République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

**UNIVERSITÉ MOHAMED KHIDER, BISKRA**

FACULTÉ des SCIENCES EXACTES et des SCIENCES de la NATURE et de la VIE

**DÉPARTEMENT DE MATHÉMATIQUES**

Mémoire présenté en vue de l'obtention du Diplôme **:**

**MASTER en Mathématiques**

Option : **Analyse**

Par

**Guercif Lamya**

Titre :

# Comparaison des Problèmes Numériques

Membres du Comité d'Examen :

| | | | |
|---|---|---|---|
| Dr. | Mokhtari Zouhir | UMKB | Président |
| Dr. | Rajah Faouzia | UMKB | Encadreur |
| Dr. | Soltani Sihem | UMKB | Examinateur |

Juin 2020

*DEDICACE*

C'est avec l'aide de mon DIEU tout puissant que travail a pu être réalisé,

DIEU qui m'a donné fois, raison et lucidité.

Merci mille fois et que ''Allah " vont garder et protèger.

Mes chers parents *"Guercif Ammar et Hammami Ghezala"* pour leurs amours, leurs patiences et leurs encouragement qui ne jamais cassé de me convenir

durant mes années d'études.

Mes sœurs *"Abida, Kanza"* et la femme de mon frère *"Sihem"*.

Mes frères *"Hamid, Adel et Mouhamed"*.

Les fils de mon frère *"Bailsan et Lokman"*.

Mes amies proches.

Toute ma famille *"Guercif et Hammami"*.

Tous mes collègues de ma promotion 2020.

## *REMERCIEMENTS*

C'est avec l'aide de mon *DIEU* tout puissant que ce modeste travail a pu être réalisé,
D*IEU* qui m'a donné fois, raison et lucidité.

Je tiens à remercier mon encadreur madame *Rajah Faouzia*, pour m'avoir donné l'opportunité de travailler sur ce projet,

pour son grand soutient scientifique et moral, pour les suggestions et les encouragements qu'il m'a apportés durant mon projet.

Mon sincère remerciement aux membres de jury monsieur *Mokhtari* et madame *Soltani*
qui ont accepté de juger mon travail.

Je remercie vivement tous les enseignants de notre département qui ont toujours donné
le meilleur d'eux-mêmes afin de nous assurer une formation de qualité.

Je n'oublie pas de remercier ma famille d'être avec nous dans tous les moments.

Enfin je m'exprime ma profonde reconnaissance à toutes les personnes qui
Ont contribué de près

Ou de loin pour la réalisation de ce travail.

# Table des matières

# Introduction

L'objectif de l'analyse numérique est de concevoir et d'étudier des méthodes de résolution de certains problèmes mathématiques ; en général issus de la modélisation des problèmes réels, et on cherche à calculer la solution à l'aide d'un ordinateur.

Mais le problème qui se pose ici quand on résout un problème donné sur un ordinateur, généralement de façon approchée ; outre la complexité de calcul de l'algorithme utilisé et l'espace mémoire nécessaire, on est souvent attentif à la précision des résultats obtenus.

Il est bien connu que cette précision dépend à la fois de la qualité *(stabilité numérique)* de l'algorithme et de la nature *(conditionnement)* même du problème à résoudre. Ces deux notions, pour l'instant qualitatives, seront outre l'imprécision due aux incertitudes sur les données, une erreur apparaît au cours de l'exécution de l'algorithme de résolution. Cette erreur provient du fait que l'algorithme est nécessairement fini et que chaque opération arithmétique s'accompagne en principe d'une *«erreur d'arrondi»*.

L'accumulation de telles erreurs peut très bien fausser grossièrement les résultats. Avec une analyse explicite de la propagation des erreurs, on arrive à déceler ces deux sources d'imprécision (imprécision due aux erreurs des données et celle due aux erreurs d'arrondi) et à les bornes séparément.

Ainsi, à partir de l'erreur sur la solution due aux erreurs des données propaguées, on étudie le conditionnement du problème *(comment varie la solution quand les données sont modifiées)* et à partir de l'erreur due aux arrondis propaguées, on peut juger la qualité *(stabilité numérique)*.

Alors, pour bien choisir entre les algorithmes proposés, on essaye ; dans ce mémoire ; d'étudier les différents critères de comparaison entres les méthodes numériques et pour démontrer l'importance de cette étude, on présente une méthode classique connue d'une complexité très élevé ; après, on propose un algorithme plus performant et plus efficace.

Pour aboutir à notre but, nous avons répartie ce travail en trois chapitres organisés comme suit :

**Chapitre 1 :** *Calcul scientifique et complexité* : Dans la premier chapitre, on donne toutes les définitions nécessaires pour réaliser notre étude ; donc, il est consacré à la représentation des nombres réels et la représentation en machine. Comme la machine a une mémoire de précision bien définie, il est naturel de faire l'arrondissement d'un nombre et donner l'erreur d'arrondi maximale pour un flottant ; enfin, on présente la notion de complexité.

**Chapitre 2 :** *Conditionnement et stabilité numérique :* On présente dans ce chapitre les deux notions de base qui nous permet de faire notre comparaison. tout d'abord, on donne la définition de conditionnement d'une façon générale et le conditionnement d'une matrice ; ensuite, on expose la stabilité numérique et la théorie des perturbations.

**Chapitre 3 :** Dans le dernier chapitre, on va faire une amélioration sur une méthode numérique de résolution d'un système linéaire mal conditionné afin d'aboutir une nouvelle méthode numériquement stable, bien conditionnée et économique en cases mémoires par l'utilisation des différences divisées.

# Chapitre 1

# Calcul scientifique et complexité

Dans ce chapitre, on essaye de donner les représentations des nombres (représentation réelle et représentation machine) pour démontrer que les calculs scientifiques en machine, on ne peut pas effectuer correctement à cause de son système discret. A cette raison, on cherche toujours à réduire le nombre d'opérations nécessaires pour effectuer un problème numérique.

## 1.1 Représentation des nombres réels

Un réel $x \in \mathbb{R}$ se compose toujours en deux parties : une partie entière $\mathbb{E}(x)$ et une partie fractionnaire $F(x)$ :

$$x = E(x) + F(x), \ E(x) \in \mathbb{Z}$$

et

$$F(x) = x - E(x) \in [0, 1],$$

ne pas confondre $E(X)$ avec la troncature à l'unité d'un nombre ; c'est-à-dire : la suppression des décimales.

Souvent, on peut pas utiliser la représentation en virgule fixe pour écrire les nombre réels, à titre d'exemple :

- La masse de la terre est de 597360000000000000000000000 $kg$ ;
- La masse du soleil est de $19891000\underbrace{\ldots\ldots\ldots\ldots}_{\text{26 Zéros}}000$ $kg$;
- La masse de l'électron est de $0,00\underbrace{\ldots\ldots\ldots\ldots}_{\text{27 Zéros}}0091093822$ grammes;
- La masse d'un proton est de $0,00\underbrace{\ldots\ldots\ldots\ldots}_{\text{23Zéros}}00167226$ grammes;

Dans ces cas, il faut mieux utiliser la notation scientifique suivante :

$$a \times 10^e,\ e \in \mathbb{Z}.$$

Pour la notation normalisée, on a :

$$0,1 \leq |a| < 1.$$

Si on se base sur cette écriture, les masses précédentes sont : $5,9736 \times 10^{24}$ $kg$ et $1,9891 \times 10^{30}$ $kg$ pour la terre et le soleil resp et $9,101093822 \times 10^{-31} kg$ et $1,6726 \times 10^{-27} kg$ pour l'électron et le proton.

Malheureusement, la machine a un système discret et une précision finie, c'est-à-dire : on peut pas représenter les nombres qu'approximativement ; donc, on doit connait la représentation machine, ce qu'on va présenter dans la section suivante.

## 1.2 Représentation des nombres réels en machine

Puisque la capacité d'un ordinateur est finie, il faut choisir la taille du mot mémoire influence la précision de la représentation des nombres.

- Pour représenter exactement un rationnel $r = \dfrac{n}{d}$, il faut garder le numérateur $n \in \mathbb{Z}$ et le dénominateur $d \in \mathbb{N}^*$ sauf si dans la base choisie, $r$ admet un développement fini.
- Un nombre irrationnel $x \in \mathbb{R}\backslash\mathbb{Q}$ ne peut pas jamais être représenter exactement sur un

ordinateur, on utilise la représentation en virgule flottante de la forme :

$$x = s \times m \times b^e,$$

où $b$ est la base utilisée, $s \in \{-1, 1\}$ est le signe, $m$ est la mantisse de $x$ où significande ; c'est-à-dire : elle précise les chiffres significatifs et $e$ est l'exposant donne l'ordre du grandeur.

**Exemple (1.1) :** En base 10 : $-37.5 = -37500 \times 10^{-3} = -0.000375 \times 10^5 = -0.375 \times 10^2$. Alors $x = -0.375 \times 10^2$; a cette dernière définition, il est facile de remarquer que :

• Le signe du nombre $s = (-1)^{s_m}$ , avec $s_m \in \{0, 1\}$ ;

• La mantisse $m$ est un réel dans $]0, 1[$ ; c'est-à-dire : tous les chiffres significatifs sont à droite de la virgule.

• L'exposant $e$ est un entier relatif.

• La virgule et la base sont représentées de façon implicite ; en machine la représentation est unique

| $s_m$ | $e$ | $d_{-1}$ | $d_{-2}$ | $\cdots$ | $d_{-p}$ |
|---|---|---|---|---|---|

$= (-1)^{s_m}\, 0, d_{-1}d_{-2}\cdots d_{-p} \times 10^e,$

avec $d_{-1} \neq 0$. Par convention, la représentation de 0 ne contient que des zéros.

On considère une représentation avec :

*1.* Une mantisse de 3 chiffres décimaux.

*2.* Un exposant de 2 chiffres décimaux.

*3.* Deux bits de signe.

**Exemple (1.2) :** $37.5 = 0.375 \times 10^2$ est représenté en machine par :

| + | + | 0 | 2 | 3 | 7 | 5 |
|---|---|---|---|---|---|---|

Les nombres strictement positifs représentables vont de : $+0,100 \times 10^{-99}$ :

| | | | | | | |
|---|---|---|---|---|---|---|
| + | − | 9 | 9 | 1 | 0 | 0 |

à : $+0,999 \times 10^{+99}$ :

| | | | | | | |
|---|---|---|---|---|---|---|
| + | + | 9 | 9 | 9 | 9 | 9 |

.

Les nombres strictement négatifs sont variés de : $-0.999 \times 10^{-99}$ :

| | | | | | | |
|---|---|---|---|---|---|---|
| − | + | 9 | 9 | 9 | 9 | 9 |

à $-0,100 \times 10^{-99}$ :

| | | | | | | |
|---|---|---|---|---|---|---|
| + | − | 9 | 9 | 1 | 0 | 0 |

.

Dans la représentation en virgule flottante, on trouve souvent les situations de dépassement (overflow, underflow). Lorsqu'on choisit un système en virgule flottante de précision finie :

• Les nombres 3 et 7 sont représentés par : $3 = 0,30 \times 10^1$ et $7 = 0,70 \times 10^1$; Mais le résultat de $3 + 7 = 10 = 0,10 \times 10^2$ n'est plus représentable, d'où overflow.

• Les nombres $0,010 \times 10^{-1}$ et $0,011 \times 10^{-1} = 0,11 \times 10^{-1}$ sont représentables, mais leur différence : $0,011 - 0,010 = 0.001 = 0.10 \times 10^{-2}$ ne l'est pas.

• De même $0,010/2 = 0,005 < 0,010$ n'est pas représentable ; on a donc affaire à un underflow.

- Si on relâche la condition que le digit de plus fort poids sont non nul, on peut représenter ces nombres : $0,001 = 0,01 \times 10^{-1}$ et $0,005 = 0,05 \times 10^{-1.}$.

- Ces nombres ; ≪*sous-normaux*≫; *subnormal* sont utilisés pour représenter des quantités très petites mais non nuls.

Un opérateur quelconque ayant comme résultat un nombre $x$ non représentable engendre

une erreur d'arrondi ; On doit approximer le *"vrai"* résultat $x$ par un réel $\tilde{x}$ représentable dans le système en virgule flottante choisi.

Dans ce qui suit, on va présenter les méthodes d'approximations d'un nombre en machine de sorte que la valeur approchée $\tilde{x}$ est plus proche de la valeur exacte $x$ ; c'est-à-dire : on cherche à réaliser une erreur d'arrondi minimale.

## 1.3 Arrondissement (Troncature) d'un nombre

Dans cette section, il est important de préoccuper de la manière d'arrondi les nombres dans une machine. On a vu précédemment qu'un nombre est représenté par un nombre fini de caractères ; fixé à l'avance ; qui dépend de l'architecture de la machine.

Pour définir les différents types d'arrondissement, on introduit la définition de l'ensemble machine. On note par $E_t$ l'ensemble suivant :

$$x \in E_t \iff x = 0 \text{ ou } x = \pm 0, d_1 d_2 \cdots d_t \times 10^e;$$

où

$$d_1, d_2, \ldots, d_t \in \{0, 1, 2, \ldots, 9\} \text{ et } d_1 \neq 0.$$

*Arrondi par défaut*

Ce type d'arrondi est très simple parce qu'on prend les "$t$" premiers chiffres significatifs lorsqu'on fait la troncature. Si

$$x = \pm 0, d_1 d_2 \cdots d_t d_{t+1} \cdots \times 10^e,$$

on applique sur ce nombre :

$$\begin{aligned} Rd_p : \ & \mathbb{R} \longrightarrow E_t \\ & x \longrightarrow Rd_p(x) = \pm 0, d_1 d_2 \cdots d_t \times 10^e. \end{aligned}$$

*Arrondi par excès*

Le deuxième type est l'arrondissement par excès qui fait lorsqu'on ajoute 1 à la t$^{ème}$ chiffre significatif de la manière suivante :

$$\begin{aligned} Rd_p^+ : \ & \mathbb{R} \longrightarrow E_t \\ & x \longrightarrow Rd_p^+(x) \pm 0, d_1 d_2 \cdots (d_t + 1) \times 10^e. \end{aligned}$$

Les deux types d'arrondissement ne répond pas a notre besoin de précision dans les calculs scientifiques.

*Arrondi pair*

on appelle arrondi pair l'application "$fl$"; cette application réalise la minimalité, c'est-à-dire : lorsqu'on tronque un réel par $fl$, ce nombre approché est le plus proche de la valeur exacte. Cette application est définie par :

$$\begin{aligned} fl : \ & \mathbb{R} \longrightarrow E_t \\ & x \longrightarrow fl(x) \end{aligned}$$

où

$$fl(x) = \begin{cases} Rd_p(x) & \text{si } 0 \leq d_{t+1} \leq 4 \\ Rd_p^+(x) & \text{si } 6 \leq d_{t+1} \leq 9. \end{cases}$$

Le cas où $d_{t+1} = 5$ : ici, on distingue deux cas :

* Si $(\exists j > t+1)$ ; tel que : $d_j \neq 0$ : $fl(x) = Rd_p^+(x)$.

* Si $(\forall j > t+1)$; tel que : $d_j = 0$ : $f(x) = \begin{cases} Rd_p & \text{si } d_t \text{ est pair} \\ Rd_p^+(x) & \text{si } d_t \text{ est impair.} \end{cases}$

*Opérations machines (Opérations flottantes)*

Il est important aussi de connaitre la manière de manipuler les opérations sur une machine. Toute opération élémentaire $(+, -, \times, /, \ldots)$ est en général entachée d'erreur ; c'est-à-dire : pour effecteur une opération quelconque sur deux nombres réels, on effectue une opération sur leurs représentations flottantes et on prend ensuite la représentation flottante du résultat.

$x \oplus y = fl(fl(x) + fl(y))$.

$x \ominus y = fl(fl(x) - fl(y))$.

$x \otimes y = fl(fl(x) \times fl(y))$.

$x \oslash y = fl(fl(x)/fl(y))$.

## 1.4 Erreur d'arrondi maximale pour un flottant donné

Dans la section précédente, nous avons vu que les réels étaient arrondis au flottant le plus proche au cours d'un processus ; donc une erreur d'arrondi est faite.

**Définition (1.1) :** *l'erreur d'arrondi est l'écart entre le réel de départ et le flottant vers lequel il est converti lors de l'opération d'arrondi. si on note : $r$ le réel de départ, $f$ le flottant résultant de l'arrondi ; et $\delta$ l'écart entre les deux, cette définition se traduit par la relation suivante :*

$$r = f \pm \delta.$$

Au cours des calculs scientifiques, on peut pas trouver généralement l'erreur d'arrondi ; exacte ; mais on peut par contre de la majorée.

Sur l'intervalle compris entre le plus grand flottant positif $+\Omega$ et le plus petit flottant négatif $-\Omega$, il est possible d'estimer l'erreur d'arrondi maximale qui est en fonction du réel arrondi. En dehors de cet intervalle, l'erreur peut être arbitrairement grande ; donc, on a un phénomène de dépassement.

Chaque flottant de l'intervalle $]-\Omega;+\Omega[$ est encadré par un prédécesseur direct et un successeur direct. Pour un flottant $f$, si on note $f^-$ son prédécesseur direct et $f^+$ son successeur direct, on a la relation suivante :

$$f^- < f < f^+.$$

Dans le mode d'arrondi au plus proche, les réels compris entre $\frac{f^- + f}{2}$ et $\frac{f^+ + f}{2}$ seront arrondis vers $f$.

On peut aussi estimer l'erreur d'arrondi maximale ; si un réel $r$ est arrondi vers un flottant $f$, alors l'erreur d'arrondi est au plus égal à :

$$|\delta_{\max}| = \left|\frac{f^+ - f}{2}\right| = \left|\frac{f^- - f}{2}\right|.$$

Il ne reste plus qu'a expliciter $f^+$ et $f^-$ en fonction de $f$ pour calculer une valeur numérique de cette erreur.

*Valeur de l'erreur d'arrondi maximale en fonction d'un flottant*

Prenons un flottant $f$ de la forme :

$$f = s \times 2^e.$$

Le successeur direct de $f$, $f^+$ est obtenu en incrémentant le dernier bit du significande de $f$ ; son significande de $s^+$ est ainsi égale à : $s^+ = s + 2^{-52}$.

Le prédécesseur direct de $f$; $f^-$ ; est obtenu de manière similaire, mais en décrémentant le dernier bit du significande de $f$ ; son significande de $s^-$ vaut alors :

$$s^- = s - 2^{-52}.$$

On peut alors utiliser la formule du paragraphe précédent pour calculer l'erreur d'arrondi

maximale :

$$\begin{aligned} |\delta_{\max}| &= \frac{s - s^-}{2} \times 2^e \\ &= \frac{s^+ - s}{2} \times 2^e \\ &= 2^{-52} \times 2^e. \end{aligned}$$

On a donc une erreur d'arrondi d'ou paire $2^{-53}$ sur le significande, l'erreur sur le flottant dans son ensemble dépond de l'exposant $e$ et vaut au plus $2^{e-53}$.

**Exemple (1.3) :** *Si l'on estime l'erreur d'arrondi pour un réel donné ; un nombre qui n' est pas représentable de manière exacte ; la question qui se pose ici est : quelle est l'erreur d'arrondi maximale qui sera fait ?. Prenons par exemple le réel* $2,2$.

La première étape pour estimer l'erreur d'arrondi maximale consiste à trouver l'exposant. Pour cela, il suffit d'encadrer le nombre entre deux flottant, d'exposants consécutifs :

$$2^1 \leq 2,2 \leq 2^2,$$

l'exposant qu'aura $2,2$ une fois converti en flottant est donc il suffit maintenant d'appliquer la formule vue précédemment pour obtenir l'erreur maximale :

$$|\delta_{\max}| = 2^{e-53} = 2^{1-53} = 2^{-52} \simeq 2,22 \times 10^{-16}.$$

Il est également possible de calculer la véritable erreur d'arrondi par exemple à l'aide d'un calculateur en précision arbitraire. On obtient approximativement la valeur suivante :

$$|\delta| = 1,78 \times 10^{-16} < \delta_{\max},$$

l'erreur exacte est donc bien inférieure à l'erreur maximale.

## 1.5 Complexité

Dans cette section, on présente un outil de comparaison entre les méthodes numériques ou les algorithmes.

**Définition (1.2) :** *Une méthode numérique ou un algorithme est une procédure de calcule bien définie qui prend en entrée une valeur ; ou ensemble de valeurs ; et qui donne en sortie une valeur ; ou un ensemble de valeurs. Un algorithme est donc une séquence d'étapes de calcule qui transforment l'entrée en sortie.*

### 1.5.1 Complexité d'un algorithme

La complexité d'un algorithme est le nombre d'opérations élémentaires qu'il doit effectuer pendant l'exécution d'un calcul en fonction de la taille des donnés d'entrées pour stockmeyer et chandre ; "l'efficacité d'un algorithme est mesurée par l'augmentation du temps de calcul en fonction du nombre des données" ; Nous avons donc deux éléments à prendre en compte :

*1.* La taille des données.

*2.* Le temps de calcul.

#### Taille des données

La taille des données (ou des entrées) va dépendre du codage de ces entrées on choisit comme taille la ou les dimensions les plus significatives.

• Des éléments : le nombre d'éléments.

• Des nombres : nombre de bits nécessaires à la représentation de ceux -là.

• Des polynômes : le degré, le nombre de coefficients non nul.

• Des matrices $m \times n$ : $\max(m, n), mn, m + n$.

• Des graphes : nombre de sommets, nombre d'arcs, produit des deux.

• Des listes, tableaux, fichiers : nombre de cases, d'éléments.

• Des mots : leurs longueurs.

### Temps de calcul

Le temps de calcul d'un programme dépend de plusieurs éléments :

*1.* La quantité de données bien sur.

*2.* Mais aussi de leurs encodages.

*3.* De la qualité du code engendré par le compilateur.

*4.* De la nature et la rapidité des instructions du langage.

*5.* De la qualité de la programmation.

*6.* Et de l'efficacité de l'algorithme.

Nous ne voulons pas mesurer le temps de calcul par rapport à toutes ces variables ; mais, nous cherchons à calculer la complexité des algorithmes qui ne dépendra ni de l'ordinateur, ni du langage utilisé, ni du programmeur ni de l'implémentation ; pour cela, nous allons mettre dans le cas ou nous utilisons un ordinateur *RAM (Random Access Machine)* :

*1.* Ordinateur idéalisé.

*2.* Mémoire infinie.

*3.* Accès à la mémoire en temps constant.

*4.* Généralement à processeur unique (pas d'opérations simultanées).

Pour connaitre le temps de calcul, nous choisissons une opération fondamentale et nous calculons le nombre d'opérations fondamentales exécutées par l'algorithme.

#### *Opération fondamentale*

C'est la nature du problème qui fait que certaines opérations deviennent plus fondamentales que d'autres dans un algorithme ;

Exemple (1.4) : Le tableau suivant donne les opérations nécessaires de quelques problèmes :

| *Problème* | *Opération fondamentale* |
|---|---|
| Recherche d'un élément dans une liste | Comparaisons |
| Tri d'une liste, d'un fichier | Comparaisons, déplacements |
| Multiplication des matrices réelles | Multiplications et additions |
| Addition des entiers binaire | Opérations binaires |

### 1.5.2 Complexité d'un problème

La complexité d'un problème $A$ est la complexité du meilleur algorithme qui résout $A$ et on distingue la complexité en trois cas différents :

*Complexité dans le meilleur des cas*

$A$ un algorithme, $n$ entier, $D_n$ l'ensemble des entrées de taille $n$ et une entrée $d \in D_n$. Posons : $coût(d)$ le nombre d'opérations fondamentales effectuées par $A$, avec l'entrée $d$.

Pour bien définir la complexité d'un algorithme, on donne la complexité dans le meilleur des cas est obtenue par :

$$Min_A(n) = \min\left[coût(d)/d \in D_n\right].$$

*Complexité dans le pire des cas*

Dans ce cas on définit la complexité dans le pire des cas par :

$$Max_A(n) = \max\left[coût(d)/d \in D_n\right].$$

*Complexité en moyenne*

Enfin, la complexité en moyenne est donnée par :

$$Moy(n) = \sum_{d \in D_A} p(d).coût(d),$$

avec $p(d)$ une loi de probabilité sur les entrées.

# Chapitre 2

# Conditionnement et stabilité numérique

A cause de l'accumulation des erreurs pendant l'exécution d'une série d'opérations d'un problème ou d'une méthode numérique, on aura besoin d'estimer notre précision de calcul ; donc, on présente les deux notions de base : conditionnement et stabilité numérique.

## 2.1 Conditionnement

Dans cette section, on va définir la notion de conditionnement d'une façon générale et le conditionnement d'une matrice, qui peut servir à établir une majoration des erreurs d'arrondi dues eux erreurs sur les données. Malheureusement, nous verrons également que cette majoration n'est pas forcément très utile dans des cas pratiques.

### 2.1.1 Conditionnement et majoration de l'erreur d'arrondi

Soient $A \in \mathcal{M}_N(\mathbb{R})$; l'ensemble des matrices réelles inversibles de dimension $(N \times N)$ et $b \in \mathbb{R}^N$. Supposons que les données $A$ et $b$ ne soient connues qu'à une erreur près ; Ceci est souvent le cas dans les applications pratiques.

**Définition (2.1)** *(conditionnement) : Soit $\mathbb{R}^N$ muni d'une norme $\|.\|$ et $\mathcal{M}_N(\mathbb{R})$ muni de la norme induite. On appelle conditionnement d'une matrice inversible $A \in \mathcal{M}_N(\mathbb{R})$ par rapport à la norme $\|.\|$ le nombre réel positif $cond(A)$ définit par :*

$$cond(A) = \|A\| \left\|A^{-1}\right\|.$$

**Proposition (2.1)** *(propriétés générales du conditionnement) : Soit $\mathbb{R}^N$ muni d'une norme $\|.\|$ et $\mathcal{M}_N(\mathbb{R})$ muni de la norme induite :*

*1. Soit $A \in \mathcal{M}_N(\mathbb{R})$ une matrice inversible, alors $cond(A) \geq 1$.*

*2. Soit $A \in \mathcal{M}_N(\mathbb{R})$ et $\alpha \in \mathbb{R}^*$, alors $cond(\alpha A) = \alpha cond(A)$.*

*3. Soient $A$ et $B \in \mathcal{M}_N(\mathbb{R})$ deux matrices inversibles, alors $cond(AB) \leq cond(A)cond(B)$.*

**Définition (2.2)** *(propriétés du conditionnement pour la norme 2) : Soit $\mathbb{R}^N$ muni de la norme euclidienne $\|.\|_2$ et $\mathcal{M}_N(\mathbb{R})$ muni de la norme induite. On considère une matrice inversible $A \in \mathcal{M}_N(\mathbb{R})$;on note par $cond_2(A)$ le conditionnement associé à la norme induite par la norme euclidienne sur $\mathbb{R}^N$.*

*1.* Soit $A \in \mathcal{M}_N(\mathbb{R})$ une matrice inversible, on note par $\sigma_N$ (resp $\sigma_1$) la plus grande (resp la plus petite) valeur propre de $A^tA$ ($A^tA$ est une matrice symétrique définie positive). Alors :

$$cond_2(A) = \sqrt{\sigma_N/\sigma_1}.$$

*2.* Si de plus $A$ est symétrique définie positive, alors

$$cond_2(A) = \frac{\lambda_N}{\lambda_1}.$$

où $\lambda_N$ est la plus grande (resp petite) valeur propre de $A$.

*3.* si $A$ et $B$ sont deux matrices symétriques définies positives, alors :

$$cond_2(A+B) \leq \max(cond_2(A), cond_2(B)).$$

*4.* Soit $A \in \mathcal{M}_N(\mathbb{R})$ une matrice inversible, alors $cond_2(A) = 1$ si et seulement si $A = \alpha Q$ où $\alpha \in \mathbb{R}^*$ et $Q$ est une matrice orthogonale (c'est-à-dire : $Q^t = Q^{-1}$).

*5.* $A$ est toujours une matrice inversible, on suppose que : $A = QR$ où $Q$ est une matrice orthogonale ; alors $cond_2(A) = cond_2(R)$.

### 2.1.2 Conditionnement d'un système linéaire

Il est très connu que la plupart des problèmes numériques aboutissent à des résolutions des systèmes linéaires ; donc pour garder la précision de ces problèmes, un grand soin doit $y$ être apporté.

Le conditionnement d'un système linéaire $Ax = b$ traduite une grande difficulté à la résolution ; parce que :

$$1 \leq cond(A) \leq +\infty\,;$$

tel que :

• $cond(A) \sim 1$ ; c'est-à-dire : le système est bien conditionné (solution stable par rapport à une erreur des données ).

• $cond(A) \gg 1$ ; ce qui implique le système est mal conditionné (solution très sensible par rapport à une erreur des données ).

• $cond(A) = +\infty$ ; ceci indique que le système est singulier (matrice non inversible, $det(A) = 0$).

Le conditionnement réciproque (reciprocal condition numbre : $rcond$) est l'inverse. $rcond = 1/cond$. cette notion de l'inverse ; $rcond$ traduit en quelques cas de sorte que la proximité avec la singularité (lien avec le cas scalaire).

La stabilité se traduit par : $\left\| \frac{\Delta x}{x} \right\| \simeq cond(A) \left\| \frac{\Delta b}{b} \right\|$.

## 2.2 Stabilité numérique

Pour faciliter la définition de la stabilité numérique, on va la présenter en trois types :

*1. La stabilité d'un problème physique.*

*2. La stabilité d'un problème mathématique.*

*3. La stabilité numérique d'une méthode de calcul.*

**Stabilité d'un problème physique : système chaotique**

Un problème est dit chaotique si une petite variation des données initiales entraîne une variation totalement imprévisible des résultats. Cette notion de chaos, liée à la physique d'un problème, est indépendante du modèle mathématique utilisé et encore plus de la méthode numérique utilisée pour résoudre ce problème mathématique. La majorité des problèmes physiques sont chaotique ; à titre d'exemple, on peut donner : la turbulence des fluides.

**Stabilité d'un problème mathématique : sensibilité**

Un problème est dit très sensible ou mal conditionné si une petite variation des données ou des paramètres entraîne une grande variation des résultats. Cette notion de conditionnement, liée au problème mathématique, est indépendante de la méthode numérique utilisée pour le résoudre. Pour modéliser un problème physique qui n'est pas chaotique, on construira un modèle mathématique qui sera le mieux conditionné possible.

**Stabilité d'une méthode numérique**

Une méthode est dite instable si elle est sujette à une propagation importante des erreurs numérique de discrétisation et d'arrondi.

Un problème peut être bien conditionné alors que la méthode numérique choisie pour le résoudre est instable. Dans ce cas, il est nécessaire de changer la méthode numérique utilisée ; par contre si le problème de départ est mal conditionné, aucune méthode numérique ne pourra y remédier. Naturellement, Il faut trouver une formulation mathématique

différente du même problème.

Pour bien comprendre, on va faire une petite comparaison entre les deux notions : conditionnement et stabilité numérique.

D'une façons générale, il est très connu dans le domaine d'informatique que le système numérique de l'ordinateur est discret, c'est-à-dire : qu'il ne comporte qu'un nombre fini de chiffres comme on a cité précédemment ; il en d'écoule que tous les calculs sont entachées d'erreurs. Alors :

Le conditionnement mesure la dépendance de la solution d'un problème numérique par rapport aux données du problème ; ceci afin de contrôler la validité d'une solution calculée par rapport à ces données.

Une autre notion importante en pratique est celle de la stabilité numérique. On dira que le calcul ou l'algorithme est numériquement stable, si les erreurs introduites dans les étapes intermédiaires ont un effet négligeable sur le résultat. Si des petits changements sur les données entraine des petits changement sur les résultats ; si non, on dira que l'algorithme est numériquement instable.

## 2.3 Théorie des perturbations

Dans la théorie des perturbations, on mesure la sensibilité d'un problème par rapport à la perturbation des données, à titre d'exemple si on prend le problème de résolution d'un système linéaire ; $Ax = b$, où $A$ est $b$ sont données et $x$ est bien sur le résultat.

Pour commencer cette théorie, on désigne par $\delta A$ et $\delta b$ pour une perturbation des données et $\delta x$ pour une erreur (perturbation) sur le résultat. Il est connu que ce problème est bien posé si $x$ est unique et il est stable si : $\|\delta x\| / \|\delta A\|$ est borné. Dans ce cas là le conditionnement $C$ est définit par :

$$C = \lim_{\|\delta x\| \to 0} \sup \|\delta x\| / \|\delta A\| .$$

Donc, l'erreur relative sur le résultat est majorée par :

$$\frac{\|\delta x\|}{\|x\|} \leq C \frac{\|\delta A\|}{\|A\|}.$$

Dans ce qui suit, on essaye de mesurer la sensibilité de la solution $x$ si on perturbe le seconde membre $b$ d'une quantité $\delta b$, $A$ restant inchangée ; dans ce cas, on a :

$$\frac{\|\delta x\|}{\|x\|} \leq cond(A) \frac{\|\delta b\|}{\|b\|}.$$

En effet, evidemment si on perturbe les données le résultat $x$ sera perturbé ; c'est-à-dire : le système $Ax = b$ devient : $A(x + \delta x) = b + \delta b$.

Si on utilise les propriétés d'une norme, on trouve que :

$$\|b\| = \|Ax\| \leq \|A\| \, \|x\| \, .$$

Par une division simple, on trouve :

$$\frac{\|b\|}{\|A\|} \leq \|x\| \, ;$$

alors :

$$\frac{1}{\|x\|} \leq \frac{\|A\|}{\|b\|}.$$

D'un autre coté, on a :

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

Ce qui est équivalent à :

$$Ax + A\delta x = b + \delta b,$$

si en tenant compte que : $Ax - b = 0$, on obtient :

$$A\delta x = \delta b.$$

Puisque $A$ est inversible, on a :

$$\delta x = A^{-1}\delta b.$$

On utilise la norme et ses propriétés, on trouve immédiatement que :

$$\|\delta x\| \leq \left\|A^{-1}\right\| \|\delta b\|.$$

En multipliant les deux formules membre par membre, on obtient :

$$\frac{\|\delta x\|}{\|x\|} \leq \left\|A^{-1}\right\| \|A\| \frac{\|\delta b\|}{\|b\|}.$$

Alors :

$$\frac{\|\delta x\|}{\|x\|} \leq cond(A)\frac{\|\delta b\|}{\|b\|}.$$

Maintenant, on étudie le cas où $b$ reste inchangé, on perturbe $A$ d'une quantité $\delta A$, de sorte que :

$$1 - \left\|A^{-1}\right\| \|\delta A\| > 0.$$

On distingue deux formules importantes, la première est :

$$\frac{\|\delta x\|}{\|x + \delta x\|} \leq cond(A)\frac{\|\delta A\|}{\|A\|}.$$

De la même manière, on peut établir cette dernière relation ; à partir des données perturbées du système $Ax = b$. On a :

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

ce qui implique que :

$$Ax + A\delta x + \delta Ax + \delta A\delta x = b.$$

Puisque $Ax - b = 0$, on trouve :

$$A\delta x + \delta Ax + \delta A\delta x = 0.$$

Par des calculs simples, on obtient :

$$A\delta x = -A\delta x - \delta A\delta x,$$

et aussi

$$\delta x = -A^{-1}(\delta Ax + \delta A\delta x);$$

c'est-à-dire :

$$\delta x = -A^{-1}\delta A(x + \delta x).$$

On introuit la norme pour trouver :

$$\|\delta A\| \leq \left\|A^{-1}\right\| \|\delta A\| \|x + \delta x\|,$$

alors

$$\frac{\|\delta(x)\|}{\|x + \delta x\|} \leq \left\|A^{-1}\right\| \|\delta A\| \left(\frac{\|A\|}{\|A\|}\right);$$

finalement, on trouve :

$$\frac{\|\delta x\|}{\|x + \delta x\|} \leq cond(A)\frac{\|\delta A\|}{\|A\|}.$$

La deuxième formule importante lorsqu'on perturbe la matrice $A$ est :

$$\frac{\|\delta x\|}{\|x\|} \leq \frac{cond(A)}{1 - cond(A)\dfrac{\|\delta A\|}{\|A\|}} \times \frac{\|\delta A\|}{\|A\|}.$$

Pour démontrer cette formule, on suppose toujours que la matrice $A$ du système $Ax = b$ est perturbée ; c'est-à-dire :

$$(A + \delta A)(x + \delta x) = b;$$

ce qui implique que :

$$Ax + A\delta x + \delta Ax + \delta A\delta x = b,$$

et

$$A\delta x + A\delta x + \delta A\delta x = 0.$$

On a :

$$A\delta x = -\delta Ax + \delta A\delta x.$$

On multiplie les deux membres par $A^{-1}$ :

$$\delta x = -A^{-1}(\delta Ax + \delta A\delta x);$$

Si on utilise la norme on trouve :

$$\|\delta x\| \leq \left\|A^{-1}\right\| \|\delta Ax + \delta A\delta x\|,$$

et

$$\|\delta x\| \leq \left\|A^{-1}\right\| \|\delta A\| \|x\| + \left\|A^{-1}\right\| \|\delta A\| \|\delta x\|.$$

Utilisant les mêmes techniques, on obtient :

$$\|\delta x\| \left(1 - \left\|A^{-1}\right\| \|\delta A\|\right) \leq \left\|A^{-1}\right\| \|\delta A\| \|x\|.$$

Si on multiplie et on divise par $\|A\|$, on a :

$$\|\delta A\| \leq \frac{\|A^{-1}\| \, \|\delta A\| \, \|x\|}{1 - \|A^{-1}\| \, \|\delta A\|} \times \frac{(\frac{\|A\|}{\|A\|})}{(\frac{\|A\|}{\|A\|})},$$

c'est-à-dire :

$$\frac{\|\delta x\|}{\|x\|} \leq \frac{cond(A)\frac{\|\delta A\|}{\|A\|}}{1 - cond(A)\frac{\|\delta A\|}{\|A\|}};$$

et enfin, on a :

$$\frac{\|\delta x\|}{\|x\|} \leq \frac{cond(A)}{1 - cond(A)\frac{\|\delta A\|}{\|A\|}} \times \frac{\|\delta A\|}{\|A\|}.$$

# Chapitre 3

# Application

Dans ce chapitre, on essaye de donner une application pour démontrer qu'on peut réaliser un problème numérique d'une manière plus efficace et plus performante ; c'est-à-dire : il vérifie les critères de comparaison qu'on a étudié dans les chapitres précédents (complexité, stabilité, conditionnement et précision).

## 3.1 Système de Vandermonde

Il est connu que les matrices de Vandermonde sont mal conditionnées ; donc, lorsqu'on utilise une méthode directe de résolution d'un système linéaire, on trouve vraiment un problème numériquement instable. Dans cette partie, on présente une méthode précise pour résoudre ce type de systèmes.

Pour aboutir à notre but, on suppose le vecteur $x(0:n) \in \mathbb{R}^{n+1}$ et la matrice de Vandermonde $V \in \mathbb{R}^{(n+1)\times(n+1)} = \mathcal{M}_{n+1,n+1}(\mathbb{R})$ de la forme :

$$V = V(x_0, \ldots, x_n) = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ x_0 & x_1 & \cdots & x_n \\ \vdots & \vdots & & \vdots \\ x_0^n & x_1^n & \cdots & x_n^n \end{pmatrix}.$$

D'habitude, pour résoudre un système linéaire, on coûte $\mathcal{O}(n^3)$ opérations ; mais dans cette partie, on démontre qu'on peut résoudre le système de Vandermonde en $\mathcal{O}(n^2)$ seulement par l'utilisation des différences divisées.

Le système de Vandermonde est connu beaucoup plus dans les problèmes d'interpolation. Alors, l'idée clée ici est que la résolution d'un système de Vandermonde de la forme :

$$V^T a = f;\ \text{où } a(0:n) \text{ et } f(0:n) \in \mathbb{R}^{n+1}$$

est équivalant à un problème d'interpolation polynomial parce que le vecteur "$a$" qui est la solution de notre système formé par les coefficients du polynôme :

$$P(x) = \sum_{j=0}^{n} a_j x^j;$$

où $P(x_i) = f_i$ pour $i = 0 : n$. Dans le cas où les $x_i$ sont distincts, la matrice $V$ est régulière et le polynôme d'interpolation interpolant $(x_0, f_0), \ldots, (x_n, f_n)$ est unique.

## 3.2 Algorithme de résolution de $V^T a = f$

Sous les conditions précédentes, on essaye de présenter étape par étape cette méthode : Premièrement, on calcule $a_j$ par l'utilisation du polynôme d'interpolation de Newton basé sur les différences divisées :

$$P(x) = \sum_{k=0}^{n} c_k (\prod_{i=0}^{n} (x - x_i).$$

Les constants $c_k$ sont les différences divisées qu'on peut les déterminer par :

$c(0:n) = f(0:n)$

*Pour* $k = 0 : n-1$

  *Pour* $i = n-1 : k+1$

    $c_i = (c_i - c_{i-1})/(x_i - x_{i-k-1})$.

  *Fin pour*

*Fin pour.*

Si en se basant sur les différences divisées, on peut exposer maintenant l'algorithme suivant qui nous permet de résoudre le système de Vandermonde $V^T a = f$ en $\mathcal{O}(n^2)$ opérations seulement au lieu de $\mathcal{O}(n^3)$ et en plus, cet algorithme est numériquement stable et on remarque après qu'il est économique en mémoire de la machine *(on utilise un stockage linéaire).*

**Algorithme rapide de résolution du système de Vandermonde** $V^T a = f$

**Données :** *Les vecteurs* $x = x(0:n)$, $f = f(0:n) \in \mathbb{R}^{n+1}$ *et la matrice* $V(x_0, \ldots, x_n)$.

**Résultats :** *Le vecteur* $a = a(0:n)$.

*Pour* $k = 1 : n-1$

  *Pour* $i = n : -1 : k+1$

    $f(i) = (f(i) - f(i-1))/(x(i) - x(i-k-1))$

  *Fin pour*

*Fin pour*

*Pour* $k = n-1 : -1 : 0$

  *Pour* $i = k : n-1$

    $f(i) = f(i) - f(i+1)x(k)$.

  *Fin pour*

*Fin pour.*

$a(0:n) = f(0:n)$.

Pour faciliter la compréhension de cet algorithme, on donne un système de Vandermonde de la forme $V^T a = f$ de dimension $(4 \times 4)$ suivant :

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 8 \\ 1 & 3 & 9 & 27 \\ 1 & 4 & 16 & 64 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 26 \\ 58 \\ 112 \end{pmatrix};$$

C'est-à-dire :

$$f = (f_0, f_1, f_2, f_3) = (10, 26, 58, 112) \text{ et } \ x = (x_0, x_1, x_2, x_3) = (1, 2, 3, 4).$$

Lorsqu'on applique les différences divisées sur les points : $(x_0, f_0)$, $(x_1, f_1)$, $(x_2, f_2)$ et $(x_3, f_3)$ on trouve le polynôme de Newton suivant :

$$P(x) = 10 + 16(x-1) + 8(x-1)(x-2) + (x-1)(x-2)(x-3).$$

On trouve maintenant un nouveau vecteur $f = (f_0, f_1, f_2, f_3) = (10, 16, 8, 1)$; en tenant compte ce vecteur et on exécute la deuxième partie de l'algorithme étape par étape pour trouver enfin la solution du système $V^T a = f$; $a = (4, 3, 2, 1)^T$; de la manière suivante :

On a $n = 3 / x = (x_0, x_1, x_2, x_3) = (1, 2, 3, 4) / f = (f_0, f_1, f_2, f_3) = (10, 16, 8, 1)$;

$*$ *Pour* $k = n - 1 = 2$; $i = 2$

- *Pour* $i = 2$

  $f_2 = f_2 - f_3 x_2 = 8 - 1 \times 3 = 5$..

  $f = (f_0, f_1, f_2, f_3) = (10, 16, 5, 1)$.

$*$ *Pour* $k = n - 2 = 1$; $i = 1 : 2$

- *Pour* $i = 1$

  $f_1 = f_1 - f_2 x_1 = 16 - 5 \times 2 = 6$.

- *Pour* $i = 2$

  $f_2 = f_2 - f_3 x_1 = 5 - 1 \times 2 = 3.$

  $f = (f_0, f_1, f_2, f_3) = (10, 6, 3, 1).$

$*$ *Pour* $k = 0$; $i = 0 : 2$

- *Pour* $i = 0$

  $f_0 = f_0 - f_1 x_0 = 10 - 6 \times 1 = 4.$

- *Pour* $i = 1$

  $f_1 = f_1 - f_2 x_0 = 6 - 3 \times 1 = 3.$

- *Pour* $i = 2$

  $f_2 = f_2 - f_3 x_0 = 3 - 1 \times 1 = 2.$

  $f = (f_0, f_1, f_2, f_3) = (4, 3, 2, 1).$

Après l'exécution des deux boucles, le dernier $f$ est égale à "$a$" *(la solution du système* $V^T a = f$*)* ; donc : $f = (f_0, f_1, f_2, f_3)^T = (4, 3, 2, 1)^T = a.$

# Conclusion

A cause de l'accumulation des erreurs d'arrondi pendant l'exécution d'un algorithme, dans les études scientifiques, on cherche toujours à découvrir des méthodes employées les critères de comparaison les plus essentiels et qu'on a étudié dans notre mémoire *(complexité, conditionnement et stabilité numérique).*

Pour répondre à notre question, on a proposé un algorithme ; de résolution d'un système de Vandermonde ; non seulement rapide *(en $O(n^2)$ opérations seulement au lieu de $O(n^3)$)* mais, il est bien conditionné, numériquement stable grâce à l'utilisation des différences divisées et en plus, il utilise un stockage linéaire.

En conclusion, on peut créer des algorithmes ou des méthodes numériques qui vérifiées les critères de comparaison qui sont des critères d'efficacité.

# Bibliographie


[1] Charles Van Loan et Gene Golub. *Matrix computations, Third edition.*

[2] Conditionnement d'un système linéaire. *Cours d'internet.*

[3] Eric Goncalves. *Méthodes, analyse et calculs numériques, institut polytechnique de Grenoble, Septembre* 2005.

[4] Eric Trichet. *Introduction à la complexité algorithmique, Centre de ressources et d'innovation pédagogique de l'université de linoges.*

[5] Fellah M et Allal H. *Exercices corrigés en analyse numérique, Office des publications universitaires.*

[6] Introduction à l'arithmétique flottant. *Cours d'internet* (2018).

[7] Koepfler G. *Numération et logique* $(2014 - 2015)$.

[8] *Raphaèle Herbin* $(2006 - 2007)$. *Cours d'analyse numérique, université Aix Marseille* 1 *licence de mathématique.*

# Abréviations et Notations

Les différentes abréviations et notations utilisées tout au long de ce mémoire sont expliquées ci-dessous :

- $\mathbb{E}(x)$ : *Partie entière de* $x$.
- $F(x)$ : *Partie fractionnaire de* $x$
- $e(x)$ *: Exposant de* $x$.
- $b$ *:* Base du système.
- $m(x)$ *: Mantisse de* $x$.
- $E_t$ : Ensemble machine.
- $\delta x$ *: Perturbation sur* $x$.
- $\delta b$ *: Perturbation sur* $b$.
- $\delta A$ *: Perturbation sur* $A$.
- *cond* : *Conditionnement.*