Université René Descartes
UFR de mathématiques et informatique

chapitre 1

Résolution des systèmes linéaires

Méthode de Gauss

Méthodes numériques 2003/2004 - D.Pastre
licence de mathématiques et licence MASS

Résolution des systèmes linéaires

Notations

$$\left| \begin{array}{l} a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + ... + a_{2n}x_n = b_2 \\ \quad ... \\ a_{n1}x_1 + a_{n2}x_2 + ... + a_{nn}x_n = b_n \end{array} \right. \quad \text{n équations} \quad \text{n inconnues}$$

$$A = \begin{bmatrix} a_{11} & a_{12} & ... & a_{1n} \\ a_{21} & a_{22} & ... & a_{2n} \\ ... & & & \\ a_{n1} & a_{n2} & ... & a_{nn} \end{bmatrix} \quad B = \begin{bmatrix} b_1 \\ b_2 \\ ... \\ b_n \end{bmatrix} \quad AX = B$$

Etude des solutions :

Si $dét(A) \neq 0$ (A régulière) solution unique

Exemple : $\left| \begin{array}{l} x + y = 3 \\ x + 2y = 5 \end{array} \right.$

Si $dét(A) = 0$ (A singulière) système dégénéré (impossible ou indéterminé)

Exemples : $\left| \begin{array}{l} 2x + 3y = 4 \\ 4x + 6y = 5 \end{array} \right.$ $\quad \left| \begin{array}{l} 2x + 3y = 4 \\ 4x + 6y = 8 \end{array} \right.$

Théorie

Expression des solutions par la règle de Cramer :

$x_k = \frac{dét_k(A)}{dét(A)}$ avec

$$dét_k(A) = \begin{vmatrix} a_{11} & ... & a_{1,k-1} & b_1 & a_{1,k+1} & ... & a_{1n} \\ a_{21} & ... & a_{2,k-1} & b_2 & a_{2,k+1} & ... & a_{2n} \\ ... & & & & & & \\ a_{n1} & ... & a_{n,k-1} & b_n & a_{n,k+1} & ... & a_{nn} \end{vmatrix}$$

Calcul théorique d'un déterminant

$$dét(A) = \sum_{i=1}^{n} (-1)^{i+j} \, a_{ij} \, m_{ij}$$

où m_{ij} est le déterminant de la sous-matrice obtenue en supprimant de A la $i^{ème}$ ligne et la $j^{ème}$ colonne

Exercice : évaluer le nombre N_n d 'opérations nécessaires pour calculer un déterminant en utilisant cette formule.
Aide : on cherchera d 'abord une relation de récurrence entre N_n et N_{n-1}.

Méthode de Gauss

Transformation de A en une matrice triangulaire supérieure

Exemple :

$$\left| \begin{array}{l} 2x + y - 4z = 8 \\ 3x + 3y - 5z = 14 \\ 4x + 5y - 2z = 16 \end{array} \right. \quad A = \begin{bmatrix} 2 & 1 & -4 \\ 3 & 3 & -5 \\ 4 & 5 & -2 \end{bmatrix} \quad B = \begin{bmatrix} 8 \\ 14 \\ 16 \end{bmatrix}$$

Notation : $\quad A = \left[\begin{array}{ccc|c} 2 & 1 & -4 & 8 \\ 3 & 3 & -5 & 14 \\ 4 & 5 & -2 & 16 \end{array} \right]$

1^{er} pivot : $\boxed{2}$
$2^{ème}$ ligne - $1^{ère}$ ligne $*$ 3/2
$3^{ème}$ ligne - $1^{ère}$ ligne $*$ 2

$$\left[\begin{array}{ccc|c} 2 & 1 & -4 & 8 \\ 0 & 3/2 & 1 & 2 \\ 0 & 3 & 6 & 0 \end{array} \right]$$

$2^{ème}$ pivot : $\boxed{3/2}$
$3^{ème}$ ligne - $2^{ème}$ ligne $*$ 2

$$\left[\begin{array}{ccc|c} 2 & 1 & -4 & 8 \\ 0 & 3/2 & 1 & 2 \\ 0 & 0 & 4 & -4 \end{array} \right]$$

$3^{ème}$ pivot : $\boxed{4}$

D'où : $\left| \begin{array}{l} 4z = -4 \\ \frac{3}{2}y - 1 = 2 \\ 2x + 2 + 4 = 8 \end{array} \right.$

$\boxed{\begin{array}{l} z = -1 \\ y = 2 \\ x = 1 \end{array}}$

Remarque : Toutes les matrices intermédiaires ont le même déterminant qui est donc égal à $2 * \frac{3}{2} * 4 = 12$

Autre façon de conduire les calculs

(ligne 1) / pivot $\boxed{2}$
(ligne 2) - (nouvelle ligne 1)$*3$
(ligne 3) - (nouvelle ligne 1)$*4$

$\boxed{\left. \begin{array}{ccc|c} 1 & 1/2 & -2 & 4 \\ 0 & 3/2 & 1 & 2 \\ 0 & 3 & 6 & 0 \end{array} \right.}$

(ligne 2) / pivot $\boxed{3/2}$
(ligne 3)-(nouvelle ligne 2)$*3$

$\boxed{\left. \begin{array}{ccc|c} 1 & \frac{1}{2} & -2 & 4 \\ 0 & 1 & \frac{2}{3} & \frac{4}{3} \\ 0 & 0 & 4 & -4 \end{array} \right.}$

(ligne 3) / pivot $\boxed{4}$

$\boxed{\left. \begin{array}{ccc|c} 1 & 1/2 & -2 & 4 \\ 0 & 1 & 2/3 & 4/3 \\ 0 & 0 & 1 & -1 \end{array} \right.}$

D 'où : $\left| \begin{array}{l} z = -1 \\ y = 4/3 - 2/3z \\ x = 4 - 1/2y - (-2)z \end{array} \right.$ $\boxed{\begin{array}{l} z = -1 \\ y = 2 \\ x = 1 \end{array}}$

Remarque : Le déterminant de A est égal au produit des pivots, soit $2 * \frac{3}{2} * 4 = 12$

$2^{ème}$ exemple

$$A = \begin{bmatrix} 2 & 1 & -4 \\ 4 & 2 & -7 \\ 2 & 1 & -1 \end{bmatrix}$$

la $1^{ère}$ étape donne : $\begin{bmatrix} 1 & 1/2 & -2 \\ 0 & 0 & 1 \\ 0 & 0 & 3 \end{bmatrix}$

Le système est impossible ou indéterminé

exemples

$$B = \begin{bmatrix} 8 \\ 15 \\ 9 \end{bmatrix} \rightarrow \begin{bmatrix} 4 \\ -1 \\ 1 \end{bmatrix}$$

$$z = \frac{1}{3} = -1$$

$$B = \begin{bmatrix} 8 \\ 15 \\ 5 \end{bmatrix} \rightarrow \begin{bmatrix} 4 \\ -1 \\ -3 \end{bmatrix}$$

$$\left\{ \begin{array}{l} z = -1 \\ y \ quelconque \\ x = 2 - y/2 \end{array} \right.$$

$3^{ème}$ exemple

$$A = \begin{bmatrix} 2 & 1 & -4 \\ 4 & 2 & -7 \\ 2 & 2 & -1 \end{bmatrix}$$

la $1^{ère}$ étape donne : $\begin{bmatrix} 1 & 1/2 & -2 \\ 0 & 0 & 1 \\ 0 & 1 & 3 \end{bmatrix}$

Il n 'y a plus qu'à échanger la $2^{ème}$ et la $3^{ème}$ligne

Résolution

2 phases :
- **éliminations** → système triangulaire équivalent
- **substitutions** → résolution

On suppose que A est de rang n

$\boxed{1}$ On suppose a_{11} non nul (sinon on fait un échange de lignes).
On résout la première équation par rapport à x_1 et on remplace dans les autres équations.

On obtient le système équivalent

$$\left|\begin{array}{l} a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n = b_1 \\ \qquad a_{22}^{(2)}x_2 + ... + a_{2n}^{(2)}x_n = b_2^{(2)} \\ \qquad ... \\ \qquad a_{n1}^{(2)}x_1 + a_{n2}^{(2)}x_2 + ... + a_{nn}^{(2)}x_n = b_n^{(2)} \end{array}\right.$$

avec $\left|\begin{array}{l} a_{ij}^{(2)} = a_{ij} - a_{i1}a_{1j}/a_{11} \\ b_i^{(2)} = b_i - a_{i1}b_1/a_{11} \end{array}\right.$ pour $2 \leq i,j \leq n$
et $\left| a_{i1}^{(2)} = 0 \right.$ pour $i \geq 2$

Itérations

On recommence avec le pivot $a_{22}^{(2)}$ supposé non nul sinon on fait un échange de lignes, etc ...

En posant $A_{.,n+1} = B$ et $A^{(1)} = A$, **à l'étape k**, avec le **pivot** $a_{k,k}^{(k)} \neq 0$, on a $A^{(k+1)} =$

$$\left[\begin{array}{ccccc|c} a_{11}^{(1)} & a_{12}^{(1)} & ... & a_{1k}^{(1)} & & & a_{1,n+1}^{(1)} \\ 0 & a_{22}^{(2)} & ... & a_{2k}^{(2)} & & & a_{2,n+1}^{(2)} \\ ... & & & ... & & & ... \\ 0 & & & a_{k,k}^{(k)} & & & a_{k,n+1}^{(k)} \\ 0 & & & 0 & a_{k+1,k+1}^{(k+1)} & ... & a_{k+1,n+1}^{(k+1)} \\ ... & & & ... & ... & & ... \\ 0 & & & 0 & a_{n,k+1}^{(k+1)} & ... & a_{n,n+1}^{(k+1)} \end{array}\right]$$

$a_{ij}^{(k+1)} = a_{ij}^{(k)} - a_{ik}^{(k)}a_{kj}^{(k)}/a_{kk}^{(k)}$
pour $k+1 \leq i \leq n$ et $k+1 \leq j \leq n+1$
$a_{ij}^{(k+1)} = 0$ pour $k+1 \leq i \leq n$ et $j = k$
$a_{ij}^{(k+1)} = a_{ij}^{(k)}$ sinon

Remarque : $d\acute{e}t(A) = \pm$ le produit des pivots.

Fin de la résolution

$\boxed{2}$ A la $(n-1)^{ème}$ étape, on a une matrice triangulaire supérieure

$$\underbrace{\begin{bmatrix} a_{11}^{(1)} & a_{12}^{(1)} & ... & a_{1k}^{(1)} & ... & a_{1,n}^{(1)} \\ 0 & a_{22}^{(2)} & ... & a_{2k}^{(2)} & ... & a_{2,n}^{(2)} \\ ... & & & ... & ... & \\ 0 & 0 & ... & a_{k,k}^{(k)} & ... & a_{k,n}^{(k)} \\ ... & & & ... & ... & \\ 0 & 0 & ... & 0 & 0 & a_{n,n}^{(n)} \end{bmatrix}}_{A^{(n)}} \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ ... \\ x_k \\ ... \\ x_n \end{bmatrix}}_{X} = \underbrace{\begin{bmatrix} a_{1,n+1}^{(1)} \\ a_{2,n+1}^{(2)} \\ ... \\ a_{k,n+1}^{(k)} \\ ... \\ a_{n,n+1}^{(n)} \end{bmatrix}}_{B^{(n)}}$$

On remonte facilement, en commençant par x_n

$x_n = a_{n,n+1}^{(n)}/a_{n,n}^{(n)}$
...
$x_i = (a_{i,n+1}^{(i)} - a_{i,i+1}^{(i)}x_{i+1} - ... - a_{i,j}^{(i)}x_j - ... - a_{i,n}^{(i)}x_n)/a_{i,i}^{(i)}$
...
$x_1 = (a_{1,n+1}^{(1)} - a_{1,2}^{(1)}x_{i+1} - ... - a_{1,j}^{(1)}x_j - ... - a_{1,n}^{(1)}x_n)/a_{1,1}^{(1)}$

Notation matricielle

Avec les matrices de Frobenius

$$G_k = \begin{bmatrix} 1 & & 0 & & & 0 \\ & ... & & & & \\ 0 & & 1 & & ... & 0 \\ & & -a_{k+1,k}^{(k)}/a_{k,k}^{(k)} & 1 & ... & 0 \\ & & ... & & & \\ 0 & & -a_{n,k}^{(k)}/a_{k,k}^{(k)} & 0 & ... & 1 \end{bmatrix}$$

de déterminant égal à 1 on a $A^{(k+1)} = G_k * A^{(k)}$

D'où $A^{(n)} = G_{n-1} * ... * G_2 * G_1 * A_{initiale}$

En posant $\quad L = (\prod_i G_i)^{-1}$ et $U = A^{(n)}$
on en déduit $\quad A_{initiale} = L * U$
avec L triangulaire inférieure
et R triangulaire supérieure
et $\quad d\acute{e}t(A_{initiale}) = d\acute{e}t(A^{(n)}) = \prod_k a_{k,k}^{(k)}$
$= $ produit des pivots

Algorithme

pour $k = 1$ à n

- si $a_{kk} = 0$ alors
  - s'il existe $i > k$ tel que $a_{ik} \neq 0$
  - alors échanger les lignes i et k
  - sinon | la matrice est singulière
  - | stop
- $\{$le pivot $a_{kk} \neq 0$ $\}$
- pour $i = k+1$ à n, retrancher à la ligne i
  - la nouvelle ligne k multipliée par a_{ik}/a_{kk}
    - (colonnes de k à n
    - ou même, seulement $k+1$ à n)

$x_n = a_{n,n+1}/a_{nn}$

pour $i = n-1$ à 1

$x_i = (a_{i,n+1} - \sum_{j=i+1}^{n} a_{ij} * x_j)/a_{ii}$

ou : en rangeant les valeurs des solutions dans la $(n+1)^{ème}$ colonne de A :

$a_{n,n+1} = a_{n,n+1}/a_{nn}$

pour $i = n-1$ à 1

$a_{i,n+1} = (a_{i,n+1} - \sum_{j=i+1}^{n} a_{ij} * a_{j,n+1})/a_{ii}$

Premier programme Scilab

```
A=[2,1,-4;3,3,-5;4,5,-2],
B=[8;14;16],
n=3

// copie de B dans la colonne n+1 de A
for i=1:n, A(i,n+1)=B(i), end

for k=1:n ,   // etape k
  for i=k+1:n ,   // nouvelle ligne i
    for j=[n+1:-1:k], // pourquoi pas [k:n+1] ?
      A(i,j) = A(i,j) - A(i,k)*A(k,j)/A(k,k),
    end, //
  end,
end,

// remontee, calcul des xi  dans le vecteur X
X(n)=A(n,n+1)/A(n,n),
for i=[n-1:-1:1],
    X(i)=A(i,n+1),
    for j=i+1:n, X(i) = X(i) - X(j)*A(i,j); end,
    X(i) = X(i)/A(i,i),
end,
```


Meilleur Scilab

La copie de B dans la colonne n+1 de A

```
for i=1:n, A(i,n+1)=B(i), end
```

peut être remplacée par `A(:,n+1) = B`

ou `A = [A,B]`

La nouvelle ligne i

```
for j=[n+1:-1:k],
     A(i,j) = A(i,j) - A(i,k)*A(k,j)/A(k,k),
end,
```

peut être remplacée par

```
A(i,:)=A(i,:)-A(i,k)*A(k,:)/A(k,k)
```

ou, mieux par

```
A(i,k:n+1)=A(i,k:n+1)-A(i,k)*A(k,k:n+1)/A(k,k)
```


puis l'ensemble des lignes i

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

par

```
A(k+1:n,k:n+1) =
 A(k+1:n,k:n+1) - A(k+1:n,k)*A(k,k:n+1)/A(k,k)
```

Dans la remontée,

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

peut être remplacé par

```
X(i) = (A(i,n+1)-A(i,i+1:n)*X(i+1:n))/A(i,i),
```

Pourquoi ne peut-on pas remplacer cette dernière boucle `for i= ...` ?

fonction Scilab

```
function X = gauss0(A,B)

n=size(A,'c') // calcul de la taille de A et
if ~ size(A,'r')==n     // verif carree
   then X='matrice non carree', return, end

// copie de B dans la colonne n+1 de A
A(:,n+1) = B // ou A = [A,B]

for k = 1 : n ,    // etape k
    if A(k,k)==0 then X='pivot nul', return end,

// nouvelles ligne i
    A(k+1:n,k:n+1) =
     A(k+1:n,k:n+1)-A(k+1:n,k)*A(k,k:n+1)/A(k,k)
end,

// remontee, calcul des xi dans le vecteur X
X(n) = A(n,n+1)/A(n,n),
for i=[n-1:-1:1],
   X(i) = (A(i,n+1)-A(i,i+1:n)*X(i+1:n))/A(i,i),
end,
```


Recherche de pivot non nul

```
if A(k,k)==0 then //recherche pivot non nul
  printf('pivot nul %d',k),
  if k<n then i=k+1,
            while(A(i,k)==0 & i<n ), i=i+1,
            end,
       else i=k,
  end,
  if A(i,k)==0 then X='matrice singuliere',
                 return
  else printf(
       'on echange les lignes %d et %d',k,i),
     aux=A(k,:),A(k,:)=A(i,:),A(i,:)=aux
     // ou mieux   A([k,i],:)=A([i,k],:),
  end,
end,


printf('pivot=%f',A(k,k)),
```


Calcul du déterminant

Si on a effectué p échanges de lignes

$$d\acute{e}t(A) = (-1)^p \, a_{11} \, a_{22}^{(2)} ... \, a_{nn}^{(n)}$$

Si le calcul n'a pas pu se poursuivre jusqu'au bout, car il n'y avait pas de pivot non nul $d\acute{e}t(A) = 0$

Cet algorithme permet ainsi de calculer, rapidement, le déterminant.
Inutile, donc de le calculer **avant**

programme complet
avec impressions intermédiaires
et calcul du déterminant

```
function X = gauss0(A,B)

n=size(A,'c') // calcul de la taille de A et verif carree
if ~ size(A,'r')==n then printf('matrice non carree'),end

//  copie de B dans la colonne n+1 de A
A(:,n+1) = B
print(6,A),

ech=1 // pour le nombre d'echanges (parite)

for k = 1 : n ,    // etape k
    if A(k,k)==0 then // recherche d'un pivot non nul
        printf('pivot nul a l''etape %d',k),
        if k<n then i=k+1,
                    while(A(i,k)==0 & i<n ), i=i+1,
                    end,
               else i=k,
        end,
        if A(i,k)==0 then X='matrice singuliere', return
          else printf('on echange les lignes %d et %d',k,i),
                          A([k,i],:)=A([i,k],:),
                          ech=-ech,
                          print(6,A),
        end,
    end,
```

```
   printf('pivot=%f',A(k,k)),

   for i = k+1 : n ,   // nouvelle ligne i
       A(i,k:n+1) = A(i,k:n+1) - A(i,k)*A(k,k:n+1)/A(k,k)
   end,
   // ou  mieux
   // A(k+1:n,k:n+1) =
   //         A(k+1:n,k:n+1)-A(k+1:n,k)*A(k,k:n+1)/A(k,k)

   printf('etape %d',k),
   print(6,A),
end,

de = ech,
for i=1:n, de=de*A(i,i), end,
// ou  mieux de = ech*prod(diag(A)),
printf('determinant=%f', de),

// remontee, calcul des xi dans le vecteur X
X(n) = A(n,n+1)/A(n,n),
for i=[n-1:-1:1],   // remontee : calcul des xi
    X(i) = (A(i,n+1) - A(i,i+1:n)*X(i+1:n))/A(i,i),
end,
printf('solution'),
```

Ce programme se trouve à l'adresse

http://

www.math-info.univ-paris5.fr/~pastre/meth-num/gauss0

Gauss $2^{ème}$ version

On remarque que dans la formule
$a_{ij}^{(k+1)} = a_{ij}^{(k)} - a_{ik}^{(k)} a_{kj}^{(k)} / a_{kk}^{(k)}$
on fait la division de $a_{kj}^{(k)}$ par $a_{kk}^{(k)}$ pour toutes les valeurs de i.

On va commencer par diviser la ligne k du pivot par le pivot $a_{kk}^{(k)}$ et retirer à la ligne i la nouvelle ligne k multipliée par $a_{ik}^{(k)}$

On calculera alors une nouvelle ligne k par

$$a_{kj}^{(k+1)} = a_{kj}^{(k)} / a_{kk}^{(k)}$$

pour $j = k$ (ou $k+1$) à $n+1$
en remarquant que $a_{kj}^{(k+1)} = 0$ si $j < k$
et que $a_{kk}^{(k+1)} = 1$
puis les nouvelles lignes i

$$a_{ij}^{(k+1)} = a_{ij}^{(k)} - a_{ik}^{(k)} a_{kj}^{(k+1)}$$

pour $i = k+1$ à n et
$j = k$ (ou $k+1$) à $n+1$ (ligne i)
en remarquant que pour $j < k$, $a_{ij}^{(k+1)} = 0$

Avec une seule matrice pour mémoriser tous les $A^{(k)}$
$a_{kj} = a_{kj}/a_{kk}$
puis $a_{ij} = a_{ij} - a_{ik}\, a_{kj}$
puisqu'il s'agit du **nouveau** a_{kj}

Pour la remontée, on économise aussi les divisions par les pivots.

De plus, x_n se trouve déjà dans $a_{n,n+1}$

Nouvel algorithme

pour $k = 1$ à n
- si $a_{kk} = 0$ alors
  - s'il existe $i > k$ tel que $a_{ii} \neq 0$
  - alors échanger les lignes i et k
  - sinon | la matrice est singulière
  - stop
- {le pivot $a_{kk} \neq 0$ }
- diviser la ligne k par a_{kk}
  - (pour les colonnes k à n ou,
  - variante $k+1$ à n)
- pour $i = k+1$ à n, retrancher à la ligne i
  - la nouvelle ligne k multipliée par a_{ik}
  - (colonnes de k à $n+1$ ou,
  - variante $k+1$ à $n+1$)

pour $i = n-1$ à 1
- (calcul des solutions dans $a_{i,n+1}$)
- $a_{i,n+1} = a_{i,n+1} - \sum_{j=i+1}^{n} a_{ij} * a_{j,n+1}$

Programmation

```
for k = 1 : n ,   // etape k
  if A(k,k)==0 then ...
  //nouvelle ligne k
  A(k,k:n+1) = A(k,k:n+1)/A(k,k),
  for i = k+1 : n ,   // nouvelle ligne i
    A(i,k:n+1) = A(i,k:n+1) - A(i,k)*A(k,k:n+1)
  end,
  ...
end,
```

Variante

```
for k = 1 : n ,   // etape k
  if A(k,k)==0 then ...
  //nouvelle ligne k
  A(k,k+1:n+1) = A(k,k+1:n+1)/A(k,k),
  for i = k+1 : n ,   // nouvelle ligne i
    A(i,k+1:n+1)=A(i,k+1:n+1)-A(i,k)*A(k,k+1:n+1)
  end,
  ...
end,
```


Exercice

Exécuter la variante avec l'exemple du début. Si A est la matrice finale, on définit alors les matrices suivantes :
L est la partie triangulaire gauche inférieure de A, diagonale comprise
U est la partie triangulaire droite supérieure de A, sans la diagonale dont tous les éléments sont égaux à 1.

Calculer le produit LU. Que constatez-vous ?

Peut-on généraliser le résultat pour une matrice A quelconque ?

Recherche du pivot maximal

On a intérêt à avoir des pivots les plus grands possibles, sinon ils peuvent devenir très petits et pris égaux à 0 en machine.
On choisira donc comme pivot les plus grand des a_{kl} en valeur absolue, et on échangera les lignes k et l

Programme

```
...
amax=0, l=k, // recherche du pivot maximal
for i=k:n
    absA=abs(A(i,k))
    if absA>amax then l=i, amax=absA, end
end
if amax==0 then X='matrice singuliere', return,
end
if l<> k then A([k,l],:)=A([l,k],:),
          // aux=A(k,:),A(k,:)=A(l,:),A(l,:)=aux
              ech=-ech,
end,
...
```


Complexité

Nombre d'opérations effectuées
(Gauss $2^{ème}$ version) :

nouvelle ligne k (ligne du pivot)

$$\left| \begin{array}{l} k = 1 : n \\ \quad i = k : n+1 \end{array} \right| \begin{array}{l} \text{1 division} \\ \text{soit au total} \end{array}$$

$\sum_{k=1}^{n}(n-k+2) = \frac{n(n+3)}{2} \simeq \frac{n^2}{2}$ divisions

nouvelles lignes i

$$\left| \begin{array}{l} k = 1 : n \\ \quad i = k+1 : n \\ \qquad j = n+1 : -1 : k \end{array} \right| \begin{array}{l} \text{1 multiplication} \\ \text{et 1 addition} \\ \text{soit au total} \end{array}$$

$\sum_{k=1}^{n}\sum_{i=k+1}^{n}(n-k+2) = \frac{n(n+1)(2n-1)}{6} \simeq \frac{n^3}{3}$

remontée

$$\left| \begin{array}{l} i = n-1 : -1 : 1 \\ \quad j = i+1 : n \end{array} \right| \begin{array}{l} \text{1 multiplication et 1 addition} \\ \text{soit au total} \end{array}$$

$\sum_{i=1}^{n-1}(n-i) = \frac{n(n-1)}{2} \simeq \frac{n^2}{2}$

Au total, de l'ordre de $\frac{n^2}{2}$ divisions, $\frac{n^3}{3}$ multiplications et additions, soit $\boxed{2\frac{n^3}{3}}$ opérations

Retour sur la notation matricielle

S'il y a des échanges de lignes, la formule avec les matrices de Frobenius sera modifiée en utilisant les matrices de transposition

$$P(i,j)_{k,l} = \left|\begin{array}{ll} 1 \text{ si} & (k=i \text{ et } l=j) \\ & \text{ou } (k=j \text{ et } l=i) \\ & \text{ou } (k=l \text{ et } k\neq i \text{ et } k\neq j \\ 0 \text{ sinon} & \end{array}\right.$$

Exercices

- Montrer que $P(i,j)A$ échange les lignes i et j de A et que $A\,P(i,j)$ échange les colonnes i et j
- Soit $E(i,j)$ telle que

$$E(i,j)_{k,l} = \left|\begin{array}{l} 1 \text{ si } k=i \text{ et } l=j \\ 0 \text{ sinon} \end{array}\right.$$

Montrer que $P(i,j) =$

$$I + E(i,j) + E(j,i) - E(i,i) - E(j,j)$$

- Montrer que si A est une matrice régulière, il existe une matrice de permutation P, une matrice triangulaire inférieure L et une matrice triangulaire supérieure U telles que $A = P\,L\,U$