Algèbre Linéaire Creuse : méthodes directes

Exercice 3 du partiel 2022

voir sujet sous Moodle

Nous allons travailler ensemble cet exercice à la fois "sur papier" et en Matlab en intercalant, entre les questions 4 et 5, l'application de l'algorithme du *degré minimum* et son résultat sur le remplissage de la matrice lors de la factorisation.

Algèbre Linéaire Creuse : méthodes directes

On cherche à résoudre le système linéaire : $Ax = b$ avec A une matrice **creuse**.

Dans tous les exercices, on identifiera la stratégie de réordonnancement la plus efficace.

Le critère consiste à minimiser le nombre de non-zéros dans les facteurs de matrice (**vous justifierez ce critère dans le rapport à rendre**).

1 Nombre d'opérations de la phase de résolution

Étant donné le facteur L d'une factorisation de Cholesky d'une matrice symétrique définie positive ($A = L.L^T$), calculer le nombre total d'opérations flottantes nécessaires à la résolution des deux systèmes linéaires triangulaires $L.y = b$ puis $L^T.x = y$.

2 Qualité de la solution

Après chaque résolution, pour évaluer la qualité de la solution calculée, on calculera l'erreur inverse «normwise» :

$$\begin{aligned} \eta^N_{A,b}(\widetilde{x}) &= min\{\varepsilon : (A + \Delta A)\widetilde{x} = b + \Delta b, \\ &\qquad \|\Delta A\| \leq \varepsilon\|A\|, \|\Delta b\| \leq \varepsilon\|b\|\} \\ &= \frac{\|b - A\widetilde{x}\|}{\|A\|\|\widetilde{x}\| + \|b\|} \end{aligned}$$

Pour calculer la norme et le nombre de flops, vous pouvez écrire des fonctions dédiées.

3 Résolution du système linéaire symétrique avec une factorisation de Cholesky

On cherche à résoudre le système linéaire : $Ax = b$ où A est une matrice symétrique et $b = (1, 2, 3, \cdots)^T$.

Les matrices que vous avez à votre disposition sont les 4 matrices `mat0`, ..., `mat3` du précédent TP ainsi que la matrice `BCSSTK27`.

1. Résoudre le système avec une factorisation de Cholesky sans réordonnancement : nombres de non-zéros dans L, nombre d'opérations pour la résolution.
2. Chercher une permutation de la matrice permettant de diminuer le remplissage et le nombre d'opérations dans la résolution [1]. À chaque fois
   — visualiser la structure de A permutée et du L obtenu,
   — comptabiliser les non-zéros,
   — calculer le nombre d'opérations flottantes pour la résolution
   — vérifier que la solution est correcte (important !).

4 Résolution d'un système linéaire associé une matrice non symétrique avec une factorisation LU

On cherche à résoudre le système linéaire non symétrique : $Ax = b$ avec $b = (1, 2, 3, \cdots)^T$.

Les matrices que vous avez à votre disposition sont les matrices `hydcar20`, `pde225_5e-1` et `piston`.

1. Réaliser une résolution du système linéaire sans réordonnancement : nombres de non-zéros dans L et U, nombres d'opérations pour la résolution.
2. Chercher une permutation de la matrice permettant de diminuer le remplissage et le nombre d'opérations dans la résolution. Comme les matrices ne sont plus symétriques, on a maintenant la possibilité d'utiliser des permutations non-symétriques.
   À chaque fois
   — visualiser la structure de A permutée et des facteurs L et U obtenus,
   — comptabiliser les non-zéros,
   — calculer le nombre d'opérations flottantes pour la résolution
   — vérifier que la solution est correcte.

1. les différentes fonctions Matlab de ré-ordonnancement sont indiquées en annexe

Déposez sous Moodle le fichier tp.m complété (et si vous avez écrit des fonctions, les fichiers de ces fonctions) ainsi qu'un petit rapport reprenant les différents résultats pour les problèmes proposés en étudiant l'influence du réordonnancement

1. qualité de la solution,
2. mémoire,
3. flops

Ce rendu est à déposer sous Moodle pour le vendredi 29 Mars.

5 Instructions MATLAB utiles

Voici quelques instructions qui vous seront utiles pendant ce TP.

Faire avant toute chose un **help sparfun**.

- **load matrice** : lit sur disque matrice.mat
- **spy** : affiche la structure d'une matrice et donne le nombre de non-zéros
- **nnz(A)** : retourne le nombre de non-zéros dans la matrice **A**.
- **lu** : factorisation lu creuse
- **chol** : factorisation Cholesky creuse
- **amd, colamd, symamd, symrcm, colperm,** $\cdots$: algorithmes de réordonnancement (liste des possibles avec **help sparfun**)
- **C = A(p , p)** : applique la permutation symétrique **p** à la matrice **A** ; **C** reçoit la matrice permutée
- **C = A(: , p)** : applique la permutation **p** sur les colonnes de la matrice **A** ; **C** reçoit la matrice permutée
- **b(p)** : applique la permutation **p** sur le vecteur **b**
- **b(p) = bp** : dé-permute **bp** permuté avec **p** dans **b**.