Département de Mathématiques
Université de Nantes

Résolution du Laplacien 1D

On s'intéresse à la résolution du problème de Laplace 1D avec conditions aux bords de Dirichlet homogène

$$\begin{cases} -u''(x) = f(x), & x \in]a, b[, \\ u(x) = 0, & x = a \text{ et } x = b. \end{cases}$$

On discrétise l'intervalle $[a, b]$ $N+2$ points $(x_i)_{i=0,\ldots,N+1}$ tels que $x_i = a + ih$ où $h = \dfrac{b-a}{N+1}$.
Soit u_i une approximation de $u(x_i)$ et on note $f_i = f(x_i)$. On considère le schéma suivant :

$$\begin{cases} \dfrac{2u_i - u_{i+1} - u_{i-1}}{h^2} = f_i, \quad i = 1, \ldots, N \\ u_0 = u_{N+1} = 0. \end{cases}$$

En définissant $U = \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_N \end{pmatrix}$ et $F = \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_N \end{pmatrix}$, on s'amène à résoudre le problème linéaire $AU = F$ où la matrice A est la matrice du Laplacien.
La résolution numérique de ce problème nécessite le développement d'une classe de matrices et de vecteurs, de préférence creux, et d'implémenter toutes les méthodes (opérations d'addition, multiplication, produit, calcul de l'inverse...). Afin de gagner du temps, on propose d'utiliser une librairie d'algèbre linéaire existante. Il en existe beaucoup, compatibles avec le C++ (non nécessairement codées en C++) :

- `Eigen` (C++) : matrices denses et creuses, en séquentiel, facile à utiliser,
- `LAPACK` (Fortran) : Matrices denses, en séquentiel, difficile à utiliser,
- `PetSc` (C) : Matrices creuses, en parallèle, ardue.

On utilisera `Eigen`, à télécharger ici et à placer dans un dossier `EigenLib` de votre répertoire courant.
Pour compiler un fichier qui fait appel à `Eigen`, il est nécessaire de faire appel à une option de compilation. Les informations nécessaires sont détaillées ici.
Pour résoudre notre problème, on utilisera des méthodes disponibles sous `Eigen` que l'on comparera à une méthode codée par nos soins.
On propose de manipuler une classe `Laplacien1D` ayant la structure suivante :

- Attributs privés de la classe :
  - les bornes du domaines x_{min} et x_{max} de type `const double`
  - le nombre d'éléments N de type `const int`
  - le pas d'espace h de type `const double`
  - la matrice du laplacien 1D $LapMat$ de type `Eigen::SparseMatrix<double>` (creuse)
  - le vecteur des points de discrétisation X de type `Eigen::VectorXd` (dense)

  - le vecteur source F de type `Eigen::VectorXd` (dense)
  - le vecteur solution U de type `Eigen::VectorXd` (dense)
  - l'erreur en norme infinie $\|U - U_{\mathsf{ex}}\|_\infty$ (on calculera une solution exacte pour une source donnée), de nom err et de type `double`

- Méthodes et opérateurs de la classe :
  - un constructeur `Laplacien1D(const double x_min, const double x_max, const int N);` pour initialiser x_{min}, x_{max}, N et h
  - une méthode `void MatLaplacien();` pour construire la matrice du laplacien
  - une méthode `void TermeSource();` pour construire la source F
  - une méthode `void SolveurDirect();` qui résout le système par une méthode directe
  - une méthode `void SolveurIteratif();` qui résout le système par une méthode itérative
  - une méthode `double CalcErreur();` qui calcule et renvoie l'erreur
  - une méthode `void Save(std::string nom_du_fichier);` qui écrit la solution dans un fichier lisible par Gnuplot

Questions et remarques :

- Il s'agit de compléter les fonctions membres de la classe.
- Pour valider les calculs, on définit le terme source à partir de la solution exacte

$$u(x) = (x - a)(x - b)e^{-x}.$$

- L'erreur correspond à la norme infinie entre la solution approchée et la solution exacte.
- La résolution du système linéaire sera faite par deux méthodes différentes dont une itérative (on laisse le choix de la méthode à implémenter).
- Le rapport doit contenir une explication des méthodes choisies (algorithme, avantage, efficacité, limite...).
- Afficher la solution approchée et la solution exacte avec `Gnuplot` pour $N = 10, 100, 500$.
- Comparer les solveurs en terme de temps de calcul en utilisant la librairie `chrono` (à ajouter).
- Calculer l'ordre de la méthode.
- Rédiger un rapport expliquant votre code, son utilisation (et l'utilisation des librairies utilisées), les méthodes numériques choisies, contenant des illustrations numériques commentées.
- Joindre votre code au rapport.