MM031 - TP9
Matrice et structure *creuse*

Maxime Chupin : chupin@ann.jussieu.fr

9 mars 2016

La structure `sparse CSR` pour les matrices creuses

Dans de nombreuses simulations numériques, la discrétisation du problème aboutit à manipuler une matrice de très grande taille (d'ordre pouvant aller jusqu'à 10^8) mais dont la plupart des coefficients sont nuls. Dans ce cas, pour optimiser l'espace mémoire nécessaire au stockage de la matrice on introduit la notion de stockage `creux` qui contient uniquement les coefficients non nuls.
Le stockage CSR utilise 2 tableaux d'entiers (`row`, `col`) qui contiennent l'information des lignes et colonnes dont les coefficients sont non nuls et un tableau de nombres réels contenant les valeurs non nulles rangées par ligne (`val`). Soit `i` l'indice de la i ème ligne de la matrice, on a :

- (`row[i+1]` - `row[i]`) = nombre de valeurs non nulles pour la i ème ligne.
- de `col[row[i]]` à `col[row[i+1] -1]` : indices colonnes des coefficients non nuls pour la ligne i.
- de `val[row[i]]` à `val[row[i+1] -1]` : valeurs non nulles des coefficients rangées dans le même ordre que les indices de colonne pour la ligne i.

Exemple
Soit la matrice

$$\begin{pmatrix} 1 & -2 & 0 & 0 & 0 \\ -4 & 1 & -2 & 0 & 0 \\ 0 & 2 & 5 & 0 & 2 \\ 0 & 0 & 1 & -3 & 3 \\ 8 & 0 & 0 & 2 & 1 \end{pmatrix}$$

Le stockage CSR équivalent est le suivant :

$$\begin{aligned} row &= \{0, 2, 5, 8, 11, 14\} \\ col &= \{0, 1, 0, 1, 2, 1, 2, 4, 2, 3, 4, 0, 3, 4\} \\ val &= \{1, -2, -4, 1, -2, 2, 5, 2, 1, -3, 3, 8, 2, 1\} \end{aligned}$$

1. Définir une classe matrice *creuse* basée sur le stockage CSR en utilisant des éléments de la classe vector pour les trois tableaux. La classe comprendra :
   - un constructeur par défaut,
   - un constructeur construisant la matrice identité,
   - un constructeur prenant en argument 3 élément de la classe vector définissant `row`, `col` et `val`.
   - un constructeur par copie,
   - une méthode d'affichage,
   - une surcharge de l'opérateur d'égalité,
   - une fonction d'ajout d'élément à la matrice avec restructuration si besoin,
   - des fonctions d'accès aux valeurs de la matrice (éventuellement en surchargeant l'opérateur ()).
2. Ajouter une méthode pour effectuer le produit matrice-vecteur.
3. Ajouter une méthode de calcul de la somme de deux matrices de profil creux identique.
4. (Bonus) Utiliser la notion de `template` pour faire varier le type de données stockées dans la matrice (`int`, `double`, ...).

Solveurs

1. Implémenter la méthode LU en utilisant la classe matrice définie précédemment.
2. De même implémenter la méthode du Gradient-conjugué.