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

Département de Mathématiques
Domaine Mathématiques et Informatique

Filière : Mathématiques

Spécialité : Analyse

Mémoire de fin d'étude en Master

Intitulé : Méthodes de Correction des Systèmes Linéaires

Présenté par :
Hamada Farida

Devant le jury

Encadreur :
Rajah Faouzia
Examinateur 1 :
Examinateur 2 :

Année Universitaire
2011-2012

Table des matières

Remerciements

Mes remerciements vont tout premièrement à mon Dieu le tout puissant pour la volonté, la santé et la patience qu'il m'a donné pour terminer mon travail de recherche.

Je souhaite tout d'abord remercier mon encadreur Madame F.Rajah. Maitre de Conférences à l'université Mohamed Khider (Biskra), de s'aide, sa disponibilité, ses orientations judicieuses et ses conseils, et pour avoir accepté de diriger ce mémoire et de sa patiente durant la période de l'encadrement.

Je remercie également Monsieur ..... à l'université Mohamed Khider (Biskra) pour m'avoir fait l'honneur de présider ce jury de mémoire de mastère.

Je voudrais aussi remercier tous les membres de jury, Monsieur.....Maitre de Conférences à l'université de Biskra, et Monsieur...Maitre de Conférences à l'université de Biskra.

Je remercier toute ma famille, mes parents qui ont toujours été présents pour me soutenir et à qui je dois d'avoir pu réaliser de si longues années d'études, ma mère qui m'a toujours encouragé à aller plus loin.

Je tiens également à remercier mes plus proches amis pour leur soutien, et toutes les personnes qui ont contribué directement ou indirectement à ce travail.

Et enfin, je m'excuse à tout les personnes qui j'oublier leurs noms et qui n'ont pas été citées.

Résumés

Résumé

A cause de la propagation des erreurs d'arrondi, effectuées pendant l'exécution d'un algorithme ou d'une méthode numérique, on propose dans ce travail deux méthodes de correction concernant les systèmes linéaires de Toeplitz et de Vandermonde confluent. Cette amélioration est basée sur le fait que ces matrices sont factorisable en terme de matrices Toeplitz triangulaires et matrices circulantes qu'on peut les multipliées par un vecteur grâce à la FFT avec un temps de calcul plus réduit.

Mots-clés : *Matrice Toepliz, Matrice de Vandermonde confluente, Analyse d'erreur.*

Abstract

Because of the spread of rounding errors during the execution of an algorithm or a numerical method, we proposed in This work two correction methods for linear systems of Toeplitz and confluent Vandermonde. These methods are based on the idea that these matrices are factors in terms of triangular Toeplitz and circulating matrices that can be multiplied by a vector with a reduced calculation time by caling of FFT.

Key Words : Toeplitz matrix, *Confluent Vandermonde matrix, Error analysis.*

Chapitre 1

Introduction générale

L'analyse numérique se propose d'étudier les propriétés mathématiques des algorithmes (des méthodes) et leur mise en œuvre (programmation), et son objectif est de concevoir et d'étudier des moyens pour donner des solutions approchées à des problèmes mathématiques dont la résolution explicite est généralement impossible.

Les solutions approchées sont les plus souvent calculées sur ordinateur au moyen d'un algorithme convenable, qui peut être direct ou itératif, avec une précision finie. Idéalement, elles devraient s'accompagner d'une majoration de l'écart avec les solutions réelles.

L'analyse numérique à aussi comme but de réaliser un problème mathématique donné par un algorithme qui est :

- Plus vite (complexité des algorithmes, complexité des problèmes) : un algorithme est d'autant plus rapide, donc plus performant, que le nombre d'opérations utilisées est réduit.
- Plus précis : c'est-à-dire l'erreur d'arrondi (liées à la machine) et l'erreur d'approximation (liées à l'algorithme) sont plus petit.
- Plus fiable (stabilité numérique) : cette propriété s'intéresse à l'aspect numérique des algorithmes. Dans ce cas, la performance d'un algorithme numérique est analysée à partir de la robustesse face à la propagation des erreurs d'arrondi provoquées par des réalisations approximatives des opérations élémentaires.

Alors, il est très rare que le traitement numérique d'un problème ; par exemple la résolution de $Ax = b$; ne donne une solution non entachée d'erreurs. Une partie importante de l'analyse numérique consiste à contenir les effets des erreurs ainsi introduites. Ces erreurs ont diverses origines qui vont des incertitudes sur les données à la représentation de ces données et proviennent de trois source essentielles :

- Erreurs de modélisation.
- Erreurs de représentation sur ordinateur.
- Erreurs de troncature.

En effet, mises à part les erreurs d'acquisition des données réelles du problème, la représentation d'un nombre réel n'utilise qu'un nombre limité de chiffres significatifs. On dit qu'on a une arithmétique à précision limitée et l'erreur commise est dite erreur d'arrondi. La prise en compte de toutes ces altérations dans le traitement de la recherche de la solution du problème $Ax = b$ nous amène en réalité à obtenir la solution du problème perturbé $(A + \triangle A)x = (b + \triangle b)$.

En général les perturbations $\triangle A$ et $\triangle b$ ne sont connues que par leurs majorations. Et comme pour un algorithme particulier, l'analyse de l'accumulation d'erreurs d'arrondi commises à chaque étape permet de majorer l'erreur finale.

Dans ce mémoire, on propose deux méthodes de corrections ou d'amélioration concernant les systèmes Toeplitz et Vandermonde confluent. Les méthodes que nous avons proposé, en plus de leur rapidité et leur stockage linéaire, elle sont numériquement stable.

Notre étude de sujet est logiquement composée des chapitres suivants :

En deuxième chapitre, on présente les différentes représentations des nombres réels ; la représentation en virgule fixe et la représentation en virgule flottante (représentation infinie), et puisque la capacité mémoire d'un ordinateur est par construction finie, quelques soient les moyens de calcule informatiques mis en œuvre, il est donc nécessaire de représenter les nombres réels d'une forme approchée et les écrivent sous la représentation machine ou la représentation finie.

Ensuite, on introduit les deux types d'erreurs ; l'erreur absolue et l'erreur relative ; pour estimer la propagation des erreurs d'arrondi commisent pendant l'exécution d'un algorithme ou d'une méthode numérique. Si les erreurs introduites dans les étapes intermédiaires ont un effet négligeable sur le résultat final, on dira que le calcul ou l'algorithme est numériquement stable ; si des petits changements sur les données entrainent des petits changements sur les résultats ; sinon on dira que l'algorithme est numériquement instable.

L'objectif de l'analyse des erreurs est de majorer l'erreur effectuée lors de l'évaluation basée sur le modèle standard.

On commence le troisième chapitre par l'arrondi pair et ses propriétés pour approchée les nombres réels de sorte que la valeur obtenue par cet arrondi est la plus proche de la valeur exacte. On définit aussi l'ensemble $\mathcal{F}$(l'ensemble des nombres flottants) et le modèle standard. Ce modèle spécifie la précision des quatre opérations arithmétiques de base (addition, soustraction, ...) et on remarque que ces opérations ne vérifient pas plusieurs propriétés (associativité, distributivité, ...) qui sont évidentes en arithmétique classique.

Puisque les algorithmes de calcul sont caractérisés par leur complexité arithmétique ; mesurée par le nombre d'opérations (additions, multiplications, ...etc.) ; et leur coût de stockage ; mesuré par le nombre de variables réelles ; on applique la FFT (Fast Fourier Transform) sur les systèmes Toeplitz et Vandermonde confluent afin d'améliorer les résultats (améliorer la précision de calcul). Cette amélioration est obtenue lorsqu'on factorise ces matrices en termes de matrices Toeplitz triangulaires et matrices circulantes ; tous cela on va se focaliser dans le dernier chapitre.

Chapitre 2

Représentations des nombres

2.1 Introduction

Le système numérique d'un calculateur quelconque est discret. C'est-à-dire ne comporte qu'un nombre fini de nombres. Il en découle que ; sauf dans les cas les plus simples ; tous les calculs seront entachés d'erreurs. Donc, il est naturel d'exposer dans le premier chapitre la représentation des nombres dans un ordinateur et incidemment sur les systèmes de numération.

2.2 Codage et système de numération

Nombres et codes

Nous manipulons tous les jours des nombres et des codes, par exemple le numéro de téléphone, le code secret d'une carte bancaire est fait de chiffres mais ce n'est pas un nombre, c'est un code.

Dans ces exemples, les chiffres sont à la base de l'expression des nombres ou des codes. Nous acceptons assez naturellement l'usage des chiffres pour exprimer des choses de nature.

Les chiffres utilisés sont ceux de la bas dix (0, 1, 2, ..., 9).

Dans les ordinateur, et plus particulièrement en électronique numérique, il se passe la même chose ; on utilise les chiffres pour représenter des nombres et des codes ; la seule différence est qu'il s'agit des chiffres de la base deux (0,1).

Qu'est-ce qu'un code ?

Le code est fabriqué en respectant une technique particulière, le nombre des chiffres est constant et ordonné.

On peut dire que le code est une représentation signifiant quelque chose (un code est la représentation d'une signification).

Qu'est-ce qu'un nombre ?

Un nombre est la représentation d'une valeur, cette valeur est représentée au moyen des chiffres écrits de façon ordonnée, la différence par rapport au code est que chaque chiffre composant le nombre est porteur d'une valeur.

2.3 Représentation des nombres réels en virgule flottante

2.3.1 Nombres en virgule fixe

Le nombre en virgule fixe possède une partie entière et une partie décimale sépares par une virgule, la position de la virgule est fixé d'où de nom. Le nombre en virgule fixe s'écrit :

$$\begin{aligned} x &= \pm \left(d_{n-1}...d_1 d_0 . d_{-1} d_{-2} .... d_{-m}\right)_\beta \\ &= \pm \left(\sum_{i=0}^{n-1} d_i \beta^i + \sum_{i=1}^{m} d_{-i} \beta^{-i}\right) \end{aligned} \tag{2.1}$$

où m est fixé et $0 \leq d_i < \beta$.

2.3.2 Nombres en virgule flottante

les nombres en virgule flottante ; souvent appelés nombres flottantes ou flottantes ; sont une représentation d'un sous-ensemble finie des nombres réels.

On note par $\mathcal{F}$ l'ensemble des nombres flottants.

Les nombres flottants sont utilisés dans les programmes de l'informatique pour approximer des valeurs de type réel. Dans un ordinateur, on est obligé de restreindre le nombre de chiffres pour les mantisses et on définit un ensemble fini de nombres $\mathcal{F}\left(\beta, p, e_{\min}, e_{\max}\right)$, où :

β est l'entier ($\beta \geq 2$) définissant la base ;
p est le nombre de chiffres de la mantisse ;
$e_{\min}$ est l'exposant minimum et $e_{\max}$ est l'exposant maximum.

Cet ensemble correspond à tous les nombres réels x qui s'écrivent :

$$x = s\left(\sum_{i=0}^{p-1} c_i \beta^{-i}\right)\beta^e \tag{2.2}$$

où $s = \pm 1$: le signe, $\forall i : 0 \leq c_i \ \leq \beta - 1$ et $e_{\min} \leq e \leq e_{\max}$

On dit que la représentation est normalisée quand la mantisse est de la forme :

$$c_0.c_1c_2c_3 \qquad \text{avec } c_0 \neq 0. \tag{2.3}$$

Une autre représentation souvent rencontrée est

$$0.c_1c_2c_3 \qquad \text{avec } c_1 \neq 0$$

Pour représenter 0, on utilise une écriture spéciale en ne mettant que des 0 dans la mantisse (ce qui est logique) et un exposant de $e_{\min} - 1$.

Remarques (2.1) :

1. Tout nombre réel (sauf zéro) peut s'écrire avec cette notation ($\pm c_0.c_1c_2c_3 \ldots \times \beta^e$; $c_0 \neq 0$) ; en virgule flottante normalisée avec en général un nombre de chiffres infini pour la mantisse. On peut aussi remarquer que certains nombres peuvent s'écrire de deux façons, par exemple $0.999... = 1$.

2. Dans les intervalles $\left[\beta^e, \beta^{e+1}\right]$ et $\left[-\beta^e, -\beta^{e+1}\right]$ l'incrément entre deux nombres flottants est constant et égal à β^{1-p+e}.

3. On notera M le plus grand nombre positif de $\mathcal{F}(\beta, p, e_{\min}, e_{\max})$

$$M = \left(\sum_{i=0}^{p-1} (\beta - 1)\,\beta^{-i}\right)\beta^{e_{\max}} = \left(1 - \beta^{-p}\right)\beta^{e_{\max}+1} \tag{2.4}$$

λ le plus petit nombre normalisé positif :

$$\lambda = (1, 00....0)\,\beta^{e_{\min}} = \beta^{e_{\min}} \tag{2.5}$$

$\lambda \leq |x| \leq M$ et μ le plus petit nombre dénormalisé positif :

$$\mu = (0, 00.....1)\,\beta^{1-p+e_{\min}} \tag{2.6}$$

4. Dans les systèmes flottants actuels, on rajoute des nombres spéciaux comme $+inf$, -inf (inf comme infini) et NaN (pour Not a Number) qui ont une représentation spéciale (utilisant en particulier un exposant de $e_{\max}$ $+1$). En fin, on notera que, du fait du bit de signe, le zéro a deux représentation qui conduisent à des résultats différents sur quelques calculs (par exemple $1\oslash + 0$ donnera +inf alors que $1\oslash - 0$ donnera -inf).

5. Soit $z \in \mathbb{R}$ le résultat exact d'une opération arithmétique entre deux flottants x et y ; il y a overflow si $|z| > M$ et underflow si $|z| < \lambda$. 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.

6. 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 ayant comme résultat NaN. L'opération $\frac{1}{0}$ déclenche un overflow et l'opération $\frac{1}{\infty}$ déclenche un underflow avec comme résultat 0.

2.4 Calcul d'erreur

Il est évident qu'un algorithme numérique est implémenté dans le but d'obtenir des résultats qui sont des nombres réels. Hélas, à cause de nombreuses limitations naturelles, l'obtention des valeurs exactes de ces résultats relève de l'impossible ; et ce malgré les travaux tentés ça et là notamment dans le cadre du calcul formel. Entre autres causes, citons en particulier le fait que la capacité mémoire de l'ordinateur est finie ; ce qui contraint le compilateur, dans implémentation, d'arrondir après chaque réalisation d'une opération élémentaire. Cette propagation des erreurs d'arrondi donne naturellement lieu à une valeur finale approchée (ou calculée) du résultat final désiré.

Ainsi, au lieu d'une valeur exacte $x \in \mathbb{R}$ d'un résultat, on n'obtient à partir de la machine qu'une valeur approchée $\hat{x} \in \mathcal{F}(\beta, p, e_{\min}, e_{\max})$ (on dit également valeur calculée). De toute évidence, il est intéressant d'avoir un moyen efficace permettant de mesurer l'erreur commise.

2.4.1 Erreur absolue

L'erreur absolue err_{abs} est la différence entre la valeur réelle et son approximation sur les flottants, on utilise généralement l'erreur absolue quand on connait une majoration a priori des valeurs intermédiaires calculées, soit $\hat{x}$ une approximation d'un nombre réel x non nul, l'erreur absolue est :

$$err_{abs} : E_a(x) = |\hat{x} - x| \tag{2.7}$$

2.4.2 Erreur relative

L'erreur relative err_{rel} est le rapport entre l'erreur absolue et la valeur réelle. On utilise l'erreur relative quand on ne connaît pas de borne a priori sur les valeurs intermédiaires calculées :

$$err_{rel} : E_{r_1}(x) = \frac{|\hat{x} - x|}{|x|} \tag{2.8}$$

Dans le cas où x et $\hat{x}$ sont deux vecteurs de même dimension, les notions d'erreur absolue et d'erreur relative s'étendent à l'aide d'une norme adéquate $\|.\|$et deviennent respectivement :

$$E_a(x) = \|\hat{x} - x\|$$

et

$$E_{r_1}(x) = \frac{\|\hat{x} - x\|}{\|x\|}$$

Dans ce contexte, il est bien connu qu'en comparant l'erreur absolue $|x - \hat{x}|$ à l'erreur relative $\dfrac{|x - \hat{x}|}{|x|}$, on est dans une situation plus confortable avec l'erreur relative qu'avec l'erreur absolue. Il est également connu que ceci est dû à l'invariance de l'erreur relative par rapport aux changements de l'échelle. Ainsi, on a du mal à accepter l'approximation.

$$1234 = 234$$

Alors, qu'on admet volontiers l'écriture

$$10000001000 \simeq 10000000000$$

Malgré le fait que l'erreur absolue, étant égale à 1000, soit la même dans les deux cas. Notre attitude vis à vis de ces deux approximations est néanmoins très bien justifiée en introduisant l'erreur relative. En effet, on observe dans la première approximation que l'erreur relative est

$$\frac{1000}{1234} \simeq 0.81$$

alors que dans la seconde, l'erreur relative est

$$\frac{1000}{10000001000} \simeq 10^{-7}$$

La différence est sans appel ! Pour avoir un formalisme général de l'erreur, on parle de l'erreur

$$|x - \hat{x}| \quad \text{relativement à} \quad |y| > 0$$

qui est tout simplement

$$\frac{|x - \hat{x}|}{|y|}.$$

En ce qui concerne notre étude où la valeur approchée $\hat{x}$ de x est telle que

$$\hat{x} \in \mathcal{F}\,(\beta, p, e_{\min}, e_{\max})$$

On distingue trois types d'erreurs relatives

(1) L'erreur $|x - \hat{x}|$ relativement à $|x|$. Posons $E_{r_1} = \dfrac{|x - \hat{x}|}{|x|}$ $(x \neq 0)$

(2) L'erreur $|x - \hat{x}|$ relativement à $|\hat{x}|$. Posons $E_{r_2} = \dfrac{|x - \hat{x}|}{|\hat{x}|}$ $(\hat{x} \neq 0)$

(3) L'erreur $|x - \hat{x}|$ relativement à $\beta^{e(x)}$, Posons $E_{r_3} = \dfrac{|x - \hat{x}|}{\beta^{e(x)}}$.

Dans les résultats suivants, on annonce les relations entre les trois types d'erreurs relatives :

Théorème (2.1) : Supposons que $E_{r_1} < 1$ *(en général* $<< 1$*). Alors :*

$$\frac{E_{r_1}}{1 + E_{r_1}} \leq E_{r_2} \leq \frac{E_{r_1}}{1 - E_{r_1}}. \quad \Diamond \tag{2.9}$$

Démonstration : Pour démontrer cette formule, on observe que :

$$\begin{aligned} |x - \hat{x}| &= |x|\, E_{r_1} \\ &\leq |x - \hat{x}|\, E_{r_1} + |\hat{x}|\, E_{r_1} \end{aligned}$$

Comme $E_{r_1} < 1$, on obtient :

$$E_{r_2} \leq \frac{E_{r_1}}{1 - E_{r_1}}$$

De même

$$\begin{aligned} |x - \hat{x}| &= |\hat{x}|\, E_{r_2} \\ &\quad |x - \hat{x}|\, E_{r_2} + |x|\, E_{r_2} \end{aligned}$$

Par suite

$$E_{r_2} \geq \frac{|x - \hat{x}|}{|x - \hat{x}| + |x|} = \frac{E_{r_1}}{1 + E_{r_1}}. \quad \blacklozenge$$

Théorème (2.2) : Supposons que $E_{r_2} < 1$ *(en général* $<< 1$*). Alors :*

$$\frac{E_{r_2}}{1 + E_{r_2}} \leq E_{r_1} \leq \frac{E_{r_2}}{1 - E_{r_2}}. \quad \Diamond \tag{2.10}$$

Démonstration : Cette formule peut être démontrer d'une façon similaire à celle précédente. $\blacklozenge$

Théorème (2.3) : On a sous l'hypothèse : $e(x) = e(\hat{x})$,

$$E_{r_3} = |m(x) - m(\hat{x})| \tag{2.11}$$

et

$$\begin{aligned} \beta^{-1} E_{r_3} &\leq E_{r_1} \leq E_{r_3} \\ \beta^{-1} E_{r_3} &\leq E_{r_2} \leq E_{r_3}. \quad \Diamond \end{aligned} \tag{2.12}$$

Démonstration : Ce théorème découle du fait que :

$$1 \leq m(x) \leq \beta \text{ et } 1 \leq m(\hat{x}) \leq \beta$$

puisque $E_{r_3} = m(x)E_{r_1} = m(\hat{x})E_{r_2}$, les inégalités s'en déduisent aussitôt. $\blacklozenge$

Ces théorèmes dites que dans les conditions normales, E_{r_1}, E_{r_2} et E_{r_3} sont équivalentes au sens que l'utilisation de l'une donne des résultats équivalentes à ceux obtenus en utilisant l'autre.

2.4.3 Erreur en ulp

L'ulp (unit in the last place) est la distance qui sépare deux flottants consécutifs et donc, l'erreur maximale qui peut être faite lors d'un arrondi, et le poids du plus petit bit significatif de la mantisse d'un flottant. Plus formellement, la valeur de l'ulp pour un nombre flottant x est :

$$ulp\,(x) = \beta^e \times \beta^{-p+1} = \beta^{e-p+1}$$

où e est l'exposant de x, β sa base de calcul et p désigne le nombre de chiffres de la mantisse.

Si $x = \pm 1.c_{-1}c_{-2}.....c_{1-p} \times 2^e$, alors $ulp(x) = 2^{e+1-p}$, si la définition de l'exposant n'est pas intrinsèque et dépend du choix de position de la virgule, celle d'ulp est intrinsèque. On peut étendre cette définition à un réel x quelconque, en disant que ulp(x) est l'ulp du flottant le plus proche de x en direction de zéro, ou encore $ulp(x) = 2^{[\log_2|x|]+1-p}$.

Définition (2.1) : On appelle chiffres significatifs d'un nombre tous les chiffres de son écriture à partir du premier chiffre différent de zéro à gauche.

Exemple : Les chiffres significatifs des nombres $\hat{x} = 0.03045$ et $\hat{x} = 0.03045000$ sont ceux soulignés. Ils sont 4 dans le premier cas et 7 dans le deuxième.

Définition (2.2) : Un chiffre significatif d'une valeur $\hat{x}$ est exact si l'erreur absolue $E_a(x)$ sur cette valeur est inférieure ou égal à$\frac{1}{2}$ *fois l'unité du rang du chiffre.*

Exemple : $x = 3.2189 \pm 0.0003$ le $'8'$ est-il un cse ?

Rang du $8 = -3$, unité du rang du $8 = 0.001$, $\frac{1}{2}$ fois cette unité$= 0.0005$; $\triangle x = 0.0003 \leq 0.0005$. Le 8 est un cse et c'est le dernier x a donc 4 cse

$x = 3.2189 \pm 0.0003$

Conséquences :

La n^{ième}décimale d'une valeur est exact $\Leftrightarrow E_a(x) \leq 0,5 \times 10^{-n}$

La n^{ième} chiffre devant le point est exact $\Leftrightarrow E_a(x) \leq 0,5 \times 10^{n-1}$

Définition (2.3) : Si tous les chiffres significatifs sont exacts, on dit que le nombre est écrit avec tous les chiffres exacts.

La précision :

Il est important de distinguer la précision d'un résultat calculé de la précision de calcul, la précision d'un résultat fait référence à l'erreur relative entachant une quantité calculée, approchant un certain résultat exact, et résultant généralement d'un algorithme de calcul.

La précision de calcul, que nous appellerons également "précision de travail" ou encore précision courante, désigne l'erreur relative commise lors de chaque opération arithmétique élémentaire $+$, $-$, $\times$ ou $/$.

Remarque (2.2) : En arithmétique flottant, la précision de chaque opération arithmétique est majorée par l'unité d'arrondi notée $u = 5 \times 10^{-p}$.

2.5 Stabilité numérique

Les méthodes numériques utilisées pour résoudre un problème approché conduisent à un résultat qui toujours entaché d'erreur. Cette erreur doit être suffisamment petite pour que la solution numérique converge vers la solution réelle. Dans ce cas l'algorithme (ou la méthode) est dit convergent. Si un raisonnement mathématique permet de montrer qu'une méthode diverge, elle ne pourra en aucun cas être utilisée sur un calculateur. En revanche, si la méthode converge il se peut qu'on pratique elle diverge.

La vitesse de convergence est un facteur important de la qualité des algorithmes. Si la vitesse de convergence est élevée, l'algorithme converge rapidement et le temps de calcul est moindre. Ces préoccupations de rapidité de convergence ont conduit à diversifier les modes de convergence et à chercher les processus optimaux.

La stabilité garantit que les erreurs ne s'amplifient pas au cours de déroulement de l'algorithme et que la méthode reste stable. On dira que le calcul ou l'algorithme est numériquement stable si de petits changements dans les données entraînent de petits changements dans les résultats. Evidemment, dans le cas contraire on dira qu'il y a instabilité numérique.

Exemple d'instabilité numérique : On veut calculer

$$f(n) = \int_0^1 \frac{x^n}{a+x} dx \qquad a = cte > 1$$

Nous allons exprimer $f(n)$ récursivement :

$$\begin{aligned} f(n) &= \int_0^1 \frac{x^{n-1}(x+a-a)}{a+x}dx = \int_0^1 x^{n-1}dx - a\int_0^1 \frac{x^{n-1}}{a+x}dx \\ &= \tfrac{1}{n} - a\ f(n-1) \qquad n \geq 1; \end{aligned}$$

$f(0) = \ln(\dfrac{1+a}{a})$

L'algorithme fourni par cette relation est numériquement instable. Voici les résultats obtenus pour $a = 10$ et $n = 0, 1, \ldots, 12$

n	$f(n)$ *calculé*	$f(n)$ *exact*
0	0.0953102	0.0953102
1	0.0468982	0.0468982
2	0.0310180	0.0310180
3	0.0261535	0.0231535
4	0.0184647	0.0184647
5	0.0153527	0.0153529
6	0.0131401	0.0131377
7	0.0114558	0.0114806
8	0.0104421	0.0101944
9	0.0066903	0.0091672
10	0.0330968	0.00832797
11	0.2400592	0.00762944
12	2.4839249	0.00703898

à partir de $n = 5$, les valeurs calculées sont de moins en moins précises à chaque itération ; pour $n \geq 10$ les résultats obtenus sont complètement erronés. Cet algorithme est d'autant plus instable que a est plus grand que 1.

Pour ce faire supposons que l'erreur d'arrondi sur $f(0)$ est égale à ε_0 et qu'aucune erreur n'est introduite dans les calculs subséquents. Notons $\widetilde{f}(n)$ les valeurs calculées.

$$\begin{aligned} \widetilde{f}(0) &= f(0) + \varepsilon_0 \\ \widetilde{f}(n) &= \frac{1}{n} - a\widetilde{f}(n-1) \qquad n = 1, 2, \ldots \end{aligned}$$

Par suite, si r_n désigne l'erreur sur $f(n)$

$$\begin{aligned} r_n &= \widetilde{f}(n) - f(n) = -a\widetilde{f}(n-1) + \frac{1}{n} - \frac{1}{n} + a\ f(n-1) \\ &= -a(\widetilde{f}(n-1) - f(n-1)) \\ &= -ar_{n-1} \qquad n = 1, 2, \ldots \end{aligned}$$

et donc, puisque $r_0 = \varepsilon_0$, nous trouvons $r_n = (-a)^n \varepsilon_0 \qquad n = 1, 2, \ldots$.L'erreur initiale est multipliée par un facteur a à chaque itération.

2.6 Analyse d'erreur

En pratique, le problème de calcul de $f(x)$ sera souvent résolu à l'aide d'un algorithme numérique, effectuant en précision finie, et dans certain ordre, les opérations définissant f. En raison des erreurs commises lors des calculs intermédiaires en précision finie, et des erreurs de données, cet algorithme numérique ne réalise pas la fonction f, mais une fonction $\widehat{f}$: l'algorithme calcule ainsi une approximation $\widehat{y} = \widehat{f}(x)$ de réel $y = f(x)$.

Le but de l'analyse d'erreur directe est de majorer, ou pus rarement d'estimer, la distance séparant le résultat calculé du résultat exacte, qui sera par conséquent appelé erreur directe. Ce type de majoration peut être obtenu en propageant les erreurs générées par chaque opération effectuée par l'algorithme numérique étudié. L'erreur directe peut être majorée de façon relative ou absolue, selon que l'on considère $E_a(\widehat{y})$ ou $E_r(\widehat{y})$.

On retiendra donc que l'analyse d'erreur directe apporte des éléments de réponse à la question : quelle est la précision du résultat calculé par l'algorithme numérique considéré ?. De plus, il est important de noter que l'analyse d'erreur directe ne permet pas de différencier l'influence du problème de celle de l'algorithme, quant à la précision du résultat calculé.

Plutôt que de chercher à majorer l'erreur directe entachant

$$\widehat{y} = \widehat{f}(x),$$

On peut dans un premier temps tenter de réponse à la question : pour quelles données le problème a t'il effectivement été résolu ?. Il s'agit précisément du but de l'analyse d'erreur inverse, dans laquelle on cherche à identifier le résultat calculé $\widehat{y}$ à l'évaluation exacte de f en une donnée perturbée :

$$x + \delta x,$$

de manière a ce que l'on ait

$$\widehat{y} = \widehat{f}(x + \delta x)$$

tout en bornant la perturbation δx. La notion d'erreur inverse permet de mettre en évidence la stabilité, ou au contraire l'instabilité d'un algorithme.

Chapitre 3

Opérations flottantes et ses propriétés

3.1 Introduction

Dans ce chapitre, on présente quelques notions de base pour analyser l'erreur de certains problèmes numériques. On commence par l'arrondi pair et ses propriétés, après, on donne le modèle standard et les opérations flottantes et on démontre par quelques exemples numériques que ces opérations ne vérifient pas plusieurs propriétés qui sont évidentes dans l'arithmétique classique.

3.2 Arrondi pair

L'ensemble des nombres réels $\mathbb{R}$ est classiquement approché par l'ensemble des flottants $\mathcal{F}$. La correspondance entre un nombre réel et sa représentation par un flottant est définie par une application appelée arrondi.

Définition (3.1) : Soit $fl : \mathbb{R} \longrightarrow \mathcal{F}$ une fonction, on dit que fl est un arrondi de $\mathbb{R}$ vers $\mathcal{F}$ si les deux propriétés suivantes sont vérifiées :

-pour tout $x \in \mathcal{F}$, $fl(x) = x$ (projections)

-pour tout x, $y \in \mathbb{R}$ tels que $x \leq y$, $fl(x) \leq fl(y)$ (monotonie).

Remarque (3.1) : On appelle l'arrondi fl, l'arrondi pair (la mantisse est pair) ou l'arrondi au plus proche.

Définition (3.2) : Le nombre $\varepsilon = \beta^{1-p}$ est appelé l'epsilon du système.

Remarque (3.2) : On remarque donc que l'epsilon du système est indépendant de $e_{\min}$ et de $e_{\max}$.

Proposition (3.1) : tout réel x est encadré par deux flottants consécutifs. Soit :

$$x^+ = \min\{y \in \mathcal{F}, y \geq x\} \text{ et } x^- = \max\{y \in \mathcal{F}, y \leq x\}. \quad \Diamond \tag{3.1}$$

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

Preuve : Si

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

soit

$$x \in \mathbb{R},\ x \notin \mathcal{F}, x > \lambda,$$

soit

$$x^- = \beta^e m \text{ alors } m \geq 1, x^- < x \text{ et } \ x^+ = \beta^e(m+\varepsilon) = x^- + \beta^e\varepsilon.$$

Donc

$$0 \leq x^+ - x^- = \beta^e\varepsilon \leq \beta^e m\ \varepsilon \leq x\ \varepsilon$$

Le cas où $x < \lambda$ se traite de même façon. ⧫

Proposition (3.2) : Tout arrondi fl vérifie :

$$\forall x \in \mathbb{R}, fl(x) = x^+ \text{ ou } \ fl(x) = x^-. \quad \Diamond \tag{3.2}$$

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 égale à l'un des deux. ⧫

Définition (3.3) : la fonction $fl(x)$ conserve le signe, c'est -à-dire :

$$fl(-x) = -fl(x). \tag{3.3}$$

Si $|x| > \sigma$ ou si $|x| < \lambda$, alors $fl(x) = NaN$.

Lemme (3.3) (Erreur de représentation) : Soit $x \in \mathbb{R}$ tel que $\lambda \leq |x| \leq \sigma$: L'arrondi fl vérifie :

$$fl(x) = x\ (1+\alpha), o\grave{u}\ \ |\alpha| \leq u \tag{3.4}$$

où est *l'unité d'erreur d'arrondi : $u = \varepsilon/2 = \frac{\beta^{1-p}}{2}$.* $\Diamond$

Preuve : Il est clair que :

$$\begin{aligned}
|x - fl(x)| &\leq \min\{|x - x^-|, |x - x^+|\} \\
&\leq \frac{|x^+ - x^-|}{2} = \frac{\beta}{2}\beta^{-p}.\beta^e \\
&\leq \frac{\beta^{1-p}}{2}.|m(x)|.\beta^e \\
&\Longleftrightarrow |x - fl(x)| \leq \frac{\beta^{1-p}}{2}.|x| \\
&\Longleftrightarrow \left|\frac{x - fl(x)}{x}\right| \leq \frac{\varepsilon}{2} = u
\end{aligned}$$

par conséquent :

$$\frac{x - fl(x)}{x} \leq u$$

si on pose :

$$\frac{x - fl(x)}{x} = \alpha \in \mathbb{R}(|\alpha| \leq u)$$

alors :

$$fl(x) = x(1+\alpha);\ \ |\alpha| \leq u. \quad \blacklozenge$$

Lemme (3.4) : Sous les mêmes hypothèses du lemme précédent, on a :

$$fl(x) = x\ /(1+\alpha), \text{ où } |\alpha| \leq u. \quad \Diamond \tag{3.5}$$

Preuve : Pour démontrer ce lemme, il suffit de démontrer que

$$\frac{x - fl(x)}{fl(x)} \leq u$$

On a toujours :

$$\begin{aligned} |x - fl(x)| &\leq \min\{|x - x^-|\,, |x - x^+|\} \\ &\leq \frac{|x^+ - x^-|}{2} = \frac{\beta^{1-p}}{2}.\beta^e \end{aligned}$$

puisque

$$fl(x) = m^+(x) \text{ où } m^-(x)$$

alors :

$$\begin{aligned} |x - fl(x)| &\leq \frac{\beta^{1-p}}{2}.|fl(x)| \\ &\Leftrightarrow \frac{|x - fl(x)|}{|fl(x)|} \leq \frac{\beta^{1-p}}{2} \\ &\Leftrightarrow \left|\frac{x - fl(x)}{fl(x)}\right| \leq u \\ &\Leftrightarrow \frac{x - fl(x)}{fl(x)} \leq u \end{aligned}$$

Si on pose

$$\alpha = \frac{x - fl(x)}{fl(x)}$$

On trouve que

$$\alpha fl(x) + fl(x) = x \Leftrightarrow fl(x)(1+\alpha) = x$$

Donc

$$fl(x) = \frac{x}{1+\alpha} \text{ où } |\alpha| \leq u. \quad \blacklozenge$$

Remarques (3.3) : *1.* La définition de l'arrondi ($fl(x) = x$) permet d'écrire $\alpha = 0$ si $x \in \mathcal{F}$.

2. Tout calcul en arithmétique flottant est susceptible de la propagation des erreurs d'arrondi.

Approximation d'un nombre réel par un nombre flottant

Etant donné $x \in \mathbb{R}$, en général $x \notin \mathcal{F}(\beta, p, e_{\min}, e_{\max})$ et il faut associer à x une approximation $fl(x) \in \mathcal{F}(\beta, p, e_{\min}, e_{\max})$:

1. Lorsque $|x| > \sigma$ ce n'est a priori pas possible (on parle souvent d'overflow) mais dans les systèmes flottants actuels, on associe à x un nombre spécial :

$$\begin{aligned} fl(x) &= +\inf \qquad \text{si } x > \sigma \\ fl(x) &= -\inf \qquad \text{si } x < -\sigma \end{aligned}$$

2. Lorsque $x \neq 0$ est tel que $|x| < \lambda$ et que seuls les nombres normalisés sont utilisés (plus zéro) alors $fl(x) = \pm 0$, selon le signe de x (on parle alors d'underflow). Si les nombres dénormalisés sont utilisés, la troncature vers zéro a lieu uniquement pour $|x| < \mu$ et pour $\mu \leq |x| < \lambda$ on procède comme dans la suite, c-à-d que $fl(x)$ est le nombre flottant le plus proche de x.

3. lorsque $\lambda \leq |x| \leq \sigma$, l'approximation $fl(x)$ est le nombre flottant le plus proche de x, c-à-d que si :

$$x = s\left(\sum_{i=0}^{+\infty} x_i \beta^{-i}\right)\beta^e \tag{3.6}$$

alors :

a. si $x_p < \beta/2$ (on choisit toujours des bases paires), on arrondit "en desssous" :

$$fl(x) = s\left(\sum_{i=0}^{p-1} x_i \beta^{-i}\right)\beta^e \tag{3.7}$$

b. si $x_p \geq \beta/2$ avec au moins un indice $j > p$ tel que $x_j \neq 0$

$$fl(x) = s\left(\sum_{i=0}^{p-2} x_i \beta^{-i} + (x_{p-1} + 1)\beta^{-(p-1)}\right)\beta^e \tag{3.8}$$

c. si $x_p = \beta/2$ avec $x_j = 0$, $\forall j > p$ (x est à égale distance entre deux nombres flottants), il existe principalement deux façons d'arrondir :

i. $fl(x)$ est donné par (3.7)

ii. on arrondit de sorte que le dernier chiffre de la représentation soit pair, c-à-d que :

1. si x_{p-1} est pair alors fl(x) est donné par (3.7)

2. si x_{p-1} est impair alors fl(x) est donné par (3.8)

d. si $x_p = \beta/2$ avec : $\exists j > p$ tel que $x_j \neq 0$ fl(x) est donné par (3.8)

Exemples : On va travailler avec l'ensemble de flottant $\mathcal{F}(10, 8, -38, 37)$ qui a l'avantage d'être en base 10 et qui est assez proche du jeu de flottante dénommé simple précision disponible sur la plupart des machines $[\mathcal{F}(2, 24, -126, 127)]$. Les nombres caractéristiques de cet ensemble sont : $\sigma = 9.9999999 \times 10^{37}$, $\lambda = 10^{-38}$ et $\mu = 10^{-45}$.

$fl(10^{38}) = +\inf$.

$fl(1.2345678 \times 10^{-41}) = +0$ si on n'utilise pas les nombres dénormalisés

$= 0.0012346 \times 10^{-38}$si on les utilise

$fl(9.9999999) = 9.9999999$

$fl(9.99999999) = 10$

$fl(1.55555555) = 1.5555556$

$fl(1.00000005) = 1.0000001$ avec le mode arrondi classique.

Soit $x \in \mathbb{R}$ tel que $|x| \in [\lambda, \sigma]$. Si $|x| \in \left[\beta^e, \beta^{e+1}\right]$, son représentant $fl(x)$ dans $\mathcal{F}(\beta, p, e_{\min}, e_{\max})$ s'écrit :

$$fl(x) = \pm c_0.c_1 \ldots c_{p-1} \times \beta^e \tag{3.9}$$

ou encore :

$$fl(x) = \pm 1.0 \ldots 0 \times \beta^{e+1}. \tag{3.10}$$

Si $|x|$ est suffisamment proche de β^{e+1}. D'après la définition de la fonction fl, l'erreur entre x et $fl(x)$, sera bornée par :

$$\begin{aligned} E_a &= |x - fl(x)| \leq \overbrace{0.0....0}^{p \text{ chiff}}\frac{\beta}{2} \times \beta^e \\ E_a &= |x - fl(x)| \leq \frac{1}{2}\beta^{1-p+e}. \end{aligned}$$

On dit souvent que $E_a \leq \frac{1}{2}ulp$. Cette majoration est constante pour tous les $|x| \in \left[\beta^e, \beta^{e+1}\right]$ et l'erreur relative commise peut être majorée en divisant E_a par le plus petit nombre de cet intervalle soit β^e :

$$E_{r_1} = \frac{|x - fl(x)|}{|x|} \leq \frac{E_a}{\beta^e} = \frac{1}{2}\beta^{1-p} \tag{3.11}$$

Remarques (3.4) :

- Lorsque $|x| \in \left[\beta^e, \beta^{e+1}\right]$ est proche de β^{e+1}, l'erreur relative maximale de x est en fait proche de $\frac{1}{2}\beta^{-p}$.

- Si la machine utilise des nombres dénormalisés et que $|x| \in [\mu, \lambda[$. Alors, on peut simplement dire que : $|fl(x) - x| \leq \frac{1}{2}\mu$, mais l'erreur relative n'est plus bornée par e_m. L'erreur relative maximum passe de e_m au voisinage de λ à $\frac{1}{2}$ au voisinage de μ.

Dans chaque opération de l'arithmétique flottant, on peut commise une erreur. A cette raison , on présente dans la section suivante, le modèle standard pour analyser l'erreur des problèmes numérique.

3.3 Modèle standard (modèle classique)

On note *op* l'une des opérations arithmétiques $(+, -, \times, /, \sqrt{}, \ldots)$ et considérons deux flottants x et y tels que x *op* y soit défini.

Le résultat x *op* y n'est pas nécessairement un flottant. Le résultat légitime attendu est $fl\,(x \text{ op } y)$; l'arrondi du résultat exact x *op* y. Le modèle standard utilisé pour analyser les erreurs décrit pendant l'exécution d'un problème numérique.

Définition (3.4) : soient x et y deux flottants de $\mathcal{F}$ tels que x op y ne provoque pas de dépassement de capacité ($\lambda \leq |x \text{ op } y| \leq \sigma$). La valeur flottante calculée par le modèle classique de l'arithmétique flottant, notée $fl\,(x \text{ op } y)$, vérifie :

$$fl(x \text{ op } y) = (x \text{ op } y)(1 + \alpha), \text{ avec } |\alpha| \leq u, \tag{3.12}$$

On a également

$$fl(x \text{ op } y) = \frac{x \text{ op } y}{1 + \delta}, \text{ avec } |\delta| \leq u, \tag{3.13}$$

Lemme (3.5) : soient x, $y \in \mathcal{F}$ et $op \in \{+, -, \times, /\}$. Si $x\ op\ y \geq 0$ alors

$$x\ op\ y \leq (1+u) fl(x\ op\ y), \tag{3.14}$$

et

$$(1-u)(x\ op\ y) \leq fl(x\ op\ y). \quad \Diamond \tag{3.15}$$

Démonstration : Pour démontrer (3.14), on utilise la deuxième version du modèle standard :

$$fl(x\ op\ y) = \frac{x\ op\ y}{1+\delta}$$

ce qui est equivalent à :

$$x\ op\ y = (1+\delta)\, fl(x\ op\ y) \leq (1+u) fl(x\ op\ y)$$

pour démontrer la formule (3.15), on utilise immédiatement (3.12). on a :

$$fl(x\ op\ y) = (x\ op\ y)(1+\alpha)$$

parce que $-u \leq \alpha \leq u$, on conclut directement le résultat :

$$fl(x\ op\ y) \geq (1-u)(x\ op\ y). \quad \blacklozenge$$

Lemme (3.6) : Soit $y \in \mathcal{F}^n$ $(n \geq 1)$ tel que $y \geq 0$. On a

$$\sum_{i=1}^{n} y_i \leq (1+u)^{n-1} fl(\sum_{i=1}^{n} y_i). \quad \Diamond \tag{3.16}$$

Démonstration : Il suffit de procéder par récurrence sur n. $\blacklozenge$

Remarque (3.5) : Le modèle standard n'implique pas dans le cas où $\alpha = 0$ (pour $x\ op\ y \in \mathcal{F}$). La notation $fl(x\ op\ y)$ de la valeur flottante calculée n'est donc pas (dans ces cas) l'arrondi $fl()$ de la valeur $x\ op\ y$.

3.4 Opérations flottantes

Les opérations arithmétiques sur $\mathcal{F}$ sont définies à l'aide de la notion d'arrondi. A toute opération arithmétique élémentaire $(+, -, \times, /)$ correspond une opération effectuée par les calculateurs électroniques, uniquement avec des éléments de $\mathcal{F}$.

Si $'.'$ désigne une opération arithmétique, l'opération flottantes correspondante notée $\odot$.

Définition (3.5) : Soit $'.'$ l'une des quatre opérations $(+, -, \times, /)$ dans $\mathbb{R}$ l'opération correspondante $\odot$ est correcte pour l'arrondi fl si elle satisfait la propriété

$$\forall x, y \in \mathcal{F} \text{ tels que } x.y \in \mathbb{R} \ , \ x \odot y = fl(x.y) \tag{3.17}$$

Le résultat flottant est l'arrondi du résultat exact s'il n'y a pas de dépassement de capacité.

Définition (3.6) : La loi $\odot$ est définie par :

$$x \odot y = fl(fl(x).fl(y)) \tag{3.18}$$

Exemples : 1. Calcul de $\pi \oplus \pi$ dans $\mathcal{F}(10, 2, -10, 10)$:

$$fl(fl(\pi) + fl(\pi)) = fl(3.1 + 3.1) = 6.2$$

2. Calcul de $2 \otimes \pi$:

$$fl(fl(2) \times fl(\pi)) = fl(2 \times 3.1) = 6.2$$

3.4.1 Addition et soustraction flottante

L'addition et la soustraction flottante sont données par

$$\begin{aligned} x_1 \oplus x_2 &= fl(fl(x_1 + fl(x_2)) \\ x_1 \ominus x_2 &= fl(fl(x_1 - fl(x_2)) \end{aligned} \tag{3.19}$$

Considérons x_1 et x_2 tels que : $|x_1| \geq |x_2|$, on a :

$$\begin{aligned} fl(x_1) &= m_1 \times 10^{e_1} \\ fl(x_2) &= m_2 \times 10^{e_2} = m_2^{'} \times 10^{e_1} \text{ avec } m_2^{'} = m_2 \times 10^{e_1 - e_2} \end{aligned}$$

Si les exposants ne sont pas les mêmes, on doit aligner, c'est-à-dire rendre le plus petit exposant égal au plus grand.

Exemples : *1.* On considère $x_1 = 0.43162 \times 10^5$, $x_2 = 0.18523 \times 10^{-1}$, $p = 4$

$$fl(x_1) = 4.316 \times 10^4, fl(x_2) = 1.852 \times 10^{-2}$$

on écrit: $fl(x_2) = 0.000001852 \times 10^4$

$$fl(x_1) + fl(x_2) = 4.316001852 \times 10^4$$

D'où :

$$x_1 \oplus x_2 = 4.316 \times 10^4 \text{ en arrondissant ou en tronquant}$$

2.

$$\begin{aligned} fl(x_1) &= 4.316 \times 10^4, fl(x_2) = 3.422 \times 10^1 \\ fl(x_1) + fl(x_2) &= 4.319422 \times 10^4 \end{aligned}$$

D'où :

$$x_1 \oplus x_2 = 4.319 \times 10^4 \text{ en arrondissant ou en tronquant}$$

3.

$$\begin{aligned} fl(x_1) &= 4.316 \times 10^4, fl(x_2) = 4.315 \times 10^4 \\ fl(x_1) - fl(x_2) &= 0.001 \times 10^4 \end{aligned}$$

D'où :

$$x_1 \ominus x_2 = 1.000 \times 10^1 \text{ en arrondissant ou en tronquant}$$

3.4.2 Multiplication flottante

La multiplication dans l'arithmétique flottant est définie par :

$$x_1 \otimes x_2 = fl(fl(x_1) \times fl(x_2)) \tag{3.20}$$

On a :

$$fl(x_1) \times fl(x_2) = m_1.m_2 \times 10^{e_1+e_2}. \tag{3.21}$$

Il sera par conséquent nécessaire, dans certains cas, de renormaliser la mantisse $m_1 \times m_2$ afin que son premier digit soit non nul.

1.

$$\begin{aligned} fl(x_1) &= 2.432 \times 10^1, fl(x_2) = 2.000 \times 10^2 \\ fl(x_1) \times fl(x_2) &= 4.864000 \times 10^3 \end{aligned}$$

d'où :

$$x_1 \otimes x_2 = 4.864 \times 10^3 \text{ par arrondissant ou tronquant}$$

2.

$$\begin{aligned} fl(x_1) &= 2.432 \times 10^1, fl(x_2) = 6.808 \times 10^2 \\ fl(x_1) \times fl(x_2) &= 16.557056 \times 10^3 = 1.6557056 \times 10^4 \end{aligned}$$

d'où :

$$x_1 \otimes x_2 = 1.656 \times 10^4 \text{ par arrondi}$$

3.4.3 Division flottante

La quatrième opération flottante (division) est définie par :

$$x_1 \oslash x_2 = fl(fl(x_1)/fl(x_2)) \tag{3.22}$$

On a :

$$\frac{fl(x_1)}{fl(x_2)} = \frac{m_1}{m_2} \times 10^{e_1-e_2}. \tag{3.23}$$

On calcule le quotient $\frac{m_1}{m_2}$ que l'on tronque ou arrondit à p chiffres . L'exposant e est égale à $e_1 - e_2$.

1.

$$\begin{aligned} fl(x_1) &= 4.323 \times 10^5, fl(x_2) = 2.000 \times 10^4 \\ \tfrac{fl(x_1)}{fl(x_2)} &= 2.1615 \times 10^1 \end{aligned}$$

d'où :

$$x_1 \oslash x_2 = 2.162 \times 10^1 \text{ par arrondi}$$

2.

$$\begin{aligned} fl(x_1) &= 2.162 \times 10^5, fl(x_2) = 3.000 \times 10^6 \\ \tfrac{fl(x_1)}{fl(x_2)} &= 7.206666... \times 10^{-2} \end{aligned}$$

d'où :

$$x_1 \oslash x_2 = 7.207 \times 10^{-2} \text{ par arrondi ou par troncature.}$$

Remarques (3.6) : - Les propriétés des opérations sur les nombres réels ne sont pas toutes vérifiées avec les nombres flottants.

- Les opérations flottantes commutatives :

$$fl(x \ op \ y) = fl(y \ op \ x) \text{ pour } op = \{+, -, \times\}$$

- elles ne sont pas associatives.

- elles ne sont pas distributives.

- 0 est l'élément neutre de l'addition flottante et 1 est l'élément neutre de la multiplication flottante.

- l'opposé de $x \in \mathcal{F}$ existe et vaut $-x$.

- par contre n'a pas toujours d'inverse dans $\mathcal{F}$.

Addition flottante n'est pas associative :

Dans l'arithmétique flattant $x + (y + z)$ peut être différent de $(x + y) + z$, par exemple, pour calculer la somme : $1 + 0.0004 + 0.0006 = 1.001$ avec un ordinateur, pour lequel $p = 4$ en procédant troncature.

On a :

$1 \oplus 0.0004 = 1$

$(1 \oplus 0.0004) \oplus 0.0006 = 1$

$0.0004 \oplus 0.0006 = 0.001$

$1 \oplus (0.0004 \oplus 0.0006) = 1.001$

Cet exemple montre que l'addition flottante peut influencer le résultat de la sommation des séries à termes positifs.

Calcule de sommation à termes positifs :

Ordre de sommation :

On veut calculer :

$$S = \sum_{i=1}^{n} a_i, \qquad a_i > 0$$

En arithmétique finie, on calcule cette somme en formant la suite des sommes partielles :

$$\begin{aligned} S_1 &= fl(a_1) \\ S_2 &= S_1 \oplus fl\,(a_2) \\ \vdots &\quad \vdots \\ \vdots &\quad \vdots \\ S_k &= S_{k-1} \oplus fl\,(a_k) \qquad\qquad k = 2, ..., n \end{aligned}$$

Le résultat est alors $S_n \simeq S$.

L'ordre dans lequel on somme les a_i peut changer la valeur de la somme S_n car l'arithmétique flottante n'est pas associative.

Exemple : soit

$$S = 1 + \sum_{i=1}^{n} \frac{1}{i^2 + i} = 2 - \frac{1}{n+1}$$

Calculer cette somme, en arithmétique flottant, de deux façons différentes :

$$\begin{aligned} S_{n.1} &= 1 + \frac{1}{2} + \cdots\cdots + \frac{1}{n^2 + n} \\ S_{n.2} &= \frac{1}{n^2 + n} + \cdots\cdots + \frac{1}{2} + 1 \end{aligned}$$

avec : $n = 9, n = 99, n = 999, n = 9999$.

n	$S_{n.1}$	$S_{n.2}$	valeur exacte S
9	1.9000000000	1.9000000000	1.9
99	1.9900000000	1.9900000000	1.99
999	1.9990000000	1.9990000000	1.999
9999	1.9998999999	1.9999000000	1.9999

On observe que l'on obtient des résultats différents selon que l'on somme de 1 à n ou de n à1 ; les meilleurs résultats étant obtenus dans le second cas. On peut dire ici que lorsqu'il s'agit de nombres flottants positifs, il faut trier tout d'abord les nombres de manière croissante et effectuer ensuite la sommation demandée. Il est très connu aussi dans l'arithmétique flottante que la multiplication flottante n'est pas associative.

Multiplication flottante n'est pas distributive par rapport à l'addition flottante :

La relation : $x \times (y + z) = x \times y + x \times z$ n'est pas vérifiée.

Par exemple : $x = 1.22 \times 10^2$, $y = 3.33 \times 10^2$, $z = 6.95 \times 10^2$

$$\begin{aligned} x \times (y + z) &= fl(1.22 \times 10^2 \times fl(3.33 \times 10^2 + 6.95 \times 10^2)) \\ &= fl(1.22 \times 10^2 \times fl(10.28 \times 10^2)) \\ &= fl(1.22 \times 10^2 \times fl(1.028 \times 10^3)) \\ &= fl(1.22 \times 10^2 \times (1.03 \times 10^3)) \\ &= fl(1.2566 \times 10^5) = 1.26 \times 10^5 \end{aligned}$$

en notre façon :

$$\begin{aligned} (x \times y) + (x \times z) &= fl(fl(1.22 \times 10^2 \times 3.33 \times 10^2) + fl(\ 1.22 \times 10^2 \times 6.95 \times 10^2)) \\ &= fl(fl(4.0626 \times 10^4) + fl(8.479 \times 10^4)) \\ &= fl(4.06 \times 10^4 + 8.48 \times 10^4) \\ &= fl(12.54 \times 10^4) = 1.25 \times 10^5 \end{aligned}$$

Il y a donc une différence entre les deux résultats.

Relations de stricte comparaison :

De la même façon, les relations avec inégalités strictes telles que :

- si $a < b$ alors $a + c < b + c$ pour tout c,
- si $a < b$ et $c < d$ alors $a + c < b + d$,
- si $b < c$ et $a > 0$ alors $ab < ac$.

doivent être affaiblies en remplaçant les dernière inégalité en inégalités larges pour rester vrais en arithmétique flottant.

La relation : $x \times (y/x) = y$ n'est plus vérifiée aussi. En choisissant par exemple $x = 4$ *et* $y = 5$, on obtient

$$\begin{aligned} fl(y/x) &= fl(5/4) \\ &= fl(1+u) \overset{\circ}{=} 1 \end{aligned}$$

et

$$fl(x \times fl(y/x)) = fl(4 \times fl(5/4)) = 4$$

Perte de chiffres significatifs dans la soustraction :

La soustraction ($x_1 \ominus x_2 = fl(fl(x_1 - fl(x_2)))$ est l'opération la plus dangereuse en calcul numérique. Elle peut amplifier l'erreur relative de façon catastrophique, comme l'indique l'exemple suivant :

Exemple :

Considérons les nombres $\sqrt{7001}$ et $\sqrt{7000}$.En arithmétique flottant, on a :

$$\begin{aligned} \sqrt{7001} &\simeq 8.3671979 \times 10^1 \\ \sqrt{7000} &\simeq 8.3666003 \times 10^1 \end{aligned}$$

Donc

$$\sqrt{7001} - \sqrt{7000} = fl((8.3671979 - 8.3666003) \times 10^1) = 5.9760000 \times 10^{-3}$$

On peut obtenir un résultat plus précis en utilisant l'identité suivante :

$$\sqrt{x} - \sqrt{y} = (\sqrt{x} - \sqrt{y}) \times \frac{\sqrt{x} + \sqrt{y}}{\sqrt{x} + \sqrt{y}}$$

On obtient alors :

$$1 \oslash (\sqrt{7001} \oplus \sqrt{7000}) = 1 \oslash (1.6733798 \times 10^2) = 5.9759297 \times 10^{-3}$$

La valeur exacte est $5.97592962824 \times 10^{-3}$.

Chapitre 4

Méthodes d'amélioration des systèmes Toeplitz et Vandermonde confluent

4.1 Introduction

Dans ce chapitre, on présente deux méthodes d'amélioration itératives des systèmes linéaires structurés de Toeplitz et de Vandermonde confluent pour réduire la complexité du calcul et aboutir à une stabilité numérique. afin de réaliser ce but on applique la FFT (Fast Fourier Transform). Il s'agit d'une opération numérique de base importante et bien documentée de par sa grande utilité pratique et théorique.

4.2 Transformation discrète de Fourier

La popularité de la transformation discrète de Fourier découle notamment de l'existence d'algorithmes réalisant l'opération de multiplication de la matrice de Fourier qu'on définira ultérieurement par un vecteur en utilisant $O(n \log n)$ opérations élémentaires seulement. Ces algorithmes rapides sont plus connus sous le nom de FFT. Ce qui nous a motivés à considérer la transformation discrète de Fourier dans ce chapitre réside d'un côté dans le fait que les algorithmes que nous proposons dans ce mémoire font appel à la FFT afin d'améliorer leur complexité, et d'un autre côté dans l'intention de bien préciser que la FFT, en plus de sa rapidité, est une opération numériquement stable.

Définition (4.1) : La matrice de Fourier d'ordre n est la matrice F suivante :

$$F(w,n) = \begin{bmatrix} 1 & \cdots & \cdots & 1 \\ 1 & w & \cdots & w^{n-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & w^{n-1} & & w^{(n-1)^2} \end{bmatrix} = (w^{(i-1)(j-1)})_{1\leq i,j\leq n}, \tag{4.1}$$

où w est une racine $n^{ème}$ principale de l'unité. C'est-à-dire que : $1, w, ..., w^{n-1}$ sont les racines de $Z^n - 1 = 0$. A titre d'exemple :

$$w_n = e^{i\frac{2\pi}{n}}, \quad w_n^{-1} = \overline{w_n} = e^{-i\frac{2\pi}{n}} \qquad i^2 = -1 \tag{4.2}$$

sont des racines $n^{ème}$ principales de l'unité.

Définition (4.2) : La transformation discrète de Fourier (en abrégé TDF) d'un vecteur $x = (x_0\ x_1\ \cdots\ x_{n-1})^T$ est le vecteur Fx.

Soit w toujours une racine $n^{ème}$ principale de l'unité. Alors il est bien connu que l'inverse $F(w,n)^{-1}$ de $F(w,n)$ peut être directement déterminé grâce à la formule suivante :

$$F(w,n)^{-1} = \frac{1}{n}F(w^{-1},n), \tag{4.3}$$

cette formule nous permet, de conclure que l'inverse de TDF est, à une constante multiplicative près, lui même une TDF.

Posons par ailleurs

$$Q = \frac{1}{\sqrt{n}}F(w,n),$$

on peut alors vérifier que

$$\begin{aligned} Q^HQ &= \tfrac{1}{\sqrt{n}}F(w^{-1},n).\tfrac{1}{\sqrt{n}}F(w,n) \\ &= I_n \end{aligned}$$

de sorte qu'on puisse déduire que Q est unitaire.

Notons par $\|.\|$ la norme euclidienne dans $\mathbb{C}^n$, et rappelons que si $A \in \mathbb{C}^{n\times n}$

$$\|A\| = \max_{\|x\|=1} \|Ax\| .$$

On peut énoncer le résultat suivant :

Théorème (4.1) [6] : Supposons que $y = F(w,n)x$. Alors

(i) $\|F(w,n)\| = \sqrt{n}$

(ii) $\|y\| = \sqrt{n}\,\|x\| .$ $\Diamond$

4.2.1 FFT itérative

Dans cette section, on suppose que $n = 2^q$ est une puissance de 2. Si $a = \begin{bmatrix} a_0 & a_1 & \dots & a_{n-1} \end{bmatrix}^T$ on pose

$$a^{[0]} = \begin{bmatrix} a_0 & a_2 & \dots & a_{n-2} \end{bmatrix}^T$$

et

$$a^{[1]} = \begin{bmatrix} a_1 & a_3 & \dots & a_{n-1} \end{bmatrix}^T .$$

D'autre part, si w est une racine $n^{ième}$ principale de l'unité, on note par Δ_w la matrice diagonale suivante :

$$\Delta_w = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & w & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & w^{\frac{n}{2}-1} \end{bmatrix}. \tag{4.4}$$

Par exemple $\Delta_{-1} = [1]\,, \qquad \Delta_i = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$ (où $i^2 = -1$).

On peut dire que l'algorithme FFT récursif de Cooley et Tukey peut être résumé dans la formule suivante :

$$F(w,n)a = \begin{bmatrix} I_{\frac{n}{2}} & \Delta_w \\ I_{\frac{n}{2}} & -\Delta_w \end{bmatrix} \begin{bmatrix} F(w^2,\frac{n}{2}).a^{[0]} \\ F(w^2,\frac{n}{2}).a^{[1]} \end{bmatrix}. \tag{4.5}$$

En développant davantage, on obtient :

$F(w,n)a =$

$$\begin{bmatrix} I_{\frac{n}{2}} & \Delta_w \\ I_{\frac{n}{2}} & -\Delta_w \end{bmatrix} \begin{bmatrix} \begin{bmatrix} I_{\frac{n}{4}} & \Delta_{w^2} \\ I_{\frac{n}{4}} & -\Delta_{w^2} \end{bmatrix} & 0 \\ 0 & \begin{bmatrix} I_{\frac{n}{4}} & \Delta_{w^2} \\ I_{\frac{n}{4}} & -\Delta_{w^2} \end{bmatrix} \end{bmatrix} \begin{bmatrix} F(w^4, \frac{n}{4}).a^{[0][0]} \\ F(w^4, \frac{n}{4}).a^{[0][1]} \\ F(w^4, \frac{n}{4}).a^{[1][0]} \\ F(w^4, \frac{n}{4}).a^{[1][1]} \end{bmatrix}.$$

Pour simplifier, posons :

$$A_1 = \begin{bmatrix} I_{\frac{n}{2}} & \Delta_w \\ I_{\frac{n}{2}} & -\Delta_w \end{bmatrix}$$

et

$$A_2 = I_2 \otimes \begin{bmatrix} I_{\frac{n}{4}} & \Delta_{w^2} \\ I_{\frac{n}{4}} & -\Delta_{w^2} \end{bmatrix}$$

où $\otimes$ est le produit de Kronecker. On a :

$$A_2 = \begin{bmatrix} \begin{bmatrix} I_{\frac{n}{4}} & \Delta_{w^2} \\ I_{\frac{n}{4}} & -\Delta_{w^2} \end{bmatrix} & 0 \\ 0 & \begin{bmatrix} I_{\frac{n}{4}} & \Delta_{w^2} \\ I_{\frac{n}{4}} & -\Delta_{w^2} \end{bmatrix} \end{bmatrix}.$$

D'une manière générale, pour $j = 1, 2, \ldots, q$, on pose

$$A_j = I_{2^{j-1}} \otimes \begin{bmatrix} I_{\frac{n}{2^j}} & \Delta_{w^{2^{j-1}}} \\ I_{\frac{n}{2^j}} & -\Delta_{w^{2^{j-1}}} \end{bmatrix}. \tag{4.6}$$

On peut alors énoncer le résultat suivant :

Théorème (4.2) : Il existe une matrice de permutation P qu'on peut déterminer telle que

$$F(w,n) = A_1.A_2.A_3 \cdots A_q P. \quad \Diamond \tag{4.7}$$

Procédons maintenant à la construction de la matrice de permutation P évoquée dans le théorème précédent. Pour cela, nous aurons besoin d'introduire la notion de renversement binaire d'un entier compris entre 0 et $n-1$. Dans cette perspective, rappelons qu'un entier $k \in \{0, 1, 2, \ldots, n-1\}$ admet la représentation binaire unique

$$k = \sum_{i=0}^{q-1} k_i 2^i \quad (k_i \in \{0,1\}),$$

de sorte qu'un tel entier puisse être identifié à un vecteur de $\{0,1\}^q$. Ainsi, on peut écrire :

$$k = \begin{pmatrix} k_0 \\ k_1 \\ \vdots \\ k_{q-1} \end{pmatrix}$$

Exemple : $(n = 8,\ q = 3)$, $0 = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$, $1 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$, $2 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$, $3 = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$, $4 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$, $5 = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$, $6 = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}$, $7 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$.

D'autre part, notons par J la matrice $(q \times q)$ suivante :

$$J = \begin{bmatrix} 0 & \cdots & 0 & 1 \\ \vdots & & 1 & 0 \\ 0 & & & \vdots \\ 1 & 0 & \cdots & 0 \end{bmatrix}.$$

Définition (4.3) : Soit k un entier compris entre 0 et $n-1$. Le renversement binaire $rev(k)$ de l'entier k est l'entier suivant :

$$rev(k) = J \cdot k.$$

Exemple : $(N = 8, q = 3)$, $rev(0) = 0$, $rev(1) = 4$, $rev(2) = 2$, $rev(3) = 6$, $rev(4) = 1$, $rev(5) = 5$, $rev(6) = 3$, $rev(7) = 7$.

Le résultat suivant détermine la matrice de permutation en question.

Théorème (4.3) [6] : On a $\forall j = 1 : n$

$$Pe_j = e_{rev(j-1)+1},$$

(e_j) *désignant la base canonique* $\diamondsuit$.

Réalisation de l'opération $y = A_j x$: Rappelons que :

$$A_j = I_{2^{j-1}} \otimes \begin{bmatrix} I_{\frac{n}{2^j}} & \Delta_{w^{2^{j-1}}} \\ I_{\frac{n}{2^j}} & -\Delta_{w^{2^{j-1}}} \end{bmatrix}$$

et supposons que $x = \begin{pmatrix} x_0 & x_1 & \cdots & x_{n-1} \end{pmatrix}^T$ et $y = \begin{pmatrix} y_0 & y_1 & \cdots & y_{n-1} \end{pmatrix}^T$. Posons $m = \frac{n}{2^{j-1}}$. Alors si $k = 0 : 2^{j-1} - 1$, on a :

$$\begin{pmatrix} y_{km} \\ y_{km+1} \\ \vdots \\ y_{(k+1)m-1} \end{pmatrix} = \begin{pmatrix} I_{\frac{m}{2}} & \Delta_{w^{2^{j-1}}} \\ I_{\frac{m}{2}} & -\Delta_{w^{2^{j-1}}} \end{pmatrix} \begin{pmatrix} x_{km} \\ x_{km+1} \\ \vdots \\ x_{(k+1)m-1} \end{pmatrix}.$$

Autrement dit, pour $t = 0 : \frac{m}{2} - 1$ et $k = 0 : 2^{j-1} - 1$, on a :

$$\begin{aligned} y_{t+km} &= x_{t+km} + w^{t \cdot 2^{j-1}} x_{t+km+\frac{m}{2}} \\ y_{t+km+\frac{m}{2}} &= x_{t+km} - w^{t \cdot 2^{j-1}} x_{t+km+\frac{m}{2}}. \end{aligned} \tag{4.8}$$

Enfin, nous pouvons présenter l'algorithme suivant que nous nommerons méthode FFT itérative basée sur la formule (4.7).

Algorithme(4.1) : FFT itérative. Dans cet algorithme, on suppose qu'on dispose des n racines de l'unité : $1, w, w^2, \ldots, w^{n-1}$ *stockées dans un tableau nommé R. Par conséquent* $R[k] = w^{k-1}$. *L'objet de cet algorithme consiste à réaliser l'opération* $b = Fa = A_1 A_2 \cdots A_q Pa$, *où* $a = \begin{pmatrix} a_0 & a_1 & \cdots & a_{n-1} \end{pmatrix}^T$ *et* $b = \begin{pmatrix} b_0 & b_1 & \cdots & b_{n-1} \end{pmatrix}^T$.

Commentaire : calcul de $Pa = c$.

1. pour $s = 0 : n - 1$

2. $\quad c_s = a_{rev(s)+1}$;

3. pour $j = q : 1$

4. *pour* $k = 0 : 2^{j-1} - 1$
5. $m = \frac{n}{2^{j-1}}$;
6. *pour* $t = 0 : \frac{m}{2} - 1$
7. $d[t + km] = c[t + km] + R\left[t \cdot 2^{j-1}\right] c\left[t + km + \frac{m}{2}\right]$;
8. $d\left[t + km + \frac{m}{2}\right] = c[t + km] - R\left[t \cdot 2^{j-1}\right] c\left[t + km + \frac{m}{2}\right]$;
9. *fin pour*
10. *fin pour*
11. *pour* $s = 0 : n - 1$
12. $c[s] = d[s]$.

4.2.2 Multiplication rapide de deux polynômes

Soient $P_1(x) = \sum_{i=0}^{n-1} a_i x^i$ et $P_2(x) = \sum_{i=0}^{n-1} b_i x^i$ deux polynômes de degré $\leq (n-1)$. Dans le cas général, pour réaliser la multiplication polynômiale $P(x) = P_1(x) \cdot P_2(x)$, on applique la formule

$$P(x) = \sum_{i=0}^{2n-2} c_i x^i$$

avec

$$c_i = \sum_{k+j=i} a_k b_j.$$

Heureusement, il est très connu dans la littérature qu'on peut effectuer cette opération en $O(n \log n)$ opérations au lieu de $O\left(n^2\right)$ grâce à la FFT comme l'indique le théorème suivant.

Théorème (4.4) [6] : Soit w une racine $(2n)^{ème}$ principale de l'unité et posons $F = F(w, 2n)$. Alors

$$Fc = (Fa) \odot (Fb) \tag{4.9}$$

où $\odot$ désigne le produit ponctuel :

$$\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{pmatrix} \odot \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix} = \begin{pmatrix} x_1 y_1 \\ x_2 y_2 \\ \vdots \\ x_m y_m \end{pmatrix}. \quad \Diamond$$

Pour réaliser le produit de deux polynômes, en utilisant l'algorithme suivant :

Algorithme (4.2) : Multiplication polynômiale de (P_1, P_2).

Données : a_0, a_1, …, a_{n-1} et b_0, b_1, …, b_{n-1}.

Résultats : c_0, c_1, …, c_{2n-2}.

1. $a^* = F(w, 2n)a \quad \left(a = \begin{bmatrix} a_0 & a_1 & \dots & a_{n-1} & 0 & \dots & 0 \end{bmatrix}^T\right)$
2. $b^* = F(w, 2n)b \quad \left(b = \begin{bmatrix} b_0 & b_1 & \dots & b_{n-1} & 0 & \dots & 0 \end{bmatrix}^T\right)$
3. $c^* = a^* \odot b^* \quad (\odot$ *est le produit ponctuel*$)$
4. $c = \frac{1}{2n} F(w^{-1}, 2n)c^*$.

4.2.3 Matrices circulantes et φ-circulantes

on appelle matrice circulante $C(a)$ généré par un vecteur $a = \begin{bmatrix} a_0 & a_1 & \dots & a_{n-1} \end{bmatrix}^T$ la matrice ayant la forme suivante :

$$C(a) = \begin{pmatrix} a_0 & a_{n-1} & \cdots & a_1 \\ a_1 & a_0 & \cdots & a_2 \\ \vdots & \vdots & \ddots & \vdots \\ a_{n-1} & \cdots & & a_0 \end{pmatrix}$$

On dit que $C^-(a)$ est une matrice anti-circulante généré par un vecteur a si

$$C^-(a) = \begin{pmatrix} a_0 & -a_{n-1} & \cdots & -a_1 \\ a_1 & a_0 & \ddots & \vdots \\ \vdots & \vdots & \ddots & -a_{n-1} \\ a_{n-1} & \cdots & & a_0 \end{pmatrix}$$

Plus généralement, on peut définir une classe des matrices qui a les mêmes propriétés de la classe des matrices circulantes. On dit qu'une matrice carrée du type $(n \times n)$; $C_\varphi(a)$ est φ-circulante si elle a la forme suivante :

$$C_\varphi(a) = \begin{pmatrix} a_0 & \varphi a_{n-1} & \cdots & \varphi a_1 \\ a_1 & a_0 & \cdots & \varphi a_2 \\ \vdots & \vdots & \ddots & \vdots \\ a_{n-1} & \cdots & & a_0 \end{pmatrix}$$

où le vecteur $a = (a_0\ a_1\ \cdots\ a_{n-1})^T$ est sa première colonne. Les matrices circulantes et φ-circulante ont les propriétés importantes suivantes

Théorème (4.5) [6] : On a :

$$F.C(a).F^{-1} = D = diag(g_1, g_2, \ldots, g_n)$$

où

$$g = Fa. \quad \Diamond$$

Proposition (4.6) [4] : Soit $C_\varphi(a)$ une matrice φ-circulante de taille $(n \times n)$. Elle est aussi diagonalisable par F, et plus précisément on a :

$$C_\varphi(a) = D_\varphi^{-1} F^{-1} D F D_\varphi$$

avec $D = diag(FD_\varphi a)$, $D_\varphi = diag(1\ \delta\ \delta^2\ \ldots\ \delta^{n-1})$, pour n'importe quel $\delta \in \mathbb{k}$ qui vérifie $\delta^n = \varphi$. $\Diamond$

Remarque : Il est important de signaler que l'inverse d'une matrice circulante (φ-circulante) est circulante (φ-circulante). C'est-à-dire, on peut effectuer *la multiplication d'une matrice circulante (φ-circulante) par un vecteur en utilisant $O(n \log n)$ opérations.*

4.3 Méthode d'amélioration des systèmes Toeplitz

En algèbre linéaire, une matrice Toeplitz ou matrice à diagonales constantes est une matrice dont les coefficients sur une diagonale descendant de gauche à droite sont les mêmes. Toute matrice T à m lignes et n colonnes de la forme :

$$T = \begin{pmatrix} t_0 & t_{-1} & t_{-2} & ..... & t_{-n+1} \\ t_1 & t_0 & \ddots & \ddots & \vdots \\ t_2 & t_1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & t_{-1} \\ t_{m-1} & ..... & t_2 & t_1 & t_0 \end{pmatrix}$$

est une matrice Toeplitz si l'élément situé à l'intersection de la ligne i et la colonne j de T est

$$T_{ij} = t_{i-j}$$

Exemple : La matrice suivante du type (3×3) *est de Toeplitz*

$$T = \begin{pmatrix} 5 & 2 & 7 \\ 4 & 5 & 2 \\ 9 & 4 & 5 \end{pmatrix}$$

La somme de deux matrices Toeplitz est une matrice Toeplitz ; mais la multiplication de deux matrices Toeplitz ou l'inverse d'une matrice Toeplitz n'est pas de Toeplitz, sauf si T est triangulaire.

4.3.1 Produit matrice Toeplitz par vecteur

Cette section à comme but de calculer la multiplication $y = T.x$; d'une matrice Toeplitz par un vecteur $x \in \mathbb{R}^n$.Dans le cas général cette opération coûte $O(n^2)$ opérations qu'on peut la réduire à $O(n \log n)$ opérations, grâce à la FFT.

On peut plonger la matrice T vers matrice circulante $C(\omega) \in \mathbb{R}^{(2n-1)(2n-1)}$ si :

$$T = \begin{pmatrix} t_0 & t_{-1} & ... & t_{-n+1} \\ t_1 & t_0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & t_{-1} \\ t_{n-1} & ... & t_1 & t_0 \end{pmatrix}$$

donc $C(\omega)$ est générer par le vecteur $\omega = (t_0, t_1, ..., t_{n-1}, t_{-n+1}, ..., t_{-1})^T$.

Exemple :

$$T = \begin{pmatrix} 5 & 2 & 7 \\ 4 & 5 & 2 \\ 9 & 4 & 5 \end{pmatrix}$$

une sous matrice du type (3×3) d'une matrice circulante C :

$$C = \begin{pmatrix} 5 & 2 & 7 & 9 & 4 \\ 4 & 5 & 2 & 7 & 9 \\ 9 & 4 & 5 & 2 & 7 \\ 7 & 9 & 4 & 5 & 2 \\ 2 & 7 & 9 & 4 & 5 \end{pmatrix}$$

En générale, si $T = (t_{ij})$ est une matrice Toeplitz du type $(n \times n)$ alors $C = \mathbb{R}^{(2n-1)(2n-1)}$ est circulante avec

$$C(:,1) = \begin{pmatrix} T(1:n,1) \\ T(1,n:-1:2)^T \end{pmatrix}$$

en faisant appelle au théorème (4.5), en affirmant que la multiplication $y = T.x$ effectuée par $O(n \log n)$ opérations.

On sait que la norme de Frobeinus

$$\|A\|_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} |A_{ij}|^2}$$

et la norme matricielle $\|A\|_2$subordonnée à la norme euclidienne sont liées par les inégalités

$$\|A\|_2 \leq \|A\|_F \leq \sqrt{n}\, \|A\|_2$$

Dans ce travaille, on aura besoin de la norme de Frobeinus. On a

$$\|T\|_F \leq \|C(\omega)\|_F \leq \sqrt{(2n-1)}\, \|T\|_F$$

On en déduit que le calcul de

$$y = C(\omega)x$$

via FFT donne une valeur approchée $\widehat{y}$ de y telle que

$$\|\widehat{y} - y\|_2 \leq c_n \varepsilon\, \|C(\omega)\|_2\, \|x\|_2$$

4.3.2 Structures de déplacement

La matrice Toeplitz satisfaite l'équation de déplacement suivante :

$$Z_1 T - T Z_{-1} = e_1 \omega^T + v e_n^T \tag{4.10}$$

où Z_s est donnée par :

$$Z_s = \begin{pmatrix} 0 & 0 & ... & 0 & s \\ 1 & \ddots & \ddots & & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 0 \end{pmatrix}$$

$$\omega^T = (t_{n-1} - t_{-1}.....t_1 - t_{-n+1}t_0),\ v = \begin{pmatrix} t_0 \\ t_{1-n} + t_1 \\ \vdots \\ t_{-1} + t_{n-1} \end{pmatrix}$$

Si on prémultiplie et postmultiplie (4.10) par T^{-1}, on obtient :

$$T^{-1} Z_1 - Z_{-1} T^{-1} = T^{-1} e_1 \omega^T T^{-1} + T^{-1} v e_n^T T^{-1} \tag{4.11}$$

Soit C^- la matrice anti-circulante générée par $T^{-1}e_1$ et

$$C = \begin{pmatrix} e_n^T T^{-1} Z_1 \\ e_n^T T^{-1} Z_1^2 \\ \vdots \\ e_n^T T^{-1} Z_1^{n-1} \\ e_n^T T^{-1} \end{pmatrix}$$

$(C^-)^{-1}$est une matrice anti-circulante vérifie la relation commutative

$$(C^-)^{-1} Z_{-1} = Z_{-1} (C^-)^{-1}$$

De même C^{-1}est circulante et qu'elle commute avec Z_1

$$C^{-}Z_1 = Z_1 C^{-1}$$

Si en prémultipliant et postmultipliant (4.11) par $(C^{-})^{-1}$et C^{-1} respectivement, on obtient :

$$\begin{aligned}(C^{-})^{-1}T^{-1}C^{-1}Z_1 - Z_{-1}(C^{-})^{-1}T^{-1}C^{-1} &= (C^{-})^{-1}T^{-1}e_1\omega^T T^{-1}C^{-1} \\ &+(C^{-})^{-1}T^{-1}ve_n^T T^{-1}C^{-1} \\ &= e_1\omega^T T^{-1}C^{-1} + (C^{-})^{-1}T^{-1}ve_n^T\end{aligned}$$

Par identification, on a :

$$(C^{-})^{-1}T^{-1}C^{-1} = T^{'}$$

est une matrice Toeplitz et

$$T^{-1} = C^{-}\acute{T}C \tag{4.12}$$

Par conséquent le produit $x = T^{-1}y$ peut se faire en $O(n \log n)$ opérations. Si $\hat{x}$ désigne la solution approchée de x, il est facile de montrer que :

$$\|\hat{x} - x\|_2 \leq c_n \varepsilon \left\|C^{-}\right\|_2 \left\|\acute{T}\right\|_2 \|C\|_2 \|x\|_2$$

4.3.3 Résolution du système Toeplitz

On considère un système linéaire $Ax = b$ avec A inversible, on utilise plusieurs méthodes pour résoudre ce système. Par exemple la méthode LU, Cholesky...

Lorsqu'il s'agit d'un système linéaire Toeplitz $Tx = b$, on peut proposer les deux approches suivantes :

Approche 1

- Réaliser la décomposition $T = LU$. En générale ceci coûte $O(n^2)$ opérations.

- Résoudre le système $L(U.x) = b$. Comme ni L ni U n'est structurée, cette résolution coûte $O(n^2)$.

Approche 2

- Calculer l'inverse T^{-1}de T, ce qui demande $O(n^2)$ opérations.

- Calculer $x = T^{-1}b$ ce qui, en utilisant la décomposition (4.12), coûte $O(n \log n)$ opérations.

Dans ces méthodes, on peut dire sur le plan numérique, que si $\widetilde{B}$ désigne la valeur approchée de T^{-1}, il existe une constante d'instabilité $K_1(T)$, telle que :

$$\left\| \widetilde{B} - T^{-1} \right\|_2 \leq c_n \varepsilon K_1(T) \left\| T^{-1} \right\|_2 \tag{4.13}$$

Dans ce qui suit, on travaille au lieu de $\widetilde{B}$ avec la matrice B vérifiant la structure :

$$BZ_1 - Z_{-1}B = \widetilde{B}e_1\omega^T\widetilde{B} + \widetilde{B}ve_n^T\widetilde{B} \tag{4.14}$$

Dans ce cas

$$B \simeq C^{-}T^{'}C \tag{4.15}$$

où C^{-} est la matrice anti-circulante générée par $\widetilde{B}e_1$et

$$C = \begin{pmatrix} e_n^T\widetilde{B}Z_1 \\ e_n^T\widetilde{B}Z_1^2 \\ \vdots \\ e_n^T\widetilde{B}Z_1^{n-1} \\ e_n^T\widetilde{B} \end{pmatrix}$$

et $T^{'}$ est la matrice Toeplitz par identification dans l'équation (4.14). Dans ces conditions, il est bien à quel point l'estimation (4.13) est altérée.

Posons pour cela

$$\phi(x) = XZ_1 - Z_{-1}X$$

ϕ est une application linéaire inversible dont la matrice par rapport à la base canonique est :

$$\phi = \begin{pmatrix} Z_{-1} & 0 & ... & 0 & I_n \\ I_n & \ddots & \ddots & & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & I_n & Z_{-1} \end{pmatrix}$$

La norme de Frobeinus $\left\|\phi^{-1}\right\|_F$ de ϕ^{-1} est un polynôme en n, par suite

$$\left\|\phi^{-1}\right\|_2 = c_n$$

est un polynôme en n. Ceci étant on a

$$\begin{aligned} \left\|B - T^{-1}\right\|_2 &\leq \left\|B - T^{-1}\right\|_F &&\leq \left\|\phi^{-1}\right\|_2 \left\|\phi(B - T^{-1})\right\|_F \\ &\leq c_n \left\|\phi(B - T^{-1})\right\|_F &&\leq c_n \left\|\phi(B - T^{-1})\right\|_2 \end{aligned}$$

On est alors conduit à estimer $\left\|\phi(B - T^{-1})\right\|_2$. On a

$$\begin{aligned} \phi(B - T^{-1}) &= (\widetilde{B} e_1 \omega^T \widetilde{B} + \widetilde{B} v e_n^T \widetilde{B}) - (T^{-1} e_1 \omega^T T^{-1} + T^{-1} v e_n^T T^{-1}) \\ &= (\widetilde{B} - T^{-1}) e_1 \omega^T \widetilde{B} + T^{-1} e_1 \omega^T (\widetilde{B} - T^{-1}) + (\widetilde{B} - T^{-1}) v e_n^T \widetilde{B} \\ &\quad + T^{-1} v e_n^T (\widetilde{B} - T^{-1}) \end{aligned}$$

Si bien que

$$\left\|\phi(B - T^{-1})\right\|_2 \leq (\left\|e_1 \omega^T \widetilde{B}\right\|_2 + \left\|T^{-1} e_1 \omega^T\right\|_2 + \left\|v e_n^T \widetilde{B}\right\|_2 + \left\|T^{-1} v e_n^T\right\|_2) \left\|\widetilde{B} - T^{-1}\right\|_2$$

Par conséquent, en utilisant (4.13)

$$\left\|B - T^{-1}\right\|_2 \leq c_n \varepsilon K_2(T) \left\|T^{-1}\right\|_2 \tag{4.16}$$

où

$$K_2(T) = K_1(T)(\left\|e_1 \omega^T \widetilde{B}\right\|_2 + \left\|T^{-1} e_1 \omega^T\right\|_2 + \left\|v e_n^T \widetilde{B}\right\|_2 + \left\|T^{-1} v e_n^T\right\|_2)$$

On considère le problème de résoudre un système linéaire

$$T.x = y$$

où T est Toeplitz et en supposant qu'on dispose de la matrice B définie par l'équation de déplacement (4.14) et vérifiant la décomposition (4.15) et l'estimation (4.16). Posons

$$\hat{x} = B.y$$

de sorte que

$$\|\hat{x} - x\|_2 \leq c_n \varepsilon K_2(T) \left\|T^{-1}\right\|_2 \|T.x\|_2$$

Pour calculer $\hat{x} = B.y$, on utilise la décomposition (4.15) afin d'atteindre la performance $O(n \log n)$, ce qui numériquement donne une solution approchée $\hat{x}$ telle que :

$$\|\hat{x} - x\|_2 \leq c_n \varepsilon \left\|C^-\right\|_2 \left\|T^{'}\right\|_2 \|C\|_2 \|T.x\|_2$$

d'où

$$\|\hat{x} - x\|_2 \leq c_n \varepsilon K(T) \|T.x\|_2$$

où

$$K(T) = K_2(T) \left\|T^{-1}\right\|_2 + \left\|C^-\right\|_2 \left\|T^{'}\right\|_2 \|C\|_2$$

Dans la correction de la solution $\hat{x}$, nous supposons que :

$$\sqrt{\varepsilon} K(T) \|T\|_2 \leq c_n$$

où c_n est la constante générique qui dépend polynômialement de n. Nous proposons alors l'algorithme de correction suivant :

Algorithme (4.3) : (Méthode d'amélioration des système Toeplitz)

1. $d = y - T.\hat{x}$

2. $\check{r} = B.d$

3. $x^{'} = \hat{x} + \check{r}$

complexité : $O(n \log n)$ opérations.

4.3.4 Matrices Toeplitz et polynômes

Une classe très importante des matrices Toeplitz est la classe des matrices triangulaires.

Exemples :

1. $\begin{bmatrix} a_0 & 0 & 0 & 0 \\ a_1 & a_0 & 0 & 0 \\ a_2 & a_1 & a_0 & 0 \\ a_3 & a_2 & a_1 & a_0 \end{bmatrix}$ est une matrice Toeplitz triangulaire inférieure.

2. $\begin{bmatrix} a_0 & a_1 & a_2 & a_3 \\ 0 & a_0 & a_1 & a_2 \\ 0 & 0 & a_0 & a_1 \\ 0 & 0 & 0 & a_0 \end{bmatrix}$ est une matrice Toeplitz triangulaire supérieure.

Il est clair que les matrices Toeplitz triangulaires inférieures sont définies à partir de ses premières colonnes, de sorte que les matrices Toeplitz triangulaires supérieures sont les transposées de ces dernières. Cette observation justifie le stockage linéaire de ces matrices d'un coté et de l'autre coté elle justifie l'écriture $L(a)$ pour désigner la matrice triangulaire inférieure dont la première colonne est le vecteur a et $R(a) = L(a)^T$ pour désigner la matrice triangulaire supérieure. Par exemple si $a = \begin{bmatrix} 1 & 2 & 0 & 1 \end{bmatrix}^T$, on a :

$$L(a) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 \\ 0 & 2 & 1 & 0 \\ 1 & 0 & 2 & 1 \end{bmatrix} \quad \text{et} \quad R(a) = \begin{bmatrix} 1 & 2 & 0 & 1 \\ 0 & 1 & 2 & 0 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

C'est-à-dire la matrice de Toeplitz T est écrite :

$$T = L(a) + R(b)$$

telle que

$$a = T(:, 1) \qquad b = (0, T(1, 2 : n)).$$

Dans ce qui suit, on verra voir que le produit d'une matrice Toeplitz triangulaire par un vecteur peut-être effectué grâce au produit de deux polynômes à une variable.

Soit $L(a)$ une matrice Toeplitz triangulaire inférieure caractérisée par sa première colonne $a = \begin{bmatrix} a_0 & a_1 & \dots & a_{n-1} \end{bmatrix}^T$ et on se donne le vecteur $b = \begin{bmatrix} b_0 & b_1 & \dots & b_{n-1} \end{bmatrix}^T$, alors le produit matrice-vecteur cherché $L(a)b = c$ équivalent au produit matriciel $L(a)L(b)$ qui donne une matrice Toeplitz triangulaire inférieure $L(c)$, où $c \in \mathbb{C}^n$ représente les n premiers coefficients du polynôme produit $z(x) = p(x).q(x)$, où $p(x)$ et $q(x)$ sont les représentations polynômiales des vecteurs a et b :

$$p(x) = \sum_{i=0}^{n-1} a_i x^i \text{ et } q(x) = \sum_{i=0}^{n-1} b_i x^i$$

c'est-à-dire le vecteur c est défini comme suit :

$$c = z\,[1 : n]\,,$$

tel que :

$$z(x) = \sum_{i=0}^{2n-2} c_i x^i$$

avec

$$c_i = \sum_{k+j=i} a_k b_j.$$

Le même vecteur $z \in \mathbb{C}^{2n}$, on peut le définir grâce à la FFT de la manière suivante :

$$z = F\left(w^{-1}, 2n\right)\left(\left(F\left(w, 2n\right)a^*\right) \odot \left(F(w, 2n)b^*\right)\right),$$

où w est une racine $(2n)^{ème}$ principale de l'unité et les vecteurs a^*, $b^* \in \mathbb{C}^{2n}$ sont donnés par :

$$a^* = \begin{bmatrix} a_0 & a_1 & \dots & a_{n-1} & 0 & \dots & 0 \end{bmatrix}^T$$

et

$$b^* = \begin{bmatrix} b_0 & b_1 & \dots & b_{n-1} & 0 & \dots & 0 \end{bmatrix}^T,$$

En notant $\tilde{a}$, le retournement vertical d'un vecteur a suivant :

$$\tilde{a} = \begin{bmatrix} a_{n-1} & a_{n-2} & \dots & a_0 \end{bmatrix}^T.$$

Soit $R(a)$ une matrice Toeplitz triangulaire supérieure, le vecteur c qui représente le résultat du produit de la matrice $R(a)$ par b, on peut le trouver grâce au produit matriciel suivant :

$$R(a)R(\tilde{b}) = R(\tilde{c}),$$

c'est-à-dire que le vecteur cherché $\tilde{c}$ forme les n premiers coefficients du polynôme produit $p(x).r(x)$. D'une autre manière, le vecteur c est le suivant :

$$\tilde{c} = z\,[1:n]\,,$$

où

$$z = F^{-1}\,(w, 2n)\left((F(w, 2n)a^*) \odot \left(F\,(w, 2n)\left(\tilde{b}\right)^*\right)\right).$$

Comme la matrice de Toeplitz peut écrit comme somme de deux matrices Toeplitz triangulaires, le produit $y = Tx$ est équivalent à la postmultiplication d'une matrice Toeplitz triangulaire inférieure ou une matrice Toeplitz triangulaire supérieure par un vecteur qu'on peut l'effectuer en $O(n \log n)$ par la FFT.

Il est clair que le produit de deux polynômes joue un rôle extrêmement important dans la prémultiplication ou la postmultiplication d'une matrice triangulaire inférieure ou triangulaire supérieure par un vecteur ligne ou un vecteur colonne. C'est pour cette raison on propose l'algorithme mult pour effectuer le produit d'une matrice triangulaire par un vecteur.

Algorithme (4.4) : mult(a,b).

données : a_0, a_1, …, a_{n-1} et b_0, b_1, …, b_{n-1}.

Résultats : d_0, d_1, …, d_{n-1}.

1. $a^* = F(w, 2n)a$.
2. $b^* = F(w, 2n)b$.
3. $c^* = a^* \odot b^*$.
4. $c = \frac{1}{2n}F(w^{-1}, 2n).c^*$.
5. $d = c\,[1:n]$.

4.4 Méthode d'amélioration des systèmes Vandermonde confluent

On dit qu'une matrice W de nd lignes et nd colonnes est de Vandermonde confluente si elle est définie comme suit

$$W = [W_1\ W_2\ \dots\ W_n] \tag{4.17}$$

telle que

$$W_i = \left[f(x_i)\ f^{(1)}(x_i)\ \cdots\ f^{(d-1)}(x_i) \right]$$

et

$$f(x_i) = \left[\ 1 \quad x_i \quad x_i^2 \quad \cdots \quad x_i^{nd-1}\ \right]^T, \quad \text{pour } i = 1 : n$$

Où x_1, x_2, $\dots x_n$ sont n nombres deux à deux distincts et n et d sont des entiers tels que $n \geq 1$ et $d \geq 2$, et $f(x_i)$, $f^{(1)}(x_i)$, $\cdots$ $f^{(d-1)}(x_i)$ désigne les dérivées successives de f.

Maintenant, on cherche à appliquer la méthode d'élimination de Gauss par blocs pour résoudre un système linéaire de Vandermonde confluent de dimension $(nd \times nd)$ en $O\left((d \log d)\, n^2\right)$ opérations au lieu de $O\left(d^2 n^2\right)$. Pour aboutir à ce but, on considère la structure de déplacement proposée et démontrée dans [9].

Donc, la matrice W est caractérisée par la structure de déplacement suivante :

$$Z^T W - W D_B = e_{nd} y^T, \tag{4.18}$$

où

$$D_B = diag(B(x_1), B(x_2), \dots, B(x_n))$$

est une matrice diagonale par blocs, telle que les matrices $B\left(x_k\right)$; $k = 1 : n$ sont définies par :

$$B(x) = xI_d + A;\ A = \left[\ 0 \quad e_1 \quad 2e_2 \quad 3e_3 \quad \cdots \quad (d-1)\, e_{d-1}\ \right], \tag{4.19}$$

et

$$y^T = \left(x_1^{nd}\, (nd)_1\, x_1^{nd-1} .. (nd)_d\, x_1^{nd-d} .... x_n^{nd}\, (nd)_1\, x_n^{nd-1} .. (nd)_d\, x_n^{nd-d} \right).$$

Si on prémultiplie et postmultiplie la structure (4.18) par Z et D_B^{-1} respectivement et en tenant compte que :

$$ZZ^T = I_{nd} - e_1 e_1^T$$

et

$$Ze_{nd} = 0,$$

on constate immédiatement que cette structure devient :

$$ZW - WD_B^{-1} = e_1 v^T, \tag{4.20}$$

où

$$v^T = -\begin{pmatrix} r(x_1) & \cdots & r(x_n) \end{pmatrix}; \tag{4.21}$$

$$r(x) = \begin{pmatrix} x^{-1} & -x^{-2} & 2x^{-3} & \ldots & (-1)^{d-1}(d-1)!x^{-d} \end{pmatrix},$$

c'est-à-dire $r(x)$ composé par x^{-1} et ses dérivées successives jusqu'à l'ordre $(d-1)$.

4.4.1 Complément de Schur

Définition (4.4) : Soit $A \in C^{(n+s)\times(n+s)}$ une matrice du type $(n+s)\times(n+s)$ qu'on peut représenter par blocs de la façon suivante :

$$A = \begin{bmatrix} M & F \\ E & D \end{bmatrix}, \tag{4.22}$$

telle que : M est une matrice régulière $\in C^{s\times s}$, E^T, $F \in C^{s\times n}$ et $D \in C^{n\times n}$, on peut définir le complément de Schur de A comme opérateur vérifiant la formule suivante :

$$S(A) = D - EM^{-1}F. \tag{4.23}$$

Si on se base sur le complément de Schur, il est facile de vérifier que la première étape de l'élimination de Gauss par blocs sur A est la suivante :

$$J_1 A = \begin{bmatrix} M & F \\ 0 & S(A) \end{bmatrix}, \tag{4.24}$$

$$J_1 = \begin{bmatrix} I_s & 0 \\ -EM^{-1} & I_n \end{bmatrix}$$

désigne la première matrice de l'élimination de Gauss par blocs et par convention, on peut écrire :

$$S^0(A) = A \qquad S^{k+1}(A) = S\left(S^k(A)\right) \quad k = 0 : n-2. \tag{4.25}$$

Dans la prochaine section, on présente une propriété du complément de Schur qui est connue dans la littérature et très importante dans l'application de l'élimination de Gauss sur les matrices structurées.

4.4.2 Stabilité de la structure de déplacement par le complément de Schur

Soit la matrice structurée A définie dans (4.22) ayant la structure de déplacement suivante :

$$FA - AB = \begin{bmatrix} u & w \end{bmatrix} \begin{bmatrix} v & z \end{bmatrix}^H, \tag{4.26}$$

où $F \in \mathbb{C}^{(n+s)\times(n+s)}$ est bidiagonale inférieure, $B \in \mathbb{C}^{(n+s)\times(n+s)}$ est bidiagonale supérieure et u, v, w, $z \in \mathbb{C}^{(n+s)}$, en concordance avec (4.22), il est nécessaire de représenter les matrices F, B et $\begin{bmatrix} u & w \end{bmatrix} \begin{bmatrix} v & z \end{bmatrix}^H$ dans (4.26) par blocs comme suit :

$$F = \begin{bmatrix} F_1 & 0 \\ X & F_2 \end{bmatrix}, \; B = \begin{bmatrix} B_1 & Y \\ 0 & B_2 \end{bmatrix}$$

et

$$\begin{bmatrix} u & w \end{bmatrix} \begin{bmatrix} v & z \end{bmatrix}^H = \begin{bmatrix} u_1 & w_1 \\ u_2 & w_2 \end{bmatrix} \begin{bmatrix} v_1 & z_1 \\ v_2 & z_2 \end{bmatrix}.$$

Le complément de Schur d'une matrice A qui constitue l'opération de base dans le processus d'éliminations de Gauss est stable par la structure de déplacement de cette dernière comme l'indique le théorème suivant :

Théorème (4.7) : Supposons que A vérifie l'équation (4.26)*. Alors :*

$$F_2 S(A) - S(A) B_2 = \begin{bmatrix} u' & w' \end{bmatrix} \begin{bmatrix} v' & z' \end{bmatrix}^H \tag{4.27}$$

avec

$$\begin{bmatrix} 0 & 0 \\ u' & w' \end{bmatrix} = \begin{bmatrix} u & w \end{bmatrix} - \begin{bmatrix} I_s \\ EM^{-1} \end{bmatrix} \begin{bmatrix} u_1 & w_1 \end{bmatrix} \tag{4.28}$$

et

$$\begin{bmatrix} 0 & 0 \\ v' & z' \end{bmatrix} = \begin{bmatrix} v & z \end{bmatrix}^H - \begin{bmatrix} v_1 & z_1 \end{bmatrix}^H \begin{bmatrix} I_s & M^{-1}F \end{bmatrix}. \quad \Diamond \tag{4.29}$$

Démonstration : Soit A la matrice structurée caractérisée par sa structure de déplacement (4.24). Si on prémultiplie et postmultiplie cette structure par J_1 et K_1 respectivement, où J_1 est la première matrice de l'élimination de Gauss et K_1 est la suivante :

$$K_1 = \begin{bmatrix} I_s & -M^{-1}F \\ 0 & I_n \end{bmatrix},$$

on obtient l'équation :

$$\left(J_1 F J_1^{-1}\right)(J_1 A K_1) - (J_1 A K_1)\left(K_1^{-1} B K_1\right) = J_1 \begin{bmatrix} u & w \end{bmatrix} \begin{bmatrix} v & z \end{bmatrix}^H K_1.$$

En tenant compte que :

($\star$) Pour le premier terme on a :

$$J_1 F J_1^{-1} = \begin{bmatrix} F_1 & 0 \\ * & F_2 \end{bmatrix},\ J_1 A K_1 = \begin{bmatrix} M & 0 \\ 0 & S(A) \end{bmatrix}$$

$$\text{et}\quad K_1^{-1} B K_1 = \begin{bmatrix} B_1 & * \\ 0 & B_2 \end{bmatrix}.$$

($\star\star$) Pour le deuxième terme on a :

$$J_1 \begin{bmatrix} u_1 & w_1 \\ u_2 & w_2 \end{bmatrix} = \begin{bmatrix} u_1 & w_1 \\ \acute{u} & \acute{w} \end{bmatrix}$$

et

$$\begin{bmatrix} v_1 & z_1 \\ v_2 & z_2 \end{bmatrix}^H K_1 = \begin{bmatrix} v_1 & z_1 \\ \acute{v} & \acute{z} \end{bmatrix}^H,$$

où

$$\begin{aligned} u' &= u_2 - EM^{-1}u_1 \\ w' &= w_2 - EM^{-1}w_1 \\ v' &= v_2 - (M^{-1}F)^H v_1 \\ z' &= z_2 - (M^{-1}F)^H z_1. \end{aligned}$$

Ce qui nous permet d'écrire :

$$\begin{bmatrix} 0 & 0 \\ u' & w' \end{bmatrix} = \begin{bmatrix} u & w \end{bmatrix} - \begin{bmatrix} I_s \\ EM^{-1} \end{bmatrix} \begin{bmatrix} u_1 & w_1 \end{bmatrix}$$

et

$$\begin{bmatrix} 0 & 0 \\ v' & z' \end{bmatrix} = \begin{bmatrix} v & z \end{bmatrix}^H - \begin{bmatrix} v_1 & z_1 \end{bmatrix}^H \begin{bmatrix} I_s & M^{-1}F \end{bmatrix}.$$

Par des calculs simples, on trouve :

$$\begin{bmatrix} F_1M - MB_1 & \star \\ \star & F_2S(A) - S(A)B_2 \end{bmatrix} = \begin{bmatrix} \star & \star \\ \star & \begin{bmatrix} u' & w' \end{bmatrix} \begin{bmatrix} v' & z' \end{bmatrix}^H \end{bmatrix},$$

alors, on peut obtenir immédiatement le résultat, c'est-à-dire :

$$F_2S(A) - S(A)B_2 = \begin{bmatrix} u' & w' \end{bmatrix} \begin{bmatrix} v' & z' \end{bmatrix}^H. \quad \blacklozenge$$

Dans l'équation de déplacement (4.20), on observe que Z est triangulaire inférieure et D_B^{-1} est diagonale par blocs, alors le théorème (4.17) indique que le complément de Schur de W prend le même type de structure. C'est pour cette raison qu'on préfère la structure (4.20).

Définition (4.5) : Soit A une matrice du type $(rd \times rd)$; $r = 1 : n$. On dit que A est matrice r structurée Vandermonde confluente, s'il existe deux vecteurs g et h de dimension rd telle que :

$$ZA - AD_{r,B}^{-1} = gh^T, \tag{4.30}$$

où :

$$D_{r,B} = diag\,(B(x_k)\,;\ k = n - r + 1 : n$$

est une matrice diagonale par blocs et $x_1, \ldots, x_n$ *sont toujours n points deux à deux distincts.*

On a bien observé que la matrice r Vandermonde confluente A définie par l'équation de déplacement (4.30) est caractérisée par r et les vecteurs g et h, alors leur identification nécessite $O(nd)$ stockages et par convention, on peut l'écrire :

$$A \longleftrightarrow [g, h]^r\,, \tag{4.31}$$

laquelle peut être représentée par blocs de la manière suivante :

$$A = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1n} \\ A_{21} & A_{22} & \cdots & A_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n1} & A_{n2} & \cdots & A_{nn} \end{bmatrix}, \tag{4.32}$$

où ses éléments $(A_{ij})\,; i, j = 1 : r$ sont des matrices du type $(d \times d)$. Tout vecteur V de dimension rd est représenté par blocs comme suit :

$$V = [V_1\ V_2\ \ldots\ V_r]^T\,, \tag{4.33}$$

où $(V_k)_{k=1:r}$ sont des vecteurs de dimension d.

En concordance avec la représentation (4.22), A peut être écrit comme suit :

$$A = \begin{bmatrix} A_{11} & G \\ H & K \end{bmatrix}, \tag{4.34}$$

où

$$G = \begin{bmatrix} A_{11} & \cdots & A_{1r} \end{bmatrix}$$

et

$$H = \begin{bmatrix} A_{21} & \cdots & Ar1 \end{bmatrix}^T,$$

A_{11} doit être non singulière. Dans ce cas, le complément de Schur de A ; $S(A)$ est défini comme suit :

$$S(A) = K - HA_{11}^{-1}G. \tag{4.35}$$

Par une application directe du théorème (4.17) sur la matrice A, on présente le résultat suivant :

Théorème (4.8)[10] : Soit A la matrice de Vandermonde confluente donnée dans (4.34) *et notée par* (4.31) *;* $A \longleftrightarrow [g,h]^r$ $(r \geq 2)$. *Le complément de Schur* $S(A)$ *est lui même une matrice de Vandermonde confluente ;* $S(A) \longleftrightarrow [g',h']^{r-1}$ *avec :*

$$\begin{aligned} g'_k &= g_{k+1} - A_{k+1,1}A_{11}^{-1}g_1 \quad k = 1 : r-1 \\ h'^T_k &= h^T_{k+1} - h_1^T A_{11}^{-1}A_{1,k+1} \quad k = 1 : r-1. \quad \Diamond \end{aligned} \tag{4.36}$$

Dans notre étude, on s'intéresse au matrice n Vandermonde confluente qu'on note par :

$$W \leftrightarrow [e_1, v]^n$$

où v est le vecteur définie dans (4.21) et on la représente par blocs de la façon suivante :

$$W = \begin{bmatrix} W_{11} & W_{12} & \cdots & W_{1n} \\ W_{21} & W_{22} & \cdots & W_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ W_{n1} & W_{n2} & \cdots & W_{nn} \end{bmatrix}. \tag{4.37}$$

Comme On a énoncé précédemment, le processus d'élimination de Gauss est une répétition finie du complément de Schur, alors on peut l'obtenir par l'application de (4.25) sur W, c'est-à-dire :

$$S^0(W) = W,\ S^{k+1}(W) = S(S^k(w)) \quad k = 0 : n-2$$

A l'aide de cette notation, on peut énoncer le résultat suivant qui nous affirme que le

$k^{ème}$ complément de Schur $S^k(W)$ est une matrice $(n-k)$ structurée de Vandermonde confluente pour $k = 0 : n-1$.

Théorème (4.9)[10] : Soit W une matrice de Vandermonde confluente du type $(nd \times nd)$ définie dans (4.17)*, alors les compléments de Schur de W, $S^k(W)$, $k = 0 : n-1$ existe ; $S^0(W) = W \longleftrightarrow [e_1, v]^n$ et $S^k(W) \longleftrightarrow [g[k], h[k]]^{n-k}$, $k = 0 : n-2$, avec*

$$\begin{aligned} g[0] &= e_1, \quad h[0]^T = v^T \\ g[k+1]_i &= g[k]_{i+1} - \left(S^k(W)\right)_{i+1,1} \left(S^k(W)\right)_{11}^{-1} g[k]_1 \\ h[k+1]_i^T &= h[k]_{i+1}^T - h[k]_1 \left(S^k(W)\right)_{11}^{-1} \left(S^k(W)\right)_{1,i+1} \end{aligned} \tag{4.38}$$

pour $i = 1 : n-k+1$. $\Diamond$

La décomposition de W en deux matrices carrées par blocs L et U du type $(nd \times nd)$; L est triangulaire inférieure et U est triangulaire supérieure, telle que :

$$W = LU,$$

lesquelles sont définies par :

$$\begin{aligned} L_{i+k,k+1} &= \left(S^k(W)\right)_{i,1} \left(S^k(W)\right)_{11}^{-1} \quad i = 1 : n-k \\ U_{k+1,j+k} &= \left(S^k(W)\right)_{1,j} \quad j = 1 : n-k \end{aligned} \tag{4.39}$$

pour $k = 0 : n-1$.

A partir des relations de récurrences (4.38), il est facile de voir que l'algorithme de réaliser l'élimination de Gauss par blocs d'une matrice de Vandermonde confluente est de complexité $O((dn)^2)$. Dans la section suivante, nous étudierons les structures des éléments blocs des matrices $r-$structurées de Vandemonde confluentes de telle sorte que la complexité de cet algorithme peut s'améliorer.

Avant de factoriser les éléments blocs, on donne quelques propriétés sur les matrices Toeplitz Triangulaires.

Soient u, $v \in \mathbb{C}^d$ deux vecteurs de dimension d génèrent les matrices Toeplitz triangulaires inférieures $L(u)$ et $L(v)$, telle que :

1. $u, v \in \mathbb{C}^d \Longrightarrow \exists z_1 \in \mathbb{C}^d$ telle que $L(u) \cdot L(v) = L(v) \cdot L(u) = L(z_1)$ (resp $\exists z_2 \in \mathbb{C}^d$ telle que $R(u) \cdot R(v) = R(v) \cdot R(u) = R(z_2)$).

2. $u \in \mathbb{C}^d$ et $u_1 \neq 0 \Longrightarrow \exists v_1 \in \mathbb{C}^d$ telle que $L(u)^{-1} = L(v_1)$ (resp $\exists v_2 \in \mathbb{C}^d$ telle que $R(u)^{-1} = R(v_2)$).

On ajoute que le produit de deux matrices Toeplitz du type $(d \times d)$ peut être réalisé par l'utilisation de $O(d \log d)$ opérations et $O(d)$ stockages. Ici, on suppose que le produit de deux matrices Toeplitz triangulaires inférieures du type $(d \times d)$ peut être réalisé par l'utilisation de $C(m)d \log d$ opérations.

Pour inverser une matrice Toeplitz triangulaire inférieure non singulière du type $(d \times d)$,on a besoin d'effectuer $O(d \log d)$ opérations. dans ce cas, on suppose que l'inverse d'une matrice Toepliz triangulaire non singulière du type $(d \times d)$ peut être réalisé par l'utilisation de $2C(m)d \log d$ opérations.

4.4.3 Factorisation des éléments blocs

Soit $V(x)$ la matrice du type $(d \times d)$ suivante :

$$\begin{cases} V(x) & = \left[\rho(x)\ \rho'(x)\ \rho^{(2)}(x)\ \ldots\ \rho^{(d-1)}(x)\right] \\ \rho(x) & = \left[1\ x\ x^2\ \ldots\ x^{d-1}\right]^T \end{cases}$$

On observe que la première ligne de W par blocs est composée des matrices :

$$W_{1j} = V(x_j) \quad \text{pour } j = 1 : n$$

Il est important de signaler que $B(x)$ et $V(x)$ peut transformer en matrices Toeplitz comme l'indique les lemmes suivants :

Lemme (4.10) : Soit D la matrice diagonale du type $(d \times d)$ suivante :

$$D = diag(1\ 1!\ 2!\ 3!\ \ldots\ (d-1)!)$$

donc :

$$D^{-1}V(x) = L\left(\eta(x)\right),$$

où

$$\eta(x) = \left(1\ x\ \frac{x^2}{2!}\ \frac{x^3}{3!}\ \ldots\ \frac{x^{d-1}}{(d-1)!}\right)^T. \quad \Diamond$$

Ce lemme nous permet de dire que les éléments blocs W_{1j} de W peut être transformés en matrices Toeplitz triangulaire inférieure par une multiplication par une matrice diagonale. Alors $W_{11}^{-1} = L\left(\eta(x_1)\right)^{-1} D^{-1}$; qu'on utilisera dans le processus d'éliminations de Gauss ; on peut l'inverser par l'utilisation de $O(d \log d)$ opérations.

Le résultat suivant, nous donne deux propriétés importantes satisfaites par $B(x)$.

Lemme (4.11) : (a) soit $D = diag(1\ 1!\ 2!\ 3!\ \ldots\ (d-1)!)$. *Alors*

$$DB(x)D^{-1} = T(x) = R(xe_1 + e_2). \tag{4.40}$$

(b) Soit $\omega(x) \in C^{d-1}$ *une fonction et* $\sigma(x) = x\omega(x)$. *Donc*

$$\begin{aligned} & \left[\ \omega(x) \quad \omega^{(1)}(x) \quad \omega^{(2)}(x) \quad \cdots \quad \omega^{(d-1)}(x)\ \right] B(x) \\ = & \left[\ \sigma(x) \quad \sigma^{(1)}(x) \quad \sigma^{(2)}(x) \quad \cdots \quad \sigma^{(d-1)}(x)\ \right]. \quad \Diamond \end{aligned}$$

Démonstration : Le premier résultat est un résultat immédiat du calcul matriciel $DB(x)D^{-1}$.

pour démontrer le deuxième résultat, on utilise la formule $(x\omega(x))^{(i)} = i\omega^{(i-1)}(x) + x\omega^{(i)}(x)$ (on utilise la formule de Leibniz). d'autre part, le deuxième membre de l'équation est un vecteur de dimension d ; $a = (a_0\ a_1\ \ldots\ a_{d-1})$. Il est facile de voir que :

$$a_i = i\omega^{(i-1)}(x) + x\omega^{(i)}(x) = \sigma^{(i)}(x). \quad \blacklozenge$$

L'idée clé dans notre étude est de factoriser les éléments de la première ligne et la première colonne de $S^k(W)$ par blocs à partir de ses structures de déplacement dont on donne dans le théorème suivant :

Théorème (4.12) : Les éléments de la première ligne et la première colonne de $S^k(W)$ *par blocs vérifient les structures de déplacement suivantes :*

$$\begin{aligned} &(i)\ ZS^k(W)_{1j} - S^k(W)_{1j}B(x_{k+j})^{-1} &&= g_1\,[k]\,h_j\,[k]^T \\ &(ii)\ ZS^k(W)_{i1} - S^k(W)_{i1}B(x_{k+1})^{-1} &&= g_i\,[k]\,h_1\,[k]^T - e_1 e_d^T S^k(W)_{i-1,1} \end{aligned} \tag{4.41}$$

Dans la cas, où $k = 0$, $g_i\,[k]\,h_1\,[k]^T = 0$. $\Diamond$

Démonstration : On a dit précédemment que

$$S^k(W) \leftrightarrow [g[k], h[k]]^{n-k}.$$

est une matrice $(n-k)$ structurée de Vandermonde confluentes. Alors la matrice $S^k(W)$ est caractérisée par l'équation de déplacement :

$$ZS^k(W) - S^k(W)D_B^{-1} = g[k]\, h[k]^T$$

On peut obtenir ces formules à partir de la représentation par blocs de la structure de déplacement associée à $S^k(W)$ et par quelques multiplications adéquates.♦

Pour aboutir à notre but à savoir factoriser les éléments blocs $S^k(W)_{1j}$ et $S^k(W)_{i1}$ en termes des matrices Toeplitz triangulaires, on considère les deux matrices $F(x)$ et $G(x)$ du type $(d \times d)$ définies par :

$$ZF(x) - F(x)B(x)^{-1} = zw^T, \quad ZG(x) - G(x)B(x)^{-1} = e_1 w^T \tag{4.42}$$

tel que $z \neq 0$. Par une prémultiplication de la deuxième équation par $L(z)$ et en tenant compte que $ZL(z) = L(z)Z$ et $L(z)e_1 = z$, on implique que $F(x) = L(z)G(x)$.

D'autre part, il est important d'étudier le cas particulier où $G(x) = V(x)$ dans lequel :

$$\begin{array}{ll} w^T & = \Phi^T = \left[\begin{array}{ccccc} \phi(x) & \phi^{(1)}(x) & \phi^{(2)}(x) & \cdots & \phi^{(d-1)}(x) \end{array}\right] \\ \phi(x) & = -x^{-1}. \end{array} \tag{4.43}$$

Il est clair que

$$\phi^{(k)}(x) = (-1)^{k+1} k! x^{-(k+1)}$$

de laquelle, on peut déduire la relation suivante :

$$\Psi^T = \Phi^T D^{-1} = \left[(-1)^{k+1} x^{-(k+1)}\right]_{0 \leq k \leq (d-1)} \tag{4.44}$$

Par une postmultiplication de l'équation qui caractérise $G(x)$ par D^{-1} et en nous basant sur la relation (4.44), on obtient l'équation de déplacement de $G(x)D^{-1}$ et $V(x)D^{-1}$ respectivement :

$$\begin{aligned} ZG(x)D^{-1} - G(x)D^{-1}T(x)^{-1} &= e_1 w^T D^{-1} \\ ZV(x)D^{-1} - V(x)D^{-1}T(x)^{-1} &= e_1 \Psi^T \end{aligned}$$

où $T(x) = R(xe_1 + e_2)$ est une matrice Toeplitz triangulaire supérieure.

Le théorème suivant donne une relation entre $G(x)$ et $V(x)$.

Théorème (4.13) : Soit u un vecteur de dimension d, tel que :

$$w^T D^{-1} = \Psi^T R(u) \tag{4.45}$$

Alors

$$G(x) = V(x)D^{-1}R(u)D. \quad \lozenge$$

Démonstration : En postmultipliant la structure de déplacement de $V(x)D^{-1}$ par $R(u)$ et en tenant compte que

$$T(x)R(u) = R(u)T(x),$$

en trouvant directement le résultat. ⧫

D'après le théorème précédent, on obtient les factorisations suivantes :

$$\begin{aligned} G(x) &= DL\left(\eta(x)\right) D^{-1}R(u)D, \\ F(x) &= L(z)DL\left(\eta(x)\right) D^{-1}R(u)D \end{aligned}$$

où $F(x)$ et $G(x)$ définies dans (4.42).

Pour calculer u, il est facile de remarquer que :

$$\widetilde{u} = R(\Psi)^{-1} J D^{-1} w. \tag{4.46}$$

Telle que

$$J = \begin{bmatrix} e_n & e_{n-1} & e_{n-2} & \cdots & e_1 \end{bmatrix}$$

Remarque : $S^k(W)_{i1} = F_1 + F_2$, *telle que :*

$$ZF_1 - F_1 B(x)^{-1} = g_i\,[k]\,h_1\,[k]^T$$

et

$$ZF_2 - F_2 B(x)^{-1} = -e_1 e_d^T S^k(W)_{i-1,1}.$$

En appliquant ces résultats, on obtient les factorisations cherchées des éléments de la première ligne et la première colonne de $S^k(W)$ respectivement.

Théorème (4.14) : On a

$$\begin{aligned} S^k(W)_{1,j} &= L\,(g_1\,[k])\,DL\,(\eta(x_{k+j}))\,D^{-1}R(u)D \\ \widetilde{u} &= R(\Psi)^{-1}JD^{-1}h_j\,[k] \end{aligned} \tag{4.47}$$

et

$$\begin{aligned} S^k(W)_{i,1} &= L\,(g_i\,[k])\,DL\,(\eta(x_{k+1}))\,D^{-1}R(z)D - DL\,(\eta(x_{k+1}))\,D^{-1}R(w)D \\ \widetilde{z} &= R(\Psi)^{-1}JD^{-1}h_1\,[k]\,, \quad \widetilde{w} = R(\Psi)^{-1}JD^{-1}S^k(W)^T_{i-1,1}e_d. \quad \Diamond \end{aligned} \tag{4.48}$$

Démonstration : est une conséquence directe des factorisations précédentes de $F(x)$ et $G(x)$. ⧫

D'après cette factorisation, il est très clair qu'on peut réaliser la méthode d'élimination de Gauss par blocs en $O\left((d\log d)\,n^2\right)$ opérations parce qu'on peut factoriser les éléments blocs en terme de matrices Toeplitz triangulaires et matrices diagonales qu'on peut les multiplier par un vecteur en $O\,(d\log d)$ au lieu de $O\left(n^2\right)$.

Conclusion

Dans ce mémoire, nous avons présenté les notions de basas pour l'analyse d'erreur en précision finie, en introduisant les notions d'erreurs absolue et l'erreur relative, ainsi le modèle standard qui joue un rôle entrêmemment important dans l'arithmétique flottante.

Dans ce travaille, nous nous sommes intéressés au problématique : comment améliorer la précision d'un résultat calculé en arithmétique flottante, plus particulièrement dans la résolution d'un système linéaire.

La précision du résultat d'un calcule en précision finie dépend de deux facteurs : la stabilité numérique du problème et la précision de l'arithmétique utilisée (arithmétique flottante). Pour aboutir à notre objectif de résoudre un système linéaire structuré de Toeplitz ou de Vandermonde confluent, en faisant appel à la FFT comme méthode privilégiée pour améliorer la précision des calculs d'une part pour réduire la complexité du calcul de $O(n^2)$ opérations à $O(nlogn)$, alors, elle nous permet de réduire le temps d'exécution (vitesse de convergence), et de l'autre part pour réaliser une stabilité numérique.

Bibliographie

[1] Anne Bourlioux, Robert G.Owens. *MAT2412.Analyse Numerique1, 02/09/2011.*

[2] Anthony Mailho. *Mémoire de licence, les Nombres a virgule flottante 2006.*

[3] B.Pinçon, J-F.Scheid. *Notes de cours Mathématiques Numériques et Analyse de Données, 2007/2008.*

[4] F.Benabba. *Mémoire de Magister en Mathématiques, Arithmétique en virgule flottante et Méthodes de correction 2005/2006.*

[5] Franck Boyer. *Cours d'analyse Numérique-Agrégation Externe de Mathématiques,01/10/2011.*

[6] G.H.Golub, C.F.van Loan. *Matrix Computations, John Hopkins.Univ, press, Baltimore, Third edition,(1996).*

[7] J.senpau Roca-A.priou. *Cours Numération et codages, 09/2003.*

[8] Jocelyne Erhel, Nabil Nassif et Bernard Philippe. Cours de calcul matriciel et systèmes linéaires, 2004.

[9] L.Melkemi. *Structures de déplacement pour les matrices de Vandermonde p-confluentes.C.R.Acad.Sci.Paris Série 1(1999)621-926.*

[10] L.Melkemi, F.Rajah. *Block LU factorization of confluent Vandermonde matrices. Applied mathematics Lehers 23(2010) 747-750.*

[11] Manfred Gilli. *Méthodes Numériques, Université de Genève, 25/03/2006.*

[12] M.R.O'Donohoe. *Numerical Analysis 1.*

[13] Mohammed Said Belaid, Claude Michel et Michel Rueher. *Articles, Résolution de contraintes sur les nombres à virgule flottante par une approximation sur les nombres réels,2010.*

[14] Nicolas Louvet.Sylvain Veloso. *Thèse, Algorithmes compensés en arithmétique flottante, 27/11/2007.*

[15] Nicolas Louvet. *Notes du cours du 16/09/2011.*

[16] Phillippe Langlois. *Analyse d'erreur en précision finie, 16/12/2004.*

[17] Pr.Souad El Bernoussi. *Cours Méthodes Numériques et Programmation.*

[18] Sylain Chevillard. Thèse, *Evaluation efficace de fonctions numériques outils et exemples, Université de Lyon, 06/07/2009.*

[19] Université Mohammed V-Agdal. *Module Analyse Numérique 1,2003/2004.*

[20] Vincent Lefèvre-Paul Zimmermann. *Arithmétique flottante, 01/2004.*