Problème de la semaine 12

Problème 1 — (Symétries d'un espace vectoriel).

L'objectif principal de ce problème est d'étudier les propriétés des symétries dans un espace vectoriel, de manière analogue à ce que nous avons vu en classe pour les projections.

Rappel des notations et définitions du cours. Soit E un $\mathbb{K}$-ev ; soient F et G deux sev supplémentaires de E (c'est à dire : $E = F \bigoplus G$).

Dans ce contexte : $\quad \forall\, \overrightarrow{v} \in E, \quad \exists!\left(\overrightarrow{f}, \overrightarrow{g}\right) \in F \times G, \quad \overrightarrow{v} = \overrightarrow{f} + \overrightarrow{g}.$

Ces hypothèses et notations étant posées, on appelle :

- **projection sur F parallèlement à G** l'application p_F de E dans E définie en posant $p_F\left(\overrightarrow{v}\right) = \overrightarrow{f}$;
- **symétrie par rapport à F parallèlement à G** l'application s_F de E dans E définie en posant $s_F\left(\overrightarrow{v}\right) = \overrightarrow{f} - \overrightarrow{g}$.

Partie I — Exemple

Dans cette partie, E désigne le $\mathbb{R}$-espace vectoriel $\mathrm{M}_2(\mathbb{R})$ des matrices carrées de taille 2 à coefficients réels.

On pose $F = \mathrm{Vect}\left(\mathrm{I}_2\right)$, et on note G l'ensemble des matrices de E de trace nulle.

1/ Montrer que G est un sev de E, et en déterminer une famille génératrice.

2/ Montrer que F et G sont supplémentaires dans E.

3/ Donner l'expression de la projection p_F sur F parallèlement à G.[1]

4/ Donner l'expression de la symétrie s_F par rapport à F parallèlement à G.

Partie II — Généralités sur les projections et symétries

On revient à présent au cas général : dans toute cette partie, E désigne un $\mathbb{K}$-ev, et F et G deux sev supplémentaires dans E ($E = F \bigoplus G$).

5/ Montrer que l'application s_F est un endomorphisme de E.

6/ Justifier que $s_F \in \mathrm{GL}(E)$.

7/ Etablir une relation entre les endomorphismes s_F, p_F et p_G.

8/ Déterminer $\ker\left(s_F - \mathrm{id}_E\right)$ et $\ker\left(s_F + \mathrm{id}_E\right)$.

1. Il s'agit d'expliciter l'image d'une matrice $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$ par l'application p_F.

Partie III — Réflexions dans un espace vectoriel

Dans cette partie, E désigne un $\mathbb{K}$-ev. Un endomorphisme f de E est appelé une **réflexion** si $f^2 = \mathrm{id}_E$; en d'autres termes, une réflexion de E est une involution linéaire de E.

Soit s un endomorphisme de E.

9/ Montrer que si s est une réflexion, alors $E = \ker(s - \mathrm{id}_E) \bigoplus \ker(s + \mathrm{id}_E)$.

10/ Réciproquement, montrer que si $E = \ker(s - \mathrm{id}_E) \bigoplus \ker(s + \mathrm{id}_E)$, alors l'endomorphisme s est une réflexion.

Partie IV — Synthèse : toute réflexion est une symétrie

Dans cette partie, E désigne toujours un $\mathbb{K}$-ev, et s un endomorphisme de E.

11/ Montrer que si s est une réflexion, alors s est une symétrie par rapport à F parallèlement à G, où F et G sont deux sev de E que l'on précisera.

Partie V — Application

On définit une application $u : \mathrm{M}_2(\mathbb{K}) \longrightarrow \mathrm{M}_2(\mathbb{K})$ en posant pour toute $M = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \mathrm{M}_2(\mathbb{K})$:

$$u(M) = \begin{pmatrix} b & a \\ d & c \end{pmatrix}$$

On admet que u est un endomorphisme de $\mathrm{M}_2(\mathbb{K})$.

12/ Etablir que $\ker\left(u - \mathrm{id}_{\mathrm{M}_2(\mathbb{K})}\right) = \mathrm{Vect}(M_1, M_2)$, où M_1 et M_2 sont deux matrices que l'on explicitera.

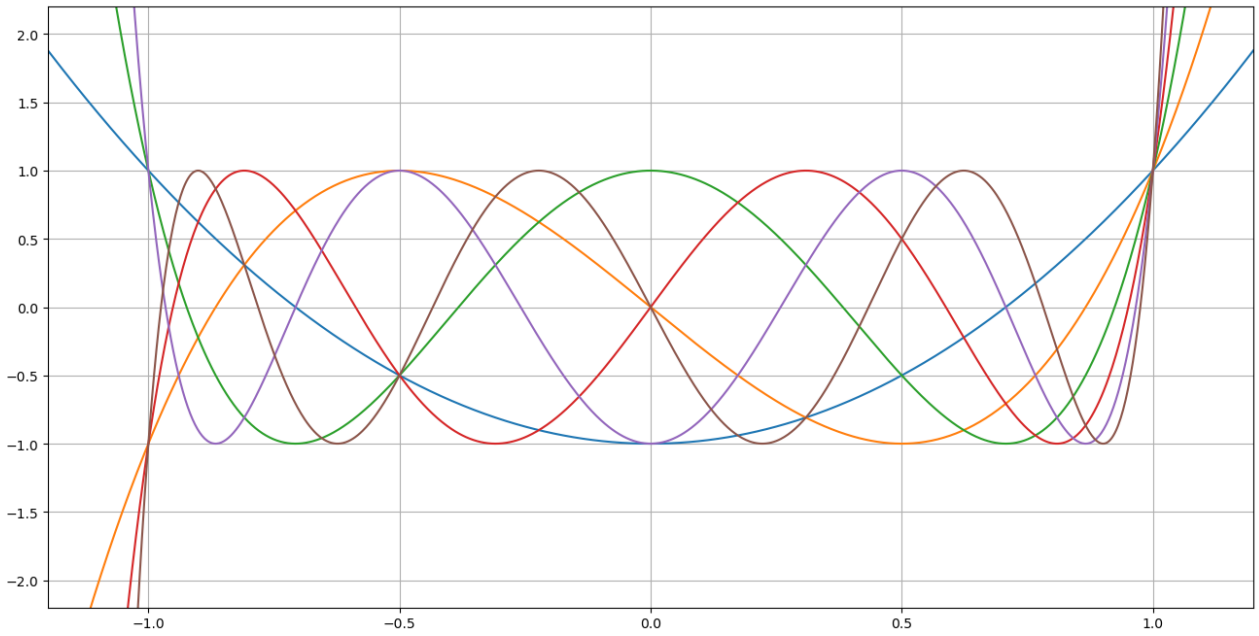
13/ Déterminer $\ker\left(u + \mathrm{id}_{\mathrm{M}_2(\mathbb{K})}\right)$, et en préciser une famille génératrice.

14/ Montrer que : $\mathrm{M}_2(\mathbb{K}) = \ker(u - \mathrm{id}_E) \bigoplus \ker(u + \mathrm{id}_E)$.

Exercice 1 — (Polynômes de Tchebychev, illustration).

Le but de cet exercice est d'illustrer une propriété de la famille $(T_n)_n$ des polynômes de Tchebychev : chacun des polynômes T_n est exactement de degré n, et chacun des polynômes T_n possède exactement n racines réelles deux à deux distinctes, toutes comprises entre -1 et 1.

La figure ci-dessous, où l'on a fait apparaître les courbes représentatives des premiers polynômes de Tchebychev (sauf $T_0 = 1$ et $T_1 = X$, qui sont un peu "à part"), illustre cette propriété.



Comme vous l'avez sans doute déjà compris, votre mission va consister à créer le graphique ci-dessus, ce qui implique primo de programmer la construction des polynômes de Tchebychev, et secundo de créer un graphique contenant les courbes représentatives des N premiers polynômes de Tchebychev, N étant un entier choisi par l'utilisateur.[2]

1/ **Construction des polynômes de Tchebychev**. Ecrire une fonction **TCHEB(n)**, qui reçoit comme paramètre un entier n, et qui renvoie le n-ème polynôme de Tchebychev T_n.

2/ **Construction du graphique**. Ecrire un programme qui demande à l'utilisateur un entier N, et qui construit les courbes représentatives des N premiers polynômes de Tchebychev (sauf éventuellement les deux premiers).

2. Des indications sont données après l'énoncé pour réaliser ces programmes.

Ce dont vous pourrez avoir besoin

➢ Un peu de maths — Définition des polynômes de Tchebychev.

On rappelle que la suite de polynômes $(T_n)_n$ peut être définie en posant :

$$T_0 = 1\,; \qquad T_1 = X\,; \qquad \forall\, n \in \mathbb{N}, \quad T_{n+2} = 2XT_{n+1} - T_n$$

➢ Et de l'informatique.

➤ Définition de polynômes avec numpy

On commence par importer la bibliothèque numpy via l'instruction classique : `import numpy as np`.

Ceci fait, on dispose d'une fonction (poly1d) qui permet de définir un polynôme, soit à l'aide de ses coefficients, soit à l'aide de ses racines. Explicitement :

le polynôme $aX^2 + bX + c$ peut être défini par l'instruction `np.poly1d([a,b,c])`.

le polynôme $(X-a)(X-b)(X-c)$ peut être défini par l'instruction `np.poly1d([a,b,c],True)`.

Dans les deux cas, l'objet obtenu est une fonction, que l'on peut ensuite évaluer numériquement. Illustration :

```
>>>import numpy as np;

>>>P =np.poly1d([1,2,3]) ; # définit le polynôme P comme P = X^2 + 2X +3

>>>P(1); # Calcule P(1)...
6

>>>Q =np.poly1d([1,2,3], True); # définit le polynôme Q comme Q = (X-1)(X-2)(X-3)

>>>Q(1); # Calcule Q(1)...
0.0
```

➤ Tracé de graphiques

Le code ci-dessous est un code possible pour créer la représentation graphique de la fonction polynomiale $P : x \longmapsto x^2+2x+3$ sur l'intervalle $[-3, 3]$. Son objet est de vous rappeler la syntaxe nécessaire pour faire des graphes en Python, et le principe de construction d'une "courbe" (une "courbe" est un nuage de points, construit à partir d'une liste des abscisses et d'une liste des ordonnées).

```
import matplotlib.pyplot as plt;
import numpy as np;

P =np.poly1d([1,2,3]); # définit le polynôme P comme P = X^2 + 2X +3

NPTS =1000 # Nombre de points
xmin =-3 # Comme son nom le suggère assez bien...
xmax =3 #

h =(xmax-xmin)/NPTS # Définition du pas

# Création de la liste des abscisses (subdivision de l'intervalle [xmin, xmax])
x =xmin
LABS =[x]
for k in range(NPTS):
    x =x+h
    LABS =LABS +[x]

# Création de la liste des ordonnées (images de P aux points de la subdivision)
LORD =[]
for k in range(len(LABS)):
        LORD =LORD +[P(LABS[k]) ]

# Création du graphique
plt.clf();
axes = plt.gca()
axes.set_xlim(xmin, xmax)
plt.plot(LABS, LORD)
plt.grid(True)
plt.show();
```