Résolution de systèmes linéaires par des méthodes directes

J. Erhel

Janvier 2014

1 Inverse d'une matrice carrée et systèmes linéaires

Ce paragraphe a pour objet les matrices carrées et les systèmes linéaires. Il définit les notions de matrice inversible et de matrice singulière.

1.1 Inverse d'une matrice

Théorème 1.1 *$A \in \mathbb{R}^{n\times n}$, sont équivalents :*
A est inversible : il existe une matrice notée A^{-1}, telle que $AA^{-1} = I$ (et $A^{-1}A = I$),
$det(A) \neq 0$,
$rang(A) = n$,
le système linéaire $Ax = b$ a une solution pour tout b,
si le système linéaire $Ax = b$ a une solution, elle est unique,
$Im(A) = \mathbb{R}^n$,
$ker(A) = \{0\}$.

Preuve. $rang(A) = n$ équivaut à $Im(A) = \mathbb{R}^n$ et à $ker(A) = \{0\}$, ce qui équivaut aussi à $Ax = b$ a une solution pour tout b. D'après la proposition précédente, $rang(A) = n$ équivaut aussi à si le système a une solution, elle est unique.

Si $rang(A) = n$, soit Cb la solution unique du système $Ax = b$, alors $ACb = b, \forall b$ donc $AC = I$ et $CAx = Cb = x, \forall x$ donc $CA = I$ et A est inversible d'inverse C.

Réciproquement, si A est inversible alors $AA^{-1}b = b, \forall b$ donc le système $Ax = b$ a une solution pour tout b.

On admet que A inversible équivaut à $det(A) \neq 0$.

$\diamond$

Définition 1.1 *Une matrice carrée non inversible est dite singulière. Une matrice inversible est dite non singulière.*

Proposition 1.1 *Si A et B sont inversibles, alors AB est inversible et $(AB)^{-1} = B^{-1}A^{-1}$.*

Si A est inversible, alors A^T est inversible et $(A^T)^{-1} = (A^{-1})^T = A^{-T}$.

Preuve. $(AB)(B^{-1}A^{-1}) = A(BB^{-1})A^{-1} = AIA^{-1} = I$ donc AB est inversible d'inverse $B^{-1}A^{-1}$.

$A^T(A^{-1})^T = (A^{-1}A)^T = I$. ◇

1.2 Matrices particulières

Proposition 1.2 *Si Q est orthogonale, alors Q est inversible et $Q^{-1} = Q^T$.*

Une matrice diagonale $D = diag(d_i)$ est inversible si et seulement si $d_i \neq 0$ et l'inverse de D est la matrice diagonale $D^{-1} = diag(1/d_i)$.

Une matrice triangulaire L est inversible si et seulement si $L_{ii} \neq 0$ et l'inverse de L est triangulaire.

Preuve. Si Q est orthogonale, alors $Q^TQ = I$ donc $Q^{-1} = Q^T$.

Si D est diagonale, $Dx = b$ équivaut à $d_ix_i = b_i, i = 1, \ldots, n$, a une solution pour tout b si et seulement si $d_i \neq 0, i = 1, \ldots, n$; la solution vaut $x_i = b_i/d_i$ donc $D^{-1} = diag(1/d_i)$.

Si L est triangulaire inférieure, alors le système $Lx = b$ s'écrit $l_{11}x_1 = b_1, \sum_{j<i} l_{ij}x_j + l_{ii}x_i = b_i, i = 2, \ldots, n$ et a une solution pour tout b si et seulement si $l_{ii} \neq 0$; la solution vaut $x_1 = b_1/l_{11}, x_i = (b_i - \sum_{j<i} l_{ij}x_j)/l_{ii}, i = 2, \ldots, n$ donc x_i ne dépend que de $b_j, j \leq i$ et L^{-1} est triangulaire inférieure.

◇

Proposition 1.3 *Calculer l'inverse d'une matrice diagonale d'ordre n est de type BLAS1, avec n opérations.*

Résoudre un système triangulaire d'ordre n est de type BLAS2, avec n^2 opérations.

Résoudre k systèmes triangulaires d'ordre n est de type BLAS3, avec kn^2 opérations.

1.3 Résolution d'un système linéaire

On considère un système linéaire de n équations à n inconnues, qui a une solution unique, autrement dit le système $Ax = b$, avec A inversible.

La méthode de résolution de Cramer, basée sur les déterminants, est à proscrire, pour deux raisons :

- la complexité est élevée, en $O(n!)$, ce qui explose très vite.
- les phénomènes de cancellation dans les calculs de déterminants amplifient les erreurs d'arrondi et l'algorithme est numériquement instable. Même avec un système d'ordre 2, le résultat peut être complètement faux.

Si la matrice est diagonalisable, on peut en théorie exprimer le système dans la base des vecteurs propres et résoudre le système diagonal :

$$A = VDV^{-1}, Ax = b \Leftrightarrow D(V^{-1}x) = V^{-1}b.$$

Mais diagonaliser une matrice est très coûteux. Ensuite, il faudrait faire les calculs avec des nombres complexes, ce qui est nettement plus coûteux. De plus, toutes les matrices ne sont pas diagonalisables.

Les méthodes basées sur l'élimination d'inconnues, avec combinaisons linéaires des lignes, sont les méthodes directes de référence. Lorsqu'on élimine les inconnues, on aboutit finalement à un système triangulaire supérieur, que l'on remonte pour calculer les coordonnées de la solution. Le processus d'élimination est en fait la résolution d'un système triangulaire inférieur. L'algorithme s'appelle la factorisation LU.

Pour garantir la stabilité numérique, il est nécessaire de modifier l'ordre d'élimination, en appliquant ce qu'on appelle une stratégie de pivot.

2 Factorisation de Cholesky

On considère la décomposition de matrices symétriques et définies positives c'est-à-dire telles que :

$$\forall x \neq 0,\ x^T Ax > 0$$

Ces matrices sont inversibles et leurs valeurs propres sont strictement positives.

2.1 Algorithme de factorisation

Théorème 2.1 Factorisation de Cholesky
Soit $A \in \mathbb{R}^{n\times n}$ une matrice symétrique et définie positive. Alors il existe une unique matrice triangulaire inférieure L qui vérifie :

$$A = LL^T \text{ et } diag(L) > 0$$

Preuve. On démontre le théorème par récurrence sur la dimension n de la matrice.

$n = 1$: le résultat est évident car alors le nombre qui représente la matrice est positif et le facteur de Cholesky est sa racine carrée.

On suppose la proposition vraie pour toute matrice d'ordre $n-1$. Soit A une matrice d'ordre n que l'on partitionne en blocs de la manière suivante où $A' \in \mathbb{R}^{(n-1)\times(n-1)}$, $a \in \mathbb{R}^{n-1}$, et $\alpha \in \mathbb{R}$:

$$A = \begin{pmatrix} A' & a \\ a^T & \alpha \end{pmatrix}.$$

On recherche le facteur de Cholesky L partitionné de la même manière :

$$L = \begin{pmatrix} L' & 0 \\ l^T & \lambda \end{pmatrix}$$

qui vérifie $LL^T = A$ ou encore

$$\begin{aligned} L'L'^T &= A' && (1)\\ L'\,l &= a && (2)\\ \|l\|^2 + \lambda^2 &= \alpha. && (3) \end{aligned}$$

La matrice A' étant une matrice principale de la matrice A est définie positive. On peut donc lui appliquer l'hypothèse de récurrence, ce qui détermine le facteur L' vérifiant l'équation (1) et à diagonale positive. L'équation (2) permet alors de calculer $l = L'^{-1}\,a$. En remplaçant dans la dernière équation les quantités déjà trouvées, il reste à résoudre :

$$\lambda^2 = \alpha - a^T A^{-1} a.$$

Il faut prouver que le terme de droite de l'égalité est positif ce qui pemet de conclure que $\lambda = \sqrt{\alpha - a^T A^{-1} a}$. On suppose que $a \neq 0$ car sinon le résultat est évident. Pour cela on étudie, pour tout vecteur $u \in \mathbb{R}^n$ non nul et tout réel ρ, la quantité :

$$\tau \;=\; (u^T\ \rho) \begin{pmatrix} A & a \\ a^T & \alpha \end{pmatrix} \begin{pmatrix} u \\ \rho \end{pmatrix} \;=\; \alpha\rho^2 + 2\rho u^T a + u^T A u$$

qui est strictement positive pour tout ρ. Le discriminant du trinôme est donc négatif :
$(u^T a)^2 - \alpha(u^T A u) < 0$. Le résultat est obtenu en choisissant $u = A^{-1}a$ et en remarquant que $a^T A^{-1} a > 0$.

L'unicité et l'existence de la décomposition sont donc prouvées. ◇

Proposition 2.1 *L'algorithme de Cholesky a une complexité en* $n^3/3 + O(n^2)$.

Preuve. Soit $C(n)$ le nombre d'opérations pour calculer le facteur de Cholesky d'une matrice d'ordre n. On obtient donc la relation de récurrence

$$C(n+1) = C(n) + n^2 + O(n)$$

car la résolution du système triangulaire qui calcule l est de complexité $n^2 + O(n)$. On en déduit que $C(n) = 1/3\ n^3 + O(n^2)$. ◇

Cette décomposition en blocs introduite dans la démonstration aboutit à une version par lignes, qui récursivement construit le facteur de Cholesky de dimension n à partir de celui de la matrice principale d'ordre $n-1$.

Nous décrivons maintenant deux versions par colonnes, obtenues à partir de la décomposition suivante :

$$A \;=\; \begin{pmatrix} \alpha & a^T \\ a & A' \end{pmatrix}.$$

On recherche le facteur de Cholesky L partitionné de la même manière :

$$L = \begin{pmatrix} \lambda & 0 \\ l & L' \end{pmatrix}$$

qui vérifie $LL^T = A$ ou encore

$$\lambda^2 = \alpha \quad (4)$$
$$\lambda l = a \quad (5)$$
$$l\ l^T + L'L'^T = A'. \quad (6)$$

Soit $B = A' - l\ l^T$. Puisqu'on a déjà prouvé l'existence de la factorisation de Cholesky pour A, cela entraîne que l'équation (6) a une solution, donc que B admet une factorisation de Cholesky. On peut aussi prouver que B est symétrique définie positive, puis par récurrence que B admet une factorisation de Cholesky, donc A aussi.

Ces équations se résolvent en

$$\begin{cases} \lambda = \sqrt{\alpha} \\ l = (1/\lambda)a \\ B = A' - l\ l^T \\ \text{Factorisation de Cholesky de } B \end{cases} \quad (7)$$

On peut dérouler la récurrence suivant l'algorithme suivant :

ALGORITHM 1: Cholesky (Right Looking)

```
L := triangle_inférieur(A)
do j = 1, n
    L(j,j) = sqrt(L(j,j))
    do i = j+1, n
        L(i,j) = L(i,j)/L(j,j)
    enddo
    do k = j+1, n
        do i = k, n
            L(i,k) = L(i,k) - L(k,j) * L(i,j)
        enddo
    enddo
enddo
```

Cet algorithme est appelé Right Looking (ou Fan Out), car dès qu'une colonne de L est prête, elle corrige toutes les colonnes suivantes. On peut aussi imaginer d'organiser autrement les calculs : on retarde chaque mise à jour d'une colonne au plus tard possible; ainsi une colonne de la matrice A n'est considérée que lorsque toutes les précédentes de L sont construites. Cela revient à inverser dans

l'algorithme 1 les boucles indicées par `j` et `k`. On obtient alors l'algorithme 2 appelé Left Looking (ou Fan In).

ALGORITHM 2: Cholesky (Left Looking)

$L :=$ triangle_inférieur(A)
do $k = 1, n$
  do $j = 1, k-1$
    do $i = k, n$
      $L(i,k) = L(i,k) - L(k,j) * L(i,j)$
    enddo
  enddo
  $L(k,k) = \sqrt{L(k,k)}$
  do $i = k+1, n$
    $L(i,k) = (1/L(k,k)) * L(i,k)$
  enddo
enddo

Puisque l'algorithme est composé de trois boucles emboîtées, il existe six manières de les ordonner: deux versions par lignes, deux versions par colonnes, et deux versions mixtes. Pour plus de précision, on peut consulter [4, 5].

La factorisation de Cholesky est un algorithme de la bibliothèque Lapack. Il existe une version par blocs, qui exploite des procédures de la bibliothèque BLAS3, pour augmenter l'efficacité.

2.2 Résolution d'un système symétrique défini positif

L'algorithme de résolution d'un système symétrique défini positif comporte les étapes décrites ci-dessous:

- Factorisation de Cholesky $A = LL^T$,
- Résolution du système triangulaire $Ly = b$ (descente),
- Résolution du système triangulaire $L^T x = y$ (remontée).

La bibliothèque utilisée et la complexité des étapes sont données ci-dessous:

Factorisation de Gauss avec pivot partiel

étape	bibliothèque	nombre d'opérations
factoriser $A = LL^T$	LAPACK (utilise BLAS3)	$n^3/3 + O(n^2)$
résoudre $Ly = b$	BLAS2	$O(n^2)$
résoudre $L^T x = y$	BLAS2	$O(n^2)$

L'algorithme a ainsi une complexité polynomiale cubique, ce qui permet de résoudre des systèmes avec plusieurs milliers d'inconnues. De plus, si on résout plusieurs systèmes avec la même matrice et des seconds membres différents, on n'effectue la factorisation qu'une fois. On a donc une complexité cubique pour le premier système puis quadratique pour les systèmes suivants.

2.3 Stabilité numérique de Cholesky

Un algorithme est stable numériquement si les erreurs d'arrondi n'explosent pas. On admet ici que l'algorithme de Cholesky est stable.

3 Factorisation LU

Dans le cas général où la matrice n'est pas symétrique définie positive, l'algorithme est plus compliqué. Il faut permuter les lignes ou les colonnes pour garantir l'existence de la factorisation et pour assurer la stabilité numérique.

Théorème 3.1 *Soit $A \in \mathbb{R}^{n\times n}$ une matrice inversible. Alors il existe une matrice de permutation P, une matrice L triangulaire inférieure avec $L_{ii} = 1$ et une matrice triangulaire supérieure U inversible telles que*

$$PA = LU \tag{8}$$

L'égalité (8) est appelée la factorisation de Gauss de A avec pivot partiel.

Preuve. admis. ◇

Le choix de la matrice de permutation P n'est pas unique; il existe plusieurs algorithmes, dits stratégies de pivot, qui sont basés sur l'analyse des erreurs d'arrondi. La stratégie de pivot partiel permet de garantir l'existence de la factorisation et de borner les erreurs d'arrondi. L'idée est de choisir un pivot non nul et le plus grand possible à chaque étape du processus d'élimination.

3.1 Résolution d'un système linéaire

L'algorithme de résolution d'un système linéaire, dans le cas général, comporte les étapes suivantes:

- Factorisation de Gauss $PA = LU$,
- Permutation $c = Pb$,
- Résolution du système triangulaire $Ly = c$ (descente),
- Résolution du système triangulaire $Ux = y$ (remontée).

La bibliothèque utilisée et la complexité des étapes sont données ci-dessous:

Factorisation de Gauss avec pivot partiel

étape	bibliothèque	nombre d'opérations
factoriser $PA = LU$	LAPACK (utilise BLAS3)	$2n^3/3 + O(n^2)$
résoudre $Ly = c$	BLAS2	$O(n^2)$
résoudre $Ux = y$	BLAS2	$O(n^2)$

Le nombre d'opérations de Gauss est environ le double de celui de Cholesky. Il faut noter que la matrice doit être non seulement symétrique, mais aussi définie positive, pour appliquer Cholesky.

3.2 Stabilité numérique de Gauss avec pivot

On admet ici que l'algorithme de Gauss avec pivot partiel est stable numériquement.

3.3 Factorisation par blocs

La bibliothèque LAPACK utilise une partition par blocs pour la factorisation LU, afin d'utiliser BLAS3. Nous décrivons ici un algorithme sans pivot, mais LAPACK inclut des stratégies de pivot.

Supposons que la matrice A soit d'ordre n, qu'elle soit à diagonale dominante (donc l'algorithme de Gauss sans pivot est possible) et qu'elle soit partagée en quatre blocs:

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}$$

où les blocs A_{11} et A_{22} sont carrés d'ordres respectifs m et $n - m$. La quantité $m \leq n$ représente la taille de bloc. Les m étapes de l'algorithme de Gauss sont condensées en équations matricielles avec les blocs. On décompose les matrices L et U de la même manière que A:

$$L = \begin{pmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{pmatrix} \text{ et } U = \begin{pmatrix} U_{11} & U_{12} \\ 0 & U_{22} \end{pmatrix}.$$

On écrit ensuite avec les blocs que $A = LU$:

$$\begin{aligned} A_{11} &= L_{11}U_{11} && (9) \\ A_{21} &= L_{21}U_{11} && (10) \\ A_{12} &= L_{11}U_{12} && (11) \\ B &= A_{22} - L_{21}U_{12} && (12) \\ B &= L_{22}U_{22} && (13) \end{aligned}$$

On peut alors écrire l'algorithme par blocs sous forme récursive:

ALGORITHM 3: BlockLU(A)

if $\dim(A) \leq m$ **then** {*On est arrivé au dernier bloc*}
  factorisation LU de A
else {*On mène une étape de factorisation par bloc*}
  définition des matrices $A_{11},\ A_{12},\ A_{21},\ A_{22}$
  factorisation LU de A_{11} pour obtenir L_{11} et U_{11}
  résolution de $n-m$ systèmes triangulaires pour obtenir L_{21} (10)
  résolution de $n-m$ systèmes triangulaires pour obtenir U_{12} (11)
  mise à jour du bloc restant pour obtenir B (12)
  BlockLU(B) (13)
endif

On peut ensuite dérouler l'algorithme pour en supprimer la récursivité, sans modifier le nombre d'opérations. On obtient un algorithme avec trois boucles imbriquées, comme pour Cholesky. Il existe aussi six versions de l'algorithme, suivant l'ordre des boucles.

4 Conditionnement d'un système linéaire

Nous étudions la sensibilité d'un système linéaire aux variations sur les données (conditionnement). Beaucoup d'ouvrages traitent de ce sujet, notamment [3, 8, 7, 6, 1, 2].

Nous introduisons d'emblée une définition qui va être justifiée par les résultats associés. Soit $\|.\|$ une norme matricielle subordonnée.

Définition 4.1 *Le conditionnement d'une matrice A pour les systèmes linéaires est donné par*

$$\kappa(A) = \|A\|\|A^{-1}\|.$$

Le conditionnement spectral de A est donné par

$$\kappa_2(A) = \|A\|_2\|A^{-1}\|_2 = \frac{\sigma_1}{\sigma_n},$$

où $\sigma_i,\ i = 1, \ldots, n$ sont les valeurs singulières non nulles de A.

La bibliothèqe LAPACK fournit des procédures pour estimer le conditionnement d'une matrice.

Théorème 4.1 *Soit A une matrice inversible et x la solution de $Ax = b$. Soit $\Delta A, \Delta b$ tels que*

$$\|\Delta A\| \leq \alpha\|A\|, \|\Delta b\| \leq \beta\|b\|, \alpha\ \kappa(A) < 1.$$

Alors $A + \Delta A$ est inversible. Soit $x + \Delta x$ la solution du système

$$(A + \Delta A)(x + \Delta x) = b + \Delta b,$$

alors

$$\frac{\|\Delta x\|}{\|x\|} \leq \frac{\kappa(A)}{1 - \alpha\kappa(A)}(\alpha + \beta).$$

Ce théorème correspond à la définition générale, où les données sont soit A et b (si seulement A est la donnée, alors $\beta = 0$).

La preuve du théorème nécessite le lemme suivant :

Lemme 4.1 *Si $\|F\| < 1$ alors $(I + F)$ est inversible et $\|(I + F)^{-1}\| \leq \frac{1}{1-\|F\|}$.*

Preuve.

$$A + \Delta A = A(I + A^{-1}\Delta A),$$
$$\|A^{-1}\Delta A\| \leq \kappa(A)\alpha < 1,$$

donc d'après le lemme, $I + A^{-1}\Delta A$ est inversible et

$$\|(I + A^{-1}\Delta A)^{-1}\| \leq \frac{1}{1 - \alpha\kappa(A)}.$$

Soit $y = x + \Delta x$, on a $(A + \Delta A)y = b + \Delta b$, $Ax = b$ donc $A(y - x) = \Delta b - \Delta A y$ et

$$\|y - x\| \leq \|A^{-1}\|(\|\Delta b\| + \|\Delta A\|\|y\|).$$
$$\text{or } \|\Delta b\| \leq \beta\|b\| \leq \beta\|A\|\|x\|,$$
$$\text{d'où } \|\Delta x\| \leq \kappa(A)(\beta\|x\| + \alpha\|y\|).$$

Il reste à majorer $\|y\|$. On a

$$A(I + A^{-1}\Delta A)y = A(x + A^{-1}\Delta b),$$
$$\|y\| \leq \|(I + A^{-1}\Delta A)^{-1}\|(\|x\| + \|A^{-1}\|\Delta b\|),$$
$$\|y\| \leq \tfrac{1}{1-\alpha\kappa(A)}(1 + \beta\kappa(A))\|x\|.$$

Par conséquent, $\beta\|x\| + \alpha\|y\| \leq \frac{1}{1-\alpha\kappa(A)}(\alpha + \beta)\|x\|$.

On en déduit que

$$\|\Delta x\| \leq \frac{\kappa(A)}{1 - \alpha\kappa(A)}(\alpha + \beta)\|x\|.$$

◇

4.1 Lien avec le résidu

Comme pour le cas général, il est possible de faire le lien avec le résidu.

Soit x la solution du système linéaire $Ax = b$. Nous supposons qu'un algorithme de résolution calcule le vecteur y, avec un résidu $r = b - Ay$.

Corollaire 4.1

$$\frac{\|x - y\|}{\|x\|} \leq \kappa(A)\frac{\|b - Ay\|}{\|b\|}. \tag{14}$$

Il est possible d'avoir un résultat plus fin en introduisant une perturbation de la matrice. Si $\|b - Ay\|$ est assez petit, alors y est solution d'un système linéaire perturbé et nous déterminons la plus petite perturbation possible.

Théorème 4.2 *Soit $Ax = b$ un système linéaire d'ordre n et soit $y \in \mathbb{R}^n$. Soit*

$$S = \{\alpha,\ \textit{il existe}\ \Delta A,\ \Delta b,\ \|\Delta A\| \leq \alpha\ \|A\|,\ \|\Delta b\| \leq \alpha\|b\|,\ (A+\Delta A)y = b+\Delta b\}$$

Soit

$$\eta = \frac{\|b - Ay\|}{\|A\|\|y\| + \|b\|}. \tag{15}$$

Si $\eta\kappa(A) < 1$ alors $\min_{\alpha \in S} = \eta$.

Preuve. Soit $r = b - Ay$ et soit $\alpha \in S$. Alors

$$\begin{array}{l}(A + \Delta A)y = b + \Delta b,\\ r = \Delta A y - \Delta b,\\ \|r\| \leq \alpha(\|A\|\|y\| + \|b\|),\\ \text{donc } \eta \leq \alpha.\end{array}$$

Réciproquement, soit

$$\begin{array}{l}\Delta A = \eta\ \frac{\|A\|}{\|r\|\ \|y\|}\ ry^T, \Delta b = -\eta\frac{\|b\|}{\|r\|}r.\\ \text{Alors } \|\Delta A\| = \eta\|A\|,\ \|\Delta b\| = \eta\|b\|.\\ \text{On a } (A + \Delta A)y = b - r + \Delta A y = b - r + \eta\ \frac{\|A\|\|y\|}{\|r\|}r,\\ (A + \Delta A)y = b - \frac{\|b\|}{\|A\|\|y\|+\|b\|}r = b + \Delta b,\\ \text{donc } \eta \in S.\end{array}$$

Donc η est le minimum de S. ◇

En combinant les deux théorèmes sur le conditionnement et l'analyse inverse, on obtient une estimation d'erreur pour la solution calculée y.

Corollaire 4.2 *Soit x la solution du système linéaire $Ax = b$ et soit y une solution calculée telle que $\eta\kappa(A) < 1$, où η est défini par (15). Alors*

$$\frac{\|x - y\|}{\|x\|} \leq 2\frac{\kappa(A)}{1 - \eta\kappa(A)}\eta \tag{16}$$

Le calcul de $\|r\|$ puis η et une estimation du conditionnement $\kappa(A)$ permettent donc d'estimer une borne de l'erreur sur la solution.

4.2 Résidu itératif

References

[1] F. Chatelin and V. Frayssé. *Lectures on Finite Precision Computations.* SIAM, 1995.

[2] J. Erhel. *Vitesse et précision en calcul scientifique.* habilitation à diriger des recherches, Université de Rennes 1, Juin 1992.

[3] G.H Golub and C.F Van Loan. *Matrix Computations. third edition.* John Hopkins, 1996.

[4] M. Hahad, J. Erhel, and T. Priol. Factorisation parallèle de Cholesky pour matrices creuses sur une mémoirevirtuelle partagée. Rapport de recherche 1988, INRIA, 1993.

[5] M. Hahad, J. Erhel, and T. Priol. Scheduling strategies for sparse cholesky factorization on a shared virtual memory parallel computer. In *International Conference on Parallel Processing (3)*, pages 290–297, 1994.

[6] N.J. Higham. *Accuracy and Stability of Numerical Algorithms.* SIAM, 1995.

[7] P. Lascaux and R. Theodor. *Analyse numérique matricielle appliquée à l'art de l'ingénieur, Tomes 1et 2.* Masson, 1986.

[8] G. W. Stewart and Ji-guang Sun. *Matrix perturbation theory.* Academic Press, 1990.