Université de Thiès **Année Académique 2019-2020**
U.F.R S.E.S **Master 1 SDA**

PROJET D'ALGEBRE LINEAIRE

Membre du Groupe **: Souhoude OUEDRAOGO**

Ismael YODA

Lassana BA

Objectifs

L'objectif principal est d'écrire une fonction Matlab qui calcule les matrices L et U de la décomposition LU d'une matrice A. On appliquera ensuite cette décomposition au calcul du déterminant de la matrice A, puis à la résolution d'un système linéaire de la forme Ax = b, par un algorithme de type descente-remontée.

Décomposition LU

1) Avant de commencer, il nous faut quelques outils supplémentaires en Matlab. Afin de tester différentes fonctions, on se donne

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$$

a) Que renvoient size(A), size(A,1) et size(A,2) ? On pourra lire la documentation dans un premier temps (en tapant help size ou doc size) puis tester cette fonction sur la matrice A.

size(A), retourne la dimension de la matrice A (le nombre de ligne et de colonne)

size(A,1), retourne le nombre de ligne de la matrice A

size(A,2), retourne le nombre de colonne de la matrice A

b) Que renvoient A(:,1), A(2, :), A(1 :2,1 :2) et A(2 :end, :) ?

A(:,1), retourne les éléments de la première colonne de la matrice A

A(2, :), retourne les éléments de la deuxième ligne de la matrice A

A(1 :2,1 :2), retourne les éléments des deux premières ligne et deux premiers colonne de la matrice A.

A(2 :end, :), retourne les éléments de la matrice A à partir de la deuxième ligne et de tous les colonnes.

c) Que font les fonctions zeros, ones et eye ?

La fonction zeros(n) renvoie une matrice carrée de n, former uniquement de 0

La fonction ones(n) renvoie une matrice carrée de n, former uniquement de 1

La fonction eye(n) renvoie une matrice carrée In

2) Écrivons l'algorithme LU en pseudo-code.

```
Pour k = 1 a n
        Pour j = k a n
    U(k,j) = A(k,j) - L(k,1:k-1)*U(1:k-1,j);
    Fin
    Pour i = k+1 a n
    L(i,k) = ( A(i,k) - L(i,1:k-1)*U(1:k-1,k))/U(k,k);
    Fin
Fin
```

3) Ecrivons une fonction Matlab [L,U]=decomp_LU(M) qui réalise la décomposition LU d'une matrice M.

```
function [L,U] = decomp_lu(A)
   n = size(A,1);
   L = eye(n);
   U = zeros(n);

   for k=1:n
      for j=k:n
         U(k,j) = A(k,j) - L(k,1:k-1)*U(1:k-1,j);
      end
      for i=k+1:n
         L(i,k) = ( A(i,k) - L(i,1:k-1)*U(1:k-1,k))/U(k,k);
      end
   end
end
```

4) Calculer la décomposition LU de la matrice suivante :

$$B = \begin{pmatrix} 5 & 2 & 1 \\ 5 & -6 & 2 \\ -4 & 2 & 1 \end{pmatrix}$$

Apres résolution avec notre fonction decomp_LU(M) nous retrouvons les résultats suivants :

$$L = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ -0{,}8 & -0{,}45 & 1 \end{pmatrix}$$

$$U = \begin{pmatrix} 5 & 2 & 1 \\ 0 & -8 & 1 \\ 0 & 0 & 2{,}25 \end{pmatrix}$$

Résolution directe d'un système linéaire

1) a) Que renvoie sum(d), où d est un vecteur ?

sum(d), renvoie la sum de tous les éléments du vecteur d.

b) Soit a = $(1\ 2\ 3\ 4)^T$ et c = (5 6 7 8), calculons $\sum_{k=1}^{3} a_k c_k$

Sur Matlab en utilisant ce code : sum (sum (transpose(a(1:3)) * c(1:3)))
nous retrouvons le résultat suivant : 108

2) a) Écrivons une fonction [y]=descente(L,b) qui renvoie la solution de Ly = b où L est triangulaire inférieure.

```
function [y] = descente(L, b)
   n = length(b);
   y = zeros(n,1);
   for i=1:n
      y(i) = (b(i) - L(i,1:i-1)*y(1:i-1,1))/L(i,i);
   end
end
```

b) Utilisons la fonction descente pour calculer la solution y de Ly = b, avec

$$L = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ -\frac{4}{5} & -\frac{9}{20} & 1 \end{pmatrix} \quad \text{et} \quad b = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

Nous retrouvons comme résultat ave la fonction descente sur Matlab : $\begin{pmatrix} 1 \\ 1 \\ 4{,}25 \end{pmatrix}$

Nous retrouvons ces mêmes résultats avec l'appel L\ b sur Matlab.

c) Écrivons une fonction [x]=remontee(U,y) qui résout Ux = y pour U triangulaire supérieure.

```
function [x] = remontee(U, y)
    n = length(y);
    x = zeros(n,1);

    for i=n:-1:1
        x(i) = (y(i) - U(i,i+1:n)*x(i+1:n,1))/U(i,i);
    end
end
```

d) Utilisons la fonction remontee pour calculer la solution y de Ux = y, avec

$$U = \begin{pmatrix} 5 & -8 & 1 \\ 0 & -8 & 1 \\ 0 & 0 & \frac{9}{4} \end{pmatrix} \quad \text{et} \quad b = \begin{pmatrix} 1 \\ 1 \\ \frac{17}{4} \end{pmatrix}$$

Nous retrouvons comme résultat ave la fonction remontee sur Matlab :
$\begin{pmatrix} -0{,}2222 \\ 0{,}1111 \\ 1{,}8889 \end{pmatrix}$

Nous retrouvons ces mêmes résultats avec l'appel U\ y sur Matlab.

e) En déduisons une fonction [x]=resol LU(A,b) qui résout Ax = b, pour A une matrice qui admet une décomposition LU.

```
function x = resolve_LU(A, b)
    tic
    [L U] = decomp_lu(A);
    y = descente(L, b);
    x = remontee(U, y);
    toc
end
```

f) Utilisons la fonction resol_LU Résoudre Bx = b, où B et b sont définies :

$$B = \begin{pmatrix} 5 & 2 & 1 \\ 5 & -6 & 2 \\ -4 & 2 & 1 \end{pmatrix} \text{ et } \; b = \begin{pmatrix} 1 \\ 1 \\ \frac{17}{4} \end{pmatrix}$$

Nous retrouvons comme résultat ave la fonction resol_LU sur Matlab :

$$\begin{pmatrix} -0{,}2222 \\ 0{,}1111 \\ 1{,}8889 \end{pmatrix}$$

Nous retrouvons ces mêmes résultats avec l'appel B\ b sur Matlab.