Devoir 09

Polynômes, intégrales généralisées
Samedi 02/03/2024

Problème

Le but de ce problème est d'explorer quelques propriétés des polynômes de HERMITE, apparaissant dans les formules de dérivées successives de la densité gaussienne.

Le tracé par ordinateur, obtenu en exécutant le script complété de la partie E, des graphes des premiers polynômes de cette famille est un complément bienvenu à la copie.

Partie A
Préliminaires

On définit d'une part la fonction $g : \mathbb{R} \to \mathbb{R}$ par la formule

$$\forall x \in \mathbb{R},\ g(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2}$$

et d'autre part la suite de polynômes $(H_n)_{n\in\mathbb{N}}$ par la récurrence

$$(\mathscr{H}) : H_0 = 1 \text{ et } \forall n \in \mathbb{N},\ H_{n+1}(X) = X.H_n(X) - H_n'(X)$$

où H_n' désigne le polynôme dérivé de H_n.

A.1.a. Ecrire les termes H_1, H_2, H_3 et H_4 de la suite $(H_n)_{n\in\mathbb{N}}$.

A.1.b. Argumenter rapidement quant à la bonne définition de la suite $(H_n)_{n\in\mathbb{N}}$.

A.1.c. Donner le degré et le coefficient dominant de chaque polynôme H_n. On effectuera une démonstration par récurrence.

A.1.d. Montrer que

1. si n est un entier pair alors H_n est pair,
2. si n est un entier impair alors H_n est impair.

A.1.e. Montrer que les coefficients de chaque polynôme H_n sont des entiers relatifs.

A.2.a. Justifier que la fonction g est de classe $\mathscr{C}^\infty$ sur $\mathbb{R}$. Pour $n \in \mathbb{N}$, on note $g^{(n)}$ sa dérivée n-ième [1], *i.e.* , en notation physicienne,

$$\forall x \in \mathbb{R},\ g^{(n)}(x) = \frac{d^n g(x)}{dx^n}$$

A.2.b. Montrer que

$$\forall n \in \mathbb{N}, \forall x \in \mathbb{R},\ g^{(n)}(x) = (-1)^n H_n(x).g(x)$$

A.2.c. Donner, sans faire de nouveaux calculs, une formule pour $g^{(4)}$.

1. lorsque $n = 0$, il s'agit de la fonction g elle-même

Partie B

Quelques propriétés intégrales

B.1. Soit P est un polynôme quelconque.

B.1.a. Montrer que lorsque $x \to +\infty$,

$$|P(x)e^{-\frac{1}{2}x^2}|.e^x \to 0.$$

B.1.b. En déduire que l'intégrale généralisée

$$\int_0^{+\infty} P(x)e^{-\frac{1}{2}x^2}\,dx$$

est convergente.

B.2. Montrer que l'intégrale généralisée

$$\int_{-\infty}^{+\infty} P(x)e^{-\frac{1}{2}x^2}\,dx$$

est convergente.

$\star\star\star$

Si P, Q sont deux polynômes, on note

$$\langle P,Q\rangle = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{+\infty} P(x).Q(x).e^{-\frac{1}{2}x^2}\,dx.$$

B.3.a. Montrer que, si $P \in \mathbb{R}[X]$ est fixé alors l'application $\phi_P : \mathbb{R}[X] \to \mathbb{R}$ définie par

$$\forall Q \in \mathbb{R}[X],\ \phi_P(Q) = \langle P,Q\rangle$$

est bien définie, à valeurs réelles et linéaire. C'est à dire que

$$\forall \lambda_1,\lambda_2 \in \mathbb{R},\ \forall Q_1,Q_2 \in \mathbb{R}[X],\ \phi_P(\lambda_1.Q_1+\lambda_2.Q_2) = \lambda_1.\phi_P(Q_1)+\lambda_2.\phi_P(Q_2).$$

Indication:Utiliser la linéarité de l'intégrale généralisée.

B.3.b. Montrer que

$$\forall P,Q \in \mathbb{R}[X],\ \langle P,Q\rangle = \langle Q,P\rangle.$$

B.3.c. Montrer que

$$\forall P,Q \in \mathbb{R}[X],\ \langle X.P,Q\rangle = \langle P,X.Q\rangle.$$

B.3.d. Montrer finalement que

$$\forall P \in \mathbb{R}[X],\ \langle P,P\rangle \geq 0$$

et que

$$\forall P \in \mathbb{R}[X],\ \langle P,P\rangle = 0 \Rightarrow P = 0.$$

B.4.a. Montrer que pour n, $m \in \mathbb{N}$,

$$\langle H_n,X^m\rangle = (-1)^n\int_{-\infty}^{+\infty} g^{(n)}(x).x^m\,dx.$$

B.4.b. Montrer, par récurrence sur $n \in \mathbb{N}^*$, que

$$\forall n \in \mathbb{N}^*,\forall m \in \mathbb{N}, 0 \leq m \leq n-1 \Rightarrow \langle H_n,X^m\rangle = 0.$$

Indication:On traitera séparément les cas $m = 0$ et $m > 0$. Une ipp peut-être utile pour traiter ce deuxième cas.

B.4.c. Montrer, par récurrence sur $n \in \mathbb{N}$, que

$$\forall n \in \mathbb{N},\ \langle H_n,X^n\rangle = n!.$$

B.5. Déduire [2] de la question précedente que

$$\forall n \in \mathbb{N}, \forall P \in \mathbb{R}_{n-1}[X], \langle H_n, P\rangle = 0$$

puis que

$$\forall n, m \in \mathbb{N}, \langle H_n, H_m\rangle = \begin{cases} 0 & \text{si } m \neq n \\ n! & \text{si } m = n \end{cases}.$$

Partie C

Un peu d'algèbre linéaire

On fixe $n \in \mathbb{N}^*$.

Pour un polynôme $P \in \mathbb{R}_n[X]$, on admet qu'il existe un $(n+1)$-uplet de $\mathbb{R}^{n+1}$ (les « coordonnées de P sur la base $\mathscr{H}_n$ ») tel que

$$P = \sum_{k=0}^{n} p_k.H_k.$$

C.1. Démontrer que

$$\forall \ell \in \{0,\ldots,n\}\,,\; p_\ell = \frac{1}{\ell!}\langle H_\ell, P\rangle\,.$$

C.2.a. Justifier, en utilisant la récurrence $\mathscr{H}$, que le polynôme $P = H_{n+1} - X.H_n$ est de degré $\leq n-1$ et montrer que, pour un certain coefficient $p_{n-1} \in \mathbb{R}$, $P = p_{n-1}.H_{n-1}$.

Indication:Calculer les coordonnées de P sur la base $\mathscr{H}_{n-1}$ en se servant notamment des relations des questions B.3 et B.5.

C.2.b. En déduire que

$$H_n' = n.H_{n-1}.$$

et que la suite (H_n) vérifie la récurrence à deux crans

$$H_0 = 1, H_1 = X \text{ et } \forall n \in \mathbb{N}^*, H_{n+1} = X.H_n - n.H_{n-1}.$$

2. On rappelle que $\mathbb{R}_{n-1}[X]$ est l'ensemble des polynômes de degré inférieur ou égal à $n-1$. Pour $n=0$, on convient que cet ensemble ne contient que le polynôme nul.

Partie D

A propos des racines

Le but de cette partie est de démontrer les résultats suivants :

1. Chaque polynôme H_n a exactement n racines réelles distinctes (et donc de multiplicité 1)
2. Les racines de H_n et de H_{n+1} sont entrelacées au sens où entre chaque couple de racines consécutives de H_{n+1} il y a exactement une racine de H_n.

D.1. Dans cette question, on fixe $n \in \mathbb{N}^*$.

D.1.a. On suppose que H_n n'a pas de racine réelle. Montrer qu'alors

$$\forall x \in \mathbb{R}, H_n(x) > 0$$

et aboutir à une contradiction. Indication:Utiliser le fait que $\langle H_0, H_n \rangle = 0$, *cf.* Question B.5

$\star\star\star$

On note $\{\alpha_k, k \in \{1,\dots,N_n\}\}$ l'ensemble des racines réelles de H_n (au nombre de N_n donc), on suppose que

$$\alpha_1 < \cdots < \alpha_{N_n}$$

et on note m_k la multiplicité de α_k en tant que racine de H_n. On a donc la factorisation

$$H_n(X) = (X-\alpha_1)^{m_1}..(X-\alpha_{N_n})^{m_{N_n}}.Q(X)$$

où Q est un polynôme unitaire n'ayant pas de racines réelles et donc $\forall x \in \mathbb{R}$, $Q(x) > 0$.

D.1.b. On traite quelques cas particuliers avant d'aborder le cas général.

D.1.b.i. On suppose que $n \geq 2$ et $N_n = 1$ et $m_1 \geq 2$ est impair. Donner le signe de $H_n(x).(x-\alpha_1)$ pour $x \in \mathbb{R}$ et conclure à une contradiction du fait que $\langle H_n, X-\alpha_1 \rangle = 0$.

D.1.b.ii. On suppose que $n \geq 3$ et $N_n = 2$ et $m_1 = 1$, $m_2 = 2$. Donner le signe de $H_n(x).(x-\alpha_1)$ pour $x \in \mathbb{R}$ et conclure à contradiction du fait que $\langle H_n, X-\alpha_1 \rangle = 0$.

D.1.b.iii. On suppose que $n \geq 4$ et $N_n = 2$ et $m_1 = 1$, $m_2 = 3$. Donner le signe de $H_n(x).(x-\alpha_1).(x-\alpha_2)$ pour $x \in \mathbb{R}$ et conclure à une contradiction du fait que $\langle H_n, (X-\alpha_1).(X-\alpha_2) \rangle = 0$.

$\star\star\star$

Pour le cas général, on pose $P(X) = \prod_{k=1}^{N_n}(X-\alpha_k)^{\varepsilon_k}$ où $\varepsilon_k = +1$ si m_k est impair et $\varepsilon_k = 0$ si m_k est pair.

On suppose par contradiction que $N_n < n$, ce qui arriverait si H_n avait une racine au moins double ou si H_n avait une racine complexe non réelle.

D.1.c. Montrer qu'alors le polynôme P est de degré $< n$.

D.1.d. Donner le signe de $H_n(x).P(x)$ pour $x \in \mathbb{R}$

D.1.e. et conclure à une contradiction du fait que $\langle H_n, P \rangle = 0$.

D.2. La conclusion de la question précédente est que, pour $n \geq 1$, l'ensemble des racines de H_n peut s'écrire $\{\alpha_{k,n}, k \in \{1,\dots,n\}\}$ alors que celui des racines de H_{n+1} peut s'écrire $\{\alpha_{k,n+1}, k \in \{1,\dots,n+1\}\}$ avec, dès que $n \geq 2$,

$$\alpha_{1,n} < \cdots < \alpha_{n,n} \text{ et } \alpha_{1,n+1} < \cdots < \alpha_{n,n+1} < \alpha_{n+1,n+1}.$$

On cherche à démontrer que, pour $n \in \mathbb{N}^*$, en notant

$$I_1 =]\alpha_{1,n+1}, \alpha_{2,n+1}[, \dots I_n =]\alpha_{n,n+1}, \alpha_{n+1,n+1}[,$$

on a

$$\forall k \in \{1,\dots,n\}, \alpha_{k,n} \in I_k.$$

D.2.a. Démontrer en utilisant le théorème des accroissements finis et la relation obtenue en Question C.2.b que dans chaque intervalle I_k, il y a au moins une racine de H_n.

D.2.b. Conclure.

Partie E

Informatique

Le but de cette partie est de développer un petit système de calcul sur les polynômes en Python, un module Python, permettant, entre autres, de calculer effectivement un certain nombre termes consécutifs d'une suite de polynômes définie par une récurrence telle que $\mathscr{H}$.

Utiliser le script `python/polynomes-empty.py`.

On code un polynôme P en Python par une liste de nombres. Cette liste peut avoir une longueur variable, le tout est qu'elle soit suffisamment longue pour pouvoir contenir tous les coefficients. Par exemple, le polynôme

$$P = 1 + 2X^2 + 7X^5$$

peut être codé par l'une des listes `P=[1.0,0,2.0,0,0,7.0]` ou `P=[1.0,0,2.0,0,0,7.0,0,0]`. Dit autrement, la liste `P`, de longueur `len(P)` code le polynôme

$$P = \sum_{k=0}^{\texttt{len(P)-1}} \texttt{P[}k\texttt{]}X^k.$$

Les fonctions dont nous avons besoin au minimum pour gérer les polynômes sont

1. `degre(P)` qui retourne le degré de P, liée à une fonction de nature technique `trim(P)` qui retourne la liste de longueur minimale représentant P
2. `somme(P,Q)` qui retourne la somme de deux polynômes, `pdt_par_scalaire(lambada,P)` qui retourne le produit du nombre `lambada` par le polynôme `P`,
3. `produitX(P)` qui retourne le polynôme $X.P$ et, plus généralement, `produit(P,Q)` qui retourne le produit de deux polynômes P et Q
4. `derive(P)` qui retourne P', le polynôme dérivé de P
5. Les fonctions de substitution
   (a) `subst_reel(P,x)` qui retourne la valeur du polynôme P en le nombre x,
   (b) `subst_poly(P,Q)` qui retourne le polynôme $P \circ Q = P(Q)$
   (c) `subst_matrice(P,A)` qui retourne la valeur du polynôme P en la matrice carrée A,

Le script en annexe est à compléter et/ou analyser. Une fois complété, le faire tourner donne les graphes de $H_0, .., H_6$.

E.1. Compléter le corps de la fonction `degre`. Il s'agit de trouver, avec indices descendants, l'indice du premier, en partant de la fin, élément non nul d'une liste.

E.2. Compléter le corps de la fonction `pdt_par_scalaire`. On peut s'inspirer du texte de la fonction `somme`.

E.3.a. Expliquer le fonctionnement de la fonction `produitX`.

E.3.b. Expliquer le fonctionnement de la fonction `produit`. Quelle formule abstraite sur les polynômes y est implémentée ?

E.4. Compléter le corps de la fonction `subst_reel`. On peut s'inspirer des fonctions (plus compliquées) `subst_poly` et `subst_matrice`.

E.5. Expliquer le fonctionnement de la fonction `Hermite`.

Partie F

Valeurs propres, informatique et racines des polynômes H_n

Pour $n \geq 2$, on construit la matrice Δ_n de taille $n \times n$ en posant,

$$\forall k, \ell \in \{1, \ldots, n\}, [\Delta_n]_{k,\ell} = \begin{cases} 0 & \text{si } k \neq \ell + 1 \text{ ou } \ell \neq k + 1 \\ \sqrt{k} & \text{si } \ell = k + 1 \\ \sqrt{\ell} & \text{si } k = \ell + 1 \end{cases}$$

F.1. Ecrire une fonction Python d'entête `Delta(n)` qui, étant donné un entier n retourne une matrice `numpy` de `shape (n,n)` et contenant la matrice Δ_n.

F.2.a. Calculer, à la main, les valeurs propres de Δ_2 et vérifier que ce sont (exactement) les racines du polynôme H_2. (L'ordre de ces calculs est laissé à votre choix).

F.2.b. Calculer, à la main, les valeurs propres de Δ_3 et vérifier que ce sont (exactement) les racines du polynôme H_2. (L'ordre de ces calculs est laissé à votre choix).

F.2.c. Calculer, à la main, les valeurs propres de Δ_4 et vérifier que ce sont (exactement) les racines du polynôme H_4. (L'ordre de ces calculs est laissé à votre choix).

F.3. Ecrire une fonction d'entête `EgaliteRacinesVp(n)` prenant en argument un entier n et imprimant, pour k variant de 1 à n, les valeurs propres de chaque Δ_k et les valeurs de H_k calculées en ces valeurs propres.

On pourra utiliser la fonction `np.linalg.eigvals` pour évaluer les valeurs propres de Δ_k et les fonctions développées en partie E pour évaluer le polynôme H_k en ces valeurs propres.

Exécuter et constater la presque nullité ou pas de H_k aux valeurs propres de Δ_k.

PYTHON
AGRO-VETO
2020

Listes

- `[]` -------- Créer une liste vide
- `[a]*n` ----- Créer une liste avec n fois l'élément `a`
- `L.append(a)` Ajoute l'élément `a` à la fin de la liste `L`
- `L1 + L2` --- Concatène les deux listes `L1` et `L2`
- `len(L)` ---- Renvoie le nombre d'éléments de la liste `L`
- `L.pop(k)` -- Renvoie l'élément d'indice k de `L` et l'enlève de `L`
- `L.remove(a)` Enlève une fois la valeur `a` de la liste `L`
- `max(L)` ---- Renvoie le plus grand élément de la liste `L`
- `min(L)` ---- Renvoie le plus petit élément de la liste `L`
- `sum(L)` ---- Renvoie la somme de tous les éléments de la liste `L`

Numpy

`import numpy as np`

- `np.array()` ------------- Transforme une liste en matrice `numpy`
- `np.linspace(a,b,n)` ----- Crée une matrice ligne de n valeurs uniformément réparties entre a et b (inclus)
- `np.zeros([n,m])` -------- Crée la matrice nulle de taille $n \times m$
- `np.eye(n)` -------------- Crée la matrice identité de taille n
- `np.diag(L)` ------------- Crée la matrice diagonale dont les termes diagonaux sont les éléments de la liste `L`
- `np.transpose(M)` -------- Renvoie la transposée de M
- `np.dot(M,P)` ------------ Renvoie le produit matriciel MP
- `np.sum(M)` -------------- Renvoie la somme de tous les éléments de M
- `np.prod(M)` ------------- Renvoie le produit de tous les éléments de M
- `np.max(M)` -------------- Renvoie le plus grand élément de M
- `np.min(M)` -------------- Renvoie le plus petit élément de M
- `np.shape(M)` ------------ Renvoie dans un couple le format de la matrice M
- `np.size(M)` ------------- Renvoie le nombre d'éléments de M

Numpy.linalg

`import numpy.linalg as la`

- `la.inv(M)` -------------- Renvoie l'inverse de la matrice M si elle est inversible
- `la.eigvals(M)` ---------- Renvoie la liste des valeurs propres de M
- `la.eig(M)` -------------- Renvoie un couple `L,P` où `L` est la liste des valeurs propres de M et `P` la matrice de passage associée
- `la.matrix_rank(M)` ------- Renvoie le rang de M

Random

`import numpy.random as rd`

- `rd.random()` ------- Simule une réalisation d'une variable $X \hookrightarrow \mathcal{U}([0,1])$
- `rd.randint(a,b)` --- Simule une réalisation d'une variable $X \hookrightarrow \mathcal{U}([\![a,b]\!])$
- `rd.gauss(0,1)` ----- Simule une réalisation d'une variable $X \hookrightarrow \mathcal{N}(0,1)$
- `rd.choice(L)` ------ Choisit aléatoirement un élément de la liste `L`

Math

`import numpy as np`

- `np.atan(x)` ------- Renvoie $\arctan(x)$
- `np.floor(x)` ------ Renvoie $\lfloor x \rfloor$
- `np.factorial(n)` -- Renvoie $n!$ si $n \in \mathbb{N}$
- `np.sqrt(x)` -- Renvoie $\sqrt{x}$ si $x \geq 0$
- `np.log(x)` --- Renvoie $\ln(x)$ si $x > 0$
- `np.exp(x)` --- Renvoie e^x

Logique

- `a == b` ---- Teste l'égalité « $a = b$ »
- `a != b` ---- Teste « $a \neq b$ »
- `a < b` ----- Teste « $a < b$ »
- `a <= b` ---- Teste « $a \leq b$ »
- `a > b` ----- Teste « $a > b$ »
- `a >= b` ---- Teste « $a \geq b$ »
- `not A` ----- Renvoie la négation de A
- `A and B` --- Renvoie « A et B »
- `A or B` ---- Renvoie « A ou B »
- `True` ------ Constante booléenne « Vrai »
- `False` ----- Constante booléenne « Faux »

Matplotlib.pyplot

`import matplotlib.pyplot as plt`

- `plt.plot(X,Y,'+-r')` ---- Génère la courbe des points définis par les listes `X` et `Y` (abscisses et ordonnées) avec les options :
  - symbole : `'.'` point, `'o'` rond, `'h'` hexagone, `'+'` plus, `'x'` croix, `'*'` étoile, ...
  - ligne : `'-'` trait plein, `'--'` pointillé, `'-.'` alterné, ...
  - couleur : `'b'` **bleu** , `'r'` **rouge** , `'g'` **vert** , `'c'` **cyan** , `'m'` **magenta** , `'k'` **noir** , ...
- `plt.bar(X,Y)` ---------- Génère l'histogramme des points définis par les listes `X` et `Y` (abscisses et ordonnées)
- `plt.axis('equal')` ------ Rend le repère orthonormé
- `plt.xlim(xmin,xmax)` ---- Fixe les bornes de l'axe des abscisses
- `plt.ylim(ymin,ymax)` ---- Fixe les bornes de l'axe des ordonnées
- `plt.show()` ------------ Affiche le graphique

Cette liste est non exhaustive. Les candidats sont libres d'utiliser les commandes de leur choix.