sum_{i=0}^{p}\binom{p}{i}p^{(k+1)i}(1+ap)^i=1+pp^{k+1}(1+ap)+\sum_{i=2}^{p}\binom{p}{i}p^{(k+1)i}(1+ap)^i=1+p^{k+2}+ap^{k+3}+bp^{k+3}$, où on a posé $b=\sum_{i=2}^{p}\binom{p}{i}p^{(k+1)i-k-3}(1+ap)^i$ qui est bien un entier puisque pour $i\geq 3$, $(k+1)i-k-3\geq 3k+3-k-3=2k\geq 0$, et pour $i=2$ on a $\binom{p}{2}p^{(k+1)2-k-3}=\frac{p(p-1)}{2}p^{k-1}=\frac{p-1}{2}p^k\in\mathbb{N}$ (car p impaire). Ce qui montre bien que $(p+1)^{p^{k+1}}\equiv 1+p^{k+2}\,[p^{k+3}]$
Ainsi, pour tout $k\in\mathbb{N}$, on a $(p+1)^{p^k}\equiv 1+p^{k+1}\,[p^{k+2}]$.
Tout d'abord on remarque que $1+p$ est un élément de $(\mathbb{Z}/p^m\mathbb{Z})^\times$ puisque $1+p$ et p^m sont premiers entre eux (sinon $1+p$ serait un multiple de p).
On a montré, pour $k=m-1$, qu'il existai $a\in\mathbb{Z}$ tel que $(p+1)^{p^{m-1}}=1+p^m+ap^{m+1}$, en regardant modulo p^m, on a $(p+1)^{p^{m-1}}\equiv 1\,[p^m]$, ainsi l'ordre de $1+p$ divise p^{m-1}. Mais la formule de récurrence pour $k<m-1$ dit aussi que $(p+1)^{p^k}$ n'est pas congru à 1 modulo p^m, ainsi $1+p$ est d'ordre p^{m-1}.

6° Pour montrer que π est bien défini, il faut montrer que si $(k,k')\in\mathbb{Z}^2$ sont tels que $\bar{k}=\bar{k'}$ alors $\tilde{k}=\tilde{k'}$ et que si $\bar{k}$ est inversible alors $\tilde{k}$ aussi (ce qui est claire : si on est premier avec p^m alors on est premier avec p). Si $\bar{k}=\bar{k'}$ alors il existe $a\in\mathbb{Z}$ tel que $k=k'+ap^m$, en posant $b=ap^{m-1}$ on a $k=k'+bp$ et donc $\tilde{k}=\tilde{k'}$. Ce qui montre que π est bien défini.

Pour $(k,k')\in\mathbb{Z}^2$, on a $\bar{k}\bar{k'}=k\bar{k'}$ et $k\tilde{k'}=\tilde{k}\tilde{k'}$, ainsi $\pi(\bar{k}\bar{k'})=\pi(k\bar{k'})=k\tilde{k'}=\tilde{k}\tilde{k'}=\pi(\bar{k})\pi(\bar{k'})$.
Soit $k\in\mathbb{Z}$ tel que $k\wedge p^m=1$, ainsi il existe $(u,v)\in\mathbb{Z}^2$ tels que $uk+vp^m=1$, ce qui montre que $\bar{u}=\bar{k}^{-1}$, or $uk+vp^{m-1}p=1$ et donc $\tilde{u}=\tilde{k}^{-1}$, ce qui montre que $\pi(\bar{k}^{-1})=\pi(\bar{u})=\tilde{u}=\tilde{k}^{-1}=\pi(\bar{k})^{-1}$.
Ainsi π est bien un morphisme de groupe, il est clairement surjectif (il suffit de considérer $\bar{1}$, $\bar{2}$, ..., $\bar{p-1}$ qui sont dans $(\mathbb{Z}/p^m)^\times$ et de remarquer que $(\mathbb{Z}/p\mathbb{Z})^\times=\{\tilde{1},\dots,\tilde{p-1}\}$.

7° Comme $(\mathbb{Z}/p\mathbb{Z})^\times$ est cyclique et est d'ordre $p-1$, alors il possède un élément y d'ordre $p-1$, notons x un antécédent de y par π. Notons ℓ l'ordre de x, ainsi $y^\ell=\pi(x^\ell)=1$ et donc ℓ divise l'ordre de y, on a donc ℓ qui est un diviseur de $(p-1)$ mais comme c'est aussi un diviseur de $(p-1)p^{m-1}$ on a que ℓ vaut 1 ou $p-1$, comme $\ell\neq 1$ (sinon on aurait $x=1$ et donc $y=1$) on a $\ell=p-1$. Ainsi x est d'ordre $p-1$.

8° Comme le produit d'un élément d'ordre p^{m-1} et d'un élément d'ordre $p-1$ (c'est deux nombres sont premiers entre eux) est un élément d'ordre $(p-1)p^{m-1}$, $(\mathbb{Z}/p^m\mathbb{Z})^\times$ possède un élément d'ordre son cardinal, il est donc cyclique.