HAP608P « Programmation pour la physique », Université de Montpellier, 2025

Examen – Session 2

Durée 2h. Documents fournis sur le site du cours et documents sur papier autorisés. Autres documents et accès à internet interdits. La barème est à titre indicatif et peut encore changer. Il faut que vos programmes soient **exécutables** par la commande `python3 nomdufichier.py` et qu'ils **affichent leurs résultats** (notamment, ils ne doivent pas dépendre de l'environnement `Spyder` pour fonctionner correctement).

Exercice 1. Programmation élémentaire, graphisme (7 points)

On définit la fonction $\mathrm{Li}_2(z)$ par

$$\mathrm{Li}_2(z) = \lim_{N\to\infty} \sum_{k=1}^{N} \frac{z^k}{k^2}\,, \qquad z \in \mathbb{C}.$$

Réalisez une fonction `Li2(z, N = 100)` qui calcule une approximation à $\mathrm{Li}_2(z)$ avec les premiers `N` termes de la somme. Tracer, sur un même repère, les courbes $\mathrm{Re}(\mathrm{Li}_2(iy))$ et $\mathrm{Im}(\mathrm{Li}_2(iy))$ en fonction de y, pour $y \in [-1, 1]$.

Rappel : La partie réelle et imaginaire d'un nombre complexe $z = x+iy$ s'obtiennent respectivement avec `x=z.real` et `y=z.imag`. L'unité imaginaire i est `1j` en Python.

Exercice 2. Recherche des zéros (5 points)

Résolvez l'équation $e^x = 5\sin(x)$ sur l'intervalle $[1, \infty[$ par une méthode numérique de votre choix, avec une précision d'au moins 10^{-6}. Vous n'avez pas le droit d'utiliser la bibliothèque `SciPy`.

Exercice 3. Algèbre linéaire, calcul matriciel (8 points)

1. Réalisez une fonction `decomp_DR(A)` qui effectue la *décomposition DR* de la matrice carrée A. Elle renverra donc deux matrices D et R telles que D est diagonale, R est zéro sur la diagonale et
$$D + R = A\,.$$
Indications : si `A` est une matrice (un tableau 2-dimensionnel), la commande `numpy.diag(A)` renvoie un vecteur (un tableau 1-dimensionnel) qui contient les éléments diagonaux de `A`. Si `v` est un vecteur, la commande `numpy.diag(v)` renvoie une matrice diagonale avec les éléments de `v` sur la diagonale principale.
2. Rendez-vous compte que l'inverse d'une matrice diagonale est également diagonale et très facile à calculer efficacement. Utilisez ce fait pour réaliser une fonction `inv_diag(D)` qui retourne l'inverse de son argument D (supposé diagonal et inversible).
3. On propose l'algorithme suivant pour résoudre le système d'équations linéaires $A\vec{x} = \vec{b}$, où A est une matrice $n \times n$ appropriée et $\vec{b}$ est un vecteur à n composantes :
   — Choisir une précision numérique souhaitée ϵ. Décomposer $A = D + R$ comme ci-dessus et calculer l'inverse D^{-1} de D. Poser $\vec{x}$ = une copie de $\vec{b}$ et $\vec{x}_{\mathrm{anc}}$ = un vecteur quelconque.
   — Remplacer $\vec{x}_{\mathrm{anc}} \leftarrow \vec{x}$, puis $\vec{x} \leftarrow D^{-1}(\vec{b} - R\vec{x})$.
   — Tant que $|\vec{x} - \vec{x}_{\mathrm{anc}}| > \epsilon$, itérer à partir de l'étape précédente. Lorsque $|\vec{x} - \vec{x}_{\mathrm{anc}}| \leq \epsilon$, terminer et renvoyer $\vec{x}$.

   Implémentez cet algorithme pour trouver la solution $\vec{x}$ du système linéaire $A\vec{x} = \vec{b}$, avec une précision de $\epsilon = 10^{-6}$ et
$$A = \begin{pmatrix} 1 & -1 & 0 \\ 2 & -5 & 2 \\ 0 & 1 & 3 \end{pmatrix}, \quad \vec{b} = \begin{pmatrix} -2 \\ 0 \\ 1 \end{pmatrix}.$$
   Servez-vous des fonctions `decomp_DR` et `inv_diag` des questions 1. et 2.