cours MAP : Mathématiques Appliquées
DIIC 1ere année

Jocelyne Erhel [1]

April 12, 2006

[1]INRIA, UR Rennes, Jocelyne.Erhel@inria.fr, http://www.irisa.fr/aladin/perso/erhel/

Contents

Chapter 1

Introduction

1.1 Notations

On note $\mathbb{R}^n$ un espace vectoriel de dimension n.
La base canonique de $\mathbb{R}^n$ est $(e_1, \ldots, e_n)$. Un vecteur $x \in \mathbb{R}^n$, de composantes $x_1, \ldots, x_n$, est noté $x = (x_i)$.
Une matrice $A \in \mathbb{R}^{m \times n}$ est notée $A = (a_{ij})$.
Le j^{eme} vecteur colonne de A est $a_j = Ae_j$.
Un système de n vecteurs $u_1, \ldots, u_n \in \mathbb{R}^m$ est noté sous forme matricielle $U = (u_1 \ldots u_n) \in \mathbb{R}^{m \times n}$, avec u_j le j^{eme} vecteur colonne de U.
On note $Im(U)$ ou $Vect(U)$ le sous-espace vectoriel engendré par les vecteurs colonnes de U.

Il est possible de définir plusieurs normes dans l'espace vectoriel $\mathbb{R}^n$. La norme vectorielle euclidienne est définie par

$$\|x\|_2 = (\sum_i x_i^2)^{\frac{1}{2}}.$$

La norme vectorielle infinie est définie par

$$\|x\|_\infty = \max_i |x_i|.$$

1.2 Objectif du cours

L'objectif de ce cours est la résolution de systèmes linéaires avec soit autant de données que d'inconnues, soit plus de données que d'inconnues (système sur-déterminé), soit plus d'inconnues que de données (système sous-déterminé). Un système linéaire est un ensemble de m équations linéaires à n inconnues. Il s'écrit sous forme matricielle

$$\text{Trouver } x \in \mathbb{R}^n, \text{ tel que } Ax = b,$$

où $A \in \mathbb{R}^{m \times n}$ et $b \in \mathbb{R}^m$ sont donnés. Selon les cas, un tel système peut ne pas avoir de solution, avoir une solution unique, avoir une infinité de solutions. Pour étudier le cas général et pour définir des algorithmes de résolution, on reformule le système sous la forme d'un problème aux moindres carrés :

$$\text{Trouver } x \in \mathbb{R}^n, \text{ tel que } \|Ax - b\|_2 \text{ soit minimum.}$$

Ces problèmes aux moindres carrés interviennent dans beaucoup de domaines tels que le traitement du signal, le traitement d'images, l'analyse statistique, etc.

Les derniers chapitres sont consacrés à une notion qui généralise celle de valeur propre. Cela permet de résoudre non seulement les problèmes aux moindres carrés, mais aussi beaucoup d'autres problèmes linéaires. Cette propriété, appelée la Décomposition en Valeurs Singulières, s'applique dans des domaines variés comme les télécommunications, la réalité virtuelle, les moteurs de recherche, etc.

Les modèles mathématiques sont définis dans l'ensemble $\mathbb{R}$ des nombres réels. Mais les calculs se font sur ordinateur, qui ne disposent que d'une mémoire finie et ne peuvent donc pas stocker l'infinité des nombres réels. Ce cours introduit les notions de calcul flottant et d'erreurs de calcul.

1.3 Application au calibrage d'un modèle

Une des premières applications à l'origine des problèmes aux moindres carrés est le calibrage d'un modèle mathématique. Ce modèle est une représentation d'un phénomène que l'on étudie. Supposons qu'il soit défini par une fonction $f(t)$, où la fonction f est caractérisée par un ensemble de n paramètres $x \in \mathbb{R}^n$ que l'on ne connait pas. On dispose d'un ensemble de m observations ou mesures b_i aux "temps" $t_i,\ i = 1, \ldots, m$. La calibration du modèle consiste à trouver x tel que b_i soit proche de $y_i = f(t_i)$. Une solution consiste à rendre minimum la somme des carrés des erreurs. Mathématiquement, cela s'écrit :

$$\text{Trouver } x \in \mathbb{R}^n, \text{ tel que } \sum_{i=1}^{m} (b_i - y_i)^2 \text{ soit minimum.}$$

Si la fonction f est une fonction linéaire par rapport à x, elle s'écrit

$$f(t) = \sum_{j=1}^{n} x_j \Phi_j(t),$$

et $y_i = \sum_{j=1}^{n} x_j \Phi_j(t_i)$. Sous forme matricielle, on a $y = Ax$ et le problème devient

$$\text{Trouver } x \in \mathbb{R}^n, \text{ tel que } \|b - Ax\|_2 \text{ soit minimum.}$$

1.3.1 Exemple : interpolation et approximation polynomiale

Dans l'interpolation ou l'approximation polynomiales, on cherche f sous la forme d'un polynôme de degré $n-1$, défini par

$$f(t) = x_1 + x_2 t + \ldots + x_n t^{n-1} = \sum_{j=1}^{n} x_j t^{j-1}.$$

La matrice $A = (a_{ij})$ est définie par $a_{ij} = t_i^{j-1}$.

Si $m = n$, on parle d'**interpolation polynomiale**, on obtient ainsi un **système linéaire**

$$Ax = b$$

de n équations à n inconnues. Voir le chapitre et la section sur les systèmes linéaires.

Sinon, on cherche une **approximation polynomiale au sens des moindres carrés** et on résout le **problème linéaire aux moindres carrés**

$$\min_x \|Ax - b\|_2.$$

Voir les chapitres sur les problèmes aux moindres carrés.

1.4 Application à un moteur de recherche

On dispose de m termes et de n documents, avec des poids a_{ij} du terme i dans le document j. On construit la matrice $A = (a_{ij})$ qui est la base de recherche. La j^{eme} colonne d_j est l'ensemble de poids attribués au document j.

Une requête est un ensemble de valeurs $q = (q_i)$ attribuées aux termes i.

L'objectif est de mesurer l'angle entre la requête et chaque document j, autrement dit les quantités

$$\cos(\theta_j) = \frac{d_j^T q}{\|d_j\|_2 \|q\|_2}$$

où $d_j^T q$ est le produit scalaire des deux vecteurs.

Si l'angle θ_j est petit, autrement dit si $\cos(\theta_j)$ est proche de 1, alors la probabilité que le document j contienne des termes de la requête q est élevée.

Il faut obtenir une réponse rapide, même sur des bases gigantesques avec des milliers ou des millions de documents et de termes. Pour cela, on utilise une approximation de la base A par une base beaucoup plus petite A_k et on calcule une approximation des coefficients $\cos(\theta_j)$ avec la base A_k. Voir le chapitre sur la décomposition en valeurs singulières et la section sur l'approximation de rang k.

Chapter 2

Bases de l'algèbre linéaire

Ce chapitre introduit les opérations de base sur l'algèbre des matrices. Dans le cas des matrices carrées, il définit les notions de matrice inversible et de matrice singulière. Les notions de complexité et de performances sont introduites. Le chapitre termine par la description des bibliothèques BLAS et LAPACK.

2.1 Matrices et vecteurs

2.1.1 Matrices particulières

Une matrice $A \in \mathbb{R}^{m \times n}$ est carrée d'ordre n si $n = m$.

La matrice identité d'ordre k, dans $\in \mathbb{R}^{k \times k}$, vaut $I_k = (e_1 \dots e_k)$.

Une matrice carrée D est diagonale si les seuls éléments non nuls sont sur la diagonale ; elle est notée $D = diag(d_i)$.

Une matrice carrée L triangulaire inférieure si les seuls éléments non nuls sont dans le triangle inférieur ; on définit de même une matrice carrée U triangulaire supérieure.

Une matrice tridiagonale a trois diagonales non nulles, une matrice bidiagonale a deux diagonales non nulles.

2.1.2 Opérations sur les matrices

L'ensemble des matrices $\mathbb{R}^{m \times n}$ est un espace vectoriel de dimension mn.

Soit $A \in \mathbb{R}^{m \times n}$ et $B \in \mathbb{R}^{n \times p}$; le produit $C = AB \in \mathbb{R}^{m \times p}$ est défini par $c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}$.

L'ensemble des matrices carrées $\mathbb{R}^{n \times n}$ est un anneau. Attention, cet anneau n'est pas commutatif (il existe A et B tels que $AB \neq BA$). Attention, l'anneau n'est pas intègre, car il existe des diviseurs de zéro.

Le produit de deux matrices diagonales est une matrice diagonale.

Le produit de deux matrices triangulaires est une matrice triangulaire.

2.1.3 Matrices carrées symétriques

La transposée $A^T \in \mathbb{R}^{n \times m}$ d'une matrice $A \in \mathbb{R}^{m \times n}$ est la matrice obtenue en interchangeant les lignes et les colonnes.

Une matrice carrée A est symétrique si $A = A^T$.

Les matrices A^TA et AA^T sont symétriques.
On a $(A^T)^T = A$ et $(AB)^T = B^TA^T$.

2.1.4 Partitions par blocs

Une matrice par blocs est définie par une partition où les éléments scalaires sont regroupés dans des sous-matrices ; on note $A = A_{ij}$.

On définit les mêmes règles d'opérations, en respectant les dimensions dans les produits. Attention à l'ordre des blocs dans les produits.

2.2 Normes matricielles

La norme matricielle euclidienne est associée à la norme vectorielle euclidienne et est définie par

$$\|A\|_2 = \max_{x \neq 0} \frac{\|Ax\|_2}{\|x\|_2} = \max_{\|x\|_2=1} \|Ax\|_2.$$

La norme matricielle infinie est associée à la norme vectorielle infinie et est définie par

$$\|A\|_\infty = \max_{x \neq 0} \frac{\|Ax\|_\infty}{\|x\|_\infty} = \max_{\|x\|_\infty=1} \|Ax\|_\infty = \max_{1 \leq i \leq m} \sum_{j=1}^{n} |a_{ij}|.$$

La norme matricielle de Frobenius est la norme prise au sens de l'espace vectoriel de dimension mn et est définie par

$$\|A\|_F = (\sum_{i,j} a_{ij}^2)^{\frac{1}{2}}.$$

Remarque 2.2.1 *La norme matricielle euclidienne n'est pas simple à calculer, contrairement à la norme infinie et la norme de Frobenius. Voir le chapitre sur la SVD.*

Proposition 2.2.1

$$|x^Ty| \leq \|x\|_2\|y\|_2$$
$$\|AB\|_2 \leq \|A\|_2\|B\|_2$$
$$\|e_j\|_2 = 1$$
$$\|I\|_2 = 1$$
$$\|D\|_2 = max_i|d_i|$$

2.3 Orthogonalité dans $\mathbb{R}^n$

Définition 2.3.1 $x \perp y \Leftrightarrow x^Ty = 0$

Définition 2.3.2 $\cos(angle(x,y)) = \frac{x^Ty}{\|x\|_2\|y\|_2}$

Proposition 2.3.1 *Théorème de Pythagore :*
Si $x \perp y$, alors $\|x+y\|_2^2 = \|x\|_2^2 + \|y\|_2^2$.

Définition 2.3.3 *$U = (u_1 \cdots u_k) \in \mathbb{R}^{n \times k}$, $k \leq n$ est un système orthonormé ssi $u_i^Tu_j = \delta_{ij}$ ssi $U^TU = I_k$.*

Remarque 2.3.1 *Attention, si $k < n$, on a $UU^T \neq I_n$. Voir preuve plus loin.*

Proposition 2.3.2 *Un système orthonormé forme un système libre.*

Remarque 2.3.2 *$(e_1, \ldots, e_n)$ est une base orthonormée de $\mathbb{R}^n$.*

Définition 2.3.4 *Soit S un sous-espace vectoriel de $\mathbb{R}^n$. L'orthogonal de S est défini par*

$$S^\perp = \{y / y^T x = 0, \quad \forall x \in S\}.$$

Proposition 2.3.3 *$S^\perp$ est un sous-espace vectoriel et les sous-espaces S et $S^\perp$ sont supplémentaires.*

Proposition 2.3.4 *Théorème de la base incomplète. Soit U un système orthonormé de taille k. On peut compléter U par U_1 de taille $n-k$, pour former une base orthonormée de $\mathbb{R}^n$. Le système U_1 est une base orthonormée de $U^\perp$. Alors $U_1^T U_1 = I_{n-k}$ et $U^T U_1 = 0$. Tout vecteur x s'écrit*

$$x = UU^T x + U_1 U_1^T x.$$

Définition 2.3.5 *$Q \in \mathbb{R}^{n \times n}$ est orthogonale ssi $Q^T Q = I_n$. Alors $QQ^T = I_n$. Les colonnes de Q forment une base orthonormée de $\mathbb{R}^n$.*

Proposition 2.3.5 *$U \in \mathbb{R}^{m \times n}$, $n \leq m$ système orthonormé, alors*

$$\forall x \in \mathbb{R}^n, \quad \|Ux\|_2 = \|x\|_2.$$

$A \in \mathbb{R}^{m \times n}, U \in \mathbb{R}^{p \times m}$ système orthonormé, alors

$$\|UA\|_2 = \|A\|_2.$$

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

$$\|AQ\|_2 = \|A\|_2.$$

Preuve. $\|Ux\|_2^2 = (Ux)^T(Ux) = x^T(U^T U)x = x^T x = \|x\|_2^2$
$\|UA\|_2 = \max_{\|x\|_2=1} \|UAx\|_2 = \max_{\|x\|_2=1} \|Ax\|_2 = \|A\|_2$.
$\|AQx\|_2 = \max_{\|x\|_2=1} \|AQx\|_2 = \max_{\|y\|_2=1} \|Ay\|_2 = \|A\|_2$ car Q est inversible $(\forall y, \exists x, Qx = y)$ et $\|Qx\| = \|x\|$. ◇

Remarque 2.3.3 *Attention, si $U \in \mathbb{R}^{n \times p}$ orthonormé, on peut avoir $\|AU\|_2 \neq \|A\|_2$.*

2.4 Image, noyau, rang d'une matrice

$A \in \mathbb{R}^{m \times n}$.

Définition 2.4.1 $Im(A) = \{y \in \mathbb{R}^m / y = Ax\} = Vect(a_1, a_2, \cdots, a_n)$
$ker(A) = \{x \in \mathbb{R}^n / Ax = 0\}$

Proposition 2.4.1 *Im(A) est un sev de $\mathbb{R}^m$ et $ker(A)$ est un sev de $\mathbb{R}^n$.*

Définition 2.4.2 $rang(A) = dim(Im(A))$

Proposition 2.4.2 $Im(A)^\perp = ker(A^T)$ *et* $Im(A^T) = ker(A)^\perp$.

Preuve. $y \in Im(A)^\perp \Leftrightarrow \forall x, \quad (Ax)^T y = 0 \Leftrightarrow \forall x, \quad x^T(A^T y) = 0 \Leftrightarrow A^T y = 0 \Leftrightarrow y \in ker(A^T)$. ◇

Proposition 2.4.3 $rang(A) = rang(A^T)$
$dim(ker(A)) + rang(A)) = n$
$dim(ker(A^T)) + rang(A^T)) = m$
$rang(A) \leq min(m, n)$.

Preuve. Soit $r = rang(A^T)$, on va montrer que $rang(A) \geq r$. Par transposition, on en déduira que $r = rang(A)$.
Soit $X = \{x_1, \ldots, x_r\}$ une base de $Im(A^T)$ et $Y = AX$. Par construction, $Y \in Im(A)$. On va montrer que Y est un système libre, ce qui implique que $r \leq rang(A)$. Pour cela, on va montrer que $Yv = 0 \Rightarrow v = 0$.

$Yv = 0 \Rightarrow AXv = 0 \Rightarrow Xv \in ker(A)$. Or $ker(A) = Im(A^T)^\perp$ donc $Xv \in Im(A^T)^\perp$. Mais X est une base de $Im(A^T)$ donc $Xv \in Im(A^T)$. Donc $Xv = 0$ et puisque X est une base, $v = 0$.

D'après la proposition précédente, on a $dim(ker(A)) + rang(A^T) = n$, on en déduit l'égalité avec $rang(A)$. ◇

Proposition 2.4.4 $rang(A) = n$ *équivaut à* $ker(A) = \{0\}$.

Preuve. évident. ◇

Proposition 2.4.5 $rang(AB) \leq rang(A),\ rang(AB) \leq rang(B)$

Preuve. $ker(B) \subset ker(AB)$ donc $rang(AB) \leq rang(B)$.
$rang((AB)^T) = rang(B^T A^T) = rang(AB) \leq rang(A^T) = rang(A)$. ◇

Proposition 2.4.6 *Soit* $U = (u_1 \ldots u_k)$ *et* $V = (v_1 \ldots v_k)$ *deux systèmes libres de* $\mathbb{R}^m$ *et de* $\mathbb{R}^n$ *respectivement, alors* $UV^T \in \mathbb{R}^{m \times n}$ *et*

$$ker(UV^T) = V^\perp,\ Im(UV^T) = Im(U),\ rang(UV^T) = k.$$

Réciproquement, si A *est une matrice de rang* k*, il existe* $U = (u_1 \ldots u_k)$ *base de* $Im(A)$ *et* $V = (v_1 \ldots v_k)$ *base de* $ker(A)^\perp$ *tels que* $A = UV^T$.

Preuve. Soit $A = UV^T$, alors $x \in ker(A) \Leftrightarrow UV^T x = 0 \Leftrightarrow V^T x = 0 \Leftrightarrow x \in V^\perp$ donc $ker(A) = V^\perp$. On en déduit que $rang(A) = k$ car $dim(V^\perp) = n - k$. D'autre part, $Im(A) \subset Im(U)$ d'où, par égalité des dimensions, $Im(A) = Im(U)$.

Réciproquement, soit $V \in \mathbb{R}^{n \times k}$ une base orthonormée de $ker(A)^\perp$, complétée par V_1 dans $ker(A)$. Alors $\forall x,\ x = VV^T x + V_1 V_1^T x$ et $Ax = AVV^T x$. Soit $U = AV$, alors $Ax = UV^T x$ donc $A = UV^T$. De plus, U est un système libre car $Uy = 0 \Leftrightarrow AVy = 0 \Leftrightarrow Vy \in ker(A) \cap ker(A)^\perp \Leftrightarrow y = 0$, donc U est une base de $Im(A)$. ◇

Remarque 2.4.1 *En particulier, les matrices de rang* 1 *sont de la forme* uv^T*, où* u *et* v *sont deux vecteurs non nuls.*

Définition 2.4.3 *Soit* $A \in \mathbb{R}^{m \times n}$ *une matrice rectangulaire avec* $m \geq n$ *;* A *est dite de rang plein si* $rang(A) = n$.

2.5 Valeurs propres et vecteurs propres

Nous considérons ici des matrices carrées d'ordre n. Avant de définir la notion de valeur propre, nous devons définir le déterminant d'une matrice carrée. Nous choisissons une définition récursive.

Définition 2.5.1 *Soit $A \in \mathbb{R}^{n\times n}$ une matrice carrée. Le déterminant de A est défini par*

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

où A_{1j} est la matrice carrée d'ordre $n-1$ obtenue en supprimant la ligne i et la colonne j dans la matrice A. D'autre part, $det(a) = a$ pour une matrice a d'ordre 1.

Proposition 2.5.1 $det(AB) = det(BA)$
$det(A^T) = det(A)$
$det(\alpha A) = \alpha^n A$
$det(I) = 1$
$det(A) \neq 0 \Leftrightarrow rang(A) = n.$

Proposition 2.5.2 *La fonction de la variable complexe λ définie par $det(A - \lambda I)$ est un polynôme de degré n. On l'appelle le polynôme caractéristique de A.*

Définition 2.5.2 *Le nombre complexe λ est valeur propre de la matrice carrée A s'il existe un vecteur x non nul tel que $Ax = \lambda x$.*

Proposition 2.5.3 *Les valeurs propres de A sont les racines du polynôme caractéristique. Il y a donc n valeurs propres, distinctes ou non. Les valeurs propres de A et de A^T sont les mêmes.*

Preuve. $Ax = \lambda x$ équivaut à $(A - \lambda I)x = 0$ équivaut à $x \in ker(A - \lambda I)$. Donc x non nul existe si et seulement si $rang(A - \lambda I) < n$ si et seulement si $det(A - \lambda I) = 0$.

Tout polynôme de degré n dans l'ensemble des nombres complexes a n racines complexes, comptées avec leur multiplicité.

On a $det(A - \lambda I) = det(A^T - \lambda I)$. ◇

Définition 2.5.3 *Le spectre de A est l'ensemble des n valeurs propres de A. Le rayon spectral de A est le plus grand module dans le spectre. La trace de A est la somme des éléments diagonaux; c'est aussi la somme des valeurs propres.*

2.6 Notions de complexité et de performances

Les algorithmes de calcul sont caractérisés par leur complexité arithmétique, mesurée par le nombre d'opérations (additions, multiplications, etc) sur des réels, ainsi que par leur coût de stockage, mesuré par le nombre de variables réelles. Le stockage des entiers et les calculs entiers sont d'un coût marginal, qui est ignoré. En pratique, les opérations arithmétiques et le stockage se font sur des nombres flottants (voir chapitre sur l'arithmétique flottante).

Les performances d'un algorithme se mesurent par sa vitesse de calcul. Celle-ci dépend de la complexité mais aussi de l'exploitation de l'architecture de l'ordinateur, notamment du parallélisme interne et de la hiérarchie des mémoires.

Pour la complexité, on donnera souvent le terme prédominant, du plus grand ordre, sous la forme $O(cn^k)$. Cela signifie que le nombre d'opérations divisé par cn^k tend vers 0 quand n tend vers l'infini. Par exemple, pour une complexité polynomiale en n, le terme prépondérant est celui du plus grand degré dans le polynôme.

2.7 Bibliothèques BLAS et LAPACK

Les opérations de base d'algèbre linéaire sont regroupées dans une bibliothèque numérique appelée BLAS : Basic Linear Algebra Subroutines. Cette bibliothèque est souvent fournie par le constructeur et optimisée pour une architecture donnée. Les opérations sont divisées en trois niveaux, appelés BLAS1, BLAS2, BLAS3. L'optimisation concerne l'ordre des opérations et l'accès à la mémoire, pour exploiter au mieux la hiérarchie (mémoire principale, mémoire cache, etc). Les performances (vitesse de calcul) sont d'autant meilleures que le niveau est élevé.

Les opérations plus complexes sont regroupées dans la bibliothèque numérique appelée LAPACK : Linear Algebra Package. Les opérations de LAPACK utilisent au maximum les opérations BLAS, surtout BLAS3, qui est le plus performant en temps de calcul.

2.7.1 Opérations BLAS1

Ce niveau concerne les opérations entre vecteurs. Quelques exemples :

- combinaison linéaire de vecteurs $z = a * x + b * y$,
- produit scalaire de vecteurs $a = x^T y$.

Toutes les opérations BLAS1 ont une complexité en $O(n)$ opérations arithmétiques (sur les nombres réels) et un accès mémoire en $O(n)$ mots (réels).

Il n'y a qu'un niveau de boucle.

2.7.2 Opérations BLAS2

Ce niveau concerne les opérations entre matrices et vecteurs. Quelques exemples :

- produit matrice-vecteur $y = a * y + b * Ax$,
- produit extérieur de vecteurs $A = xy^T$.

Toutes les opérations BLAS2 ont une complexité en $O(n^2)$ et un accès mémoire en $O(n^2)$.

Il y a deux niveaux de boucle imbriqués, ce qui permet de construire deux variantes suivant l'ordre des boucles. On peut ainsi choisir un calcul par lignes ou par colonnes. On peut aussi définir une partition de la matrice et effectuer les opérations par blocs, pour optimiser l'accès hiérarchique à la mémoire.

2.7.3 Opérations BLAS3

Ce niveau concerne les opérations entre matrices. Quelques exemples :

- produit de matrices $C = a * C + b * AB$,

- produit $C = AA^T$.

La complexité est ici en $O(n^3)$, avec un accès mémoire en $O(n^2)$.

Les trois niveaux de boucle imbriqués permettent de définir six variantes suivant l'ordre des boucles. On peut choisir de parcourir chacune des matrices par lignes ou par colonnes. Comme dans le niveau BLAS2, on optimise l'accès à la mémoire en définissant une partition par blocs. De plus, comme l'algorithme fait plus d'opérations arithmétiques que d'accès aux données, on peut ré-utiliser des valeurs qui sont dans la mémoire cache. C'est pour cette raison que le niveau 3 est le plus performant et permet pratiquement d'atteindre la performance maximale d'une machine.

2.7.4 bibliothèque LAPACK

La bibliothèqe LAPACK regroupe la plupart des algorithmes d'algèbre linéaire. LAPACK utilise une partition par blocs des matrices, de façon à exploiter les performances des opérations BLAS3. Les algorithmes de résolution des systèmes linéaires et des problèmes aux moindres carrés sont codés dans LAPACK.

Chapter 3

Arithmétique flottante

3.1 Erreurs de calcul

3.1.1 Sources d'erreur

Résoudre un problème signifie trouver une solution x qui dépend de données a et de formules. Ces formules sont issues en général d'une modélisation d'un problème scientifique (en physique, chimie, biologie, économie, etc). Nous ne discutons pas ici de l'erreur de modélisation qui est l'écart entre le phénomène réel et le phénomène simulé. Dans la suite, les variables x et a appartiennent à des espaces vectoriels normés et il existe une fonction F telle que le problème s'écrive

$$F(x, a) = 0.$$

En général, il n'est possible de résoudre qu'un problème approché. Ce problème discret, formulé en dimension finie, doit être résolu par un algorithme, qui peut être direct ou itératif. Enfin, l'algorithme est exécuté sur un ordinateur avec une précision finie. Dans le résultat d'un calcul scientifique, il apparaît ainsi plusieurs sources d'erreur :

- l'erreur due à l'approximation des données,
- l'erreur d'approximation ou de discrétisation,
- l'erreur due à l'algorithme s'il est itératif,
- l'erreur d'arrondi due à la précision finie.

Les différentes sources d'erreur interfèrent avec les erreurs d'arrondi lors d'un calcul. Par exemple, la résolution d'un problème très sensible aux variations sur les données donne lieu à des calculs avec de grandes erreurs d'arrondi.

3.1.2 Mesures de l'erreur

Les exemples précédents montrent qu'il ne faut pas avoir une confiance aveugle dans les résultats d'un calcul. Tout calcul devrait s'accompagner d'une estimation des erreurs d'approximation commises. Pour mesurer celles-ci, nous définissons les erreurs absolues et relatives, ainsi que la notion de chiffres significatifs.

Définition 3.1.1 *Soit $x \in \mathbb{R}$ et $\tilde{x} \in \mathbb{R}$ une approximation de x. L'erreur absolue sur $\tilde{x}$ est $|x - \tilde{x}|$. Si $x \neq 0$, l'erreur relative est $|x - \tilde{x}|/|x|$.*

Définition 3.1.2 *Le nombre de chiffres significatifs de x est le nombre de chiffres à partir du premier chiffre non nul.*

Par exemple, le nombre de chiffres significatifs de $x = 0.0034560$ est 5. L'erreur relative sur $\tilde{x} = 0.00346$ vaut environ $1.16 \quad 10^{-3}$ et le nombre de chiffres significatifs exacts est ici 2.

Dans le cas de vecteurs, on définit les erreurs sur les normes ou les erreurs composante par composante.

Définition 3.1.3 *Soit $x \in \mathbb{R}^n$, $\tilde{x} \in \mathbb{R}^n$ une approximation de x et $\|.\|$ une norme définie sur $\mathbb{R}^n$. L'erreur absolue sur la norme est $\|x - \tilde{x}\|$.*
Si $x \neq 0$, l'erreur relative sur la norme est $\|x - \tilde{x}\|/\|x\|$.
Si $x_i \neq 0,\ i = 1, \ldots, n$, l'erreur relative composante par composante est $\max_{1 \leq i \leq n} |x_i - \tilde{x}_i|/|x_i|$.

3.2 Arithmétique flottante

Avant d'analyser l'impact des erreurs d'arrondi et des différentes sources d'erreur sur des calculs, nous allons détailler les caractéristiques arithmétiques d'un ordinateur. L'arithmétique flottante est définie en détails dans plusieurs ouvrages, citons par exemple [1, 4, 5, 6, 8, 9].

Un logiciel de calcul scientifique utilise plusieurs types de variables, qui correspondent à un codage spécifique. Les types arithmétiques les plus usuels sont le type *entier*, le type *réel* et le type *complexe*. Un complexe est formé de deux réels, la partie réelle et la partie imaginaire. Un réel est codé par un nombre flottant.

3.2.1 Format flottant

Définition 3.2.1 *Un système flottant est défini par une base b, un exposant minimal e_{min}, un exposant maximal e_{max}, un nombre de chiffres p. L'ensemble des nombres flottants normalisés est noté $\mathcal{F}$. Un nombre flottant non nul normalisé est*

$$x = (-1)^s m b^e$$

où $s \in \{0, 1\}$ est le bit de signe, l'exposant e est un entier tel que $e_{min} \leq e \leq e_{max}$, la mantisse m vérifie $1 \leq m < b$ et s'écrit

$$\left\{ \begin{array}{l} m = a_0 + a_1 * b^{-1} + a_2 * b^{-2} + \ldots + a_p * b^{-p} \\ avec \ \ 0 \leq a_i \leq b - 1 \ i = 0, ..p \ \ et \ \ a_0 \neq 0 \end{array} \right.$$

Par convention, le nombre 0 a un exposant et une mantisse nuls.

Proposition 3.2.1 *L'ensemble $\mathcal{F}$ est symétrique par rapport à 0.*
Le plus grand nombre de $\mathcal{F}$ vaut

$$x_{max} = b^{e_{max}}(b - b^{-p})$$

et le plus petit nombre strictement positif vaut

$$u = b^{e_{min}}$$

de sorte que

$$\mathcal{F} \subset \mathbb{R}_{\mathcal{F}} = [-x_{max}, -u] \cup \{0\} \cup [u, x_{max}].$$

L'ensemble des nombres entiers de l'intervalle $[-x_{max}, x_{max}]$ *est inclus dans* $\mathcal{F}$.

Remarque 3.2.1 *Nous ne définissons pas ici les nombres dénormalisés qui sont des nombres flottants dans l'intervalle* $]-u, u[$.

L'écart minimal entre deux mantisses consécutives est b^{-p}. Ce nombre caractérise la précision du système flottant.

Définition 3.2.2 *Le nombre* $\epsilon = b^{-p}$ *est appelé la précision du système flottant.*

La proposition suivante détermine les deux nombres flottants qui encadrent un nombre réel.

Proposition 3.2.2 *Tout réel de* $\mathbb{R}_{\mathcal{F}}$ *est encadré par deux flottants consécutifs. Soit*

$$x^- = \max\{y \in \mathcal{F},\ y \leq x\} \text{ et } x^+ = \min\{y \in \mathcal{F},\ y \geq x\}.$$

Il n'existe pas de flottant entre x^- *et* x^+ *et* $0 \leq x^+ - x^- \leq |x|\epsilon$.

Preuve. Si $x \in \mathcal{F}, x^- = x^+ = x$.

Soit $x \in \mathbb{R}_{\mathcal{F}},\ x \notin \mathcal{F},\ x > u$, soit $x^- = b^e m$ alors $m \geq 1,\ x^- < x$ et $x^+ = b^e(m+\epsilon) = x^- + b^e\epsilon$. Donc $0 \leq x^+ - x^- = b^e\epsilon \leq b^e m\epsilon \leq x\epsilon$.

Le cas $x < u$ se traite de la même façon. ◇

On peut remarquer que le nombre flottant qui précède l'entier 1 vaut $1 - \epsilon/b$ et celui qui le suit vaut $1 + \epsilon$.

3.2.2 Arrondis

Définition 3.2.3 *Un arrondi est une application* fl *de* $\mathbb{R}_{\mathcal{F}}$ *dans* $\mathcal{F}$ *vérifiant les propriétés de monotonie et de projection suivantes.*

$$\begin{array}{l} \forall x, y \in \mathbb{R}_{\mathcal{F}}, \quad x \leq y \Rightarrow fl(x) \leq fl(y), \\ \forall x \in \mathcal{F}, \quad fl(x) = x. \end{array}$$

Tout arrondi est défini à l'aide des deux flottants x^+ et x^-.

Proposition 3.2.3 *Tout arrondi* fl *vérifie*

$$\begin{array}{l} \forall x \in \mathbb{R}_{\mathcal{F}}, \quad fl(x) = x^+ \text{ ou } fl(x) = x^- \\ \forall x \in \mathbb{R}_{\mathcal{F}}, \quad |fl(x) - x| \leq \epsilon|x| \end{array}$$

Preuve. Par définition, $x^- \leq x \leq x^+$, d'où par monotonie et par projection $x^- \leq fl(x) \leq x^+$. Or il n'existe aucun flottant strictement compris entre x^+ et x^- donc $fl(x)$ est égal à l'un des deux.

la deuxième partie se déduit de la proposition 3.2.2. ◇

3.2.3 Opérations arithmétiques

L'ensemble $\mathcal{F}$ est non clos pour les opérations arithmétiques de $\mathbb{R}$ et il faut donc définir le résultat flottant d'une opération flottante. Soit fl l'arrondi choisi.

Définition 3.2.4 *Soit $\cdot$ l'une des 4 opérations $\{+,-,\times,/\}$ dans $\mathbb{R}$. L'opération flottante correspondante $\odot$ est correcte pour l'arrondi fl si elle satisfait la propriété*

$$\forall x, y \in \mathcal{F} \text{ tels que } x \cdot y \in \mathbb{R}_{\mathcal{F}}, \quad x \odot y = fl(x \cdot y), \tag{3.1}$$

où fl désigne un mode d'arrondi. Autrement dit, le résultat flottant est l'arrondi du résultat exact, s'il n'y a pas de dépassement de capacité.

Proposition 3.2.4 *Si l'opération $\odot$ est correcte, alors*

$$\forall x, y \in \mathcal{F} \text{ tels que } x \cdot y \in \mathbb{R}_{\mathcal{F}}, \quad x \odot y = (x \cdot y)(1+\alpha) \text{ avec } |\alpha| \leq \epsilon.$$

Proposition 3.2.5 *Les propriétés des opérations sur les nombres réels ne sont pas toutes vérifiées avec les nombres flottants.*

- *Les opérations flottantes sont commutatives,*
- *elles ne sont pas associatives,*
- *elles ne sont pas distributives,*
- 0 *est élément neutre de l'addition flottante et* 1 *est élément neutre de la multiplication flottante,*
- *l'opposé de $x \in \mathcal{F}$ existe et vaut $-x$,*
- *par contre $x \in \mathcal{F}$ n'a pas toujours d'inverse dans $\mathcal{F}$.*

Par exemple, avec Matlab, $49 \times (1/49) = 1 - \epsilon/2$.

3.2.4 Exceptions

Il peut arriver que le résultat exact d'une opération ne soit pas dans $\mathbb{R}_{\mathcal{F}}$. Pour fermer le système arithmétique flottant, l'ensemble $\mathcal{F}$ est doté de nombres spéciaux, notés $+\infty, -\infty, NaN$ (Plus l'infini, Moins l'infini, Not a Number). Le nombre 0 a un signe. Les opérations sont définies avec ces nombres spéciaux. Une opération ayant pour résultat un des nombres spéciaux déclenche une exception.

Définition 3.2.5 *Soit $z \in \mathbb{R}$ le résultat exact d'une opération arithmétique entre deux flottants x et y ; il y a overflow si $|z| > x_{max}$ et underflow si $|z| < u$. Le résultat flottant est respectivement ∞ et* 0 *avec le signe adéquat. L'opération est invalide si le résultat ne peut être ni un flottant, ni l'infini. Le résultat flottant d'une opération invalide est NaN.*

Remarque 3.2.2 *Les nombres dénormalisés permettent de mettre en place un underflow graduel vers* 0.

Par exemple, les opérations $\frac{0}{0}$ et $(+\infty - \infty)$ sont invalides, avec comme résultat NaN. L'opération $\frac{1}{0}$ déclenche un overflow avec comme résultat $\pm\infty$ et l'opération $\frac{1}{\infty}$ déclenche un underflow avec comme résultat 0.

Type	Base b	Mantisse p	Exposant e_{min}	Exposant e_{max}
Simple	2	24	-126	+127
Double	2	53	-1022	+1023

Table 3.1: Formats simple et double précision de la norme IEEE-754

Type	précision ϵ	Seuil d'overflow x_{max}	Seuil d'underflow u
Simple	10^{-7}	10^{+38}	10^{-38}
Double	10^{-16}	10^{+308}	10^{-308}

Table 3.2: Valeurs caractéristiques approchées des formats simple et double précision de la norme IEEE-754

3.2.5 Norme IEEE-754

La norme IEEE-754 [13], publiée en 1985, spécifie un système flottant et l'arithmétique flottante associée. La plupart des constructeurs respectent aujourd'hui cette norme, ce qui facilite grandement le transfert de logiciels entre machines. Cette norme permet aussi d'établir des preuves sur le comportement des erreurs d'arrondi dans un algorithme.

La norme IEEE-754 spécifie deux formats, appelés simple et double précision, qui sont résumés dans les tables 3.2.5 et 3.2.5. La norme définit 4 arrondis.

Définition 3.2.6 *L'arrondi vers moins l'infini de x vaut x^-,*
l'arrondi vers plus l'infini de x vaut x^+,
l'arrondi vers zéro ou par troncature de x vaut x^+ si $x < 0$ et x^- si $x > 0$,
l'arrondi au plus près de x vaut x^- si $x - x^- < x^+ - x$, il vaut x^+ sinon. Dans le cas d'égalité, une règle de parité fixe le choix.

Remarque 3.2.3 *L'arrondi par défaut est l'arrondi au plus près, noté $\tilde{x}$ dans ce qui suit. Il vérifie*

$$|x - \tilde{x}| \leq |x|\epsilon/2.$$

Les 4 opérations arithmétiques $\{+, -, \times, /\}$ ainsi que la racine carrée et l'opération de congruence (remainder) sont correctes.

Les exceptions overflow, underflow, invalide, inexact ainsi que les nombres spéciaux et les opérations entre nombres spéciaux sont également spécifiés par la norme. Les opérations continuent après une exception. La norme utilise les nombres dénormalisés et l'underflow graduel.

3.2.6 Phénomène d'absorption

L'absorption se produit lors de l'addition de deux quantités avec des ordres de grandeur très différents.

Proposition 3.2.6 *Soit $x, y \in \mathcal{F}$, tels que $u < |x| < x_{max}$ et $x + y \in \mathbb{R}_{\mathcal{F}}$. Soit $x = b^e m$ avec $m \geq 1 + \epsilon$. Soit $\oplus$ l'addition flottante, avec arrondi au plus près. Alors*

$$x \oplus y \;=\; x \quad \Leftrightarrow \quad |y| \leq b^e \; \epsilon/2$$

Preuve. Le résultat est immédiat par définition de l'arrondi. ◇

Ce résultat conduit à la définition suivante.

Définition 3.2.7 *Soit deux réels x et y dans $\mathbb{R}_{\mathcal{F}}$. Il y a absorption de y par x si*

$$|y| \leq |x| \ \epsilon/2.$$

Le résultat d'une addition après absorption est en général le plus grand des deux opérandes.

Par exemple, en double précision, le résultat flottant de $1+10^{-16}$ est 1 donc le résultat flottant de $(1+10^{-16})-1$ vaut 0. Ici, le phénomène d'absorption est associé à un phénomène de cancellation, décrit ci-dessous.

Un exemple classique d'absorption se produit lors du calcul de la somme $S_n = \sum_{i=1}^{n} 1/i$ par l'algorithme avec indices croissants $S_{n+1} = S_n + 1/(n+1)$, si bien que cette série converge numériquement avec l'arithmétique flottante. Un changement d'algorithme avec un ordre de sommation différent permet de retrouver la divergence de la série. Pour plus de détails, voir par exemple [8].

3.2.7 Phénomène de cancellation

Ce phénomène se produit lors de la soustraction de deux nombres voisins. Il faut alors renormaliser le résultat car les chiffres de gauche s'éliminent (d'où le nom cancellation). On dit aussi élimination. Les chiffres de droite deviennent les premiers chiffres significatifs. Si les opérandes sont exacts, la cancellation est bénigne car la soustraction est en général exacte. Mais si les opérandes sont entachés d'erreur, l'ordre de grandeur du résultat exact est celui des incertitudes, donc le résultat flottant risque de n'avoir aucun sens ; la cancellation est dite catastrophique. Il faut noter que la cancellation est un révélateur des erreurs précédentes.

Définition 3.2.8 *Soit x et y deux réels dans $\mathbb{R}_{\mathcal{F}}$. Il y a élimination de x et y si*

$$0 < |x - y| \leq \epsilon/2 \ (|x| + |y|)$$

Soit $x(1+\alpha)$ et $y(1+\beta)$ deux approximations de x et y et soit E l'erreur relative commise sur la soustraction $x - y$. Alors

$$E = \frac{|x\alpha - y\beta|}{|x-y|} \leq F, \text{ avec } F = \max\{|\alpha|, |\beta|\}\frac{|x|+|y|}{|x-y|}.$$

Dans le cas de cancellation, $F \geq 2\max\{|\alpha|, |\beta|\}/\epsilon$ donc l'erreur relative E risque d'être supérieure à 1.

Par exemple, en base 10, $3.1451 - 3.1449 = 0.0002 = 210^{-4}$ mais, si l'on arrondit les opérandes à deux chiffres après la virgule, on obtient $3.15 - 3.14 = 0.01 = 110^{-2}$ soit un résultat avec deux ordres de grandeur en trop.

De manière générale, il y a cancellation catastrophique entre plusieurs nombres si la somme de ces nombres est beaucoup plus petite que la somme de leurs valeurs absolues, autrement dit si

$$0 < |\sum x_i| \leq \epsilon/2 \ (\sum |x_i|)$$

Exemples de cancellation

Les phénomènes de cancellation se rencontrent fréquemment dans le calcul scientifique et sont la source d'erreurs grossières. Citons par exemple le comportement d'un missile anti-missile durant la guerre du golfe [12], le calcul d'une racine d'une équation du second degré, le calcul de l'aire d'un triangle par le déterminant (voir par exemple [1]). Il est parfois possible de modifier l'algorithme pour supprimer le phénomène de cancellation ; l'exemple ci-dessous en est une illustration.

L'exponentielle d'un nombre réel x est la limite de la série convergente

$$\exp(x) = \sum_{k=0}^{\infty} \frac{x^k}{k!}$$

et peut donc être approchée par la somme

$$P_n = \sum_{k=0}^{n} \frac{x^k}{k!}.$$

Si $x < 0$, les termes de la série sont de signes alternés et le résultat qui approche $\exp(x)$ est très petit par rapport à la somme des valeurs absolues qui approche $exp(|x|)$: il y a cancellation catastrophique.

Supposons par exemple que l'on calcule $t_k = \frac{x^k}{k!}$ par l'algorithme $t_{k+1} = t_k * x/(k+1)$ et que l'erreur d'arrondi soit telle que $fl(t_k) = t_k(1+\alpha_k)$ avec $\alpha_k \leq c\epsilon$. Même s'il n'y a pas d'erreur d'arrondi dans la sommation, le résultat $fl(P_n)$ est tel que

$$|fl(P_n) - P_n| \leq c\epsilon \sum |x_k|/k! \leq c\epsilon exp(|x|)$$

et l'erreur relative risque d'être grande dès que

$$\epsilon exp(|x|)/exp(x)$$

est grand.

Si $x \geq 0$ et x pas trop grand, l'erreur d'arrondi reste de l'ordre de la précision machine. Pour $x < 0$, tel que $|x|$ n'est pas trop grand, il est donc préférable de calculer $exp(|x|)$ puis d'utiliser la formule $exp(x) = 1/exp(-x)$. Cet artifice permet de supprimer le phénomène de cancellation.

3.2.8 Extensions de la norme IEEE

Le calcul précis des fonctions mathématiques usuelles telles que l'exponentielle n'est toujours pas complètement résolu [10]. Il n'existe toujours pas de norme à ce sujet, qui reste un domaine de recherche.

La norme IEEE définit la simple et la double précision. En fait, de nombreux ordinateurs proposent aussi une précision étendue pour les calculs intermédiaires grâce à des registres de 80 bits. L'analyse d'une approche hybride utilisant différentes précisions lors des calcul reste aussi un sujet de recherche [9].

Parfois, il est nécessaire de garantir une très grande précision, c'est pourquoi il existe des bibliothèques d'arithmétique dite multi-précision, où la précision des calculs peut être arbitrairement grande [1]. Néanmoins, la précision reste finie et des erreurs d'arrondi sont inéluctables. La précision infinie est apportée par le calcul formel tant qu'il manipule des symboles et des formules. Mais

l'évaluation d'une formule se fait de nouveau en précision finie. Il est important de comprendre les interactions entre calcul symbolique et calcul numérique [1].

Une solution sûre pour garantir la précision d'un résultat est de l'inclure dans un intervalle. L'arithmétique d'intervalles permet de réaliser ces encadrements. Afin de réduire la largeur de l'intervalle, il est en général nécessaire d'adapter les méthodes de calcul. Voir par exemple [1, 2].

3.3 Stabilité des problèmes

Nous allons maintenant étudier l'une après l'autre les différentes sources d'erreur. La théorie de la perturbation étudie l'impact de variations sur les données. De nombreux ouvrages traitent de cette sensibilité aux données, voir par exemple [8, 7, 14, 4, 1].

La définition d'un problème bien posé ci-dessous est celle de Hadamard.

Définition 3.3.1 *Un problème bien posé au voisinage de a possède une solution unique continue par rapport aux donnéees dans un voisinage de a. Il est singulier sinon.*

Définition 3.3.2 *Soit $F(x, a) = 0$ un problème bien posé. Soit x la solution associée à a et $x + \Delta x$ la solution associée à $a + \Delta a$. On suppose que $x \neq 0$. Le problème est stable si*

$$\kappa(a) = \overline{\lim_{\alpha \to 0}} \frac{1}{\alpha} \frac{\|\Delta x\|}{\|x\|} \text{ où } \|\Delta a\| \leq \alpha \|a\|$$

est fini. Dans ce cas, $\kappa(a)$ est le conditionnement relatif du problème au point a.

Au voisinage d'une donnée ou d'une solution nulle, on utilise un conditionnement absolu. Le conditionnement d'un problème singulier est infini. Un problème très mal conditionné, c'est-à-dire avec un grand conditionnement, est en général proche de la singularité.

La définition du conditionnement est liée au choix des métriques dans l'espace des données et dans l'espace des solutions.

Pour α suffisamment petit, on a

$$\|\Delta x\| \leq \kappa(a) \ \|x\| \ \alpha + O(\alpha^2). \tag{3.2}$$

3.3.1 Lien avec le résidu

Considérons le problème $F(x) = a$ de solution x et soit y une solution approchée. Le résidu $r = a - F(y)$ peut être considéré comme une perturbation de a puisque $F(y) = a - r$. Si le problème a un conditionnement $\kappa(a)$, il vient, pour $\|r\|$ assez petit,

$$\|y - x\| \leq \kappa(a) \ \|x\| \ \|r\|/\|a\| + O(\|r\|^2).$$

3.3.2 Exemple : effet papillon

Les systèmes dynamiques sont souvent modélisés par un ensemble de variables, qui évoluent selon des lois mathématiques à partir de valeurs initiales. Dans certains cas, cette évolution est très sensible au choix des valeurs de départ. Il s'agit d'un problème très mal conditionné, voire instable ou mal posé. La théorie du chaos permet d'étudier de tels systèmes. Cette sensibilité aux conditions

initiales s'observe indirectement par une très grande sensibilité aux erreurs d'arrondi. En effet, les erreurs d'arrondi agissent en quelque sorte comme une perturbation des données, voir section 3.6.

Par exemple, les conditions météorologiques sont perturbées par ce qu'on appelle communément l'effet papillon. La très grande sensibilité aux valeurs initiales est illustrée symboliquement par un changement de météo dû à un battement d'ailes de papillon à l'autre bout de la planète.

Cet effet papillon a été découvert par Lorenz qui avait modélisé la météo par un système d'équations différentielles, dont la solution s'appelle maintenant l'attracteur de Lorenz. Si l'on simule l'évolution de ce système par la même méthode sur différents ordinateurs, on obtient des résultats qui diffèrent de plus en plus au cours du temps, à cause des erreurs d'arrondi qui se comportent différemment. Voir par exemple la page Web

`http://www.lactamme.polytechnique.fr/Mosaic/descripteurs/demo_14.html`

Le phénomène El Nino est un exemple d'effet papillon. En effet, une inversion du courant océanique près des côtes d'Amérique du Sud modifie de façon notoire le climat global, à l'échelle de la planète. Voir par exemple la page Web
http://www.pmel.noaa.gov/tao/elnino/nino-home.html

3.4 Stabilité et ordre des schémas de discrétisation

La deuxième source d'erreur que nous analysons est la discrétisation d'un problème.

Pour simuler l'évolution de systèmes dynamiques par exemple, il faut résoudre un système d'équations différentielles ou d'équations aux dérivées partielles. Un schéma de discrétisation définit un problème approché, fonction en général d'un paramètre h. L'ordre du schéma traduit la vitesse de convergence de la solution approchée vers la solution exacte en fonction de h. Il peut arriver que la solution approchée explose, le schéma est alors instable. Dans d'autres cas, la solution approchée oscille, ou certains invariants ne sont plus respectés, etc.

De plus, les erreurs de discrétisation peuvent interférer avec les erreurs d'arrondi. Le choix du paramètre de discrétisation dépend alors non seulement de la précision de l'approximation mais aussi de la précision des calculs flottants.

3.5 Convergence des algorithmes itératifs

Les problèmes discrets sont ensuite résolus par une méthode qui peut être directe ou itérative. Dans le cas itératif, une troisième source d'erreur vient s'ajouter. Nous considérons ici les méthodes itératives dans l'espace vectoriel normé de dimension finie $\mathbb{R}^n$. L'étude des méthodes itératives est traitée par exemple dans [11].

Une des difficultés dans les méthodes itératives est le choix du critère d'arrêt. Celui-ci doit tenir compte des effets de l'arithmétique flottante. Il peut arriver que les erreurs d'arrondi modifient le comportement d'une méthode itérative.

Définition 3.5.1 *Soit $x_k \in \mathbb{R}^n$ une suite et $x \in \mathbb{R}^n$. La suite converge vers x si $\lim_{k\to\infty} \|x_k - x\| = 0$.*

La suite converge linéairement vers x s'il existe un entier K et une constante c, $0 \leq c < 1$, tels que, pour tout $k \geq K$,

$$\|x_{k+1} - x\| \leq c\|x_k - x\|.$$

La suite a une convergence superlinéaire s'il existe une suite c_k telle que

$$\lim_{k\to\infty} c_k = 0 \text{ et } \|x_{k+1} - x\| \leq c_k \|x_k - x\|.$$

La suite a une convergence d'ordre au moins $p > 1$ si x_k converge vers x et s'il existe un entier K, une constante $c \geq 0$ tels que, pour tout $k \geq K$,

$$\|x_{k+1} - x\| \leq c\|x_k - x\|^p.$$

Une des difficultés dans les méthodes itératives est le choix du critère d'arrêt. Celui-ci doit être relatif pour éviter les effets d'échelle, ou mieux être un hybride relatif/absolu pour éviter aussi les risques de non convergence.

Très souvent, il est possible d'associer un résidu à un problème, tel que le résidu est nul pour la solution. Pour un système non linéaire $F(x) = 0$, le résidu est $F(x)$. Un critère d'arrêt usuel est alors

$$\|F(x)\| \leq \tau_r \|F(x_0)\| + \tau_a$$

où τ_r et τ_a sont respectivement les seuils de convergence relatif et absolu.

Ce critère d'arrêt est justifié par la propriété suivante qui lie le résidu $F(x)$ à l'erreur $\|x - x^*\|$. Cette propriété utilise la notion de conditionnement d'une matrice.

Proposition 3.5.1 *Nous faisons les hypothèses suivantes :*

- *F est une fonction de $\mathbb{R}^n$ dans $\mathbb{R}^n$, différentiable, de matrice jacobienne $F'(x)$.*
- *le problème $F(x) = 0$ a une solution x^*.*
- *La fonction F' est Lipschitzienne de constante γ.*
- *La matrice $F'(x^*)$ est inversible.*

Alors il existe δ tel que pour tout x, $\|x - x^*\| < \delta$

$$\frac{1}{4\ \kappa(F'(x^*))} \frac{\|x - x^*\|}{\|x_0 - x^*\|} \leq \frac{\|F(x)\|}{\|F(x_0)\|} \leq \frac{4\kappa(F'(x^*))\|x - x^*\|}{\|x_0 - x^*\|}$$

où $\kappa(F'(x^*)) = \|F'(x^*)\|\|F'(x^*)^{-1}\|$ est le conditionnement de la matrice jacobienne $F'(x^*)$.

Preuve. Voir par exemple []. ◇

Il est également possible, assez souvent, d'utiliser la différence $\|x_{k+1} - x_k\|$ entre deux itérés, lorsqu'elle est liée à l'erreur. Par exemple, pour une méthode à convergence linéaire, on a

$$(1 - c)\|x_k - x\| \leq \|x_{k+1} - x_k\| \leq (1 + c)\|x_k - x\|.$$

Un critère d'arrêt usuel est alors

$$\|x_{k+1} - x_k\| \leq \tau_r \|x_k\| + \tau_a.$$

Parfois, il est nécessaire de tester la divergence de la méthode. Si l'écart $\|x_{k+1} - x_k\|$ devient trop grand pendant plusieurs itérations consécutives, on peut considérer que la méthode diverge.

En pratique, il est utile de limiter le nombre d'itérations pour éviter une explosion du temps de calcul.

Les erreurs d'arrondi peuvent parfois modifier sensiblement la courbe de convergence d'un algorithme, ce qui complique d'autant plus le choix d'un critère d'arrêt.

3.6 Stabilité des algorithmes directs

Nous nous plaçons ici dans le cadre des algorithmes directs et nous étudions l'impact des erreurs d'arrondi sur le résultat d'un calcul scientifique. Plus précisément, nous supposons que la solution exacte x du problème $F(x, a) = 0$ peut être calculée en un nombre fini d'opérations arithmétiques. Le cas de l'impact des arrondis sur les algorithmes itératifs est plus complexe. Voir par exemple [8, 4].

Nous voulons mesurer l'effet des erreurs d'arrondi. Pour cela, nous supposons que les opérations arithmétiques usuelles sont correctes, par exemple que l'arithmétique flottante satisfait la norme IEEE.

Soit $x(\epsilon)$ la solution calculée en arithmétique flottante avec une précision ϵ. Il existe deux façons de mesurer la qualité du résultat : une analyse directe et une analyse inverse.

Définition 3.6.1 *Un algorithme est stable au sens direct s'il existe $\psi(a)$ tel que*

$$\|x(\epsilon) - x\|/\|x\| \leq \psi(a)\ \epsilon.$$

Un algorithme est stable au sens inverse s'il existe $\phi(a)$ et $a(\epsilon)$ tels que

$$F(x(\epsilon), a(\epsilon)) = 0,\ \textit{avec}\ \|a(\epsilon) - a\|/\|a\| \leq \phi(a)\epsilon.$$

Proposition 3.6.1 *Si le problème est stable et si l'algorithme de résolution est stable au sens inverse avec $\phi(a)$ assez petit, alors l'algorithme est stable au sens direct et*

$$\|x(\epsilon) - x\|/\|x\| \leq \kappa(a)\ \phi(a)\ \epsilon + O(\epsilon^2). \tag{3.3}$$

Preuve. La formule se déduit immédiatement du résultat (3.2). ◇

La formule précédente montre qu'en général la précision relative sur un résultat de calcul est la précision machine multipliée par le conditionnement du problème à résoudre. Le résultat risque donc d'être d'autant plus imprécis que le problème est mal conditionné. Les erreurs d'arrondi sont amplifiées sous l'effet du conditionnement.

3.6.1 Exemple : produit scalaire

Soit $x \in \mathbb{R}^n$ et $y \in \mathbb{R}^n$ deux vecteurs non nuls et $s = x^T y$ leur produit scalaire.

Proposition 3.6.2 *Le conditionnement relatif composante par composante du produit scalaire vaut*

$$\kappa(x, y) = 2|x|^T|y|/|x^T y|.$$

Preuve. Soit $s + \Delta s = (x + \Delta x)^T(y + \Delta y)$, avec $|\Delta x| \leq \alpha|x|$ et $|\Delta y| \leq \alpha|y|$ alors $\Delta s = x^T\Delta y + y^T\Delta x + \Delta x^T\Delta y$ et $|\Delta s| \leq 2|x|^T|y|\alpha + |x|^T|y|\alpha^2$ d'où le résultat. ◇

Nous considérons l'algorithme suivant, écrit en syntaxe Matlab, qui fixe l'ordre des opérations :

ALGORITHM : Produit scalaire

```
s=0 ;
for i=1:n,
   s = s+x(i) * y(i) ;
end ;
```

Pour démontrer la stabilité numérique de cet algorithme, nous avons besoin d'un lemme très souvent utilisé en analyse d'erreur.

Lemme 3.6.1 *Si $n\epsilon < 0.1$ et si $|\delta_i| \leq \epsilon$ alors*

$$\Pi_{i=1}^{n}(1+\delta_i) = 1+\theta \text{ avec } \theta \leq 1.06\ n\epsilon$$

Proposition 3.6.3 *Si $n\epsilon < 0.1$, l'algorithme est stable au sens inverse et on a*

$$fl(x^T y) = x^T(y+\Delta y)$$
$$|\Delta y| \leq 1.06\ n\epsilon |y|.$$

Preuve. Soit $s_i = s_{i-1} + x_i * y_i$ pour $i \geq 2$ et $s_1 = x_1 * y_1$. Alors

$fl(s_1) = x_1 * y_1(1+\alpha_1)$,
$fl(s_i) = (fl(s_{i-1}) + x_i * y_i(1+\alpha_i))(1+\beta_i)$,
avec $|\alpha_i| \leq \epsilon$ et $|\beta_i| \leq \epsilon$,
d'où par récurrence
$fl(s_n) = x_1 * y_1(1+\alpha_1)(1+\beta_2)\ldots(1+\beta_n) +$
$x_2 * y_2(1+\alpha_2)(1+\beta_2)\ldots(1+\beta_n) +$
$x_n * y_n(1+\alpha)_n)(1+\beta_n)$
et par application du lemme,
$fl(s) = fl(s_n) = x_1 * y_1(1+\theta_1) + \ldots + x_n * y_n(1+\theta_n)$ avec $|\theta_i| \leq 1.06 n\epsilon$.

◇

Une preuve similaire peut être faite pour tout ordre de sommation, avec une constante différente.

Les erreurs d'arrondi sont donc de l'ordre du conditionnement multiplié par la précision machine. Lorsque $|x^T y| << |x|^T|y|$, le conditionnement est élevé. Ce très grand conditionnement se manifeste par le biais d'un phénomène de cancellation globale dans le calcul du produit scalaire. Cette cancellation globale amplifie les erreurs d'arrondi.

3.6.2 Exemple : produit extérieur

Définition 3.6.2 *Soit $x \in \mathbb{R}^n$ et $y \in \mathbb{R}^n$ deux vecteurs non nuls. Leur produit extérieur est $A = xy^T$.*

Proposition 3.6.4 *Le conditionnement relatif composante par composante du produit extérieur vaut*

$$\kappa(x,y) = 2|x||y|^T/|xy^T|.$$

Proposition 3.6.5 *Le calcul du produit extérieur n'est pas stable au sens inverse. Il est stable au sens direct (composante par composante) et*

$$fl(xy^T) = xy^T + \Delta A \text{ avec } |\Delta A| \leq \epsilon |x||y|^T.$$

Preuve. Le résultat flottant $\tilde{A} = fl(xy^T)$ n'est pas en général une matrice de rang 1 donc ne peut être égal à $(x+\Delta x)(y+\Delta y)^T$. Par contre, $\tilde{A}_{ij} = x_i y_j(1+\alpha_{ij})$ avec $|\alpha_{ij}| \leq \epsilon$ donc $\tilde{A} = A + \Delta A$ avec $|\Delta A|_{ij} \leq |x_i||y_j|\epsilon$. ◇

Chapter 4

Résolution de systèmes linéaires carrés

Ce chapitre a pour objet les matrices carrées et les systèmes linéaires de n équations à n inconnues. Il définit les notions de matrice inversible et de matrice singulière. Il rappelle les algorithmes de résolution d'un système linéaire. Le chapitre suivant étudiera les matrices rectangulaires de rang plein et définira les algorithmes de résolution des problèmes aux moindres carrés de m équations à n inconnues, avec $m \geq n$.

4.1 Inverse d'une matrice

Théorème 4.1.1 *$A \in \mathbb{R}^{n \times n}$, 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 $Im(A) = \mathbb{R}^n$, ce qui équivaut aussi dire que le système $Ax = b$ a une solution pour tout b. D'autre part, $rang(A) = n$ équivaut à $ker(A) = \{0\}$, aussi à dire que 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 4.1.1 *Une matrice carrée non inversible est dite singulière. Une matrice inversible est dite non singulière.*

Proposition 4.1.1 *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 $B^{-1}A^{-1}$.
$A^T(A^{-1})^T = (A^{-1}A)^T = I$. ◇

La méthode qu'on utilise souvent pour la résolution symbolique d'un système linéaire utilise le déterminant et les formules de Cramer. Cette méthode est à proscrire pour la résolution numérique. En effet, d'une part le nombre d'opérations est très élevé car chaque calcul de déterminant a une complexité en $O(n!)$, ce qui est prohibitif. D'autre part, les erreurs d'arrondi se propagent de manière catastrophique, à cause du phénomène d'élimination qui risque de se produire dans le calcul des sommes alternées.

Il existe d'autres méthodes adaptées au calcul numérique, qui sont basées sur une transformation de la matrice dite factorisation et sur la résolution de systèmes particuliers. Nous allons voir d'abord ces cas particuliers, puis les algorithmes de factorisation.

4.1.1 Matrices particulières

Proposition 4.1.2 *Si Q est orthogonale, alors Q est inversible et $Q^{-1} = Q^T$. Résoudre un système avec une matrice orthogonale est une opération BLAS2, avec $2n^2 + O(n)$ opérations.*

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 $D^{-1} = diag(1/d_i)$. Résoudre un système diagonal est une opération BLAS1, avec n opérations.

Une matrice triangulaire L est inversible si et seulement si $L_{ii} \neq 0$ et l'inverse de L est triangulaire. Résoudre un système triangulaire est une opération BLAS2, avec $n^2/2 + O(n)$ opérations.

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.

◇

4.2 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. Nous étudions maintenant la factorisation de la matrice qui permet de transformer le système de départ en deux systèmes triangulaires.

Théorème 4.2.1 *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*

$$PA = LU \tag{4.1}$$

L'égalité (4.1) est appelée la factorisation de Gauss de A avec pivot partiel. La factorisation de PA existe et est unique.

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. Voir paragraphe ci-dessous.

L'algorithme de résolution d'un système linéaire comporte les étapes suivantes:

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 $Dz = y$	BLAS1	$O(n)$
résoudre $L^T x = z$	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.

Si la matrice est symétrique, on peut parfois utiliser une variante de Gauss, sous la forme

$$A = LDL^T, \tag{4.2}$$

où L est une matrice L triangulaire inférieure avec $L_{ii} = 1$ et D est une matrice diagonale D inversible. Cette factorisation n'existe pas toujours. Nous allons voir maintenant un cas important où elle existe.

4.2.1 Cas des matrices symétriques définies positives

Définition 4.2.1 *Soit $A \in \mathbb{R}^{n\times n}$ une matrice symétrique d'ordre n, alors A est définie positive si et seulement si*

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

et est semi-définie positive si et seulement si

$$\forall x,\ x^T Ax \geq 0.$$

Un exemple qui sera utile dans la suite est celui des matrices $A^T A$.

Proposition 4.2.1 *Soit $A \in \mathbb{R}^{m\times n}$, alors la matrice $A^T A$ est symétrique semi-définie positive. Elle est définie positive si et seulement si $rang(A) = n$.*

Preuve. $\forall y,\ y^T(A^T A)y = (Ay)^T(Ay) = \|Ay\|_2^2 \geq 0$ donc $A^T A$ est semi-définie positive.

$rang(A) = n \Leftrightarrow ker(A) = \{0\} \Leftrightarrow \forall y \neq 0,\ Ay \neq 0 \Leftrightarrow \forall y \neq 0,\ \|Ay\|_2 > 0 \Leftrightarrow A^T A$ définie positive. ◇

La factorisation des matrices symétriques définies positives est toujours possible.

Théorème 4.2.2 *Soit $A \in \mathbb{R}^{n\times n}$ une matrice symétrique définie positive. Alors il existe une unique matrice L triangulaire inférieure avec $L_{ii} = 1$ et une unique matrice D diagonale avec $d_i > 0$, telles que*

$$A = LDL^T \tag{4.3}$$

L'égalité (4.3) est appelée la factorisation de Cholesky de A et la matrice L est appelée le facteur de Cholesky de A.

Preuve. admis. ◇

Soit S la matrice diagonale définie par $s_i = \sqrt{d_i}$ et $\overline{L} = LS$; alors

$$A = LDL^T = \overline{LL}^T.$$

Cette variante, qui est aussi appelée factorisation de Cholesky, est moins utilisée que la précédente car elle nécessite le calcul de racines carrées.

L'algorithme de résolution d'un système linéaire avec une matrice symétrique définie positive comporte ainsi les étapes suivantes:

	Factorisation de Cholesky	
étape	bibliothèque	nombre d'opérations
factoriser $A = LDL^T$	LAPACK (utilise BLAS3)	$n^3/3 + O(n^2)$
résoudre $Ly = b$	BLAS2	$O(n^2)$
résoudre $Dz = y$	BLAS1	$O(n)$
résoudre $L^T x = z$	BLAS2	$O(n^2)$

On constate qu'on a réduit de moitié le nombre d'opérations de la factorisation, qui est le terme prédominant.

4.3 Analyse des erreurs pour un système linéaire

4.3.1 Stabilité numérique de Gauss et de Cholesky

La matrice de permutation P dans la factorisation de Gauss permet d'éviter des divisions par 0 et de garantir l'existence de L et U. En fait, pour réduire les erreurs d'arrondi, on choisit la matrice P pour éviter des divisions par de petits nombres; dans la stratégie de pivot partiel, on choisit ainsi le diviseur le plus grand possible (en valeur absolue), grâce à une permutation des lignes. On peut alors faire une analyse de la propagation des erreurs d'arrondi et on obtient le résultat suivant:

Théorème 4.3.1 *Soit $A \in \mathbb{R}^{n\times n}$ une matrice symétrique définie positive et $PA = LU$ la factorisation de Cholesky. On résout le système $Ax = b$ par l'algorithme de Gauss avec pivot partiel. Le résultat obtenu y est solution du système perturbé*

$$(A + E)y = b$$

avec $\|E\|_\infty \leq c\rho\|A\|_\infty\epsilon$, où c est une constante (qui dépend de n) et ρ, qui est appelé le facteur de croissance, dépend des valeurs calculées au cours de l'algorithme.

Dans le cas des matrices symétriques définies positives, on montre qu'il n'y a pas besoin de stratégie de pivot; on obtient le résultat suivant après analyse des erreurs d'arrondi:

Théorème 4.3.2 *Soit $A \in \mathbb{R}^{n\times n}$ une matrice symétrique définie positive et $A = LL^T$ la factorisation de Cholesky. On résout le système $Ax = b$ par l'algorithme de Cholesky. Le résultat obtenu y est solution du système perturbé*

$$(A + E)y = b$$

avec $\|E\|_2 \leq c\|A\|_2\epsilon$, où c est une constante (qui dépend de n).

Les erreurs d'arrondi ne calculent pas la solution exacte, mais la solution d'un problème perturbé, où la perturbation E de la matrice est bornée par la précision machine. Il n'y a pas explosion des erreurs d'arrondi, on sait borner les perturbations. On dit que les algorithmes de Gauss avec pivot partiel et de Cholesky sont stables au sens inverse. Ensuite, une analyse de perturbation permet de borner l'erreur sur la solution.

4.3.2 Analyse de perturbation d'un système linéaire

Ici, nous étudions l'erreur induite sur la solution d'un système linéaire, suite à une perturbation de la matrice et du second membre. Nous introduisons d'abord la notion de conditionnement de la matrice. Nous verrons au chapitre sur la SVD comment relier ce conditionnement aux valeurs singulières.

Définition 4.3.1 *Soit $A \in \mathbb{R}^{n\times n}$ avec $rang(A) = n$, c'est-à-dire A inversible. Le conditionnement de la matrice A pour les systèmes linéaires est défini par $cond(A) = \|A\|_2\|A^{-1}|_2$.*

Nous établissons à quelle condition suffisante la matrice $A + E$ est inversible.

Lemme 4.3.1 *Si $\|F\|_2 < 1$ alors $(I + F)$ est inversible et $\|(I + F)^{-1}\|_2 \leq \frac{1}{1-\|F\|_2}$.*

Preuve. admis. ◇

Proposition 4.3.1 *Soit E telle que $\|E\|_2 \leq \alpha\|A\|_2$ et $cond(A)\alpha < 1$; alors $A + E$ est inversible et*

$$\|(I + A^{-1}E)^{-1}\|_2 \leq \frac{1}{1 - \alpha cond(A)}.$$

Preuve. $\|A^{-1}E\|_2 \leq \|(A^{-1}\|_2\|E\|_2 \leq cond(A)\alpha < 1$ donc d'après le lemme, $I + A^{-1}E$ est inversible et en multipliant par A, on en déduit que $A + E$ est inversible.

Il suffit d'appliquer le lemme pour obtenir l'inégalité cherchée. ◇

Nous calculons une borne pour l'erreur sur la solution, qui utilise le conditionnement et les amplitudes des perturbations.

Théorème 4.3.3 *Soit $A \in \mathbb{R}^{n\times n}$ une matrice inversible; E telle que $\|E\|_2 \leq \alpha\|A\|_2$ et $cond(A)\alpha < 1$; Soit $b \neq 0$; e tel que $\|e\|_2 \leq \beta\|b\|_2$ soit x la solution du système $Ax = b + e$, y la solution du système $(A + E)y = b + e$. Alors*

$$\frac{\|y - x\|_2}{\|x\|_2} \leq \frac{cond(A)}{1 - \alpha cond(A)}\ (\alpha + \beta).$$

Preuve. On va calculer $y - x$, on a $Ay + Ey = b + e = Ax + e$ d'où $A(y - x) = -Ey + e$ et $y - x = -A^{-1}(Ey - e)$ donc

$$\|y - x\|_2 \leq cond(A)(\alpha\|y\|_2 + \beta\|x\|_2).$$

Il reste à majorer $\|y\|_2$ en fonction de $\|x\|_2$.

On a $A(I + A^{-1}E)y = Ax + e$ d'où $y = (I + A^{-1}E)^{-1}(x + A^{-1}e)$ et d'après la proposition précédente

$$\|y\|_2 \leq \frac{1}{1 - \alpha cond(A)}(1 + \beta cond(A))\|x\|_2.$$

On obtient ainsi le résultat du théorème. ◇

Corollaire 4.3.1 *Avec les hypothèses du théorème,*

$$\frac{\|y - x\|_2}{\|x\|_2} \leq O(cond(A))(\frac{\|E\|_2}{\|A\|_2} + \frac{\|e\|_2}{\|b\|_2}).$$

L'erreur relative sur la solution est majorée par un terme de l'ordre de la perturbation relative sur les données multipliée par le conditionnement de la matrice.

Comme pour le cas général, il est possible de faire le lien avec le résidu $r = b - Ay$, en choisissant $e = -r$ et $E = 0$.

Corollaire 4.3.2 *Le résidu $r = b - Ay$ vérifie*

$$\frac{\|x - y\|_2}{\|x\|_2} = O(cond(A))\frac{\|r\|_2}{\|b\|_2}. \tag{4.4}$$

Il est possible d'affiner ce résultat en déterminant une perturbation de la matrice, la plus petite possible.

Théorème 4.3.4 *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

$$\eta = \frac{\|b - Ay\|}{\|A\|\|y\| + \|b\|}. \tag{4.5}$$

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

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

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

Réciproquement, soit

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

Donc η est le minimum de S. ◇

En combinant les deux théorèmes précédents, on obtient une estimation d'erreur pour la solution calculée y.

Corollaire 4.3.3 *Soit x la solution du système linéaire $Ax = b$ et soit y une solution calculée telle que $\eta cond(A) < 1$, où η est défini par (4.5). Alors*

$$\frac{\|x - y\|}{\|x\|} \leq 2 \frac{cond(A)}{1 - \eta cond(A)} \eta \tag{4.6}$$

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

4.3.3 Précision de la résolution d'un système linéaire

En combinant les résultats sur l'analyse inverse de stabilité de Gauss ou Cholesky, puis les résultats sur l'analyse de perturbation des systèmes linéaires, on obtient une majoration de l'erreur relative sur la solution.

Proposition 4.3.2 *Soit x la solution du système linéaire $Ax = b$ et soit y la solution calculée avec l'algorithme de Gauss avec pivot partiel, en utilisant une arithmétique flottante de précision ϵ. Alors*

$$\frac{\|x - y\|_2}{\|x\|_2} \leq O(cond(A)\epsilon) \tag{4.7}$$

En général, cette borne est atteinte et l'erreur relative sur la solution est de l'ordre de la précision machine multipliée par le conditionnement. Le conditionnement est le facteur qui dégrade la précision.

Chapter 5

Problèmes aux moindres carrés - cas du rang plein

Soit $A \in \mathbb{R}^{m\times n}$ et $b \in \mathbb{R}^m$. Le problème aux moindres carrés est de trouver

$$x \in \mathbb{R}^n, \quad \text{tel que } \|b - Ax\|_2 \text{ est minimum} \tag{5.1}$$

L'objet de ce chapitre est d'étudier les problèmes aux moindres carrés dans le cas où la matrice A est de rang plein. Il introduit la notion de pseudo-inverse d'une matrice rectangulaire de rang plein qui généralise la notion d'inverse. Pour plus de détails, voir par exemple [7, 3].

5.1 Existence et unicité d'une solution

Ce paragraphe montre l'existence d'une solution et définit les conditions d'unicité.

5.1.1 Existence d'une solution dans le cas général

Définition 5.1.1 *Le résidu est $r = b - Ax$.*

Théorème 5.1.1 *Le problème (5.1) a au moins une solution.*
Soit $b = b_1 + b_2$, avec $b_1 \in Im(A)$ et $b_2 \in Im(A)^\perp$.
L'ensemble des solutions est $\{x \in \mathbb{R}^n,\ Ax = b_1\}$.

Preuve. $b = b_1 + b_2$, avec $b_1 \in Im(A)$ et $b_2 \in Im(A)^\perp$. Alors $b - Ax = (b_1 - Ax) + b_2$ et $\|b - Ax\|_2^2 = \|b_1 - Ax\|_2^2 + \|b_2\|_2^2$ donc $\|b - Ax\|_2 \geq \|b_2\|_2$. Soit x tel que $Ax = b_1$, alors $\|b - Ax\|_2$ est minimum. Donc il existe au moins une solution.

Réciproquement, soit x tel que $\|b-Ax\|_2$ est minimum, alors $\|b-Ax\|_2 = \|b_2\|_2$ et $\|b_1-Ax\|_2 = 0$ d'où $Ax = b_1$.

◇

Proposition 5.1.1 *x est solution de (5.1) si et seulement si le résidu r est dans l'orthogonal de l'image de A, ssi $A^T r = 0$.*
Le problème (5.1) est équivalent à

$$(A^T A)x = A^T b \tag{5.2}$$

Preuve. Si x est solution, alors $Ax = b_1$ et $b - Ax = b_2 \in Im(A)^\perp = ker(A)^T$.

Réciproquement, si $r \in Im(A)^\perp$, alors $A^T r = 0$ donc $A^T(Ax - b) = A^T(Ax - b_1) = 0$ d'où $Ax - b_1 \in Im(A) \cap Im(A)^\perp$ et $Ax = b_1$ et x solution de (5.1).

$A^T r = 0 \Leftrightarrow A^T Ax = A^T b$. ◇

5.1.2 Condition d'unicité de la solution

Proposition 5.1.2 *La solution est unique si et seulement si $rang(A) = n$.*

Preuve. La solution est unique ssi $Ax = b_1$ a une unique solution ssi $rang(A) = n$. ◇

Proposition 5.1.3 *On a $rang(A) = rang(A^T A)$. Soit $m \geq n$, alors $rang(A) = n \Leftrightarrow (A^T A)$ inversible.*

Preuve. $y \in ker(A) \Rightarrow y \in ker(A^T A)$, et $y \in ker(A^T A) \Rightarrow y^T A^T Ay = 0 \Rightarrow \|Ay\|_2 = 0 \Rightarrow Ay = 0 \Rightarrow y \in ker(A)$. Donc $ker(A) = ker(A^T A)$ et $rang(A) = rang(A^T A)$.

Soit $m \geq n$ alors $rang(A) = n \Leftrightarrow rang(A^T A) = n \Leftrightarrow A^T A$ inversible. ◇

Définition 5.1.2 *Soit $m \geq n$ et $rang(A) = n$. La pseudo-inverse de A est*

$$A^\dagger = (A^T A)^{-1} A^T \tag{5.3}$$

Proposition 5.1.4 *Soit $m \geq n$ et $rang(A) = n$. L'unique solution de (5.1) est*

$$x = A^\dagger b \tag{5.4}$$

Preuve. évident ◇

5.2 Equations normales

Le système (5.2) est appelé système des équations normales. Nous allons construire un algorithme de résolution du problème (5.1) à l'aide de (5.2).

Pour cela, nous étudions les propriétés de la matrice symétrique $A^T A$.

L'algorithme de résolution de (5.1) par les équations normales est le suivant :

	Équations normales	
étape	bibliothèque	nombre d'opérations
calculer $B = A^T A$	BLAS3	$mn(n+1)$
calculer $c = A^T b$	BLAS2	$2mn$
factoriser $B = LL^T$	LAPACK (utilise BLAS3)	$O(n^3/3)$
résoudre $Ly = c$	BLAS2	$O(n^2)$
résoudre $L^T x = y$	BLAS2	$O(n^2)$

Le coût total est donc en $mn^2 + n^3/3 + O(mn)$, en particulier si $m >> n$, le coût total est en $mn^2 + O(mn)$.

5.3 Factorisation QR

Nous allons voir maintenant une autre méthode de résolution du problème (5.1), qui n'utilise pas les équations normales. Ensuite, nous comparerons les complexités et les précisions des deux méthodes.

Théorème 5.3.1 *Soit $A \in \mathbb{R}^{m\times n}, m \geq n$. Il existe une matrice orthogonale $Q = (Q_1 \;\; Q_2) \in \mathbb{R}^{m\times m}$ avec $Q_1 \in \mathbb{R}^{m\times n}$ et une matrice triangulaire supérieure $R \in \mathbb{R}^{n\times n}$ avec $R_{ii} \geq 0$ telles que*

$$A = Q \begin{pmatrix} R \\ 0 \end{pmatrix} = Q_1 R \tag{5.5}$$

L'équation (5.5) est appelée factorisation QR de A.

Preuve. admis ◇

Proposition 5.3.1 *Soit $Q = (Q_1 \;\; Q_2)$ et R les matrices de la factorisation QR.*
Alors $ker(A) = ker(R)$ et $rang(A) = rang(R)$.
Si $rang(A) = n$, alors R est inversible et $Im(A) = Im(Q_1)$, $Im(A)^\perp = Im(Q_2)$.

Preuve. $Ax = 0 \Leftrightarrow Q_1 Rx = 0 \Leftrightarrow Rx = 0$ donc $ker(A) = ker(R)$ et $rang(A) = rang(R)$.
Si $rang(A) = n$, alors $rang(R) = n$ donc R est inversible.
De plus, $Im(A) = Im(Q_1)$ d'où $Im(A)^\perp = Im(Q_1)^\perp = Im(Q_2)$. ◇

Le résultat suivant fait le lien avec la factorisation de Cholesky de A^TA.

Proposition 5.3.2 *Si $rang(A) = n$ alors les matrices Q_1 et R dans la factorisation $A = Q_1R$ sont uniques : R^T est le facteur de Cholesky de A^TA et $Q_1 = AR^{-1}$. On a donc la factorisation de Cholesky $A^TA = R^TR$.*

Preuve. Si $rang(A) = n$, alors A^TA est définie positive et R est inversible. Si $A = Q_1R$, alors $A^TA = R^TR$ avec $R_{ii} > 0$ donc par unicité de la factorisation de Cholesky, $R = L^T$ d'où $Q_1 = AL^{-T}$. ◇

Remarque 5.3.1 *Il n'y a pas unicité de la matrice Q_2.*

Nous pouvons maintenant résoudre le problème (5.1).

Théorème 5.3.2 *Si $rang(A) = n$, soit $A = Q_1R$, le problème (5.1) est équivalent au système*

$$Rx = Q_1^Tb \tag{5.6}$$

Preuve. Soit $b = b_1 + b_2$ avec $b_1 \in Im(A)$ et $b_2 \in Im(A)^\perp$, alors $b_1 = Q_1Q_1^Tb$ et $b_2 = Q_2Q_2^Tb$.
Le problème (5.1) équivaut à $Ax = b_1$ donc à $Q_1Rx = Q_1Q_1^Tb$ donc à $Rx = Q_1^Tb$.
On peut aussi vérifier que $\|b - Ax\|_2^2 = \|Q_1(Q_1^Tb - Rx) + Q_2Q_2^Tb\|_2^2 = \|Q_1^Tb - Rx\|_2^2 + \|Q_2^Tb\|_2^2$ donc que $\|b - Ax\|_2$ est minimum si et seulement si $Rx = Q_1^Tb$, le minimum valant $\|Q_2^Tb\|_2$. ◇

Proposition 5.3.3 *Si $rang(A) = n$, la pseudo-inverse de A vaut $A^\dagger = R^{-1}Q_1^T$.*

Preuve. Pour tout b, l'unique solution de (5.1) vaut $A^\dagger b$ et $R^{-1}Q_1^Tb$.
On peut aussi vérifier que $A^\dagger = (A^TA)^{-1}A^T = (R^TR)^{-1}R^TQ_1^T = R^{-1}Q_1^T$. ◇

L'algorithme de résolution de (5.1) par la factorisation QR est le suivant :

factorisation QR

étape	bibliothèque	nombre d'opérations
factoriser $A = Q_1 R$	LAPACK (utilise BLAS3)	$2mn^2 - 2n^3/3$
calculer $c = Q_1^T b$	BLAS2	$O(mn)$
résoudre $Rx = c$	BLAS2	$O(n^2)$

Le coût total est donc en $2mn^2 - 2n^3/3 + O(mn)$, en particulier, pour $m >> n$, il est en $2mn^2 + O(mn)$.

Donc, lorsque $m >> n$, la factorisation QR nécessite deux fois plus d'opérations que les équations normales.

D'un autre côté, nous allons voir ci-dessous que la factorisation QR est en général plus précise que les équations normales.

5.4 Analyse d'erreur pour les problèmes aux moindres carrés de rang plein

5.4.1 Stabilité numérique de la factorisation QR

Théorème 5.4.1 *Soit y le résultat obtenu lorsqu'on exécute l'algorithme de factorisation QR. Alors y est solution du problème aux moindres carrés*

$$\min \|(A+E)y - (b+e)\|_2$$

avec $\|E\|_2 \leq c\|A\|_2\epsilon$ et $\|e\|_2 \leq c\|b\|_2\epsilon$ où c est une constante.

Comme pour Gauss et Cholesky, on montre que QR est stable au sens inverse; à cause des erreurs d'arrondi, on résout un problème perturbé, avec une amplitude de perturbation bornée par la précision machine.

5.4.2 Analyse de perturbation et conditionnement

Comme pour les systèmes linéaires, nous introduisons la notion de conditionnement.

Définition 5.4.1 *Soit $A \in \mathbb{R}^{m\times n}$ avec $rang(A) = n$. Le conditionnement de la matrice A pour les problèmes aux moindres carrés est défini par $cond(A) = \|A\|_2\|A^\dagger\|_2$.*

Dans ce qui suit, $A \in \mathbb{R}^{m\times n}$ avec $m \geq n$ et $rang(A) = n$. Nous donnons une condition suffisante pour qu'une matrice perturbée $A + E$ reste de rang plein. Ensuite, nous donnons un résultat d'analyse de perturbation.

Proposition 5.4.1 *Soit E telle que $\|E\|_2 \leq \alpha\|A\|_2$ et $cond(A)\alpha < 1$; alors $rang(A+E) = n$.*

Preuve. admis. ◇

Théorème 5.4.2 *Avec les mêmes hypothèses et $\|e\|_2 \leq \beta\|b\|_2$, soit x la solution du problème $\min \|Ax - b\|_2$ avec $r = b - Ax$ et y la solution du problème $\min \|(A+E)y - (b+e)\|_2$; alors*

$$\frac{\|y-x\|_2}{\|x\|_2} \leq \frac{cond(A)}{1 - \alpha cond(A)} \left[(1 + cond(A)\frac{\|r\|_2}{\|A\|_2}\|x\|_2)\alpha + \frac{\|b\|_2}{\|A\|_2}\beta\right].$$

Preuve. admis. ◇

Remarque 5.4.1 *Dans le cas des systèmes linéaires, on a* $r = 0$.

Corollaire 5.4.1 *Avec les hypothèses du théorème, on distingue deux cas :*

$$\begin{array}{l} \textit{si } \|r\|_2 \textit{ est petit } \frac{\|y-x\|_2}{\|x\|_2} = O(cond(A)) \max(\alpha, \beta), \\ \textit{si } \|r\|_2 \textit{ est grand } \frac{\|y-x\|_2}{\|x\|_2} = O(cond(A)^2) \max(\alpha, \beta). \end{array}$$

L'erreur relative sur la solution est de l'ordre de la perturbation relative sur la matrice multipliée par le conditionnement de la matrice si la norme du résidu est petite ou par le conditionnement au carré si la norme du résidu est grande.

5.4.3 Précision d'une résolution de problème aux moindres carrés de rang plein

Nous énonçons un résultat qui sera prouvé dans le chapitre sur la SVD.

Proposition 5.4.2 $cond(A^T A) = cond(A)^2$.
Soit $A = Q_1 R$ *alors* $cond(A) = cond(R)$.

Théorème 5.4.3 *Soit* y *le résultat obtenu par l'algorithme des équations normales appliqué au problème des moindres carrés. On suppose que les étapes de calcul de* $A^T A$ *et de* $A^T b$ *se font sans erreur. Alors*

$$\frac{\|y - x\|_2}{\|x\|_2} = O(cond(A)^2)\epsilon.$$

Preuve. On applique d'abord l'analyse de stabilité à l'algorithme de Cholesky avec la matrice $A^T A$ puis on applique le théorème sur le conditionnement des systèmes linéaires avec $cond(A^T A) = cond(A)^2$. ◇

Théorème 5.4.4 *Soit* y *le résultat obtenu par l'algorithme de la factorisation* QR *appliqué au problème des moindres carrés. On suppose que les étapes de calcul de* $c = Q_1^T b$ *et de* $R^{-1}c$ *se font sans erreur. Alors deux cas sont possibles :*

$$\begin{array}{l} \textit{si } \|r\|_2 \textit{ est petit } \frac{\|y-x\|_2}{\|x\|_2} = O(cond(A))\epsilon, \\ \textit{si } \|r\|_2 \textit{ est grand } \frac{\|y-x\|_2}{\|x\|_2} = O(cond(A)^2)\epsilon. \end{array}$$

Preuve. Il suffit d'appliquer les théorèmes précédents. ◇

On constate que la méthode de factorisation QR est plus précise que la méthode des équations normales lorsque la norme du résidu est petite. Il peut donc être intéressant d'utiliser cette méthode, bien que sa complexité soit le double de la complexité des équations normales.

Chapter 6

Décomposition en Valeurs Singulières

Ce chapitre introduit la Décomposition en Valeurs Singulières (Singular Value Decomposition, abrégée en SVD). Auparavant, nous rappelons la notion de diagonalisation d'une matrice.

6.1 Diagonalisation d'une matrice carrée

Définition 6.1.1 *Deux matrices carrées A et B sont semblables s'il existe une matrice inversible X telle que $A = XBX^{-1}$.*

La matrice X peut être considérée comme une matrice de changement de base; les matrices A et B représentent la même application linéaire dans la base canonique et la base X.

Proposition 6.1.1 *Deux matrices semblables ont le même polynôme caractéristique. Elles ont donc le même spectre et leurs vecteurs propres sont transformés par le changement de base.*

Définition 6.1.2 *Une matrice est diagonalisable si elle est semblable à une matrice diagonale.*

Proposition 6.1.2 *Une matrice A est diagonalisable si et seulement s'il existe une base V de n vecteurs propres; on a alors $A = VDV^{-1}$, où $D = diag(\lambda_i)$ est la matrice diagonale constituée des n valeurs propres.*

Preuve. $A = VDV^{-1} \Leftrightarrow (AV = VD$ et V inversible$) \Leftrightarrow (Av_i = \lambda_i v_i, i = 1, \ldots, n)$ et V inversible. $\diamond$

Toutes les matrices ne sont pas diagonalisables; nous allons étudier maintenant la multiplicité des valeurs propres pour caractériser les matrices diagonalisables.

Définition 6.1.3 *La multiplicité algébrique d'une valeur propre est sa multiplicité dans le polynôme caractéristique; si la racine est simple, la valeur propre est simple, sinon la valeur propre est multiple.*

Définition 6.1.4 *La multiplicité géométrique d'une valeur propre λ est la dimension du sous-espace vectoriel $ker(A - \lambda I)$. Elle est au plus égale à la dimension algébrique.*

Définition 6.1.5 *Une valeur propre est semi-simple si sa multiplicité géométrique est égale à sa multiplicité algébrique; c'est une valeur propre défective sinon.*

Proposition 6.1.3 *Les valeurs propres de deux matrices semblables ont la même multiplicité géométrique.*

Proposition 6.1.4 *Une matrice est diagonalisable si et seulement si toutes ses valeurs propres sont semi-simples.*

6.1.1 Diagonalisation des matrices symétriques

Théorème 6.1.1 *Les valeurs propres d'une matrice symétrique A sont réelles. Il existe une base orthonormée V de vecteurs propres réels. On a donc*

$$A = VDV^T.$$

Preuve. admis. ◇

Proposition 6.1.5 *Une matrice symétrique A est semi-définie positive si et seulement si ses valeurs propres sont positives ou nulles. Elle est définie positive si et seulement si ses valeurs propres sont strictement positives.*

Une matrice symétrique est diagonalisable, avec de plus une base orthonormée de vecteurs propres. Dans le cas non symétrique, les valeurs propres peuvent être complexes, la matrice est alors diagonalisable dans l'ensemble des complexes si les valeurs propres sont semi-simples; la matrice n'est pas diagonalisable s'il existe une valeur propre défective.

6.2 Décomposition SVD (Singular Value Decomposition) d'une matrice rectangulaire

Nous allons voir maintenant une décomposition qui généralise la diagonalisation et qui existe pour toute matrice rectangulaire.

Théorème 6.2.1 *Soit $A \in \mathbb{R}^{m\times n}$ et $p = \min(m, n)$.*
Il existe $U \in \mathbb{R}^{m\times m}$ orthogonale, il existe $V \in \mathbb{R}^{n\times n}$ orthogonale, il existe $\sigma_1 \geq \sigma_2 \geq \sigma_p \geq 0$, $\Sigma_1 = diag(\sigma_1, \sigma_2, \sigma_p) \in \mathbb{R}^{p\times p}$ matrice diagonale, tels que

$$\begin{array}{l} si\ n \geq m,\ \Sigma = \begin{pmatrix} \Sigma_1 & 0 \end{pmatrix} \\ si\ m \geq n, \Sigma = \begin{pmatrix} \Sigma_1 \\ 0 \end{pmatrix} \end{array}$$

et

$$A = U\Sigma V^T \tag{6.1}$$

Les valeurs singulières σ_i sont uniques.
Si les valeurs singulières sont toutes distinctes, les vecteurs singuliers à gauche u_i et à droite v_i sont uniques.
$\sigma_1 = \|A\|_2 = \max_{\|x\|_2=1} \|Ax\|_2$.

Preuve. Si $A = U\Sigma V^T$ alors $\|A\|_2 = \|\Sigma\|_2 = \sigma_1$. Donc si la décomposition existe, σ_1 est unique. On va montrer l'existence de la décomposition.

Soit $\sigma_1 = \|A\|_2$ et v_1 tel que $\|v_1\|_2 = 1, \|u_1\|_2 = 1, Av_1 = \sigma_1 u_1$.
Soit $U_1 = (u_1, X)$ et $V_1 = (v_1, Y)$ deux matrices orthogonales d'ordre m et n (théorème de la base incomplète).

$$U_1^T A V_1 = \begin{pmatrix} u_1^T \\ X^T \end{pmatrix} A \begin{pmatrix} v_1 & Y \end{pmatrix} = \begin{pmatrix} u_1^T A v_1 & u_1^T AY \\ X^T A v_1 & X^T AY \end{pmatrix} = \begin{pmatrix} \sigma_1 & w^T \\ 0 & B \end{pmatrix}$$

On va montrer que $w = 0$.
On a $\|U_1^T A V_1\|_2 = \|A\|_2 = \sigma_1$.
Soit $x = \begin{pmatrix} \sigma_1 \\ w \end{pmatrix}$ et $y = (U_1^T A V_1)x$. Alors

$$\|x\|_2^2 = \sigma_1^2 + w^T w \geq \sigma_1^2 \text{ et } \|y\|_2 \leq \|U_1^T A V_1\|_2 \|x\|_2 = \sigma_1 \|x\|_2$$

D'autre part,

$$y = \begin{pmatrix} \sigma_1 & w^T \\ 0 & B \end{pmatrix} \begin{pmatrix} \sigma_1 \\ w \end{pmatrix} = \begin{pmatrix} \sigma_1^2 + w^T w \\ Bw \end{pmatrix}$$

donc $\|y\|_2 \geq \sigma_1^2 + w^T w = \|x\|_2^2$ d'où $\|x\|_2 \leq \sigma_1$ et $w = 0$.
Donc

$$U_1^T A V_1 = \begin{pmatrix} \sigma_1 & 0 \\ 0 & B \end{pmatrix}$$

Nous allons maintenant faire une preuve par récurrence.
Si $A = u \in \mathbb{R}^{p\times 1}$ ou $A = v^T \in \mathbb{R}^{1\times p}$ alors la décomposition existe.
On a $B \in \mathbb{R}^{(m-1)\times(n-1)}$. Supposons que $B = U_2 \Sigma_2 V_2^T$. Soit

$$U = U_1 \begin{pmatrix} 1 & 0 \\ 0 & U_2 \end{pmatrix}, \; V = V_1 \begin{pmatrix} 1 & 0 \\ 0 & V_2 \end{pmatrix}, \; \Sigma = \begin{pmatrix} \sigma_1 & 0 \\ 0 & \Sigma_2 \end{pmatrix}$$

alors U et V sont des matrices orthogonales et

$$A = U\Sigma V^T$$

On a $\sigma_2 = \|B\|_2 \leq \sigma_1$.

Il reste à montrer l'unicité des vecteurs singuliers. On l'admet ici. ◇

Nous avons obtenu une caractérisation de la norme euclidienne de la matrice A: la norme euclidienne d'une matrice quelconque est sa plus grande valeur singulière.

Remarque 6.2.1 *Si A est symétrique, la diagonalisation de A définie par $A = VDV^T$ n'est pas une SVD. En effet, les valeurs singulières sont positives et sont les valeurs absolues des valeurs propres de A. Par contre, si A est symétrique semi-définie positive, les valeurs propres sont positives (ou nulles) et la SVD coïncide avec la diagonalisation.*

6.2.1 SVD réduite. Rang, image et noyau

Les résultats suivants permettent de caractériser le rang, l'image et le noyau de A et A^T. Ils serviront à caractériser la pseudo-inverse de A.

Proposition 6.2.1 *Soit r le nombre de valeurs singulières non nulles. Soit U_1 et V_1 les r premiers vecteurs singuliers*
$U_1 = (u_1, \dots, u_r)$ et $V_1 = (v_1, \dots, v_r)$. Soit D la matrice diagonale $D = diag(\sigma_1 \dots \sigma_r)$. Alors

$$A = U_1 D V_1^T = \sum_{j=1}^{r} \sigma_j u_j v_j^T.$$

Cette décomposition s'appelle la SVD réduite.

Proposition 6.2.2 *Soit $U = (U_1\ U_2)$ avec $U_2 = (u_{r+1}, \dots, u_m)$ et $V = (V_1\ V_2)$ avec*
$V_2 = (v_{r+1}, \dots, v_n)$. Alors
$Im(A) = Vect(U_1)$ et $ker(A) = Vect(V_2)$,
$Im(A^T) = Vect(V_1)$ et $ker(A^T) = Vect(U_2)$,
$rang(A) = r$.

Preuve.

$$A = U\Sigma V^T = (U_1\ U_2) \begin{pmatrix} D & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix} = (U_1\ U_2) \begin{pmatrix} DV_1^T \\ 0 \end{pmatrix} = U_1 D V_1^T$$

On a démontré la SVD réduite. On peut aussi l'écrire sous la forme suivante:

$$A = U_1 D V_1^T = (u_1 \dots u_r) diag(\sigma_1 \dots \sigma_r)(v_1 \dots v_r)^T = (u_1 \dots u_r)(\sigma_1 v_1 \dots \sigma_r v_r)^T = \sum_{j=1}^{r} \sigma_j u_j v_j^T.$$

$y \in Im(A) \Leftrightarrow y = U_1(DV_1^T)x \Rightarrow y \in Vect(U_1)$
$y \in Vect(U_1) \Rightarrow y = U_1 z$, soit $x = V_1 D^{-1} z$ alors $Ax = U_1 D V_1^T V_1 D^{-1} z = U_1 z = y$ donc $y \in Im(A)$.
Donc $Im(A) = Vect(U_1)$.

$x \in ker(A) \Leftrightarrow U_1 D V_1^T x = 0 \Leftrightarrow DV_1^T x = 0 \Leftrightarrow V_1^T x = 0 \Leftrightarrow x \perp V_1 \Leftrightarrow x \in Vect(V_2)$. Donc $ker(A) = Vect(V_2)$.

$Im(A^T) = ker(A)^\perp = V_2^\perp = Vect(V_1)$,
$ker(A^T) = Im(A)^\perp = U_1^\perp = Vect(U_2)$.

$rang(A) = dim(Vect(U_1)) = r$. ◇

Nous avons obtenu une caractérisation du rang d'une matrice: le rang de la matrice A est le nombre de valeurs singulières non nulles.

6.3 Approximation de rang k et rang numérique

La SVD permet aussi de déterminer des approximations d'une matrice afin de simplifier la résolution d'un problème. Elle permet aussi de calculer ce qu'on appelle le rang numérique d'une matrice.

Théorème 6.3.1 *Soit $r = rang(A)$ et $\sigma_1 \geq \sigma_r > 0 = \sigma_{r+1} = \sigma_p$ les valeurs singulières de A. Soit $k < r$ et $A_k = \sum_{j=1}^{k} \sigma_j u_j v_j^T$. Alors $rang(A_k) = k$ et*

$$\|A - A_k\|_2 = \min_{B, rang(B)=k} \|A - B\|_2 = \sigma_{k+1}.$$

Autrement dit, A_k est la meilleure approximation de A avec un rang k.

Preuve. Les valeurs singulières de A_k sont $\sigma_i, i = 1, \ldots, k$ et 0 donc $rang(A_k) = k$. On a

$$A - A_k = \sum_{j=k+1}^{r} \sigma_j u_j v_j^T, \text{ donc } \|A - A_k\|_2 = \sigma_{k+1}.$$

Soit B une matrice de rang k, alors $dim(ker(B)) = n-k$. Puisque $dim(Vect(v_1, \ldots, v_{k+1})) = k+1$, il existe $z \neq 0$ tel que $z \in ker(B) \cap Vect(v_1, \ldots, v_{k+1})$ donc tel que $Bz = 0$ et $z = \sum_{j=1}^{k+1} (v_j^T z) v_j$, soit $v_j^T z = 0,\ j > k+1$. Alors $(A - B)z = Az = \sum_{j=1}^{r} \sigma_j u_j v_j^T z = \sum_{j=1}^{k+1} \sigma_j u_j v_j^T z$ et $\|Az\|_2^2 = \sum_{j=1}^{k+1} \sigma_j^2 (v_j^T z)^2 \geq \sigma_{k+1}^2 \sum_{j=1}^{k+1} (v_j^T z)^2 = \sigma_{k+1}^2 \|z\|_2^2$. Donc

$$\|A - B\|_2 \geq \frac{\|(A - B)z\|_2}{\|z\|_2} \geq \sigma_{k+1}.$$

◇

Le rang d'une matrice est le nombre de valeurs singulières non nulles. Numériquement, il est difficile de déterminer si une très petite valeur calculée est nulle. On introduit la notion de rang numérique.

Définition 6.3.1 *Le rang numérique associé à la tolérance δ est l'entier k tel que*

$$k = \min(rang(B),\ \|A - B\|_2 \leq \delta).$$

Par application du théorème précédent, on peut caractériser k :

Proposition 6.3.1 *Le rang numérique de A vaut k si et seulement si*

$$\sigma_1 \geq \ldots \sigma_k > \delta \geq \sigma_{k+1} \geq \ldots \geq \sigma_n.$$

Preuve. Il suffit d'appliquer le théorème et la définition ci-dessus. ◇

En pratique, il peut être difficile de choisir correctement la tolérance δ.

6.4 Calcul de la SVD

Nous montrons d'abord que tout algorithme de calcul doit être itératif.

Proposition 6.4.1 *Les carrés des valeurs singulières de A sont des valeurs propres de $A^T A \in \mathbb{R}^{n \times n}$ et de $AA^T \in \mathbb{R}^{m \times m}$. Les autres valeurs propres, s'il y en a, sont nulles.*

Théorème 6.4.1 *Pour* $\min(m,n) \geq 5$*, le calcul des valeurs singulières d'une matrice de* $\mathbb{R}^{m\times n}$ *doit se faire par un algorithme itératif.*

Preuve. Les carrés des valeurs singulières sont racines des polynômes caractéristiques de A^TA de degré n et de AA^T de degré m. Or, d'après la théorie de Galois, il est impossible de calculer par formules les racines d'un polynôme de degré supérieur ou égal à cinq. ◇

Le calcul des valeurs singulières est finalement le calcul des valeurs propres de $B = A^TA$. Le calcul des valeurs propres d'une matrice symétrique se fait en deux temps : on effectue une transformation (algorithme direct) pour obtenir une matrice tridiagonale qui a les mêmes valeurs propres puis on calcule les valeurs propres (algorithme itératif) de la matrice tridiagonale.

Mais si l'on calcule explicitement B, le calcul des petites valeurs singulières devient très imprécis et l'algorithme est instable numériquement. Pour éviter cela, on choisit une autre méthode : on effectue une transformation (algorithme direct) sur A pour obtenir une matrice bidiagonale C qui a les mêmes valeurs singulières puis on calcule les valeurs singulières de cette matrice bidiagonale.

Il est à noter que la matrice C^TC est tridiagonale et peut être obtenue par tridiagonalisation de B donc la méthode suit implicitement les étapes d'une méthode de calcul de valeurs propres mais sans former explicitement les matrices B et C^TC, ce qui permet de garantir la stabilité numérique.

6.4.1 Bidiagonalisation

Ici, on suppose $m \geq n$, dans le cas contraire on peut utiliser A^T.

Théorème 6.4.2 *Soit* $A \in \mathbb{R}^{m\times n}$ *avec* $m \geq n$*. Il existe deux matrices orthogonales* $Q \in \mathbb{R}^{m\times m}$ *et* $P \in \mathbb{R}^{n\times n}$ *telles que*

$$Q^TAP = \begin{pmatrix} C \\ 0 \end{pmatrix},$$

où $C \in \mathbb{R}^{n\times n}$ *est une matrice bidiagonale.*

Preuve. admis. ◇

L'algorithme de bidiagonalisation est codé dans une procédure de LAPACK qui fait appel à BLAS3. La complexité est $4n^2(m - n/3)$.

Proposition 6.4.2 *Les valeurs singulières de* C *sont celles de* A*.*

Preuve. Soit $C = UDV^T$ la SVD de C, alors

$$A = \begin{pmatrix} Q_1U & Q_2 \end{pmatrix} \begin{pmatrix} D \\ 0 \end{pmatrix} (PV)^T$$

est une SVD de A. ◇

Les valeurs singulières de C sont calculées par une méthode itérative, qui n'est pas détaillée ici. C'est aussi une procédure de LAPACK. Il est difficile de donner une complexité, car celle-ci dépend du nombre d'itérations qui est très variable.

6.5 Analyse d'erreur du calcul de la SVD

6.5.1 Stabilité numérique du calcul d'une SVD

Nous commençons par établir la stabilité numérique de la bidiagonalisation.

Théorème 6.5.1 *Soit D la matrice bidiagonale obtenue en exécutant l'algorithme de bidiagonalisation. Alors D est le résultat d'une transformation orthogonale de la matrice $A + E$ avec $\|E\|_F \leq c\epsilon\|A\|_F$, où c est une constante.*

Preuve. admis. ◇

Le calcul itératif des valeurs singulières d'une matrice bidiagonale est également numériquement stable. L'algorithme de calcul de SVD est globalement stable.

Preuve. admis. ◇

Théorème 6.5.2 *Les valeurs singulières calculées par bidiagonalisation puis SVD sont les valeurs singulières de $A + E$ avec $\|E\|_2 \leq c\epsilon\|A\|_2$, où c est une constante.*

6.5.2 Analyse de perturbation des valeurs singulières

Le résultat suivant montre que le conditionnement absolu des valeurs singulières est inférieur à 1.

Théorème 6.5.3 *Soit A et $A+E$ deux matrices de $\mathbb{R}^{m\times n}$ avec $m \geq n$ dont les valeurs singulières sont $\sigma_i, i = 1, \ldots, n$ et $\tau_i, i = 1, \ldots, n$ respectivement. Alors*

$$|\sigma_i - \tau_i| \leq \|E\|_2.$$

Preuve. admis. ◇

Corollaire 6.5.1 *Le conditionnement relatif d'une valeur singulière σ_i est inférieure à $\frac{\sigma_1}{\sigma_i}$.*

Preuve. On a $\|A\|_2 = \sigma_1$ et $|\sigma_i - \tau_i| \leq \|E\|_2$ donc

$$\frac{|\sigma_i - \tau_i|}{\sigma_i} \leq \frac{\sigma_1}{\sigma_i}\frac{\|E\|_2}{\|A\|_2}$$

◇

6.5.3 Précision du calcul des valeurs singulières

Théorème 6.5.4 *Soit σ_i les valeurs singulières de A et τ_i les valeurs calculées par SVD avec bidiagonalisation et méthode itérative. Alors*

$$|\sigma_i - \tau_i| \leq c\epsilon\sigma_1.$$

Preuve. Il suffit d'appliquer le théorème de stabilité de l'algorithme et le théorème du conditionnement des valeurs singulières. ◇

Remarque 6.5.1 *La précision relative du calcul des petites valeurs singulières peut être dégradée dans certains cas. Il existe un algorithme plus complexe qui permet de calculer les petites valeurs singulières avec une grande précision relative.*

Chapter 7

Problèmes aux moindres carrés - cas général

Dans ce chapitre, nous utilisons la SVD pour résoudre les problèmes aux moindres carrés dans le cas général. Nous considérons toujours une matrice rectangulaire $A \in \mathbb{R}^{m\times n}$, et $p = \min m, n$. La SVD fournit une méthode de résolution de (5.1), même dans le cas où $rang(A) < n$. Le chapitre généralise la notion de pseudo-inverse.

7.1 Problèmes aux moindres carrés - cas du rang plein

7.1.1 SVD, équations normales et factorisation QR

Nous commençons par faire le lien avec les équations normales et la factorisation QR.

Proposition 7.1.1 *Les valeurs propres de A^TA et de AA^T sont les carrés des n valeurs singulières de A et 0 (s'il en reste). La SVD de A correspond aux diagonalisations de A^TA avec V base de vecteurs propres de A^TA et U base de vecteurs propres de AA^T.*

Preuve. Soit $m \geq n$, le cas $m \leq n$ est similaire.

$A = U\Sigma V^T$ d'où $A^TA = V\Sigma_1^2V^T$ et $A^TAV = V\Sigma_1^2$ donc σ_i^2 est valeur propre de A^TA et on a une diagonalisation de A^TA. ◇

Proposition 7.1.2 *Soit $A = Q\begin{pmatrix} R \\ 0 \end{pmatrix}$ la factorisation QR de A, alors les valeurs singulières de R sont celles de A et les vecteurs singuliers à droite sont les mêmes.*

Preuve. Soit $R = U\Sigma V^T$ alors

$$A = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix}\begin{pmatrix} R \\ 0 \end{pmatrix} = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix}\begin{pmatrix} U\Sigma \\ 0 \end{pmatrix}V^T = \begin{pmatrix} Q_1U & Q_2 \end{pmatrix}\begin{pmatrix} \Sigma \\ 0 \end{pmatrix}V^T$$

ce qui est une SVD de A. Par unicité des valeurs singulières, on obtient le résultat. ◇

7.1.2 SVD, inverse et pseudo-inverse

Nous faisons le lien avec l'inverse d'une matrice carrée inversible et la pseudo-inverse d'une matrice rectangulaire de rang plein.

Proposition 7.1.3 *Soit A une matrice carrée d'ordre n, A est inversible ssi $\sigma_i \neq 0, i = 1, \ldots, n$. Si A est inversible, alors*

$$A^{-1} = V\Sigma^{-1}U^T = \sum_{j=1}^{n} \frac{1}{\sigma_j} v_j u_j^T \text{ et } \|A^{-1}\|_2 = \frac{1}{\sigma_n}.$$

Preuve. U et V inversibles donc A inversible $\Leftrightarrow \Sigma$ inversible. Or Σ inversible $\Leftrightarrow \sigma_i \neq 0$.
Si A est inversible, $A^{-1} = (U\Sigma V^T)^{-1} = V\Sigma^{-1}U^T$. ◇

Théorème 7.1.1 *Soit $A \in \mathbb{R}^{m\times n}$ avec $m \geq n$, $rang(A) = n$. La matrice $D = diag(\sigma_1 \ldots \sigma_n) \in \mathbb{R}^{n\times n}$ est inversible. Par la SVD, $A = U_1 D V^T = \sum_{j=1}^{n} \sigma_j u_j v_j^T$. Alors la pseudo-inverse de A vaut*

$$A^\dagger = VD^{-1}U_1^T = \sum_{j=1}^{n} \frac{1}{\sigma_j} v_j u_j^T \text{ et } \|A^\dagger\|_2 = \frac{1}{\sigma_n}.$$

Preuve. $A^TA = VD^2V^T$ donc $(A^TA)^{-1} = VD^{-2}V^T$ et $A^\dagger = (A^TA)^{-1}A^T = VD^{-2}V^TVDU_1^T = VD^{-1}U_1^T$.

On peut aussi faire une preuve directe, connaissant $Im(A)$ et $Im(A)^\perp$. On a $b = U_1U_1^Tb + U_2U_2^Tb$ d'où $b - Ax = U_1(U_1^Tb - DV^Tx) + U_2U_2^Tb$ et $\|b - Ax\|_2^2 = \|U_1^Tb - DV^Tx\|_2^2 + \|U_2^Tb\|_2^2$ est minimum pour $DV^Tx = U_1^Tb$ et vaut alors $\|U_2^Tb\|_2$. ◇

7.2 Problèmes aux moindres carrés - cas général

Nous considérons maintenant le cas où $rang(A) \leq n$.

Théorème 7.2.1 *Soit $A \in \mathbb{R}^{m\times n}$ avec $rang(A) = r \leq min(m, n)$. Le problème*

$$\min_x \|b - Ax\|_2 \text{ avec } \|x\|_2 minimal$$

a une solution unique, qui vaut $V_1D^{-1}U_1^T\ b$.

Ce problème est équivalent à

$$b - Ax \in Im(A)^\perp, x \in ker(A)^\perp.$$

Preuve. Par la SVD, $A = U_1DV_1^T$ avec D inversible d'ordre r. On a vu que $\|b - Ax\|_2$ minimum est équivalent à $b - Ax \in Im(A)^\perp$, soit $A^T(b - Ax) = 0$.
Or $A^T(b - Ax) = V_1D(U_1^Tb - DV_1^Tx)$ d'où $A^T(b - Ax) = 0 \Leftrightarrow V_1^Tx = D^{-1}U_1^Tb$.

Donc x minimise $\|b - Ax\|_2 \Leftrightarrow V_1^Tx = D^{-1}U_1^Tb$.

Or x se décompose en $x = V_1V_1^Tx + V_2V_2^Tx$ et $\|x\|_2^2 = \|V_1^Tx\|_2^2 + \|V_2^Tx\|_2^2$. Donc, parmi tous les x qui minimisent $\|b - Ax\|_2$, celui de norme minimale est unique et tel que $V_2^Tx = 0$, autrement dit $x \in ker(A)^\perp$.

La solution unique est donc $x = V_1D^{-1}U_1^Tb$. ◇

Définition 7.2.1 *La pseudo-inverse de A est*

$$A^\dagger = V_1 D^{-1} U_1^T = \sum_{j=1}^{r} \frac{1}{\sigma_j} v_j u_j^T.$$

Remarque 7.2.1 *On retrouve bien la définition de la pseudo-inverse pour $rang(A) = n$.*

Proposition 7.2.1 *La solution unique de norme minimale qui minimise $\|b - Ax\|_2$ est*

$$x = A^\dagger b.$$

Preuve. évident d'après le théorème précédent. ◇

Proposition 7.2.2 *Soit $A \in \mathbb{R}^{m\times n}$ et $A = U\Sigma V^T$ la SVD de A, avec*

$$\Sigma = \begin{pmatrix} D & 0 \\ 0 & 0 \end{pmatrix}.$$

Alors

$$\Sigma^\dagger = \begin{pmatrix} D^{-1} & 0 \\ 0 & 0 \end{pmatrix} \text{ et } A^\dagger = V\Sigma^\dagger U^T.$$

Preuve. Dans la SVD de Σ, les vecteurs singuliers sont les colonnes des matrices identité d'où la valeur de $\Sigma^\dagger$. On a

$$V\Sigma^\dagger U^T = (V_1\ V_2) \begin{pmatrix} D^{-1} & 0 \\ 0 & 0 \end{pmatrix} (U_1\ U_2)^T = V_1 D^{-1} U_1^T = A^\dagger.$$

◇

Remarque 7.2.2 *L'égalité $A^\dagger = V\Sigma^\dagger U^T$ sur la pseudo-inverse généralise l'égalité $A^{-1} = V\Sigma^{-1}U^T$ sur l'inverse.*

La pseudo-inverse vérifie certaines propriétés, résumées ci-dessous.

Proposition 7.2.3 $\|A^\dagger\|_2 = \frac{1}{\sigma_r}$,
$(A^\dagger)^\dagger = A$,
$(A^\dagger)^T = (A^T)^\dagger$,
$A^\dagger A = V_1 V_1^T$ *et* $AA^\dagger = U_1 U_1^T$,
Si $rang(A) = n$ *alors* $A^\dagger A = I_n$,
Il existe A telle que $A^\dagger A \neq AA^\dagger$,
Il existe A, B telles que $(AB)^\dagger \neq B^\dagger A^\dagger$.

Preuve.
Soit $A = U_1 D V_1^T$ alors la décomposition $A^\dagger = V_1 D^{-1} U_1^T$ est issue de la SVD de $A^\dagger$.
Donc $\|A^\dagger\|_2 = \frac{1}{\sigma_r}$ et $(A^\dagger)^\dagger = U_1 D V_1^T = A$.
$(A^T)^\dagger = (V_1 D U_1^T)^\dagger = U_1 D^{-1} V_1^T = (A^\dagger)^T$.
Soit $A = (1\ \ 0)$ alors $A^\dagger = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ et $AA^\dagger = 1$ tandis que $A^\dagger A = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$.
Soit $B = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$ alors $B^\dagger = (B^T B)^{-1} B^T = \frac{1}{2}(11)$ et $(AB)^\dagger = 1$ tandis que $B^\dagger A^\dagger = \frac{1}{2}$. ◇

7.2.1 Résolution d'un problème aux moindres carrés avec la SVD

La solution du problème aux moindres carrés (5.1) est donnée par

$$x = V_1 D^{-1}(U_1^T b),$$

où $A = U_1 D V_1^T$ est la SVD réduite de A ; les étapes de résolution et leurs complexités sont résumées dans le tableau ci-dessous.

étape	bibliothèque	nombre d'opérations
bidiagonalisation	LAPACK	$4mn^2 + O(mn) + O(n^3)$
SVD bidiagonale	LAPACK	inconnu
$c = U_1^T b$	BLAS2 $O(rm)$	
$y = D^{-1}c$	BLAS1	$O(r)$
$x = V_1 y$	BLAS2	$O(rn)$

Donc, pour un problème aux moindres carrés, la complexité est nettement supérieure à celles des méthodes des équations normales et de la factorisation QR. Par contre, la SVD permet de traiter les cas avec un rang non plein, alors que les deux autres méthodes ne peuvent être utilisées que pour les matrices de rang plein.

7.3 SVD et conditionnement des matrices

Nous définissons le conditionnement d'une matrice quelconque, que nous relions aux valeurs singulières.

Définition 7.3.1 *Soit $A \in \mathbb{R}^{m\times n}$ avec $rang(A) = r \leq \min(m,n)$. Le conditionnement de la matrice A pour les problèmes aux moindres carrés est défini par $cond(A) = \|A\|_2\|A^\dagger\|_2$.*

Proposition 7.3.1 *Soit $\sigma_1 \geq \ldots \geq \sigma_r > 0$ les valeurs singulières non nulles de A, alors $cond(A) = \frac{\sigma_1}{\sigma_r}$.*

Preuve. $\sigma_1 = \|A\|_2$ et $\|A^\dagger\|_2 = 1/\sigma_r$. ◇

Proposition 7.3.2 *Soit $A \in \mathbb{R}^{n\times n}$ une matrice inversible; soit $\sigma_1 \geq \ldots \geq \sigma_n > 0$ les valeurs singulières de A, alors*

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

Preuve. Le résultat est immédiat d'après la proposition précédente. ◇

Proposition 7.3.3 *$cond(A^TA) = cond(A)^2$.*
Soit $A = Q_1 R$ alors $cond(A) = cond(R)$.

Preuve. Les valeurs singulières de A^TA sont les carrés des valeurs singulières de A, qui sont égales aux valeurs singulières de R. ◇

Chapter 8

Conclusion

On veut résoudre le problème

$$\min_x \|b - Ax\|_2, \text{ où } A \in \mathbb{R}^{m \times n}.$$

Trois cas se présentent :

- la matrice est carrée et inversible: on résout $Ax = b$, qui a une solution unique.
- la matrice est rectangulaire et de rang plein: on résout $\min_x \|b - Ax\|_2$, qui a une solution unique.
- la matrice peut ne pas être de rang plein: on résout $\min_x \|b - Ax\|_2$ et $\|x\|_2$ minimal, qui a une solution unique.

La matrice a un conditionnement $cond(A)$ et on effectue les calculs avec une arithmétique flottante respectant la norme IEEE et de précision machine ϵ.

Dans le premier cas (matrice carrée inversible), on distingue deux sous-cas:

- la matrice est symétrique définie positive: on utilise la factorisation de Cholesky $A = LDL^T$ et on résout les systèmes $Ly = b, Dz = y, L^T x = z$. La complexité est $n^3/3 + O(n^2)$. La précision est approximativement en $0(cond(A)\epsilon$.
- la matrice est quelconque: on utilise la factorisation de Gauss avec pivot partiel $PA = LU$ et on résout les systèmes $Ly = Pb, Ux = y$. La complexité est $2n^3/3 + O(n^2)$. La précision est approximativement en $0(cond(A)\epsilon$.

Pour le deuxième cas (matrice de rang plein), il existe principalement deux méthodes de résolution:

- la méthode des équations normales : on effectue la factorisation de Cholesky

  $$A^T A = LDL^T$$

  puis on résout les systèmes triangulaires

  $$Ly = A^T b, Dz = y, L^T x = z$$

  Cette méthode a une complexité en $mn^2 + O(mn)$ et une précision en $O(cond(A)^2\epsilon)$.

- la factorisation QR : on effectue la factorisation

$$A = Q_1 R$$

et on résout le système triangulaire

$$Rx = Q_1^T b$$

Cette méthode a une complexité en $2mn^2 + O(mn)$ et une précision en $O(cond(A)\epsilon)$ si la norme du résidu $\|Ax - b\|_2$ est petite (x est la solution exacte, sans arrondi). Donc elle est plus coûteuse mais plus précise que la méthode des équations normales.

Pour le troisième cas (matrice de rang quelconque), il existe principalement une méthode de résolution. On effectue la Décomposition en Valeurs Singulières Réduite :

$$A = U_1 D V_1^T$$

puis on résout le système diagonal

$$x = V_1 D^{-1} U_1^T b$$

Cette méthode est aussi valide dans les deux premiers cas. Cette méthode (avec une partie itérative) a une complexité en $O(mn^2)$ et une précision comme QR. Elle est plus coûteuse que QR pour la même précision. Par conséquent, elle est surtout utilisée lorsque la matrice n'est pas de rang plein.

Bibliography

[1] J-C. Bajard, O. Beaumont, J-M. Chesneaux, M. Daumas, J. Erheland D. Michelucci, J-M. Muller, B. Philippe, N. Revol, J-L.Roch, and J. Vignes. *Qualité des Calculs sur Ordinateurs. Vers des arithmétiques plusfiables?* Masson, 1997.

[2] O. Beaumont. *Algorithmique pour les intervalles : Comment obtenir un résultat sûr quandles données sont incertaines.* thèse de doctorat, université de Rennes 1, January 1999.

[3] A. Björck. *Numerical Methods for Least Squares Problems.* SIAM, 1996.

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

[5] Jean-Marie Chesnaux. *L'Arithmétique Stochastique et le Logiciel CADNA.* Habilitation à diriger des recherche, Université Paris VI, 1995.

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

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

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

[9] Philippe Langlois. *Précision finie et méthodes automatiques.* Habilitation à diriger des recherche, Université de Lyon 1, 2001.

[10] J-M. Muller. *Elementary Functions, Algorithms and Implementation.* Birkhauser Boston,, 1997.

[11] J.M. Ortega and W.C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables.* Computer science and applied mathematics. Academic Press, 1970.

[12] Douglas M. Priest. Algorithms for arbitrary precision floating point arithmetic. In Peter Kornerup and David Matula, editors, *Proceedings of the 10th Symposium on Computer Arithmetic*, pages 132–144, Grenoble, France, 1991. IEEE Computer Society Press.

[13] American National Standard. Ieee standard for binary floating-point arithmetic. ANSI/IEEE Std. 754, IEEE, 1985.

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