LINMA1170 **Devoir 2 (v1)**
Analyse Numérique **Factorisations matricielles**
Prof. J-F Remacle & F. Henrotte **Echéance, le lundi 18 novembre 2019 (16h)**

Nous nous intéressons pour ce devoir à différents types de factorisations matricielles et à l'usage de ces décompositions pour résoudre des systèmes linéaires d'équations.

1 Factorisation LU

Implémentez l'algorithme de factorisation LU en place avec pivotage partiel sous forme d'une fonction PYTHON3 `LU(A)`. Cette fonction doit retourner une matrice `LU` contenant la factorisation et un vecteur de pivotage `P`. Ce dernier est un vecteur d'entiers de taille $m+1$ où le $(m+1)$ème élément contient le nombre de pivotages effectués (m est la taille de la matrice `A`). Indiquez comment évaluer à partir de ces éléments la matrice inverse de `A` ainsi que son déterminant.

Implémentez également l'algorithme de factorisation incomplète ILU(0) de la matrice `A`, sous la forme d'une fonction PYTHON3 `ILU0(A)` retournant la factorisation `ILU0` et un vecteur de pivotage `P`.

2 Factorisation QR

Implémentez l'algorithme de Gram-Schmidt réalisant la décomposition `QR` d'une matrice $A \in \mathbb{R}^{m\times n}$ avec $m \geq n$ sous forme d'une fonction PYTHON3 nommée `QR(A)`. Cette fonction doit retourner deux matrices `V` et `R`. La matrice $V \in \mathbb{R}^{m\times n}$ contient les vecteurs v_k normés (les w_k's) permettant de construire Q.

3 Résolution

Pour chacun des deux types de factorisation, implémentez une fonction PYTHON3 nommée respectivement `LUsolve(A,b,P)` et `QRsolve(A,b)` effectuant la résolution d'un système linéaire $A\mathbf{x} = \mathbf{b}$.

4 Analyse

- Evaluez la complexité temporelle de vos fonctions `QRsolve(A,b)` et `LUsolve(A,b,P)` en fonction de la taille du système linéaire dans le cas du régime **statique** défini pour le Devoir1.
- Utilisez ces deux fonctions pour résoudre le système linéaire généré par le modèle éléments finis `ndt.py` dans les 4 régimes du Devoir1 avec le même maillage. Comparez les temps de calculs et les solutions obtenues.
- Comparez le nombre de conditionnement de la matrice A à celui de la matrice $M^{-1}A$, où M est la factorisation LU incomplète de A. Considérez également les 4 régimes définis pour le Devoir1, mais un seul maillage.

Consignes

Ce devoir est un travail **personnel**.

Les implémentations sont à réaliser en PYTHON 3. Les librairies admises sont :

- NUMPY
- MATPLOTLIB
- TIME

Toute autre librairie (p.ex. SCIPY) n'est pas acceptée. Toutes les matrices utilisées seront des `numpy.array`. Veillez à fournir des implémentations lisibles, dûment commentées, avec des noms de variables explicites.

Les algorithmes à implémenter se trouvent dans le livre de référence du cours, ou dans le document `ILU(0).pdf` disponible sur Moodle. Les implémentations sont à soumettre sur le `Moodle` du cours d'Analyse Numérique pour le lundi 18 novembre 2019 en un seul fichier MYSOLVE.PY incluant vos fonctions `QRfactorize(A)` et `QRsolve(A,b)`.

Un rapport papier d'environ deux pages est également à remettre pour le mardi 18 novembre 2019 à 16h à l'Euler A.108 (Astrid Leduc). Le rapport ne doit pas contenir de page de garde, seulement une entête reprenant au moins le nom de l'auteur. Le rapport ne doit pas spécialement contenir de code source mais décrire brièvement votre démarche lors de l'implémentation des solveurs et de l'analyse numérique. Il contiendra en outre les résultats chiffrés de celle-ci.

La langue de rédaction est le français. Le rapport doit être réalisé avec LaTeX, avec la `documentclass article [11pt]` en `pagestyle plain`. Le .PDF et le source .TEX de ce rapport sont également à remettre sur `Moodle` sous forme d'un dossier (`.zip`) compressé.

Toutes les implémentations seront soumises à un logiciel anti-plagiat.