Analyse Numérique

Gérald MONARD

Université Henri Poincaré - Nancy I

D.E.A. de Chimie Informatique et Théorique

Année Universitaire 2004 - 2005

Références :

– *Matrix computations*, Gene H. Golub et Charles F. Van Loan, Johns Hopkins Univ. Pr., USA.
– *Numerical Recipes in C : The Art of Scientific Computing*, William H. Press, Saul A. Teukolsky, William T. Vetterling, Brian P. Flannery, Cambridge Univ. Pr., USA.

Chapitre 1

Opérations sur les matrices

1.1 Rappels mathématiques

– Soit $\mathbf{x} = (x_1, x_2, \cdots, x_n)$ un élément de $\mathbb{R}^n$, $\mathbf{x}$ est appelé *vecteur* de $\mathbb{R}^n$ et on le note :

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

– Soit f une fonction de $\mathbb{R}^n$ dans $\mathbb{R}^m$, f est dite *linéaire* si et seulement si :

$$\forall\ \mathbf{x} = (x_1, x_2, \cdots, x_n) \in \mathbb{R}, \qquad \forall\ \lambda \in \mathbb{R}, \qquad f(\lambda.\mathbf{x}) = \lambda.f(\mathbf{x})$$

et

$$\forall\ \mathbf{x} \in \mathbb{R}^n, \quad \forall\ \mathbf{y} \in \mathbb{R}^n, \quad f(\mathbf{x}+\mathbf{y}) = f(\mathbf{x}) + f(\mathbf{y})$$

– Soit $\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_m$ un ensemble de m vecteurs de $\mathbb{R}^n$,

1. $\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_m$ sont dits *linéairement indépendants* si et seulement si :

$$\exists\ \lambda_1, \lambda_2, \cdots, \lambda_m \in \mathbb{R}, \qquad \lambda_1\mathbf{u_1} + \lambda_2\mathbf{u_2} + \cdots \lambda_m\mathbf{u_m} = 0$$
$$\Updownarrow$$
$$\lambda_1 = \lambda_2 = \cdots = \lambda_m = 0$$

2. $\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_m$ sont dits *générateurs* de $\mathbb{R}^n$ si et seulement si :

$$\forall\ \mathbf{x} \in \mathbb{R}^n, \qquad \exists\ \lambda_1, \lambda_2, \cdots, \lambda_m \in \mathbb{R}, \qquad \mathbf{x} = \lambda_1\mathbf{u_1} + \lambda_2\mathbf{u_2} + \cdots \lambda_m\mathbf{u_m}$$

3. $\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_m$ sont dits une *base* de $\mathbb{R}^n$ si et seulement si $\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_m$ sont à la fois linéairement indépendants et générateurs de $\mathbb{R}^n$.
   Dans ce cas, on démontre que $m = n$. n est la *dimension* associée à l'*espace vectoriel* $\mathbb{R}^n$.

– Soit $\mathbf{e}_1, \mathbf{e}_2, \cdots, \mathbf{e}_n$ une base de $\mathbb{R}^n$,

$$\mathbf{u} \in \mathbb{R}^n \qquad \Rightarrow \qquad \exists!\ u_1, u_2, \cdots, u_n \in \mathbb{R} \qquad \mathbf{u} = u_1\mathbf{e_1} + u_1\mathbf{e_1} + \cdots + u_n\mathbf{e_n}$$

Le vecteur $\begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix}$ est la représentation associée à $\mathbf{u}$ dans la base $\mathbf{e}_1, \mathbf{e}_2, \cdots, \mathbf{e}_n$ On note :

$$u = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix}$$

– Soit f une fonction linéaire de $\mathbb{R}^n$ dans $\mathbb{R}^m$, on montre que :

$$\forall\ \mathbf{x} = (x_1, x_2, \cdots, x_n) \in \mathbb{R}^n \qquad \forall\ \mathbf{y} = (y_1, y_2, \cdots, y_m) \in \mathbb{R}^m \qquad \exists!\ (a_{ij})_{\substack{i=1,m \\ j=1,n}},$$

$$\mathbf{y} = f(\mathbf{x}) \qquad \Longleftrightarrow \qquad \begin{array}{ccccccccc} y_1 & = & a_{11}x_1 & + & a_{12}x_2 & + & \cdots & + & a_{1n}x_n \\ y_2 & = & a_{21}x_1 & + & a_{22}x_2 & + & \cdots & + & a_{2n}x_n \\ \vdots & & \vdots & & \vdots & & & & \vdots \\ y_m & = & a_{m1}x_1 & + & a_{m2}x_2 & + & \cdots & + & a_{mn}x_n \end{array}$$

– On note :

$$A \in \mathbb{R}^{m\times n} \Longleftrightarrow A = [a_{ij}] = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix}$$

A est la *matrice* associé à la fonction f.

Notations : Lorsque une lettre capitale est utilisée, elle réfère à la matrice (*e.g.*, A, B, Δ), la lettre minuscule correspondante avec les indices ij indique les éléments de cette matrice (*e.g.*, a_{ij}, b_{ij}, δ_{ij}).

1.2 Opérations élémentaires sur les matrices

1.2.1 Addition

$$\begin{array}{ccccc} \mathbb{R}^{m\times n} \times \mathbb{R}^{m\times n} & \longrightarrow & \mathbb{R}^{m\times n} & & \\ A, \quad B & \longrightarrow & C = A + B & \qquad & c_{ij} = a_{ij} + b_{ij} \end{array}$$

1.2.2 Multiplication par un scalaire

$$\begin{array}{ccccc} \mathbb{R} \times \mathbb{R}^{m\times n} & \longrightarrow & \mathbb{R}^{m\times n} & & \\ \alpha, \quad A & \longrightarrow & C = \alpha.A & \qquad & c_{ij} = \alpha.a_{ij} \end{array}$$

1.2.3 Multiplication de deux matrices

$$\begin{array}{ccccc} \mathbb{R}^{m\times n} \times \mathbb{R}^{n\times p} & \longrightarrow & \mathbb{R}^{m\times p} & & \\ A, \quad B & \longrightarrow & C = A.B & & c_{ij} = \sum_{k=1}^{n} a_{ik}.b_{kj} \end{array}$$

1.2.4 Transposition

$$\begin{array}{ccccc} \mathbb{R}^{m\times n} & \longrightarrow & \mathbb{R}^{n\times m} & & \\ A & \longrightarrow & C = A^{\mathrm{T}} & & c_{ij} = a_{ji} \end{array}$$

1.2.5 Matrice identité

– une matrice de dimension $n \times n$ est dite *carré*.
– La matrice *identité* est noté I_n et sa k-ième colonne $e_k^{(n)}$:

$$I_n = \begin{bmatrix} 1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 1 \end{bmatrix} \qquad e_k^{(n)} = \begin{bmatrix} 0, & \cdots, & 0, & \underline{1}, & 0, & \cdots, & 0 \end{bmatrix}^{\mathrm{T}}$$

1.2.6 Produit scalaire

$$\mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix}, \qquad \mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}$$

$$\mathbf{u}.\mathbf{v} = \sum_{i=1}^{n} u_i v_i = u^{\mathrm{T}}.v$$

Deux vecteurs sont dits *orthogonaux* si et seulement si leur produit scalaire est nul.
D'une manière générale, on définit pour une série de vecteurs $\{u_1, \ldots, u_n\}$, l'*orthogonalité* comme :

$$\forall i \neq j, \qquad {u_i}^{\mathrm{T}} u_j = 0$$

et l'*orthonormalité* comme :

$$\forall i, \forall j, \qquad {u_i}^{\mathrm{T}} u_j = \delta_{ij}$$

1.2.7 Inverse d'une matrice

Si A et B sont deux matrices telles que $AB = I$, alors B est dit *inverse* de A. On le note $B = A^{-1}$.
Pour une matrice A donnée, si A^{-1} existe, A est dite *non-singulière*, sinon A est dite *singulière*.
On a la relation :

$$\left(A^{-1}\right)^{\mathrm{T}} = (A^{\mathrm{T}})^{-1}$$

1.2.8 Déterminant d'une matrice

Si $A = (a)$ dans $\mathbb{R}^{1\times 1}$, alors son *déterminant* est donnée par $det(A) = a$.

Pour $A \in \mathbb{R}^{n\times n}$, on a :

$$det(A) = \sum_{j=1}^{n} (-1)^{j+1} a_{1j} det(A_{1j})$$

où A_{1j} est une matrice $n-1 \times n-1$ obtenu en retirant la première ligne et la j-ième colonne de A.

Propriétés des déterminants :

1. $det(AB) = det(A)det(B)$
2. $det(A^{\mathrm{T}}) = det(A)$
3. $det(cA) = c^n det(A)$
4. $det(A) \neq 0 \iff A$ est non singulère

1.2.9 Normes

La *norme* d'un vecteur de $\mathbb{R}^n$ est une fonction $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ qui possède les propriétés suivantes :

1. $$\forall x \in \mathbb{R}^n, \qquad f(x) \geq 0$$
   avec l'égalité si et seulement si $x = 0$
2. $$\forall x, y \in \mathbb{R}^n, \qquad f(x+y) \leq f(x) + f(y)$$
3. $$\forall \alpha \in \mathbb{R}, \forall x \in \mathbb{R}^n, \qquad f(\alpha x) = |\alpha| f(x)$$

On note $f(x)$ par $||x||$

Norme de *Hölder* ou *norme p* :

$$||x||_p = (|x_1|^p + \cdot + |x_n|^p)^{1/p}$$

avec

$$||x||_1 = |x_1| + \cdot + |x_n|$$

$$||x||_2 = \sqrt{|x_1|^2 + \cdot + |x_n|^2} \qquad \text{norme euclidienne}$$

et

$$||x||_{\inf} = \max_i |x_i|$$

De même on peut définir la norme d'une matrice. Exemple :

$$\text{norme de Frobenius :} \qquad ||A||_F = \left[\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^2 \right]^{1/2}$$

1.2.10 Quelques définitions

Soit $A \in \mathbb{R}^{m\times n}$ une matrice à coefficients réels. Quelques définitions :

diagonale	$\forall\, i \neq j,$	$a_{ij} = 0$
tridiagonale	$\forall\, \lvert i-j\rvert > 1,$	$a_{ij} = 0$
bidiagonale supérieure	$\forall\, i > j$ ou $j > i+1,$	$a_{ij} = 0$
triangulaire supérieure	$\forall\, i > j$	$a_{ij} = 0$
triangulaire strictement supérieure	$\forall\, i \geq j$	$a_{ij} = 0$

symétrique	$A^{\mathrm{T}} = A$	
anti-symétrique	$A^{\mathrm{T}} = -A$	
définie positive	$\forall\, x \in \mathbb{R}^n,$	$x^{\mathrm{T}}Ax > 0$
définie non-négative	$\forall\, x \in \mathbb{R}^n,$	$x^{\mathrm{T}}Ax \geq 0$
indéfinie	$\exists\, x, y \in \mathbb{R}^n$	$(x^{\mathrm{T}}Ax)(y^{\mathrm{T}}Ay) < 0$
orthogonale	$A^{\mathrm{T}}A = I_n$	
nilpotente	$\exists\, k \in \mathbb{N}$	$A^k = 0$
idempotente	$A^2 = A$	
positive	$\forall\, i \in \mathbb{N}, \forall\, j \in \mathbb{N}$	$a_{ij} > 0$
non négative	$\forall\, i \in \mathbb{N}, \forall\, j \in \mathbb{N}$	$a_{ij} \geq 0$
diagonalement dominante	$\forall\, i \in \mathbb{N}$	$\lvert a_{ii}\rvert > \sum_{j\neq i} \lvert a_{ij}\rvert$

sous-matrice principale : on appelle *sous-matrice principale* d'ordre k d'une matrice A de $\mathbb{R}^{n\times n}$ la sous-matrice A_k obtenue en enlevant les $n-k$ dernières colonnes et les $n-k$ dernières lignes de A.

1.3 Représentation matricielles

1.3.1 Matrices pleines

exemple :

$$A = \begin{bmatrix} 1. & 2. & 3. \\ 4. & 5. & 6. \\ 7. & 8. & 9. \\ 10. & 11. & 12. \end{bmatrix}$$

traduction en C :

```
double A[4][3];


A[0][0] = 1.0d0;
A[0][1] = 2.0d0;
A[0][2] = 3.0d0;
A[1][0] = 4.0d0;
A[1][1] = 5.0d0;
A[1][2] = 6.0d0;
A[2][0] = 7.0d0;
A[2][1] = 8.0d0;
A[2][2] = 9.0d0;
A[3][0] = 10.0d0;
A[3][1] = 11.0d0;
A[3][2] = 12.0d0;
```

en mémoire dans l'ordinateur :

1. 2. 3.	ligne 1
4. 5. 6.	ligne 2
7. 8. 9.	ligne 3
10. 11. 12.	ligne 4

traduction en Fortran :

```
      double precision A
      dimension A(4,3)

      A(1,1) = 1.0d0
      A(1,2) = 2.0d0
      A(1,3) = 3.0d0
      A(2,1) = 4.0d0
      A(2,2) = 5.0d0
      A(2,3) = 6.0d0
      A(3,2) = 7.0d0
      A(3,1) = 8.0d0
      A(3,3) = 9.0d0
      A(4,1) = 10.0d0
      A(4,2) = 11.0d0
      A(4,3) = 12.0d0
```

en mémoire dans l'ordinateur :

1. 4. 7. 10.	colonne 1
2. 5. 8. 11.	colonne 2
3. 9. 6. 12.	colonne 3

1.3.2 Matrices symétriques

Exemple :

$$A = \begin{bmatrix} 3. & 5. & -9. & 0. \\ 5. & 1. & 2. & -5. \\ -9. & 2. & 4. & 3. \\ 0. & -5. & 3. & 8. \end{bmatrix}$$

Stockage 1 :

u	3.	5.	1.	-9.	2.	4.	0.	-5.	3.	8.
index	1	2	3	4	5	6	7	8	9	10

1	2	4	7
2	3	5	8
4	5	6	9
7	8	9	10

$$a_{ij} \quad \longleftrightarrow \quad u_k \qquad \text{avec} \qquad k = \frac{i(i-1)}{2} + j \qquad (i > j)$$

Stockage 2

u	3.	5.	-9.	0.	1.	2.	-5.	4.	3.	8.
index	1	2	3	4	5	6	7	8	9	10

1	2	3	4
2	5	6	7
3	6	8	9
4	7	9	10

$$a_{ij} \quad \longleftrightarrow \quad u_k \qquad \text{avec} \qquad k = (i-1)n - \frac{i(i-1)}{2} + j \qquad (i < j)$$

1.3.3 Matrices creuses

Exemple :

$$A = \begin{bmatrix} 3. & 0. & 1. & 0. & 0. \\ 0. & 4. & 0. & 0. & 0. \\ 0. & 7. & 5. & 9. & 0. \\ 0. & 0. & 0. & 0. & 2. \\ 0. & 0. & 0. & 6. & 5. \end{bmatrix}$$

Stockage 1

u	3.	1.	4.	7.	5.	9.	2.	6.	5.
ligne	1	1	2	3	3	3	4	5	5
col	1	3	2	2	3	4	5	4	5

Stockage 2

index k	1	2	3	4	5	6	7	8	9	10	11
ija	7	8	8	10	11	12	3	2	4	5	4
u	3.	4.	5.	0.	5.	x	1.	7.	9.	2.	6.
	diagonale						non diagonale				

1.4 Opérations avancées sur les matrices

1.4.1 Addition de deux matrices

traduction en C

```
double A[n][m], B[n][m], C[n][m];
int i, j;

for (i = 0; i < n; i++)
{
   for (j = 0; j < m; j++)
   {
      C[i][j] = A[i][j] + B[i][j];
   }
}
```

traduction en Fortran

```
double precision A, B, C
dimension A(n,m), B(n,m), C(n,m)
integer i,j

do i = 1,m
   do i = 1,n
      C(i,j) = A(i,j) + B(i,j)
   end do
end do
```

1.4.2 Multiplication de deux matrices

traduction en C

```
double A[n][p], B[p][m], C[n][m];
int i, j, k;

for (i = 0; i < n; i++)
{
   for (k = 0; k < p; k++)
   {
      for (j = 0; j < m; j++)
      C[i][j] = C[i][j] + A[i][k] * B[k][j];
   }
}
```

traduction en Fortran

```
double precision A, B, C
dimension A(n,p), B(p,m), C(n,m)
integer i, j, k

do j = 1, m
   do k = 1, p
      do i = 1, n
         C(i,j) = C(i,j) + A(i,k) * B(k,j)
      end do
   end do
end do
```

1.4.3 Décomposition LU

Soit $A \in \mathbb{R}^{n \times n}$, si $\forall k \in 1, \ldots, n$ les sous-matrices principales A_k de A sont toutes non singulières, alors

$\exists ! \, U \in \mathbb{R}^{n \times n}$ une matrice triangulaire supérieure et

$\exists ! \, L \in \mathbb{R}^{n \times n}$ une matrice triangulaire inférieure unitaire telles que

$$A = LU$$

Exemple :

$$A = \begin{bmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 11 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 2 & 1 \end{bmatrix} \begin{bmatrix} 1 & 4 & 7 \\ 0 & -3 & -6 \\ 0 & 0 & 2 \end{bmatrix}$$

$$Ax = b \Longleftrightarrow Ax = (LU)x = L(Ux) = Ly = b$$

$Ly = b$ **:** forward elimination

Algorithme :

For i = 1, ... , n

y_i := b_i

For j = 1, ..., $i-1$

y_i := $y_i - l_{ij} y_j$

y_i := y_i / l_{ii}

$Ux = y$ **:** back substitution

Algorithme :

For i = n , ... , 1

x_i := y_i

For j = $i+1$, ..., n

x_i := $x_i - u_{ij} x_j$

x_i := x_i / u_{ii}

$$A = LU$$

$$\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix} = \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1n} \\ v_{21} & u_{22} & \cdots & u_{2n} \\ \vdots & & \ddots & \vdots \\ v_{n1} & v_{n2} & \cdots & u_{nn} \end{bmatrix}$$

Elimination de Gauss : remplace A par LU

For $k = 1, ..., n-1$

If $a_{kk} = 0$

Then *quit*

Else

w_j := a_{kj} pour tous les $j = k+1, ...n$

For $i = k+1, ...n$

$\eta := a_{ik} / a_{kk}$

$a_{ik} := \eta$

For $j = k+1, ...n$

$a_{ij} := a_{ij} - \eta w_j$

Problème : $a_{kk} = 0$

Remède : Elimination de Gauss avec pivot complet

For $k = 1, ..., n$

Détermine indices p et q tels que

$$|a_{pq}| = \max_{\substack{k \le i \le n \\ k \le j \le n}} |a_{ij}|$$

$r_k := p$

$c_k := q$

Echange a_{kj} et a_{pj} $\quad (j = k, ..., n)$

Echange a_{ik} et a_{iq} $\quad (i = 1, ..., n)$

w_j := a_{kj} pour tous les $j = k + 1, ...n$

For $i = k + 1, ...n$

$\eta := a_{ik}/a_{kk}$

$a_{ik} := \eta$

For $j = k + 1, ...n$

$a_{ij} := a_{ij} - \eta w_j$

Numerical Recipes : routine `gaussj`

pivot partiel : on n'echange que les lignes (mais pas les colonnes)

1.4.4 Décomposition QR

Soit $A \in \mathbb{R}^{n \times n}$ une matrice non singulière,

$\exists\, Q \in \mathbb{R}^{n \times n}$ une matrice orthogonale

$\exists\, R \in \mathbb{R}^{n \times n}$ une matrice triangulaire supérieure telles que

$$A = QR$$

$$Ax = b \Longleftrightarrow Rx = Q^{\mathrm{T}} b$$

Numerical Recipes : routine `qrdcmp`

1.4.5 Décomposition de Cholesky

Si A est une matrice symétrique et définie positive, alors il existe L une matrice triangulaire inférieure telle que :

$$A = L^{\mathrm{T}} L$$

En écrivant les équations :

$$L_{ii} = \left(a_{ii} - \sum_{k=1}^{i-1} L_{ik}^2 \right)^{1/2}$$

et

$$L_{ji} = \frac{1}{L_{ii}} \left(a_{ij} - \sum_{k=1}^{i-1} L_{ik} L_{jk} \right) \qquad j = i + 1, i + 2, ..., n$$

Numerical Recipes : routine `choldc`

1.4.6 Valeurs propres/Vecteurs propres

$$Ax = \lambda x$$

λ : valeur propre de A

x : vecteur propre de A

Ensemble des valeurs propres de A $\{\lambda_1, ...\lambda_n\}$: *spectre* de la matrice

$$det(A) = \lambda_1.\lambda_2.\cdots.\lambda_n$$
$$trace(A) = \sum_{i=1}^{n} a_{ii} = \sum_{i=1}^{n} \lambda_i$$

Si les λ_i sont tous distincts : le système est dit *non-dégénéré*, sinon il est dit *dégénéré*.

Si $A \in \mathbb{R}^{n\times n}$ est une matrice réel et symétrique, alors A admet n valeurs propres réelles, et il existe $Q \in \mathbb{R}^{n\times n}$, une matrice orthogonale, telle que :

$$Q^{\mathrm{T}} AQ = diag(\lambda_1 \cdots \lambda_n)$$

Numerical Recipes : routine `jacobi`

Remarque : Une matrice symétrique est définie-positive si et seulement si toutes ses valeurs propres sont strictement positive.

1.5 Utilisation de librairies mathématiques

1.5.1 Un exemple : la multiplication de deux matrices

Combien de multiplication pour une multiplication de deux matrices ?

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} \\ a_{21} & b_{22} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix}$$

8 ?

$$\begin{aligned} c_{11} &= a_{11} \times b_{11} + a_{12} \times b_{21} \\ c_{12} &= a_{11} \times b_{12} + a_{12} \times b_{22} \\ c_{11} &= a_{21} \times b_{11} + a_{22} \times b_{21} \\ c_{22} &= a_{21} \times b_{12} + a_{22} \times b_{22} \end{aligned}$$

On peut faire mieux ! (Strassen)

$$
\begin{aligned}
Q_1 &= (a_{11} + a_{22}) \times (b_{11} + b_{22}) \\
Q_2 &= (a_{21} + a_{22}) \times b_{11} \\
Q_3 &= a_{11} \times (b_{12} - b_{22}) \\
Q_4 &= a_{22} \times (-b_{11} + b_{21}) \\
Q_5 &= (a_{11} + a_{12}) \times b_{22} \\
Q_6 &= (-a_{11} + a_{21}) \times (b_{11} + b_{12}) \\
Q_7 &= (a_{12} - a_{22}) \times (b_{21} + b_{22})
\end{aligned}
$$

puis

$$
\begin{aligned}
c_{11} &= Q_1 + Q_4 - Q_5 + Q_7 \\
c_{12} &= Q_2 + Q_4 \\
c_{21} &= Q_3 + Q_5 \\
c_{22} &= Q_1 + Q_3 - Q_2 + Q_6
\end{aligned}
$$

Ces opérations sont valides aussi lorsque a et b sont des matrices. Une récursivité est donc possible.
Nombre de multiplications pour une matrice de taille N très grand :

$$N^3 \longrightarrow N^{\log_2 7}$$

Il vaut mieux utiliser ce qui a déjà été résolu : bibliothèques mathématiques.

1.5.2 BLAS

http ://www.netlib.org
B.L.A.S. : Basic Linear Algebra Subprograms
Ce sont des routines pour effectuer des opérations de bases sur les vecteurs et les matrices (Fortran 77).
BLAS niveau 1 : opérations vecteurs-vecteurs
BLAS niveau 2 : opérations matrices-vecteurs
BLAS niveau 3 : opérations matrices-matrices
Avantage de BLAS : efficace, portable, facilement accessible.
+ $\exists$ des versions machine-spécifique $\longrightarrow$ très efficace car utilise au mieux les possibilités des machines.
ATLAS : BLAS qui s'optimise pour une architecture donnée.
Limité aux opérations élémentaires.
exemple avec BLAS-3 :

$$
\begin{aligned}
C &\leftarrow \alpha AB + \beta C \\
C &\leftarrow \alpha A^{\mathrm{T}} B + \beta C \\
C &\leftarrow \alpha AB^{\mathrm{T}} + \beta C \\
C &\leftarrow \alpha A^{\mathrm{T}} B^{\mathrm{T}} + \beta C
\end{aligned}
$$

Si T est triangulaire

$$\begin{aligned} B &\leftarrow \alpha T B \\ B &\leftarrow \alpha T^{\mathrm{T}} B \\ B &\leftarrow \alpha B T \\ B &\leftarrow \alpha B T^{\mathrm{T}} \end{aligned}$$

...

Convention d'appel :

Caractère 1 : type donnée dans la matrice

S : REAL, D : DOUBLE PRECISION, C : COMPLEX, Z : COMPLEX*16 ou DOUBLE COMPLEX

Caractère 2 et 3 : type de matrice

GE : matrice rectangulaire (général), HE : l'une des matrices est hermitienne ($A = A^{\mathrm{T}^*}$), SY : l'une des matrices est symétrique, TR : l'une des matrices est triangulaire

Caractère 4 et 5 : type d'opérations

MM : produit de matrices, SM : résolution d'une systèmes d'équations linéaires, ...

Exemple : _GEMM

1.5.3 LAPACK

Bibliothèques de routines portables (Fortran 77) permettant de résoudre la plupart des problèmes numériques en algèbre linéaire.

Ex. : systèmes d'équations linéaires, valeurs propres, décomposition en valeurs singulières, (LU, Cholesky, SVD, QR, ...)

Haute performance, notamment sur les architectures parallèles, vectorielles, à mémoires partagées, ...

LAPACK fait appel a BLAS $\longrightarrow$ très efficace et portable.

Version Parallèle : PBLAS et ScaLAPACK.

Chapitre 2

Equations Différentielles

2.1 Rappels mathématiques

2.1.1 Dérivées d'une fonction réelle

2.1.1.1 Dérivée première

Soif f une fonction de $\mathbb{R}$dans $\mathbb{R}$:

$$f : \begin{array}{ccc} \mathbb{R} & \longrightarrow & \mathbb{R} \\ x & \Longrightarrow & f(x) \end{array}$$

$f'(x_0)$, ou $f^{(1)}(x_0)$ est la dérivée première de f en x_0 et se définit par :

$$f'(x_0) = \lim_{x \to x_0} \frac{f(x) - f(x_0)}{x - x_0}$$

f est dite *dérivable* si et seulement en tout point x où f est définie il existe une dérivée $f'(x)$.

2.1.1.2 Dérivée seconde

$$f''(x_0) = f^{(2)}(x_0) = \lim_{x \to x_0} \frac{f'(x) - f'(x_0)}{x - x_0}$$

Notations :

$$f(x) = f^{(0)}(x)$$
$$f'(x) = \frac{\mathrm{df}}{\mathrm{dx}}$$
$$f^{(2)}(x) = \frac{\mathrm{d^2f}}{\mathrm{dx^2}}$$

2.1.2 Développement de Taylor

Développement de Taylor d'une fonction f continue et dérivable infiniment :

$$a, b \in \mathbb{R}, \qquad f(b) = f(a) + (b-a)f^{(1)}(a) + \frac{(b-a)^2}{2} f^{(2)}(a) + \cdots + \frac{(b-a)^n}{n!} f^{(n)}(a) + \cdots$$

ou encore

$$a, h \in \mathbb{R}, \qquad \begin{aligned} f(a+h) &= f(a) + hf^{(1)}(a) + \frac{h^2}{2} f^{(2)}(a) + \cdots \\ &\quad + \frac{h^n}{n!} f^{(n)}(a) + \cdots \end{aligned}$$

2.1.3 Développement limité

2.1.3.1 Développement limité d'ordre n

$$a, b \in \mathbb{R}, \qquad \begin{aligned} f(b) &= f(a) + (b-a)f^{(1)}(a) + \frac{(b-a)^2}{2} f^{(2)}(a) + \cdots \\ &\quad + \frac{(b-a)^n}{n!} f^{(n)}(a) + O((b-a)^{n+1}) \end{aligned}$$

$$a, h \in \mathbb{R}, \qquad \begin{aligned} f(a+h) &= f(a) + hf^{(1)}(a) + \frac{h^2}{2} f^{(2)}(a) + \cdots \\ &\quad + \frac{h^n}{n!} f^{(n)}(a) + O(h^{n+1}) \end{aligned}$$

2.1.3.2 Discrétisation des dérivées

Dérivée première à droite :

$$f'(a) = \frac{f(a+h) - f(a)}{h} + O(h)$$

Dérivée première à gauche :

$$f'(a) = \frac{f(a) - f(a-h)}{h} + O(h)$$

Dérivée première symétrique :

$$f'(a) = \frac{f(a+h) - f(a-h)}{2h} + O(h^2)$$

Dérivée seconde (symétrique) :

$$f''(a) = \frac{f(a+h) - 2f(a) + f(a-h)}{h^2} + O(h)$$

2.1.4 Equations différentielles

Une équation différentielle est une équation du type

$$f(y^{(n)}, y^{(n-1)}, \cdots, y^{(1)}, y, \mathbf{x}) = 0$$

où y est une fonction de $\mathbb{R}^m$ dans $\mathbb{R}^p$:

$$y : \begin{array}{ccc} \mathbb{R}^m & \longrightarrow & \mathbb{R}^p \\ \mathbf{x} & \Longrightarrow & y(\mathbf{x}) \end{array}$$

2.2 Equations différentielles ordinaires

2.2.1 Cas général

On s'intéresse au cas où y ne dépend que d'une seule variable x. On a alors affaire à une *équation différentielle ordinaire* (ODE).

Dans le cas général d'une équation différentielle ordinaire d'ordre n, il est toujours possible de la réduire à l'étude de n équations différentielles couplées du premier ordre.

$$\frac{\mathrm{dy_i(x)}}{\mathrm{dx}} = f_i(x, y_1, \cdots, y_n) \qquad i = 1, \cdots, n$$

Exemple :

$$\frac{\mathrm{d^2y}}{\mathrm{dx^2}} + q(x)\frac{\mathrm{dy}}{\mathrm{dx}} = r(x)$$

peut s'écrire

$$\begin{aligned} \frac{\mathrm{dy}}{\mathrm{dx}} &= z(x) \\ \frac{\mathrm{dz}}{\mathrm{dx}} &= r(x) - q(x)z(x) \end{aligned}$$

Conditions aux limites : ce sont des conditions particulières sur des valeurs de y_i ou y_i'.

Deux grands cas :
– problèmes aux valeurs initiales : on connait les y_i (ou y_i') pour une valeur de départ $\mathbf{x_D}$ et on cherche les y_i pour une valeur finale $\mathbf{x_F}$ ou pour un ensembre d'intervalles $\mathbf{x_j}$.
– problèmes de valeurs limites en deux points : on connait les y_i en plusieurs points (par exemple en x_D et x_F) et on cherche les y_i en d'autres points (typiquement en x_d et x_F).
On se limitera dans ce cours au premier grand cas.

2.2.2 Résolutions d'une ODE aux conditions initiales

$$\frac{\mathrm{dy}}{\mathrm{dx}} = f(y, x) \qquad \text{avec} y(x_0) \text{connu}$$

On discrétise l'espace des x.

$$x_0, x_1, \cdots, x_n \qquad \text{avec} x_{i+1} - x_i = \delta x$$

$$y(x_0) = y_0$$

$$y(x_i) = y_i$$

$$\left(\frac{\mathrm{dy}}{\mathrm{dx}}\right)_{x_n} = f(y_n, x_n) = f_n$$

2.2.2.1 Méthode d'Euler

Approximation du la dérivée :

$$\left(\frac{\mathrm{dy}}{\mathrm{dx}}\right)_{x_n} = \frac{y_{n+1} - y_n}{\delta x} = f_n$$

d'où

$$y_{n+1} = y_n + \delta x f_n$$

On obtient une suite récurrente.

Précision de la méthode d'Euler ? On compare avec un développement de Taylor :

$$\text{Taylor :} \qquad y_{n+1} = y_n + \delta x \left(\frac{\mathrm{dy}}{\mathrm{dx}}\right)_n + \frac{\delta x^2}{2}\left(\frac{\mathrm{d^2y}}{\mathrm{dx^2}}\right)_n + \cdots$$

$$\text{Euler :} \qquad y_{n+1} = y_n + \delta x f_n = y_n + \delta x \left(\frac{\mathrm{dy}}{\mathrm{dx}}\right)_n$$

La méthode est exacte au premier ordre en δx.

Stabilité : une méthode est stable si la petite déviation par rapport à la vrai solution n'augmente pas lorsqu'on itère la solution.

Trois ODEs types ($\alpha > 0$) :

– équation décroissante

$$\frac{\mathrm{dy}}{\mathrm{dt}} + \alpha y = 0 \quad \Longrightarrow \quad y = y_0 \exp(-\alpha t)$$

– équation croissante

$$\frac{\mathrm{dy}}{\mathrm{dt}} - \alpha y = 0 \quad \Longrightarrow \quad y = y_0 \exp(+\alpha t)$$

– équation oscillante

$$\frac{\mathrm{dy}}{\mathrm{dt}} \pm i\omega y = 0 \quad \Longrightarrow \quad y = y_0 \exp(\pm i\omega t)$$

Stabilité de ces trois équations différentielles ordinaires dans le cas d'une résolution par la méthode d'Euler :

décroissante	croissante	oscillante
$\delta t \leq \frac{2}{\alpha}$	instable	instable

2.2.2.2 Runge-Kutta du second ordre

$$\begin{aligned} y_{n+\frac{1}{2}} &= y_n + \frac{\delta x}{2} f(y_n, x_n) \\ y_{n+1} &= y_n + \delta x f(y_{n+\frac{1}{2}}, x_{n+\frac{1}{2}}) \end{aligned}$$

C'est une méthode du second ordre.

2.2.2.3 Runge-Kutta du quatrième ordre

$$\begin{aligned} k_1 &= \delta x f(x_n, y_n) \\ k_2 &= \delta x f(x_n + \frac{1}{2}\delta x, y_n + \frac{k_1}{2}) \\ k_3 &= \delta x f(x_n + \frac{1}{2}\delta x, y_n + \frac{k_2}{2}) \\ k_4 &= \delta x f(x_n + \delta x, y_n + k_3) \\ y_{n+1} &= y_n + \frac{k_1}{6} + \frac{k_2}{3} + \frac{k_3}{3} + \frac{k_4}{6} \end{aligned}$$

C'est une méthode du quatrième ordre.

Stabilité :

décroissante	croissante	oscillante
$\delta t \leq \frac{2}{\alpha}$	instable	$1 + \frac{1}{4}(\delta t \omega)^4 \leq 1$

2.2.2.4 Predictor-Corrector

Cela ressemble un peu à Runge-Kutta du deuxième ordre ... en mieux.

On aimerait (intégration en trapèze) :

$$y_{n+1} = y_n + \frac{\delta x}{2}\left[f(y_{n+1}, x_{n+1}) + f(y_n, x_n)\right]$$

Or, on ne connait pas $f(y_{n+1}, x_{n+1})$.

– Prédictor :

$$y^*_{n+1} = y_n + \delta x f(y_n, x_n)$$

– Corrector :

$$y_{n+1} = y_n + \frac{\delta x}{2}\left[f(y^*_{n+1}, x_{n+1}) + f(y_n, x_n)\right]$$

remarque : il existe en fait une multitude de méthode du type *corrector-predictor*, chacune ayant des caractéristiques différentes et pouvant être employée pour résoudre des problèmes précis.

Exemple (en tenant compte des résultats antérieurs dans une résolution) :

– Prédictor :

$$y^*_{n+1} = y_n + \frac{\delta x}{12}(23y'_n - 16y'_{n-1} + 5y'_{n-2})$$

– Corrector :

$$y_{n+1} = y_n + \frac{\delta x}{12}(5y'_{n+1} + 8y'_n - y'_{n-1})$$

remarque 2 : comme en algèbre linéaire, il existe des bibliothèques de routines mathématiques permettant de résoudre toute sorte d'ODEs.

Exemple : ODE et ODEPACK disponibles sur `http ://www.netlib.org`.

2.3 Equations différentielles partielles

On se restreint à deux dimensions (*i.e.* : $u : (x, y) \longrightarrow u(x, y)$ et au second ordre.

$$a\frac{\partial^2 u}{\partial x^2} + 2b\frac{\partial^2 u}{\partial x \partial y} + c\frac{\partial^2 u}{\partial y^2} + d\frac{\partial u}{\partial x} + e\frac{\partial u}{\partial y} + fu + g = 0$$

Condition	Type	Exemple
$b^2 < ac$	elliptique	équation de Poisson
$b^2 > ac$	hyperbolique	équation d'ondes
$b^2 = ac$	parabolique	équation de la diffusion

– Equation de Poisson :

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = \rho(x, y)$$

– Equation de la diffusion :

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}\left(D\frac{\partial u}{\partial x}\right)$$

– Equation d'ondes

$$\frac{\partial^2 u}{\partial t^2} = v^2\frac{\partial^2 u}{\partial x^2}$$

On regarde plus en détail deux exemples : l'équation de Poisson et l'équation de la diffusion.

2.3.1 Equation de Poisson

utilisation d'une méthode par différences finies.

$$\begin{aligned} x_j &= x_0 + j\Delta \qquad & j &= 0, 1, \cdots, J \\ y_l &= y_0 + l\Delta & l &= 0, 1, \cdots, L \end{aligned}$$

$$\frac{1}{\Delta^2}\left(u_{j+1,l} - 2u_{j,l} + u_{j-1,l}\right) + \frac{1}{\Delta^2}\left(u_{j,l+1} - 2u_{j,l} + u_{j,l-1}\right) = \rho_{j,l}$$

Soit

$$u_{j+1,l} + u_{j-1,l} + u_{j,l+1} + u_{j,l-1} - 4u_{j,l} = \Delta^2\rho_{j,l}$$

On transforme u en vecteur :

$$i = j(L+1) + l \qquad \text{pour } \begin{matrix} j = 0, 1, \cdots, J \\ l = 0, 1, \cdots, L \end{matrix}$$

$$u_{i+L+1} + u_{i-(L+1)} + u_{i+1} + u_{i-1} - 4u_i = \Delta^2\rho_i$$

L'équation fonctionne pour $j = 1, 2, \cdots, J-1$ et $l = 1, 2, \cdots L-1$.

$$\begin{aligned} &j = 0 \quad [i.e. \quad i = 0, \cdots, L] \\ &j = J \quad [i.e. \quad i = J(L+1), \cdots, J(L+1)+L] \\ &l = 0 \quad [i.e. \quad i = 0, L+1, \cdots, J(L+1)] \\ &l = L \quad [i.e. \quad i = L, L+1+L, \cdots, J(L+1)+L] \end{aligned}$$

Ce sont les conditions aux limites. u ou sa dérivée doivent être connus en ces points.

On obtient :

$$A.u = b$$

A est "diagonale avec des franges" (c.f. Numerical Recipes).

$\rightarrow$ bande

$\rightarrow$ creuse

Résolution :
– matricielle
– méthode de relaxation

$$A = E - F$$

où E est facilement inversible et F est le reste. On doit résoudre $Eu = Fu + b$. On itère à partir d'une solution initiale $u^{(0)}$: $E.u^{(r)} = F.u^{(r-1)} + b$

2.3.2 Equation de la diffusion

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}\left(D\frac{\partial u}{\partial x}\right)$$

On note F le flux :

$$F = -D\frac{\partial u}{\partial x}$$

On suppose $D \geq 0$.

Si D est constante :

$$\frac{1}{\Delta t}(u_j^{n+1} - u_j^n) = D\left[\frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{(\Delta x)^2}\right]$$

C'est un schéma explicite en x ou schéma FTCS (Forward Time Centered Space). Il est du premier ordre.

Condition de stabilité :

$$\frac{2D\Delta t}{(\Delta x)^2} \leq 1$$

$\rightarrow$ stable pour des petits pas de temps.

Schéma implicite (premier ordre) :

$$\frac{1}{\Delta t}(u_j^{n+1} - u_j^n) = D\left[\frac{u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}}{(\Delta x)^2}\right]$$

$\rightarrow$ stable pour des petits pas de temps.

Crank-Nicholson (second ordre en t et en x) :

$$\frac{1}{\Delta t}(u_j^{n+1} - u_j^n) = \frac{D}{2}\left[\frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{(\Delta x)^2} + \frac{u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}}{(\Delta x)^2}\right]$$

$\rightarrow$ stable pour des grands pas de temps.

Table des matières