Sup Galilée **Méthodes numériques**
MACS 1 **Année 2008-2009**

TD/TP - 4
Factorisation LU par points et par blocs

But : 1) Ecrire la fonction *LUsansPivot* permettant de calculer la factorisation LU d'une matrice A inversible, puis résoudre un système linéaire en utilisant cette factorisation.
2) Ecrire la fonction *LUsansPivotBloc* permettant de calculer la factorisation LU par blocs d'une matrice A inversible.

0.1 Factorisation LU (par points)

Ecrire la fonction Matlab `LUsansPivot` dont l'entête est le suivant :

```
function [L,U]=LUsansPivot(A)
%
%    myLU : [L,U]=LUsansPivot(A)
%
%     But : calcule une matrice triangulaire inferieure L,
%           une matrice triangulaire superieure U telles que A=L*U,
%           en utilisant l'elimination de Gauss sans pivot.
%
```

Tester votre code sur une matrice de petite taille, puis sur la matrice A d'ordre n^2 issue de la discrétisation par un schéma aux différences finies du Laplacien en 2D (cf tp 3).

0.2 Résolution de systèmes linéaires

1) Ecrire une fonction qui prend en entrée L, U et un second membre b, et qui donne en sortie la solution x du système linéaire $Ax = b$ (L et U sont les matrices de la factorisation $A = LU$). On écrira un programme pour chaque résolution.

2) Résoudre le système linéaire $Ax = b$ où A est la matrice définie en 1), et b est un vecteur de $\mathbb{R}^{n^2}$ que l'on choisira. Vérifier que $\|b - Ax\|$ est de l'ordre du epsilon machine " *eps* ". Comparer votre résultat avec celui obtenu en utilisant l'opérateur de division à gauche "*mldivide* " de Matlab : $A\backslash b$.

0.3 Méthode LU par blocs

Refaire les questions des sections 1 et 2, dans le cas où la matrice est découpée en blocs :

$$A = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1p} \\ A_{21} & A_{22} & \cdots & A_{2p} \\ \vdots & & \ddots & \vdots \\ A_{p1} & A_{p2} & \cdots & A_{pp} \end{bmatrix}.$$

On suppose ici que les blocs diagonaux sont de même taille q, donc la matrice A est de taille pq.