Feuille de TP n°2

Math3-Num2 — Méthodes numériques : LU, systèmes, EDO

Algèbre linéaire : méthodes directes, méthodes itératives

Exercice 1 – Résolution de systèmes triangulaires, décomposition LU.

1) Programmer deux fonctions, dont chacune prend en entrée une matrice A triangulaire inférieure (resp. triangulaire supérieure) et un vecteur b et qui résout le système $Ax = b$ par descente (resp. par remontée) et renvoie le vecteur solution x en sortie. Le tester sur une matrice triangulaire inversible choisie au hasard.

2) Vérifier l'hypothèse des mineurs principaux pour la matrice

$$M_1 = \begin{pmatrix} 2 & 1 & 1 & 0 \\ 4 & 3 & 3 & 1 \\ 8 & 7 & 9 & 5 \\ 6 & 7 & 9 & 8 \end{pmatrix}.$$

Que peut-on en déduire ?

3) Utiliser la décomposition LU pré-programmée en Python (commande `lu` de la bibliothèque `scipy.linalg`). Qu'obtient-on ? Utiliser cette décomposition et les fonctions de remonté et de descente programmées à la question précédente pour résoudre le système $Ax = b$, où $A = M_1$ et b est un vecteur ne contenant que des 1.

Exercice 2 – Décomposition de Cholesky.

1) Écrire l'algorithme qui permet de calculer la décomposition de Cholesky d'une matrice A. Pour cela, on remarquera que l'égalité $A = LL^T$ s'écrit coefficient par coefficient :

$$\begin{cases} l_{jj} = \sqrt{a_{jj} - \sum_{k=1}^{j-1} l_{jk}^2}, & j \geq 1, \\ l_{ij} = \frac{1}{l_{jj}} \left(a_{ij} - \sum_{k=1}^{j-1} l_{ik} l_{jk} \right), & j \geq 1, \quad i \geq j+1. \end{cases}$$

On pourra ensuite calculer les coefficients de L par colonne : de haut en bas puis de gauche à droite.

2) L"appliquer à la matrice

$$M_2 = \begin{pmatrix} 1 & 1 & -1 & 1 \\ 1 & 3 & -1 & 5 \\ -1 & -1 & 2 & -2 \\ 1 & 5 & -2 & 12 \end{pmatrix}.$$

Pour cela en commencera par vérifier que M_2 est symmétrique définie positive (on pourrait pour cela utiliser la commande `eig` de la bibliothèque `numpy.linalg`). Que calcule la commande `cholesky` de `numpy.linalg` ?

Exercice 3 – Étude comparée de méthodes itératives.

1) Soit

$$M_3 = \begin{pmatrix} 1 & 2 & -2 \\ 1 & 1 & 1 \\ 2 & 2 & 1 \end{pmatrix}.$$

Calculer grâce à numpy $\rho(J)$ et $\rho(G)$ et vérifier que $\rho(J) < 1 < \rho(G)$. Que peut-on déduire pour les méthodes de Jacobi et Gauss Seidel ?

2) Soit

$$M_4 = \begin{pmatrix} 2 & -1 & 1 \\ 2 & 2 & 2 \\ -1 & -1 & 2 \end{pmatrix}.$$

Calculer grâce à numpy $\rho(J)$ et $\rho(G)$. Que peut-on déduire pour les méthodes de Jacobi et Gauss Seidel ?

3) Implémenter une fonction `Jacobi` qui prend en entrée la matrice A, le second membre b, une tolérance ε et renvoie le vecteur x construit par la méthode de Jacobi, qui approche la solution de l'équation "à tolérance près" en choissisant un critère d'arrêt de résidu. Appliquer cette méthode au cas précédent qui vous paraît le plus judicieux.