SUP'COM

Ecole Supérieure des Communications de Tunis

Méthodes directes pour la résolution des systèmes algébriques

A. SAADA

Exercice

1- **Donnez le nombre d'opérations pour le calcul du déterminant d'une matrice d'ordre $n \geq 2$.**
2- **Soit un ordinateur qui fait 100 millions d'opérations par seconde. Combien d'années sont nécessaires pour qu'il arrive à calculer le déterminant d'une matrice d'ordre 20 par la méthode classique ? ☺**

Réponse :

1- Le déterminant d'une matrice d'ordre 2 nécessite deux multiplications et une addition (soustraction).
Le déterminant d'une matrice d'ordre 3 nécessite $3\big(2(\times)+1(\pm)\big)+2(\pm)=6(\times)+5(\pm)$.
On démontre par récurrence que le déterminant d'une matrice d'ordre n nécessite $n!\,(\times)+(n!-1)(\pm)$.
C'est en effet vérifié pour $n=2$. On suppose que c'est vrai pour n. Le déterminant d'une matrice d'ordre $(n+1)$ nécessite alors $(n+1)\big(n!\,(\times)+(n!-1)(\pm)\big)+n(\pm)=\Big((n+1)!\,(\times)+\big((n+1)!-1\big)(\pm)\Big)$. CQFD
2- Un tel déterminant nécessite 2*20 ! -1 = 4865804016353279999 opérations
Il faut 1541,88 années pour son calcul !

Rappels :

Soient $n\in\mathbb{N}^*$, $1\leq k<n$, $(\alpha_i)_{k+1\leq i\leq n}\in\mathbb{R}^{n-k}$, $A=\left(a_{ij}\right)_{1\leq i,j\leq n}\in M_n(\mathbb{R})$, $H^{(k)}\in M_n(\mathbb{R})$ avec $\left(H^{(k)}\right)_{ij}=\alpha_i\delta_{k<i}\delta_{jk}$ et $M^{(k)}=I_n+H^{(k)}$ où I_n est la matrice identité dans $M_n(\mathbb{R})$, δ_{jk} le symbole de Kronecker et $\delta_{k<i}=\begin{cases}0 & \text{si} \quad i\leq k\\ 1 & \text{si} \quad k<i\end{cases}$.

1- **Soit $\tilde{A}=M^{(k)}A$. Donnez les coefficients $\tilde{a}_{ij}$ de $\tilde{A}$ pour $1\leq i,j\leq n$.**
2- **Soit $1\leq l\leq n$, $l\neq k$ et soit $P^{(kl)}\in M_n(\mathbb{R})$, la matrice obtenue par permutation des lignes l et k de I_n et soit $\breve{A}=P^{(kl)}A$. Donnez les coefficients $\breve{a}_{ij}$ de $\breve{A}$ pour $1\leq i,j\leq n$.**

3- **Faites un programme Matlab qui calcule $\tilde{A}$ et $\breve{A}$, en fonction de $k, (\alpha_i)_{k+1 \leq i \leq n}$, et A. Donnez le nombre d'opérations élémentaires nécessaires pour les obtenir.**

Réponses :

1- Pour $i \leq k, \tilde{a}_{ij} = a_{ij}$. Pour pour $k < i, \tilde{a}_{ij} = \alpha_i a_{kj} + a_{ij}$. Autrement dit, si L_p représente la p-ème ligne de A et $\tilde{L}_p$ la p-ème ligne de $\tilde{A}$ alors $\tilde{L}_i = \begin{cases} L_i & \text{si} \quad i \leq k \\ \alpha_i L_k + L_i & \text{si} \quad k < i \end{cases}$

2- En prenant la même notation que la question précédente nous avons $\breve{L}_i = \begin{cases} L_i & \text{si} \quad i \notin \{k,l\} \\ L_k & \text{si} \quad i = l \\ L_l & \text{si} \quad i = k \end{cases}$. C'est donc une permutation entre les lignes l et k. Remarque : $A\,P^{(kl)}$ correspond à une permutation entre les colonnes l et k.

3- Il faut absolument éviter de programmer l'opération. Il faut juste programmer le résultat. Pour $\tilde{A}$ le programme est :

```
function [ A ] = combinaisonsLineairesDeLignesDeMatrice( n,k,alpha,A )
%
% n est l'ordre de A
% alpha un vecteur de longueur (n-k) dont le p-ème coefficient sont alpha_p
% Soit L_p est la p-ème ligne de la matrice A à l'entrée
% La matrice A à la sortie a les mêmes lignes de A pour p < k+1,
% et alpha_p L_(p+k) + L_k pour la p-ème ligne avec p=1...(n-k)
%
for p=1:n-k
    A(k+p,:)=alpha(p)*A(k,:)+A(k+p,:);
end
end
```

Le nombre d'opérations est $(n-k) * \big(n(\times) + n(+)\big)$

Pour $\breve{A}$ le programme est :

```
function [ A ] = permuationDesLignesLetKduneMatrice( k,l,A )
%
% Permutation entre les lignes l et k de la matrice A
%
c = A(l,:);
A(l,:)=A(k,:);
A(k,:)=c;

end
```

Cette opération ne nécessite aucune opération.

Méthode de Gauss pour la résolution de (S) :

Soient $n \in \mathbb{N}^*$, $A = A^{(0)} \in M_n(\mathbb{R})$ inversible et $b \in \mathbb{R}^n$.

1- **Supposons que $a_{11}^{(0)} \neq 0$. Construisez $M^{(1)}$ telle que $A^{(1)} = M^{(1)} A^{(0)}$ vérifie :**
$$\begin{cases} a_{1j}^{(1)} = a_{1j}^{(0)} & \text{pour} \quad 1 \leq j \leq n \\ a_{i1}^{(1)} = 0 & \text{pour} \quad 1 < i \leq n \end{cases}$$

2- **Montrez que $\boldsymbol{A^{(0)}x = b \Leftrightarrow A^{(1)}x = M^{(1)}b}$**

3- **Supposons que $\boldsymbol{a_{11}^{(0)} = 0}$. Montrez qu'il existe $\boldsymbol{i} > 1$ tel que $\boldsymbol{a_{i1}^{(0)} \neq 0}$. Monter alors que $\boldsymbol{A^{(0)}x = b \Leftrightarrow P^{(1i)}A^{(0)}x = P^{(1i)}b}$**

4- **Supposons que $\boldsymbol{a_{22}^{(1)} \neq 0}$. Construire $\boldsymbol{M^{(2)}}$ telle que $\boldsymbol{A^{(2)} = M^{(2)}A^{(1)}}$ vérifie : $\begin{cases} \boldsymbol{a_{2j}^{(2)} = a_{2j}^{(1)}} & \textbf{pour} \quad \boldsymbol{2 \leq j \leq n} \\ \boldsymbol{a_{i2}^{(2)} = 0} & \textbf{pour} \quad 2 < i \leq n \end{cases}$. Donnez les valeurs de $\boldsymbol{a_{i1}^{(2)}, i = 2, \dots, n}$.**

5- **Déduisez un algorithme qui permet de remplacer la résolution de $\boldsymbol{A^{(0)}x = b}$ par celle de $\boldsymbol{Tx = c}$ où $\boldsymbol{T \in M_n(\mathbb{R})}$ est une matrice triangulaire supérieure et $\boldsymbol{c \in \mathbb{R}^n}$ à expliciter.**

6- **Donnez une méthode pour résoudre $\boldsymbol{Tx = c}$.**

7- **Mettez en œuvre un code pour la résolution de $\boldsymbol{Ax = b}$ utilisant les algorithmes des deux questions précédentes pour $\boldsymbol{n, A}$ et $\boldsymbol{b}$ quelconques. Donnez le nombre d'opérations élémentaires nécessaires pour les deux algorithmes.**

Réponses :

1- $M^{(1)} = I_n + H^{(1)}$ avec $\left(H^{(1)}\right)_{ij} = -\frac{a_{i1}^{(0)}}{a_{11}^{(0)}}\delta_{1<i}\delta_{j1}$

2- $det\left(M^{(1)}\right) = 1 \Longrightarrow \left(A^{(0)}x = b \Leftrightarrow M^{(1)}A^{(0)}x = M^{(1)}b \Leftrightarrow A^{(1)}x = M^{(1)}b\right)$

3- Si $a_{i1} = 0 \; \forall 1 \leq i \leq n$ alors $det\left(A^{(0)}\right) = 0$ en contradiction avec $A^{(0)}$inversible. $det\left(P^{(\boldsymbol{1i})}\right) = \pm 1$ donc inversible donc $A^{(0)}x = b \Leftrightarrow P^{(1i)}A^{(0)}x = P^{(1i)}b.$

4- On pose $M^{(2)} = I_n + H^{(2)}$ avec $\left(H^{(2)}\right)_{ij} = -\frac{a_{i1}^{(1)}}{a_{11}^{(1)}}\delta_{2<i}\delta_{j2}$. $a_{i1}^{(2)} = 0 \; \forall i = 2, \dots, n.$

5- Pour i allant de 1 à $(n-1)$

Soit $i_0 / \left|a_{i_0 i}^{(i-1)}\right| = \max_{i \leq k \leq n}\left|a_{ki}^{(i-1)}\right|$

$.A^{(i)} = P^{(ii_0)}A^{(i)}$

$.b = P^{(ii_0)}b$

$.M^{(i)} = I_n + H^{(i)}$ avec $\left(H^{(i)}\right)_{lm} = -\frac{a_{li}^{(i)}}{a_{ii}^{(i)}}\delta_{i<l}\delta_{mi}.$

$.b = M^{(i)}b$

Fin Pour

6- Remarquons d'abord que $det(T) = \pm det(A) \Longrightarrow T_{ii} \neq 0 \; \forall \; 1 \leq i \leq n.$

$$Tx = c \Longrightarrow x_n = \frac{c_n}{T_{nn}} \Longrightarrow x_{n-1} = \frac{c_{(n-1)} - T_{(n-1)n}x_n}{T_{(n-1)(n-1)}} \Longrightarrow \cdots \Longrightarrow x_1 = \frac{c_1 - \sum_{j=2}^{n} T_{1j}x_j}{T_{11}}$$

7- Le code comporte une fonction de lecture des données $A \; et \; b$, une fonction de résolution qui utilise la méthode de Gauss avec pivot partiel et une fonction de visualisation de la solution. A la résolution, nous faisons appel aux fonctions, de permutation et de combinaison linéaires entre deux lignes d'une matrice, développées ci-dessus :

```
%
Programme principal
% le programme principal est foirmée de 3 étapes:
% 1- Lecture des données
% 2- Resolution du problème
% 3- Illustration des résultats
%
clear all
close all
% 1- Lecture des données
[n,A,b]=donnees();
%
% 2- Résolution du problème
[x]=resolution(n,A,b);
%
3- Illustration des résultats
illustration(x);

function [n,A,b]=donnees()
%
% Fonction de lecture des données
%
A=[1 3 3 4; 5 6 -7 8; 9 10 7 -12; 13 -14 15 16];
b=[10;2;3;4];
n=size(b,1);
end

function [x]=resolution(n,A,b);
%
% Programme pour la résolution de Ax=b par la méthode de Gauss
%
% Trigonalisation de la matrice
[T,c]=gauss(n,A,b);
prod(diag(T))
%
% Resolution par remontée
x=remontee(n,T,c);
end

function [A,b]=gauss(n,A,b)
%
% Trigonalisation de la matrice A et modifocation du second membre b
%
for i=1:n-1
    [coef,i0]=max(abs(A(i:n,i)));
    [ A ] = permuationDesLignesLetKduneMatrice( i,i+i0-1,A );
    [ b ] = permuationDesLignesLetKduneMatrice( i,i+i0-1,b );
    alpha =-A(i+1:n,i)/A(i,i);
    [ A ] = combinaisonsLineairesDeLignesDeMatrice( n,i,alpha,A );
    [ b ] = combinaisonsLineairesDeLignesDeMatrice( n,i,alpha,b );
end

function x=remontee(n,T,c);
%
% Resolution de Tx=c (T triangulaire superieure) par une remontée
%
x=sparse(n,1);
for i=n:-1:1
    x(i,1)=(c(i,1)-T(i,i+1:n)*x(i+1:n,1))/T(i,i);
end
```

```
function illustration(x)
%
% visualisation des valeurs algébriques composantes d'un vecteur par un
% histogramme
%*****************************************************************
bar(x);
title('solution du système linéaire');
xlabel('indice des coefficients de x');
ylabel('valeurs des coefficients de x');
end
```

Le nombre d'opérations est celui de la méthode de Gauss et de la remontée.

Méthode LU pour la résolution de (S) :

8- Supposons que pour $a_{(i+1)(i+1)}^{(i)} \neq 0$ pour $i = 1, \dots, n-2$. Montrez alors qu'il existe $L \in M_n(\mathbb{R})$ triangulaire inférieure à diagonale unité et $U \in M_n(\mathbb{R})$, triangulaire supérieure, tels que $A = LU$.

9- Effectuer le produit LU et donner par identification avec $A = A^{(0)}$ les coefficients de la première ligne de U et de la première colonne de L.

10- Calculez $LU - (l_{i1})_{1\le i\le n}(u_{1j})_{1\le j\le n}$, puis $A^{(1)} = A^{(0)} - (l_{i1})_{1\le i\le n}(u_{1j})_{1\le j\le n}$. En déduire les valeurs de la deuxième ligne de U et de la deuxième colonne de .

11- Développez un algorithme pour le calcul des coefficients de U puis ceux de L en se basant sur les deux questions précédentes, et procédez à sa mise en œuvre.

12- Donnez le nombre d'opérations élémentaires pour faire la décomposition.

13- Que faut-il faire si un des coefficients $a_{(i+1)(i+1)}^{(i)}$ pour $i = 1, \dots, n-2$ est nul ?

14- Comment utiliser cette méthode pour résoudre $Ax = b$?

8- Si $a_{(i+1)(i+1)}^{(i)} \neq 0$ pour $i = 1, \dots, n-2$, alors d'après la deuxième question nous avons : $A^{(0)}x = b \iff Ux = M^{(n-1)}M^{(n-2)} \dots M^{(1)}A^{(0)}x = M^{(n-1)}M^{(n-2)} \dots M^{(1)}b$ $\implies U = M^{(n-1)}M^{(n-2)} \dots M^{(1)}A^{(0)}$avec T matrice triangulaire supérieure et $M^{(n-1)}M^{(n-2)} \dots M^{(1)} = \tilde{L}$ matrice triangulaire inférieure à diagonale unité. $\tilde{L}$ est inversible et son inverse est une matrice L triangulaire inférieure à diagonale unité $\implies$ $A = A^{(0)} = LU$.

9-

$$\begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1n} \\ l_{21}u_{11} & l_{21}u_{12} + u_{22} & \cdots & l_{21}u_{1n} + u_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ l_{n1}u_{11} & \begin{matrix} l_{n1}u_{12} \\ + l_{n2}u_{22} \end{matrix} & \cdots & \sum_{k=1}^{n} l_{nk}\,u_{kn} \end{bmatrix} = \begin{bmatrix} a_{11}^{(0)} & \cdots & \cdots & a_{11}^{(0)} \\ a_{11}^{(0)} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ a_{1n}^{(0)} & \cdots & \cdots & a_{nn}^{(0)} \end{bmatrix} = A^{(0)}$$

$\Longrightarrow u_{1j} = a_{1j}^{(0)}$ pour $1 \leq j \leq n$ et $\Longrightarrow u_{i1} = \frac{a_{i1}^{(0)}}{a_{11}^{(0)}}$ pour $2 \leq i \leq n$.

10- $(Lu)_{pq} = \sum_{k=1}^{n} l_{pk}\, u_{kq}$; $\left((l_{i1})_{1\leq i\leq n}(u_{1j})_{1\leq j\leq n}\right)_{pq} = l_{p1}u_{1q} \Longrightarrow$

$\left(LU - (l_{i1})_{1\leq i\leq n}(u_{1j})_{1\leq j\leq n}\right)_{pq} = \sum_{k=2}^{n} l_{pk}\, u_{kq} = \sum_{k=2}^{min(p,q)} l_{pk}\, u_{kq}$ et $a_{pq}^{(1)} = \left(A^{(0)} - (l_{i1})_{1\leq i\leq n}(u_{1j})_{1\leq j\leq n}\right)_{pq} = a_{pq}^{(0)} - l_{p1}u_{1q}$, autrement dit :

$$\begin{bmatrix} 0 & 0 & \dots & \dots\ \dots & 0 \\ 0 & u_{22} & u_{23} & \dots\ \dots & u_{2n} \\ \vdots & l_{32}u_{22} & \sum_{k=2}^{3} l_{3k}\, u_{k3} & & \vdots \\ \vdots & \vdots & & \ddots & \vdots \\ \vdots & \vdots & & \ddots & \vdots \\ 0 & l_{n2}u_{22} & \dots & \dots\ \dots & \sum_{k=2}^{n} l_{nk}\, u_{kn} \end{bmatrix} = \begin{bmatrix} 0 & 0 & \dots\ \dots\ \dots & 0 \\ 0 & a_{22}^{(0)} - l_{21}u_{12} & \dots\ \dots\ \dots & a_{2n}^{(0)} - l_{21}u_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ 0 & a_{n2}^{(0)} - l_{n1}u_{12} & \dots\ \dots\ \dots & a_{nn}^{(0)} - l_{n1}u_{1n} \end{bmatrix} = A^{(1)}$$

11-
```
% Pour i allant de 1 à n-1 faire
%    U(i,i:n) = A(i,i:n);
%    L(i+1:n,i) = A(i+1:n,i)/U(i,i)
%    A(i:n,i:n) = A(i:n,i:n)-[1,L(i+1:n,i)]*U(i,i:n);
% fin pour i
```

```
function   A=LU(A);
%
% Décomposition de A en produit de L (triangulaire inférieure à diagonale
% unité) et U (triangulaire supérieure). L est stockée dans la partie
% triangulaire inférieure stricte de A et U dans la partie triangulaire
% supérieur
%
n = size(A,1);
for i = 1:n-1
    % i-ème colonne de L à partir du coefficient i+1.
    A(i+1:n,i) = A(i+1:n,i)/A(i,i);
    A(i+1:n,i+1:n) = A(i+1:n,i+1:n)- [A(i+1:n,i)]*[A(i,i+1:n)];
End
```

12- Le calcul de la i-ème colonne de L à l'itération i nécessite $(n-i)$ divisions, et la détermination des coefficients de $A^{(i)}$ nécessite $(n-i)^2$ multiplications et $(n-i)^2$ soustractions. La décomposition coute alors $\sum_{i=1}^{n-1}(n-i) = \frac{n(n-1)}{2}$ divisions, et $\sum_{i=1}^{n-1}(n-i)^2 = \frac{n(n-1)(2n-1)}{6}$ multiplications et soustractions.

13- Si un des coefficients $a_{(i+1)(i+1)}^{(i)}$ est nul, il faut permuter la $(i+1)$-ème ligne de $A^{(i)}$ par une ligne dont le coefficient $a_{(i+1)k}^{(i)}, k > i+1$ est non nul. La fonction Matlab de décomposition devient :

```
function   [A,b]=LU(A,b);
%
% Décomposition de A en produit de L (triangulaire inférieure à
% diagonale unité) et U (triangulaire supérieure). L est stockée
% dans la partie triangulaire inférieure stricte de A et U dans la
% partie triangulaire supérieur
%
n = size(A,1);
for i = 1:n-1
    % Recherche du coefficient de plus grand module et permutation de
    % lignes
    %
    [coef,i0]=max(abs(A(i:n,i)));
    [ A ] = permuationDesLignesLetKduneMatrice( i,i+i0-1,A );
    [ b ] = permuationDesLignesLetKduneMatrice( i,i+i0-1,b );
    %
    % ieme colonne de L à partir du coefficient i+1.
    A(i+1:n,i)  = A(i+1:n,i)/A(i,i);
    A(i+1:n,i+1:n)  = A(i+1:n,i+1:n)-  [A(i+1:n,i)]*[A(i,i+1:n)];
end
```

14- Pour résoudre $LUx = b$, nous commençons par résoudre $Ly = b$ par une descente puis $Ux = y$ par une remontée. La code par Matlab de la descente est :

```
function x=descente(n,T,c);
%
% Resolution de Tx=c (T triangulaire inferieure) par une descente
%
x=sparse(n,1);
for i=1:n
    x(i,1)=(c(i,1)-T(i,1:i-1)*x(1:i-1,1))/T(i,i);
end
```

Méthode de Cholesky pour la résolution de (S) dans le cas où Aest symétrique définie positive :

Nous supposons A symétrique définie positive.

15- Montrer qu'il existe B triangulaire inférieure à diagonale positive, telle que $A = BB^t$.
16- Mettez en œuvre un programme qui puisse calculer B.
17- Donnez le nombre d'opérations élémentaires nécessaires pour le calcul de B.

15- Nous pouvons montrer par ce qui précède que si A est inversible nous pouvons la décomposer, à une multiplication par une matrice de permutation près, en un produit d'une matrice triangulaire inférieure à diagonale unité L et une matrice triangulaire supérieure L. $A = LU$. Nous pouvons poser $U = D\tilde{L}$ avec D la matrice diagonale formée par les éléments diagonaux de U et $\tilde{L}/\tilde{l}_{ij} = \frac{u_{ij}}{u_{ii}}$. Nous pouvons montrer que les coefficients u_{ii} sont tous strictement positifs (Exercice). Comme A est symétrique alors $A^T = \tilde{L}^T D L^T$, ce qui montre par unicité de la décomposition que $\tilde{L}^T = L$ et $A = LDL^T = BB^T$ avec $B = L\sqrt{D}$ une matrice triangulaire inférieure.

16- L'algorithme de décomposition de Cholesky ne change que très peu par rapport à la décomposition LU pour le calcul des coefficients de la matrice triangulaire inférieure qui n'est plus à diagonale unité. Le code par Matlab est :

```
function   [A]=cholesky(A);
%
% Décomposition de A symétrique définie positive en produit de BB^t
%(B triangulaire inférieure à diagonale positive)
%
n = size(A,1);
A= tril(A);
for i = 1:n-1
    %
    % Calcul du i-ème coefficient de la diagonale de B
    A(i,i) = sqrt(A(i,i));
    %
    % ieme colonne de L à partir du coefficient i+1.
    A(i+1:n,i) = A(i+1:n,i)/A(i,i);
    A(i+1:n,i+1:n) = A(i+1:n,i+1:n)...
        - tril(A(i+1:n,i)*ones(1,n-i))*diag(A(i+1:n,i));
end
%
% Calcul du n-ème coefficient de la diagonale de B
A(n,n)=sqrt(A(n,n));
```

17- Le calcul de B nécessite n racines, $\frac{n\,(n-1)}{2}$ divisions et $\frac{n^3-n}{6}$ multiplications et soustractions.