Projet “Techniques d’Analyse Numérique” - L3 - 2021

Le travail sera à rendre sur Moodle le 21 mai 2021

Exercice 1 : Gauss-Legendre

Le but de cet exercice est dans un premier temps de programmer et de tester la méthode composite de Gauss-Legendre à trois points. Il s’agit ensuite de l’appliquer au calcul de projection sur une base de polynômes.

1. On considère une fonction régulière $f : [a, b] \to \mathbb{R}$. Rappeler les principes de la méthode composite de Gauss-Legendre à trois points pour évaluer $I(f) = \int_a^b f(x)dx$. On notera N le nombre de sous-intervalles, h le pas des sous-intervalles
$$h = \frac{b-a}{N}.$$
On notera $(\xi_i)_{i=0\dots2}$ les trois points de Gauss-Legendre sur l’intervalle $[-1, 1]$ et $(\omega_i)_{i=0\dots2}$ les poids correspondants. On notera $J_N(f)$ l’approximation obtenue par la méthode composite.
2. Écrire une fonction Python, nommée `intnum(a,b,N,f)` qui renvoie l’approximation $J_N(f)$ de $I(f)$.
3. On admet qu’il existe une constante $C > 0$ et un entier $\beta > 0$, tels que
$$e_N(f) = |I(f) - J_N(f)| \sim Ch^\beta.$$
4. Montrer que
$$\frac{\ln\left(\frac{e_N(f)}{e_{2N}(f)}\right)}{\ln 2} \to \beta \text{ quand } N \to \infty.$$
5. Application numérique : $f(x) = \exp(x)$, $[a, b] = [0, 1]$. Évaluer numériquement β. Combien faut-il en pratique de sous-intervalles N pour que $e_N(f) \leq 10^{-8}$.
6. Calculer P_0, P_1, P_2 et P_3 les quatre premiers polynômes de Legendre sur $[-1, 1]$. Calculer aussi L_i les polynômes de Legendre normalisés
$$L_i = \frac{P_i}{\langle P_i, P_i\rangle^{1/2}}.$$
7. Soit une fonction $g : [-1, 1] \to \mathbb{R}$, et soit Πg le projeté de g sur les premiers polynômes de Legendre
$$\Pi g = \sum_{i=0}^{3} \alpha_i L_i, \quad \alpha_i = \langle g, L_i\rangle.$$
En prenant $g(x) = \exp(x)$, calculer numériquement les valeurs des α_i avec une précision de 10^{-8} en utilisant le programme écrit à la question 2.
8. Tracer avec Python la courbe $y : x \in [-1, 1] \mapsto g(x) - (\Pi g)(x)$. Que constatez-vous ?

Exercice 2 : Modèle de Lotka-Volterra

On considère le modèle Lotka-Volterra qui modélise l’évolution d’une population de proies $x(t)$ et de prédateurs $y(t)$ au cours du temps t :

$$x' = ax - bxy,$$
$$y' = -cy + dxy.$$

Les constantes du modèle sont

$$a = \frac{\ln 2}{90}, \quad c = \frac{\ln 10}{30}, \quad b = \frac{a}{10}, \quad d = \frac{c}{100}.$$

1. Soit la fonction de deux variables

$$F(x, y) = -a \ln y - c \ln x + dx + by.$$

   Montrer que $F(x(t), y(t))$ est une constante.

2. Écrire deux fonctions Python `euler(T,x0,y0,N)` et `euler2(T,x0,y0,N)` qui calculent une approximation (x_N, y_N) de $(x(T), y(T))$ à l'instant T par les méthodes d'Euler explicite et d'Euler améliorée avec un pas de temps

$$\Delta t = T/N$$

   en partant de la condition initiale

$$x(0) = x_0, \quad y(0) = y_0.$$

3. Évaluer numériquement $F(x_N, y_N) - F(x_0, y_0)$ pour les méthodes `euler` et `euler2` et pour les paramètres $T = 100$ jours, $N = 100$, $x_0 = 100$, $y_0 = 50$. Conclusion ?

Exercice 3 : Méthode QR

1. Écrire une fonction Python `qr_dec(A)` qui renvoie la décomposition QR de A :

$$A = QR,$$

   où Q est une matrice unitaire et R une matrice triangulaire supérieure. On utilisera la méthode de Givens. Vérifier sur un exemple que le programme fonctionne.

2. La méthode QR est une méthode pour calculer les valeurs propres d'une matrice A. Elle consiste à construire une suite de matrices $(A_n)_{n \in \mathbb{N}}$ définie par

$$A_0 = A,$$

   puis

$$Q_n R_n = A_n, \quad A_{n+1} = R_n Q_n,$$

   où $Q_n R_n$ est la décomposition QR de A_n. On choisit

$$A = \begin{pmatrix} 2 & -1 & & \\ -1 & 2 & -1 & \\ & -1 & 2 & -1 \\ & & -1 & 2 \end{pmatrix}.$$

   Vérifier numériquement que A_n tend vers une matrice diagonale qui contient les valeurs propres de A. On comparera avec les valeurs propres données par la fonction Python `numpy.linalg.eig`