Sur les méthodes rapides de résolution de systèmes de Toeplitz bandes

Marwa Dridi

THESE

THÈSE EN COTUTELLE

présentée et soutenue publiquement le 17 mai 2016
au Laboratoire de Mathématiques Pures et Appliquées Joseph Liouville (LMPA), Calais
pour obtenir le grade de docteur de

Université du Littoral Côte d'Opale-France
et
École Nationale d'Ingénieurs de Tunis-Tunisie
(Spécialité Mathématiques Appliquées)

par

DRIDI MARWA

Sur les méthodes rapides pour la résolution de systèmes de Toeplitz bandes

dirigée par : **BELHAJ SKANDER, MOAKHER MAHER** et **SALAM AHMED**

Membres du Jury

M. REICHEL LOTHAR,	Professeur, Université de Kent State	Rapporteur
M. MESSAOUDI ABDERRAHIM,	Professeur, Université Mohammed V de Rabat	Rapporteur
M. JBILOU KHALIDE,	Professeur, Université du Littoral Côte d'Opale	Président
M. CHEHAB JEAN-PAUL,	Professeur, Université de Picardie Jules Verne	Examinateur
M. SALAM AHMED,	Maître de Conférences, HDR, ULCO	Co-directeur
M. MOAKHER MAHER,	Professeur, École nationale d'ingénieurs de Tunis	Co-directeur

LMPA, Centre Universitaire de la Mi-Voix
50, rue Ferdinand Buisson
CS 80699 - 62228 Calais Cedex - France

THESE

Remerciements

Je tiens en premier lieu à remercier mes co-directeurs de thèse M. MOAKHER Maher et M. SALAM Ahmed pour leur encadrement, leur disponibilité et leur aide considérable tout au long de mes années de thèse.

Je remercie vivement mon co-encadrant M. BELHAJ Skander de m'avoir fait partager ses connaissances et son expérience. Tous mes remerciements pour sa confiance et son aide précieuse. Enfin, j'ai été extrêmement sensible à ses qualités humaines d'écoute et de compréhension tout au long de ce travail.

Je tiens à remercier M. REICHEL Lothar et M. MESSAOUDI Abderrahim pour avoir accepté d'être rapporteurs de mes travaux de thèse et pour leurs observations qui m'ont permis d'améliorer la qualité de ce mémoire. Je tiens à leur exprimer mes remerciements pour l'honneur qu'ils me font en participant à ce jury.

Mes sincère remerciements et ma gratitude vont aussi à M. JBILOU Khalide pour avoir accepté de juger ce travail et d'en présider le jury.

C'est également avec plaisir que je remercie M. CHEHAB Jean-Paul pour m'avoir fait l'honneur de faire partie de jury.

Un grand merci à tous les membres du LAMSIN & LMPA qui ont été toujours à mes cotés par leurs conseils et leur aide.
Je remercie tous mes amis et proches, qui ont contribué de près ou de loin à l'accomplissement de cette thèse.

Mes remerciements les plus profonds vont naturellement à tous les membres de ma famille, qui m'ont soutenu constamment durant toutes ces longues années d'études : Papa, Maman et mes

aimables frères. Merci pour vous tous pour votre encouragement, votre amour et le support moral que vous m'avez accordé. Que Dieu vous protège et vous donne chance, santé et longue vie.

Enfin, je ne me lasserais jamais de remercier mon fiancé qui a toujours été présent à mes côtés.

Je dédie ce travail à mes parents.

Table des matières

Table des figures

Liste des tableaux

List of Algorithms

Table de notations

Notation	Explication
$\mathbb{K}$	corps arbitraire
$\mathbb{R}$	corps réel
$\mathbb{C}$	corps complexe
w	$e^{2\mathrm{i}\pi/n}$
$\lceil x \rceil$	partie entière de n
$\log n$	$\lfloor \log_2 n \rfloor$
A^T	la transposée de A
A^*	la transposée conjuguée de A, ($A^* = \bar{A}^T$)
e_i	le $i^{\text{ème}}$ vecteur canonique
$I = I_n$	la matrice identité de taille $n \times n$
O	la matrice nulle
$D(v)$	la matrice diagonale avec $v_1, \ldots, v_n$ sur la diagonale
$K(M)$	nombre de conditionnement de la matrice M
$\Delta_{A,B}(M)$	le déplacement de type Stein
$O(n)$	notation de Landau
flop	opération en arithmétique flottante

Introduction

Les matrices de Toeplitz émergent dans beaucoup d'applications et ont été largement étudiées dans la littérature. Elles sont omniprésentes dans de très nombreux problèmes tels que le traitement de signal numérique (le filtrage numérique, la théorie d'estimation, le traitement de la parole, etc . . .), ainsi que le traitement d'images, la théorie de contrôle, les équations intégrales (convolution du type Volerra ou Fredholm), les polynômes orthogonaux, les équations aux dérivées partielles (elliptique ou parabolique), l'approximation de Padé, et d'autres domaines de l'analyse numérique. La structure est souvent due à l'utilisation d'un schéma d'approximation ou simplement à des propriétés inhérentes à la physique du problème initial.

A la structure des matrices s'ajoute parfois le problème de grande taille. En effet, les modèles mathématiques employés à présent ont une taille qui ne fait que croître à cause de la précision de calcul typiquement requise : une meilleure précision demande une discrétisation plus fine et donc des matrices de très grandes tailles. Malheureusement, les méthodes classiques comme par exemple la fameuse élimination de Gauss, qui sont de l'ordre de $O(n^3)$ flops ne sont plus les meilleures méthodes de choix dans ce cas. Par conséquent, la conception d'algorithmes efficaces, rapides et robustes, est donc cruciale.

Pour cela, pour une matrice possédant une structure particulière, il est conseillé d'utiliser toujours des algorithmes spécifiques, qui sont plus performants que les algorithmes standards. Parmi les structures, on peut citer les matrices symétriques, bandes, Toeplitz, circulantes, Hankel, Gauss, Vandermonde, etc . . ..

Dans cette thèse, nous allons traiter la résolution numérique des systèmes (triangulaires inférieures, bande et bande par blocs) de Toeplitz dans le but d'améliorer le coût de calcul, l'efficacité et la stabilité numérique des algorithmes.

Dans le premier chapitre, nous étudions quelques notions de stabilité numérique et d'erreur d'arrondis, en parallèle nous présentons les concepts d'algorithmes efficaces afin de choisir le meilleur selon plusieurs critères liés au calcul scientifique.

Le deuxième chapitre est une introduction générale aux matrices structurées. Nous introduisons le concept du rang de déplacement (F-déplacement) qui est un outil puissant pour traiter les ma-

trices structurées. Nous rappelons aussi les propriétés des matrices f-circulantes, l'algorithme de transformation de Fourier rapide (FFT), et la transformée en cosinus discrète (DCT). À la fin de ce chapitre, nous décrivons la multiplication rapide d'une matrice de Toeplitz par un vecteur et nous citons quelques algorithmes pour la résolution rapide d'un système de Toeplitz.

Dans le troisième chapitre, nous focalisons notre attention sur l'amélioration de l'algorithme de Bini [10] pour le calcul de l'inverse d'une matrice triangulaire de Toeplitz. L'idée est inspirée de l'article de Lin, Ching et Ng [55] pour une inversion approchée d'une matrice triangulaire de Toeplitz. Nous proposons un algorithme rapide [8] en se basant sur la notion d'interpolation polynomiale. Cet algorithme nécessitant uniquement deux FFT$(2n)$ est manifestement efficace par rapport à ses prédécesseurs. Des expérimentations numériques illustrent l'efficacité, le temps de calcul et la stabilité numérique de cette approche. Une étude d'erreur théorique a été aussi proposée.

Nous nous intéressons plus particulièrement, dans le chapitre quatre, aux méthodes directes de résolutions de systèmes linéaires de Toeplitz bandes. Une intention principale consiste à décrire et analyser certains algorithmes rapides, plus ou moins classiques, pour la résolution de systèmes d'équations linéaires avec la structure de Toeplitz bande. En effet la méthode de la réduction cyclique a prouvé son efficacité dans le cas symétrique définie positive et dans le cas non symétrique à diagonale dominante [15, 18], qui est souvent la méthode la plus rapide et la plus stable dans ces cas. Les auteurs de [35] ont comparé soigneusement la réduction cyclique de [15] avec une variante de la stratégie de "divide-and-conquer" qui est introduite dans [77] pour les matrices de Hessenberg par blocs et avec la modification de la réduction cyclique proposée dans [57]. Le coût théorique des trois algorithmes est presque identique lorsque la bande n'est pas trop grande. D'autres approches basées sur l'algorithme de [32] qui divise la matrice de Toeplitz bande par blocs en une somme d'une matrice circulante et d'une matrice de rang petit auquel les techniques efficaces de [44] sont ensuite appliquées.

L'approche dans [60] initialement proposée dans [37] pour le cas scalaire du système symétrique de Toeplitz bande combine la factorisation spectrale de la fonction génératrice associée à la matrice de Toeplitz avec l'utilisation de la formule de Sherman-Morrison-Woodbury.

La majorité de ces algorithmes devient coûteux si la taille de bande augmente, même ils échouent parfois. D'où, nous proposons une approche qui se base sur deux techniques : la première technique est basée sur le prolongement, initialement proposée par D. Bini, et M. Capavani [11]. Plus spécifiquement, l'idée de cette technique et d'augmenter la matrice de Toeplitz bande en une matrice triangulaire de Toeplitz afin de minimiser le coût de calcul. La deuxième technique est l'utilisation de l'inverse d'une matrice triangulaire inférieure de Toeplitz qui a été largement étudié dans la littérature et puis pour profiter des résultats développés dans le troisième chapitre de cette thèse.

Dans le cinquième chapitre, nous allons aborder le cas de la résolution d'un système de Toeplitz

bande par blocs Toeplitz bandes. Cette approche est basée sur la même technique utilisée dans le Chapitre 4 (extension de la matrice de Toeplitz bande en une matrice triangulaire inférieure de Toeplitz). Le coût de cette méthode est environ $O(m^2 n \log n + m^3 n + M^3)$. Des tests numériques ont été élaborés pour approuver notre contribution. Ceci étant primordial pour faire la connexion de nos algorithmes au traitement d'images, un domaine d'application phare en mathématiques appliquées.

Le dernier chapitre est consacré à la conclusion et à quelques perspectives.

CHAPITRE 1

MESURE DE LA QUALITÉ ARITHMÉTIQUE DES ALGORITHMES NUMÉRIQUES

1.1 Introduction

Bien que les performances des ordinateurs de nos jours, ça ne peut pas nous empêcher de chercher de plus en plus à raffiner les méthodes de calculs existantes. Ainsi l'évolution dans la capacité de calcul nous a permis d'avoir des problèmes de très grandes tailles. Pour cela, nous allons traiter des problèmes qui nécessitent de calcul énorme par rapport à la capacité de l'ordinateur. Pour les résoudre, il est conseillé d'utiliser à la fois des machines très puissantes et des algorithmes efficaces.
Généralement, la calculabilité n'indique pas la fiabilité. Lorsqu'on résout un problème donné par un ordinateur, il peut y avoir plusieurs algorithmes pour le résoudre. Afin d'utiliser le meilleur d'entre eux et pour déterminer lequel est préférable, il y a trois critères d'optimalité qui doivent se présenter dans un algorithme, pour trouver le bon choix. D'une part, l'algorithme doit s'exécuter le plus rapidement possible (complexité en temps/ temps de calcul). Ensuite, il doit avoir peu d'encombrement en espace mémoire (complexité en espace). Enfin, il doit fournir un résultat numérique précis. Malheureusement, il n'est pas toujours évident de rassembler ces trois critères dans un même algorithme.

1.2 Les normes

Les normes sont un outil indispensable dans l'algèbre linéaire numérique. Elles demeurent un instrument précieux pour l'analyse d'erreur. Pour cela, dans cette section, nous décrivons certaines de leurs propriétés les plus utiles et les plus intéressantes.

- La norme d'un vecteur est définie par la fonction $\|.\| : \mathbb{C}^n \longrightarrow \mathbb{R}^+$. Les trois normes les plus utilisées dans l'analyse numérique et dans le calcul numérique sont :

$$\|x\|_1 = \sum_{i=1}^{n} |x_i|,$$

$$\|x\|_2 = \left(\sum_{i=1}^{n} |x_i|^2 \right)^{\frac{1}{2}},$$

$$\|x\|_\infty = \max_{1 \leq i \leq n} |x_i|.$$

Une norme p d'un vecteur peut être représentée par l'inégalité suivante :

$$\|x\|_{p_2} \leq \|x\|_{p_1} \leq n^{(\frac{1}{p_1} - \frac{1}{p_2})} \|x\|_{p_2}, \quad p_1 \leq p_2, \quad p_1, p_2 \in \mathbb{N}$$

- La norme matricielle est définie par la fonction $\|.\| : \mathbb{C}^{m \times n} \longrightarrow \mathbb{R}^+$. Les normes 1, 2 et ∞ sont présentées comme suit :

$$\|A\|_1 = \max_{1 \leq j \leq n} \sum_{i=1}^{m} |a_{i,j}|,$$

$$\|A\|_\infty = \max_{1 \leq i \leq m} \sum_{j=1}^{n} |a_{i,j}|,$$

$$\|A\|_2 = (\rho(A^* A))^{\frac{1}{2}} = \sigma_{\max}(A), \quad \text{(norme spectrale)},$$

où

$$\rho(B) = \max\{\lambda / \det(B - \lambda I) = 0\}$$

et $\sigma_{\max}(A)$ est la plus grande valeur singulière de A.

Les normes 1, 2 satisfont certaines inégalités qui sont fréquentées dans le calcul d'analyse matricielle tel que :

$$\frac{1}{\sqrt{m}} \|A\|_1 \leq \|A\|_2 \leq \sqrt{n} \|A\|_1.$$

Pour plus de détails, voir [46, p.106-110].

1.3 "Backward error" et "forward error"

Nous rappelons ici quelques définitions de base sur la précision d'un algorithme, qui dépendent de l'exactitude avec laquelle les opérations ($+$, $-$, $*$, $/$) sont effectuées. Nous définissons aussi la notion de l'arithmétique en virgule flottante mesurée par u qui est la précision de la machine définie dans 1.3.1. Pour plus des détails, voir [46].

Supposons qu'une approximation $\hat{y}$ de $y = f(x)$ est calculée dans une arithmétique à u précision, où f est une fonction à une variable réelle. Comment devons-nous mesurer la "qualité"

de $\hat{y}$? Dans la plupart des calculs, nous estimons l'erreur relative, $E_{rel}(\hat{y}) \approx u$ mais cela ne peut pas être toujours atteint. Au lieu de se concentrer sur l'erreur relative de $\hat{y}$, nous pouvons demander, pour l'ensemble de ces données, avons-nous effectivement résolu notre problème ? Pour combien de Δx, avons-nous $\hat{y} = f(x + \Delta x)$? En général, il peut y avoir de nombreux Δx, de sorte que nous devrions demander la plus petite. La valeur $\|\Delta x\|$ (ou $\min \|\Delta x\|$), est appelée erreur inverse ("Backward error"). Les erreurs absolues et relatives de $\hat{y}$ sont appelées erreurs directes ("forward errors"), pour les distinguer de l'erreur inverse. La Figure 1.1 illustre ces concepts. Le processus de délimitation de l'erreur inverse d'une solution calculée est appelé

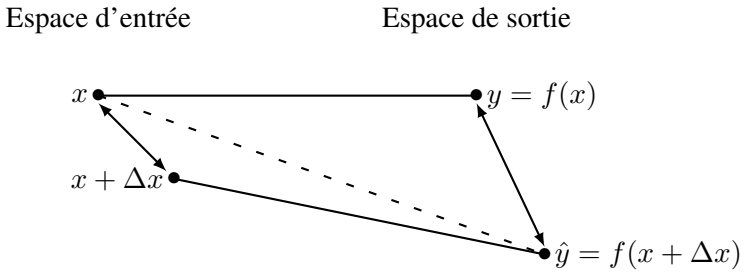


FIGURE 1.1 – "Backward" et "forward" pour $y = f(x)$. Ligne continue : "exacte", ligne pointillé : "calculée".

analyse inverse "backward analysis", et sa motivation est double. Tout d'abord, on interprète les erreurs d'arrondis comme des perturbations dans les données. Les données contiennent souvent des incertitudes dues aux mesures, des calculs précédents, ou des erreurs commises dans le stockage de numéro dans l'ordinateur. Si l'erreur inverse "backward error" n'est pas plus grande que ces incertitudes alors la solution calculée peut être difficilement critiquée, elle peut être la solution que nous cherchons, pour tout ce que nous savons. La seconde attraction de l'analyse d'erreur inverse "backward error analysis", c'est qu'elle réduit l'estimation de l'erreur directe à la théorie de perturbation.

1.3.1 Résultats classiques sur l'analyse d'erreurs

Après avoir défini un modèle pour les erreurs "backward error" et "forward error" dans la Figure 1.1. Nous allons maintenant appliquer ce modèle à certains calculs de base comme le calcul matriciel. En commençant par le produit scalaire, cette première application est assez simple pour mettre une brève analyse assez riche pour illustrer l'idée de l'analyse d'erreurs directes et l'analyse des erreurs inverses.

L'analyse des produits scalaires nous conduit immédiatement à des résultats pour le calcul de produit de matrice par un vecteur et le produit d'une matrice par une matrice.

Également dans ce chapitre nous déterminons des modèles d'études d'erreurs, dériver quelques divers résultats qui vont être utilisés dans les chapitres suivants.

Définition 1.3.1 *L'expression d'un point arithmétique en virgule flottante est notée $fl(.)$. L'opération arithmétique de base $op = +, -, *, /$ satisfait*

$$fl(x\ op\ y) = (x\ op\ y)(1+\delta), \qquad |\ \delta\ | \leqslant u \tag{1.1}$$

La quantité u signifie l'arrondi de l'unité (ou la précision de la machine).

Remarque 1.3.1 *Dans MATLAB, la valeur de u est donnée par la variable eps.*

Dans l'analyse qui suit, un chapeau représente une quantité calculée. Soit $x \in \mathbb{R}$, $\hat{x} = fl(x) = x(1+\delta)$ où $\hat{x}$ est une approximation de x. Nous rappelons maintenant quelques notions importantes, les conventions et les résultats de l'analyse de l'erreur d'arrondi développés dans [46]. Nous allons utiliser ces résultats dans ce qui suit pour obtenir une analyse des erreurs d'arrondi pour les algorithmes proposés tout au long de cette thèse.

Lemme 1.3.1 *Si $|\delta_i| \leq u$, et $\rho_i = \pm 1$ pour $i = 1, \ldots, n$ et $nu < 1$, alors*

$$\prod_{j=1}^{n}(1+\delta_i)^{\rho_i} = 1 + \theta_n,$$

avec

$$|\theta_n| < \frac{nu}{1-nu} := \gamma_n.$$

Preuve. Voir [46, pp. 63]. ■

Les notations θ_n et γ_n sont utilisées tout au long de cette section. On suppose que $nu < 1$ (cette condition est généralement satisfaite), chaque fois que nous écrivons γ_n.
Le Lemme 1.3.2 fournit quelques règles nécessaires à la manipulation des termes $1+\theta_k$ et γ_k impliqués dans le Lemme 1.3.1.

Lemme 1.3.2 *Pour tout entier positif k, θ_k désigne une quantité bornée en fonction de*

$$|\theta_k| \leq \gamma_k = \frac{ku}{1-ku}. \tag{1.2}$$

Les relations suivantes sont obtenues :

$$\begin{aligned}
(1+\theta_k)(1+\theta_j) &= (1+\theta_{k+j}),\\
\frac{1+\theta_k}{1+\theta_j} &= \begin{cases} 1+\theta_{k+j}, & j \leqslant k,\\ 1+\theta_{k+2j}, & j > k,\end{cases}\\
\gamma_k\gamma_j &\leq \gamma_{\min(k,j)}, \quad \textit{pour } \max(j,k)u \leqslant 1/2,\\
i\gamma_k &\leq \gamma_{ik},\\
\gamma_k + u &\leq \gamma_{k+1},\\
\gamma_k + \gamma_j + \gamma_k\gamma_j &\leq \gamma_{k+j}.
\end{aligned}$$

Preuve. Voir [46, pp. 73]. ■

Nous considérons le produit scalaire interne $s_n = x^T y$, tel que $x, y \in \mathbb{R}^n$.
Soit $s_i = x_1 y_1 + \ldots + x_i y_i$ et d'après le modèle standard (1.1), nous avons

$$\begin{aligned}
\hat{s}_1 &= fl(x_1 y_1) = x_1 y_1 (1 + \delta_1) \\
\hat{s}_2 &= fl(\hat{s}_1 + x_2 y_2) = (\hat{s}_1 + x_2 y_2 (1 + \delta_2))(1 + \delta_3) \\
&= (x_1 y_1 (1 + \delta_1) + x_2 y_2 (1 + \delta_2))(1 + \delta_3) \\
&= x_1 y_1 (1 + \delta_1)(1 + \delta_3) + x_2 y_2 (1 + \delta_2)(1 + \delta_3)
\end{aligned}$$

où $|\delta_i| < u$, $i = 1, \ldots, 3$. Afin de simplifier les expressions soit $1 + \delta_i = 1 \pm \delta$, alors nous pouvons écrire

$$\begin{aligned}
\hat{s}_3 &= fl(\hat{s}_2 + x_3 y_3) = (\hat{s}_2 + x_3 y_3 (1 \pm \delta))(1 \pm \delta) \\
&= (x_1 y_1 (1 \pm \delta)^2 + x_2 y_2 (1 \pm \delta)^2 + x_3 y_3 (1 \pm \delta))(1 \pm \delta) \\
&= x_1 y_1 (1 \pm \delta)^3 + x_2 y_2 (1 \pm \delta)^3 + x_3 y_3 (1 \pm \delta)^2.
\end{aligned}$$

Nous avons

$$\hat{s}_n = x_1 y_1 (1 \pm \delta)^n + x_2 y_2 (1 \pm \delta)^n + x_3 y_3 (1 \pm \delta)^{n-1} + \ldots + x_n y_n (1 \pm \delta)^2.$$

Pour simplifier de plus cette expression, il suffit d'appliquer le Lemme 1.3.1. La valeur calculée est donnée par

$$\hat{s}_n = x_1 y_1 (1 + \theta_n) + x_2 y_2 (1 + \theta_n^{'}) + x_3 y_3 (1 + \theta_{n-1}) + \cdots + x_n y_n (1 + \theta_2), \tag{1.3}$$

avec $|\theta_i| \leqslant \gamma_i \leqslant \gamma_n$, $i = 1, \ldots, n$, $\theta_n^{'} < \gamma_n$ et ainsi l'erreur relative est bornée par γ_n. Par conséquent, nous obtenons

$$fl(x^T y) = \hat{s}_n = (x + \Delta x)^T y = x^T (y + \Delta y), \quad |\Delta x| < \gamma_n |x|, \quad |\Delta y| < \gamma_n |y|. \tag{1.4}$$

Une erreur directe "forward error" provient de (1.4),

$$| \, x^T y - fl(x^T y) \, | \leqslant \gamma_n \, | \, x^T \, | \, | \, y \, | \tag{1.5}$$

où $|x|$ représente le vecteur de telle sorte que $|x|_i = |x_i|$ et les inégalités entre les vecteurs (ou matrices) sont faites composante par composante. L'équation (1.4) signifie que le calcul d'un produit scalaire est inversement stable "backward stable".

1.3.2 Multiplication de matrices

Compte tenue de l'analyse d'erreur pour le produit scalaire, il est simple d'analyser la multiplication matrice-vecteur et matrice-matrice.

Soit $A \in \mathbb{R}^{m\times n}$, $x \in \mathbb{R}^n$, et $y \in \mathbb{R}^m$ tel que $y = Ax$. Le vecteur y peut s'écrire comme m produit scalaire, $y_i = a_i^T x$, $i = 1, \ldots, m$ où a_i^T est la i-ème ligne de A. D'après (1.4), nous avons

$$\hat{y}_i = (a_i + \Delta a_i)^T x, \quad |\Delta a_i| < \gamma_n |a_i|.$$

Cela nous donne le résultat d'erreur inverse "backward error" suivant :

$$\hat{y} = (A + \Delta A)x, \qquad |\Delta A| \leqslant \gamma_n |A|. \tag{1.6}$$

Ce qui implique une erreur directe bornée

$$| \, y - \hat{y} \, | \leqslant \gamma_n \, | \, A \, | \, | \, x \, | \tag{1.7}$$

où $|A| = (|a_{i,j}|)$ qui sera fréquemment employé plus tard.
Alors, nous avons

$$\|y - \hat{y}\|_p \leqslant \gamma_n \|A\|_p \|x\|_p, \qquad p = 1, \infty \tag{1.8}$$

et pour la norme 2, on obtient

$$\|y - \hat{y}\|_2 \leqslant \min(m, n)^{\frac{1}{2}} \gamma_n \|A\|_2 \|x\|_2. \tag{1.9}$$

Lemme 1.3.3 *Pour $x, y \in \mathbb{C}$, les opérations arithmétiques de base calculées selon le modèle standard* (1.1) *satisfont*

$$\begin{aligned} fl(x \pm y) &= (x + y)(1 + \delta), & |\delta| &\leqslant u, \\ fl(xy) &= xy(1 + \delta), & |\delta| &\leqslant \sqrt{2}\gamma_2, \\ fl(x/y) &= (x/y)(1 + \delta), & |\delta| &\leqslant \sqrt{2}\gamma_4. \end{aligned}$$

Preuve. Tout au long de la preuve, δ_i désigne un nombre bornée par $|\delta_i| < u$, $x = a + \mathrm{i}\, b$, $y = c + \mathrm{i}\, d$ et on a

$$\begin{aligned} a \pm y &= a \pm c + \mathrm{i}\,(b \pm d), \\ xy &= ac - bd + \mathrm{i}\,(ad + bc), \\ x/y &= \frac{ac + bd}{c^2 + d^2} + \mathrm{i}\,\frac{bc - ad}{c^2 + d^2}. \end{aligned}$$

- Pour l'addition/ la soustraction :

$$\begin{aligned} fl(x + y) &= (a + c)(1 + \delta_1) + \mathrm{i}\,(b + d)(1 + \delta_2) \\ &= x + y + (a + c)\delta_1 + \mathrm{i}\,(b + d)\delta_2, \end{aligned}$$

alors

$$|fl(x+y)-(x+y)|^2 \leq (|a+c|^2+|b+d|^2)u^2 = (|x+y|u)^2,$$

d'où le résultat.

- La multiplication :

$$\begin{aligned} fl(xy) &= (ac(1+\delta_1)-bd(1+\delta_2))(1+\delta_3) \\ &\quad + \mathrm{i}\,(ad(1+\delta_4)+bc(1+\delta_5))(1+\delta_6)) \\ &= ac(1+\theta_2)-bd(1+\theta_2')+\mathrm{i}\,(ad(1+\theta_2'') \\ &\quad + bc(1+\theta_2'')) \\ &= xy+e, \end{aligned}$$

où

$$\begin{aligned} |e|^2 &\leq \gamma_2^2(|ac|+|bd|)^2+(|ad|+|bc|)^2) \\ &\leq 2\gamma_2^2(a^2+b^2)(c^2+d^2) \\ &= 2\gamma_2^2|xy|, \end{aligned}$$

d'où le résultat.

- La division :

$$\begin{aligned} fl(c^2+d^2) &= (c^2(1+\delta_1)+d^2(1+\delta_2))(1+\delta_3) \\ &= c^2(1+\theta_2)+d^2(1+\theta_2') \\ &= (c^2+d^2)(1+\theta_2''). \end{aligned}$$

Alors

$$\begin{aligned} fl(\mathrm{Re}(x/y)) &= \frac{(ac(1+\delta_4)+bd(1+\delta_5))(1+\delta_6)}{(c^2+d^2)(1+\theta_2'')} \\ &= \mathrm{Re}(x/y)+e_1, \end{aligned}$$

en utilisant le Lemme 1.3.2

$$|e_1| \leq \frac{|ac|+|bd|}{c^2+d^2}\gamma_4.$$

En appliquant la même démarche sur $fl(\mathrm{Im}(x/y)$, nous obtenons

$$\begin{aligned} |fl(x/y)-x/y|^2 &\leq \frac{(|ac|+|bc|)^2+(|bc|+|ad|)^2}{(c^2+d^2)^2}\gamma_4^2 \\ & 2\gamma_4^2|x/y|^2. \end{aligned}$$

d'ou le résultat.

Voir plus de détails [46, pp. 79]. ■

Lemme 1.3.4 *Si $X_j + \Delta X_j \in \mathbb{R}^{n\times n}$ satisfait $|\Delta X_j| \leqslant \delta_j |X_j|$ pour tous j, alors*

$$\Big| \prod_{j=0}^{p}(X_j + \Delta X_j) - \prod_{j=0}^{p} X_j \Big| \leqslant \Big(\prod_{j=0}^{p}(1+\delta_j) - 1 \Big) \prod_{j=0}^{p} \Big| X_j \Big|.$$

Preuve. Voir [46, Lemme 3.8]. ■

La FFT est mieux cernée (au moins par un numéricien) en l'interprétant comme l'application d'une factorisation intelligente de la matrice F_n de transformation de Fourier discrète (DFT).

Théorème 1.3.1 *Soit $\hat{y} = fl(F_n x)$, où $n = 2^t$, la valeur calculée en utilisant FFT dans la base binaire. Alors*

$$\frac{\|y - \hat{y}\|}{\|y\|} \leq \frac{t\eta}{1 - t\eta}, \qquad \eta := \mu + \gamma_4(\sqrt{2} + \mu) \tag{1.10}$$

et

$$\hat{y} = (F_n + \Delta F_n)x, \qquad \|\Delta F_n\|_2 \leqslant n^{1/2}\frac{t\eta}{1 - t\eta} =: f(n,u). \tag{1.11}$$

De plus, si $\hat{x} = fl(F_n^{-1}y)$, alors

$$\hat{x} = (F_n + \Delta F_n^{-1})y, \qquad \|\Delta F_n^{-1}\|_2 \leqslant n^{-1} f(n,u). \tag{1.12}$$

avec $\mu = cu$ où c est une constante, voir [82].

Preuve. Voir [46, pp. 453]. ■

Conclusion 1.3.2 *Dans ce chapitre, nous avons introduit les outils élémentaires pour le calcul de "backward error" et "forward error". Ces résultats vont être développés et bien évidemment utilisés tout au long de cette thèse.*

CHAPITRE 2

LES MATRICES STRUCTURÉES

2.1 Introduction

Les matrices structurées (telles que les matrices circulantes, Toeplitz, Hankel, Frobenius, Sylvester, Bézout, Vandermonde, et Cauchy) sont omniprésentes dans les calculs algébriques et numériques dans les diverses sciences, l'ingénierie, et les statistiques. Parmi leurs nombreux domaines d'application importants, nous choisissons des calculs fondamentaux, y compris la multiplication.

Dans ce chapitre, nous révélons la corrélation entre les calculs avec des polynômes et les matrices structurées, dans le but de réduire temps de calcul de la multiplication d'une matrice structurée par un vecteur. Nous définissons aussi les notions de la FFT (transformation de Fourier rapide) et de la DCT (transformée en cosinus discrète) qui sont des transformations très rapides permettant une mise en œuvre efficace.

Ensuite, nous introduisons une classe de matrices f-circulantes vue son importance dans le domaine de matrices structurées et qui va être présente dans le troisième chapitre. Nous étudions également quelques algorithmes de calcul de l'inverse bien connus pour les matrices de Toeplitz. La Table 2.1 représente les quatre classes les plus populaires des matrices structurées, et de nombreuses autres classes et sous-classes sont bien étudiées, comme les matrices de Bézout, matrices f-circulantes, les matrices bandes de Toeplitz, matrices par blocs : Toeplitz, Hankel et Vandermande, . . .

Ainsi, les matrices structurées :

1. peuvent être représentées avec seulement un petit nombre de paramètres (une matrice structurée de taille $n \times n$ dépend de $O(n)$ coefficients (à la place de n^2),
2. peuvent être rapidement multipliées par des vecteurs, en temps presque linéaire,
3. ont une étroite relation algorithmique à des calculs avec des polynômes en particulier à leur multiplication, division, interpolation, et évaluation multipoint,

4. sont associées avec les opérateurs de déplacement linéaire et peuvent être facilement récupérées à partir d'opérateurs F et l'image des matrices $F(M)$, appelée le déplacement de M, qui ont un rang r plus petit.

Les propriétés 1. et 2. sont bien précisées dans la Table 2.2, pour plus des détails sur la propriété 3. voir [21, 69]. Nous détaillons davantage la propriété 4. dans la section 2.3. Toutes ces propriétés rendent ces matrices particulièrement importantes pour les grands calculs matriciels.

Toeplitz $T = (t_{i-j})_{i,j=0}^{n-1}$ $\begin{pmatrix} t_0 & t_{-1} & \dots & t_{-n+1} \\ t_1 & t_0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & t_{-1} \\ t_{n-1} & \dots & t_1 & t_0 \end{pmatrix}$	Hankel $H = (h_{i+j})_{i,j=0}^{n-1}$ $\begin{pmatrix} h_0 & h_1 & \cdots & h_{n-1} \\ h_1 & h_2 & \iddots & h_n \\ \vdots & \iddots & \iddots & \vdots \\ h_{n-1} & h_n & \dots & h_{2n-2} \end{pmatrix}$
Vandermonde $V = (V_i^j)_{i,j=0}^{n-1}$ $\begin{pmatrix} 1 & v_0 & \cdots & v_0^{n-1} \\ 1 & v_1 & \cdots & v_1^{n-1} \\ \vdots & \vdots & & \vdots \\ 1 & v_{n-1} & \cdots & v_{n-1}^{n-1} \end{pmatrix}$	Cauchy $C = (\frac{1}{s_i - t_j})_{i,j=0}^{n-1}$ $\begin{pmatrix} \frac{1}{s_0-t_0} & \cdots & \frac{1}{s_0-t_{n-1}} \\ \frac{1}{s_1-t_0} & \cdots & \frac{1}{s_1-t_{n-1}} \\ \vdots & & \vdots \\ \frac{1}{s_{n-1}-t_0} & \cdots & \frac{1}{s_{n-1}-t_{n-1}} \end{pmatrix}$

TABLE 2.1 – Les quatre classes de matrices structurées

Les matrices de Toeplitz et de Hankel définies dans la Table 2.1 ci-dessus, sont les matrices les plus connues parmi les matrices structurées. Remarquons qu'une matrice de Hankel carrée d'ordre n est une matrice symétrique et que les produits HJ et JH d'une matrice de Hankel H carrée d'ordre n par la matrice de Hankel particulière J représentée dans la définition 2.2.1 sont des matrices de Toeplitz. Cette matrice de permutation d'ordre n permet de renverser l'ordre des n colonnes (respectivement des n lignes) d'une matrice lorsque celle-ci est multipliée à droite (respectivement à gauche) par la matrice J : c'est pourquoi on l'appelle matrice de renversement du fait qu'elle permet d'écrire de droite à gauche les colonnes que l'on lit de gauche à droite et inversement. Inversement, les produits TJ et JT d'une matrice de Toeplitz T carrée d'ordre n par la matrice J sont des matrices de Hankel.

Nature de la matrice M	Nombre de paramètres pour une matrice M de taille $m \times n$	Nombre de flops pour calculer Mv
Générale	mn	$2mn - n$
Toeplitz	$m + n - 1$	$O((m+n)\log(m+n))$
Hankel	$m + n - 1$	$O((m+n)\log(m+n))$
Vandemande	m	$O((m+n)\log^2(m+n))$
Cauchy	$m + n$	$O((m+n)\log^2(m+n))$

TABLE 2.2 – Les paramètres et le nombre de flops pour la représentation matricielle et la multiplication par un vecteur.

2.2 Préliminaires

Nous allons rappeler quelques notations et formules auxiliaires qui vont être utiles pour la suite.

On note e_i le i-ème vecteur de base canonique, son i-ème composante 1 et les autres composantes sont nulles. Ainsi, $e_1 = (1, \ldots, 0)^T$, et $\sum_{i=1}^{n} e_i = (1, 1, \ldots, 1)^T$.

Définition 2.2.1 *$D(v)$ est une matrice diagonale avec $v = (v_0, v_1, \ldots, v_{n-1})^T$*

$$D(v^T) = \mathrm{diag}((v_i)_{i=0}^{n-1}) = \begin{pmatrix} v_0 & & & \\ & v_1 & O & \\ & O & \ddots & \\ & & & v_{n-1} \end{pmatrix}. \tag{2.1}$$

I est la matrice identité de taille $n \times n$,

$$I = (e_1, \ldots, e_n) = \begin{pmatrix} 1 & 0 & \ldots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \ldots & 0 & 1 \end{pmatrix}. \tag{2.2}$$

J *est la matrice de taille $n \times n$ remplie avec des zéros, sauf sur l'antidiagonale,*

$$\mathrm{J} = (e_n, \ldots, e_1) = \begin{pmatrix} 0 & \cdots & 0 & 1 \\ \vdots & \iddots & \iddots & 0 \\ 0 & \iddots & \iddots & \vdots \\ 1 & 0 & \cdots & 0 \end{pmatrix}. \tag{2.3}$$

Théorème 2.2.1 *La matrice* J *est symétrique et orthogonale. Par conséquence* $\mathrm{J}^2 = I$, $\mathrm{J}\, v = (v_{n-1-i})_{i=0}^{n-1}$, *et* $\mathrm{J}\, D(v^T) J = D(\mathrm{J}\, v)$, *pour tout vecteur colonne* $v = (v_i)_{i=0}^{n-1}$.

Définition 2.2.2 *On définit Z_f l'unité de matrice f-circulante,*

$$Z_f = (e_2, \ldots, e_n, fe_1) = \begin{pmatrix} 0 & \cdots & \cdots & f \\ 1 & \ddots & & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 1 & 0 \end{pmatrix} \tag{2.4}$$

avec f est un scalaire quelconque, Si $f = 0$, Z_0 s'appelle matrice "shift" vers le bas, ainsi Z_0 s'écrit :

$$Z_0 = (e_2, \ldots, e_n, 0) = \begin{pmatrix} 0 & \cdots & \cdots & 0 \\ 1 & \ddots & & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 1 & 0 \end{pmatrix}. \tag{2.5}$$

Sans confusion on note Z_n à la place de Z_0.

Théorème 2.2.2 *Pour une matrice Z_e de taille $n \times n$ et un scalaire e, nous avons $Z_e^n = eI$, $JZ_eJ = Z_e^T$, $Z_eU = \sum_{i=0}^{n-1} u_i Z_e^i$ pour le vecteur colonne $U = (u_i)_{i=0}^{n-1}$. Pour $f \neq 0$, nous avons $Z_{1/f}^T = Z_f^{-1}$.*

Définition 2.2.3 *Soit M une matrice par blocs de taille $(p+q) \times (p+q)$, partitionnée comme suit :*

$$M = \begin{pmatrix} M_{1,1} & M_{1,2} \\ M_{2,1} & M_{2,2} \end{pmatrix}$$

où les blocs $M_{1,1}$, $M_{1,2}$, $M_{2,1}$, et $M_{2,2}$ sont des matrices de taille $p \times q$, $p \times p$, $q \times q$, et $q \times p$ respectivement, avec $M_{2,1}$ inversible. Le bloc S de taille $p \times p$

$$S = M_{1,2} - M_{1,1}M_{2,1}^{-1}M_{2,2}$$

est le complément de Schur du bloc $M_{2,1}$ de la matrice M.

Théorème 2.2.3 *(Formule de Sherman-Morrison-Woodbury [40])*
Soit $A \in \mathbb{C}^{n\times n}$, $G, H \in \mathbb{C}^{n\times r}$, $r \leq n$, avec A et $I_r + H^T A^{-1} G$ sont des matrices inversibles, alors

$$(A + GH^T)^{-1} = A^{-1} - A^{-1}G(I_r + H^T A^{-1} G)^{-1} H^T A^{-1}. \tag{2.6}$$

Définition 2.2.4 ***Matrice "Pencil" [40]***
Soit A et B deux matrices de tailles $n \times n$. L'ensemble de toutes les matrices de la forme $A - \lambda B$ avec $\lambda \in \mathbb{C}$ est dit "pencil". Les valeurs propres sont les éléments $\lambda(A, B)$ définis par

$$\lambda(A, B) = \{z \in \mathbb{C}, det(A - zB) = 0\}$$

si $\lambda \in \lambda(A, B)$ *et*

$$Ax = \lambda Bx \quad x \neq 0$$

alors x *est dénommée un vecteur propre de* $A - \lambda B$.
Notons que si $0 \neq \lambda \in \lambda(A, B)$ *alors* $(1/\lambda) \in \lambda(B, A)$. *Aussi si* B *est inversible alors* $\lambda(A, B) = \lambda(B^{-1}A, I) = \lambda(B^{-1}A)$.

2.3 Opérateurs de déplacement

L'étude moderne des matrices structurées a été en grande partie motivée par le papier proposé par Kailath [48] et en particulier par le concept du rang de déplacement. L'idée était de mesurer la structure d'une matrice par le rang de son déplacement.
Dans cette partie, nous allons introduire brièvement la notion de rang de déplacement. Soit A une matrice dans $\mathbb{K}^{n\times n}$ tel que $A = GH^T$, avec G, $H \in \mathbb{K}^{n\times d}$. Le couple (G, H) est appelé un générateur de la matrice A de longueur d. Le rang de A est la longueur minimale de d, et on le note $r = \text{rang}(A)$. D'où, si $r \ll n$ (r très petit devant n), alors A peut être représentée par $2nr$ coefficients. Cela nous permettra de réduire l'espace de stockage. En outre, pour calculer Av (multiplication de A par un vecteur v), nous avons besoin seulement de $4nr - n - r$ flops à la place $2n^2 - n$ flops.
Pour une matrice structurée A le plus facile c'est de chercher un opérateur $\mathcal{L}$ qui transforme A en matrice $\mathcal{L}(A)$ de petit rang, de telle sorte que nous pourrions facilement récupérer A à partir de son image $\mathcal{L}(A)$. Cela nous permettra de profiter de tous les avantages d'exploitation de matrices de petit rang, même si la matrice structurée de départ peut avoir le rang maximal.
Le concept des opérateurs de déplacement et le rang de déplacement introduits dans [21, 44, 48, 49], est un outil puissant pour traiter ce genre de matrices structurées. Nous rappelons ici les principales idées et résultats.

Définition 2.3.1 *Soit* $F : \mathbb{K}^{n\times n} \to \mathbb{K}^{n\times n}$ *un opérateur, soit* $A \in \mathbb{K}^{n\times n}$ *et soient* G, $H \in \mathbb{K}^{n\times d}$, *sont deux matrices telles que* $F(A) = GH^T$. *Alors* $r = \text{rang}(F(A))$, *le rang de la matrice* $F(A)$ *est appelée le* F-$rang$ *et le couple* (G, H) *est appelé* F-*générateur de* A *de longueur* d.

2.3.1 Opérations de base

Définition 2.3.2 *Soit* M *et* N *deux matrices qui sont appelées les matrices opérateurs. Nous définissons deux types d'opérateurs tels que*
$F(A) : \mathbb{K}^{n\times n} \to \mathbb{K}^{n\times n}$
opérateur de type Sylvester

$$F(A) = \nabla_{M,N}(A) = MA - AN, \tag{2.7}$$

opérateur de type Stein

$$F(A) = \Delta_{M,N}(A) = A - MAN. \tag{2.8}$$

Les opérateurs de ces deux types sont étroitement liés les uns aux autres.

Théorème 2.3.1 $\nabla_{M,N} = M^{-1}\Delta_{M^{-1},N}$ *si la matrice opérateur* M *est inversible, et* $\nabla_{M,N} = -\Delta_{M,N^{-1}}N$ *si la matrice opérateur* N *est inversible.*

Preuve. Voir [21, 69]. ■

Théorème 2.3.2 *Soient* A, B, M *et* $N \in \mathbb{K}^{n\times n}$, *nous avons :*

1. $\Delta_{M,N}(\alpha A + \beta B) = \alpha\Delta_{M,N}(A) + \beta\Delta_{M,N}(B)$,
2. $\Delta_{M,N}(A^T) = [\Delta_{N^T,M^T}(A)]^T$,
3. $\Delta_{M,N}(A^{-1}) = MA^{-1}\Delta_{M,N}(A)M^{-1}N$ *si* M *est inversible,*
4. $\Delta_{M,N}(A^{-1}) = A^{-1}N^{-1}\Delta_{N,M}(A)A^{-1}N$ *Si* N *est inversible,*
5. $\mathrm{rang}_{M,N}(A) = \mathrm{rang}_{N^T,M^T}(A)$,
6. $\mathrm{rang}_{M,N}((A^{-1})) = \mathrm{rang}_{N,M}(A)$, *avec* $\mathrm{rang}_{M,N}(A)$ *désigne le* $\mathrm{rang}(\Delta_{M,N}(A))$.

Preuve. Pour plus de détails voir [21, 69]. ■

2.3.2 Matrice de Toeplitz

Nous allons traiter les cas particuliers des opérateurs de déplacement de type Hankel et Toeplitz. Soit T une matrice de Toeplitz comme définie dans la Table 2.1, Z_ψ et Z_φ deux matrices définies par l'équation (2.4). On a

$$\Delta_{Z_\varphi,Z_\psi^T}(T) = \begin{pmatrix} t_0 - \psi\varphi t_0 & t_{-1} - \varphi t_{n-1} & \cdots & t_{-n+1} - \varphi t_1 \\ t_1 - \psi t_{-n+1} & & & \\ \vdots & & O & \\ t_{n-1} - \psi t_{-1} & & & \end{pmatrix}, \tag{2.9}$$

ainsi nous obtenons $\Delta_{Z_\varphi,Z_\psi^T}(T) \leq 2$. D'une manière générale pour toute matrice A de taille $n \times n$ qui vérifie $\mathrm{rang}(\Delta_{Z_\varphi,Z_\psi^T}(T)) \ll n$, elle est appelée une matrice de type Toeplitz.

Remarque 2.3.1 *L'inversion d'une matrice de Toeplitz n'est pas une matrice de Toeplitz (seulement dans le cas d'une matrice triangulaire de Toeplitz, voir Chapitre 3) mais c'est une matrice de type Toeplitz de rang de déplacement au plus* 2. *De plus, la multiplication de deux matrices de Toeplitz est une matrice de type Toeplitz avec un rang de déplacement au plus* 4.

Théorème 2.3.3 *Soit A une matrice de Toeplitz, pour G, $H \in \mathbb{K}^{n\times r}$ et φ, $\psi \in \mathbb{K}^*$, vérifiant :*

$$\Delta_{Z_\varphi, Z_\psi^T}(A) = GH^T = \sum_{i=1}^{r} g_i h_i^T$$

alors

$$A = \frac{1}{1-\varphi\psi} \sum_{i=1}^{r} C_\varphi(g_i) C_\psi^T(h_i) \tag{2.10}$$

où h_i, g_i, $i = 1, \ldots, r$ sont des vecteurs de dimension n, et $C_f(\boldsymbol{a})$ est la matrice f-circulante avec f égale à ψ ou φ définie par sa première colonne $\boldsymbol{a}$, voir (2.12).
En particulier, si

$$\Delta_{Z, Z^T}(A) = GH^T$$

alors

$$A = \sum_{i=1}^{r} L(g_i) L^T(h_i) \tag{2.11}$$

où $L(\boldsymbol{a})$ est une matrice triangulaire inférieure de Toeplitz définie par sa première colonne $\boldsymbol{a}$.

Preuve. Pour plus de détails, voir [71], et [21, théorème 2.11.2]. ■

Les résultats ci-dessus nous permettent de faire des opérations élémentaires entre les matrices de type Toeplitz à faible coût par l'intermédiaire de la FFT. En particulier, le calcul du produit de matrices qui ont un rang de déplacement égal à r nécessite $O(rn \log n)$ flops. L'inverse d'une matrice non singulière avec un rang de déplacement égal à r nécessite $O(r^2 n \log^2 n)$ flops, en utilisant un algorithme ultra-rapide comme l'algorithme de Bitmead Anderson [22].

Orthogonale représentation pour les matrices structurées

Pour une paire fixe d'un opérateur F et une matrice A, le mieux est de choisir un F-générateur orthogonal. Un F-générateur orthogonal est un F-générateur calculé à base de la SVD [1]. Autrement dit, on calcule d'abord la SVD du déplacement $F(A) = W$.
$W = U\Sigma^2 V^T$,
$U^*U = V^*V = I_\rho, \quad \Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_\rho)$,
$\sigma_1 \geq \ldots \geq \sigma_\rho > 0, \quad \rho = \mathrm{rang}(W)$,
où U et V sont des matrices de taille $n \times \rho$ et $\rho \times n$, respectivement. Et $\sigma_1^2, \ldots, \sigma_\rho^2$ sont les valeurs singulières de la matrice W, alors nous pouvons écrire $G = U\Sigma$, $H = V\Sigma$.
Le calcul d'un F-générateur orthogonal d'une matrice A de longueur minimale r est numériquement plus stable, voir [67,68]. Souvent une matrice structurée A est connue par son déplacement $F(A)$ de petit rang. En général, le calcul d'un générateur est une étape coûteuse qui demandera $O(n^3)$. D'où, l'idée d'intervenir le déplacement $F(A)$ qui joue le rôle de médiateur, qui baisse le

1. Décomposition en valeurs singulières

coût de calcul d'un générateur en $O(r^2 n)$ flops, pour plus de détails voir [21, Problème 2.2.11b]. La Table 2.3 représente quelques exemples de matrices d'opérateurs associés naturellement avec les matrices A. Dans la Table 2.1, les matrices dont le $F-\mathrm{rang}$ est petit pour l'opérateur F des lignes respectives Table 2.3 seront appelées les matrices du type Toeplitz, Hankel, Vandermonde, et Cauchy, respectivement.

Type de matrice	La paire (M, N) de l'opérateur $\Delta_{M,N}$	Le $\mathrm{rang}-\Delta_{M,N}$
Toeplitz	(Z_e, Z_f^T) ou (Z_e^T, Z_f)	au plus 2
Hankel	(Z_e, Z_f) ou $(Z - e^T, Z_f^T)$	au plus 2
Vandermonde	$(D(v), Z_e^T)$ ou $(D(v), Z_e)$	1
Cauchy	$(D^{-1}(s), D(t))$	1

TABLE 2.3 – Les matrices opérateurs M, N de l'opérateur $\Delta_{M,N}$ associé aux quatre classes de base de matrices structurées.

2.3.3 FFT

La transformée de Fourier rapide, FFT (Fast Fourier Transform), est l'un des algorithmes dont la publication a provoqué une véritable révolution dans le domaine du calcul, comme la décrit Charles Van Loan dans son livre [82]. Le célèbre algorithme FFT, est publié par James Cooley et John Tuckey en 1965. Ils ont montré que le calcul de la DFT, qui exige $O(n^2)$ flops peut être calculé par l'algorithme de la FFT en $O(n \log n)$ flops. L'importance de la FFT provient du fait qu'on peut à la fois avoir l'efficacité et la rapidité du calcul. Pour plus de détails sur la FFT, voir [43].
Pour introduire la FFT, on commence par donner quelques définitions :

Définition 2.3.3 *La matrice de Fourier d'ordre n est la matrice F_n suivante :*

$$F_n = \sqrt{\frac{1}{n}}(w^{ij})_{0\le i,j\le n-1} = \sqrt{\frac{1}{n}}\begin{pmatrix} 1 & \cdots & \cdots & 1 \\ 1 & w & \cdots & w^{(n-1)} \\ \vdots & \vdots & & \vdots \\ 1 & w^{(n-1)} & \cdots & w^{(n-1)^2} \end{pmatrix}$$

avec $w = \exp(2\mathrm{i}\pi/n)$, w est une racine primitive n-ème de l'unité, c-à-d $w^n = 1$.

Définition 2.3.4 *F_n est une matrice de la transformation de Fourier discrète qui est un cas spécial de matrices de Vandermande (voir la Table 2.1). La transformation de Fourier discrète d'un vecteur p de dimension n est le vecteur $DFT(p) = F_n p$.*

Proposition 2.3.1 *La matrice F_n est unitaire.*

Preuve. Soit $F_n F_n^* = (b_{ij})_{0\leq i,j\leq n-1}$. Comme $F_n^* = \frac{1}{\sqrt{n}}(w^{-ij})_{0\leq i,j\leq n-1}$ alors on a

$$b_{ij} = \frac{1}{n}\sum_{k=0}^{n-1} w^{ik}w^{-kj} = \frac{1-w^{(i-j)n}}{1-w^{i-j}} = \begin{cases} 1 & \text{si} \quad i = j \mod n \\ 0 & \text{si} \quad i \neq j \mod n \end{cases}.$$

■

Proposition 2.3.2 *(Algorithme FFT)*
Si $n = 2^h$ alors on peut calculer la $DFT(p)$ introduite dans la définition 2.3.4 en $\frac{3}{2}n\log n$ flops.

Preuve. Le problème est résolu en appliquant la transformée de Fourier rapide (FFT) qui est un outil important d'algorithmes récursifs sur la base de méthode diviser-pour-régner ("divide and conquer").
Soient les données suivantes $p_0, p_1, \ldots, p_{n-1}$, $1, w, \ldots, w^{n-1}$, n est pair, $y = x^2$ et notons $t_A(n)$ le nombre minimum des opérations suffisantes pour la DFT. Nous pouvons écrire :

$$\begin{aligned} p(x) &= (p_0 + p_2x^2 + \ldots + p_{n-2}x^{(n-2)}) + x(p_1 + p_3x^2 + \ldots + p_nx^{(n-2)}) \\ &= q(x^2) + xs(x^2) \\ &= q(y) + xs(y). \end{aligned}$$

Le polynôme $q(y) = p_0 + p_2y + \ldots + p_{n-2}y^{(\frac{n-2}{2})}$ et $s(y) = p_1 + p_3y + \ldots + p_ny^{(\frac{n-2}{2})}$ ont des degrés au plus $\frac{n-2}{2}$.
Par conséquent, pour évaluer $p(x)$ avec $x = w^k$ pour $k = 0, 1 \ldots, n-1$, il suffit de calculer $q(y)$ et $s(y)$ où $y = (w^2)^h$ et puis $q((w^2)^h) + xs((w^2)^h)$ pour $x = w^h$. En remarquant que w^2 est racine $\frac{n}{2}$ de l'unité, il existe seulement $\frac{n}{2}$ valeurs distinctes parmi les entrées de w^2. Ce qui nous permettons de réduire le nombre d'opération en $2DFT$ de taille $\frac{n}{2}$, $n/2$ multiplications (50% des multiplications peuvent être éleminer, car $x_i = -x_{i+\frac{n}{2}}$ pour tout i) et n additions.
Ainsi,

$$t_A(n) \leq 2t_A(\frac{n}{2}) + \frac{2}{3}n.$$

En appliquant récursivement cette estimation de n, $\frac{n}{2}$, $\frac{n}{4}$, ..., on a

$$\begin{aligned} t_A(n) &\leq 2t_A\left(\frac{n}{2}\right) + 1.5n \\ &\leq 2\left(2t_A\left(\frac{n}{2^2}\right) + 1.5\frac{n}{2}\right) + 1.5n \\ &\vdots \\ &\leq 2^h t_A\left(\frac{n}{2^h}\right) + 1.5hn \\ &\leq 2^h t_A(1) + 1.5hn. \end{aligned}$$

En remarquant que $t_A(1) = 0$ (car elle ne demande aucune opération) et comme $n = 2^h \Rightarrow \log_2 n = h$ d'où $t_A(n) = \frac{3}{2}n\log_2 n$. ■

2.3.4 DCT

La transformée en cosinus discrète, DCT (Discrete Cosine Transform), est une transformation similaire à la transformée de Fourier discrète (DFT). Le noyau de projection est un cosinus et génère donc des coefficients réels, contrairement à la DFT, dont le noyau est une exponentielle complexe et qui génère donc des coefficients complexes. On peut cependant exprimer la DCT en fonction de DFT, Voir [30] pour plus de détails sur la DCT.

La variante la plus courante de la transformée en cosinus discret est de Type-II, souvent simplement appelée la "DCT" qui est deux fois plus rapide que la FFT. Son inverse, qui correspond au type-III est souvent simplement appelée "IDCT".

La DCT est une fonction linéaire inversible de $\mathbb{R}^n \longrightarrow \mathbb{R}^n$ ou de manière équivalente une matrice carrée $n \times n$ inversible. Il existe plusieurs légères variantes de la DCT. Voici les quatre types les plus connus.

	T_Q(Transformée discrète)	T_Q^{-1}(Transformée inverse)
DCT-I	$C_n^I = \sqrt{\frac{2}{n-1}}(\mu_k \mu_{n-1-k} \mu_{n-1-j} \cos \frac{kj\pi}{n-1})_{k,j=0}^{n-1}$	$(C_n^I)^T = C_n^I$
DCT-II	$C_n^{II} = \sqrt{\frac{2}{n}}(\mu_k \cos \frac{(2j+1)k\pi}{2n})_{k,j=0}^{n-1}$	$(C_n^{II})^T = C_n^{III}$
DCT-III	$C_n^{III} = \sqrt{\frac{2}{n}}(\mu_j \cos \frac{(2k+1)j\pi}{2n})_{k,j=0}^{n-1}$	$(C_n^{III})^T = C_n^{II}$
DCT-IV	$C_n^{IV} = \sqrt{\frac{2}{n}}(\cos \frac{(2k+1)(2j+1)\pi}{4n})_{k,j=0}^{n-1}$	$(C_n^{IV})^T = C_n^{IV}$

TABLE 2.4 – $\mu_0 = \mu_n = 1/\sqrt{2},\ \mu_i = 1$ pour $0 < i < n$.

2.4 Matrice f-circulante, et diagonalisation par la DFT

Définition 2.4.1 *Chaque matrice f-circulante est définie par sa première colonne $v = (v_i)_{i=0}^{n-1}$ et par un scalaire f.*

$$Z_f(v) = \sum_{i=0}^{n-1} v_i Z_f^i = \begin{pmatrix} v_0 & fv_{n-1} & \dots & fv_1 \\ v_1 & v_0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & fv_{n-1} \\ v_{n-1} & \dots & v_1 & v_0 \end{pmatrix}. \tag{2.12}$$

Les matrices circulante, anti-circulante, et triangulaire inférieure de Toeplitz sont trois sous-classes de matrices f-circulantes où $f = 1$, $f = -1$, et $f = 0$, respectivement.

Lemme 2.4.1 *Une matrice circulante Z_1 donnée par la définition 2.2.2 est diagonalisable par la matrice de Fourier F :*

$$Z_1 = F_n^* D_{n,w} F_n \tag{2.13}$$

où $D_{n,w} = \mathrm{diag}(1, w, w^2, \ldots, w^{n-1})$, F_n [2] *définie en 2.3.3.*

Preuve. La matrice Z_1 est diagonalisable sur $\mathbb{C}$ et a pour valeurs propres des racines n-ème de l'unité. Soit $w = e^{\frac{2i\pi}{n}}$ la racine primitive de l'unité.

Nous pouvons vérifier que $X_k = \begin{pmatrix} 1 \\ w^k \\ w^{2k} \\ \vdots \\ w^{(n-1)k} \end{pmatrix}$ est le vecteur propre à gauche de Z_1 associé à la valeur propre w^k(en effet $X_k^T Z_1 = (w^k, \ldots, 1) = w^k(1, \ldots, w^{(n-1)k}) = w^k X_k^T$).

On a donc exhibé, pour k allant de 0 à $n-1$, une famille de n vecteurs propres associés à des valeurs propres distinctes, alors :

$$Z_1 = Q_n D_{n,w} P_n^T$$

où

$$D_{n,w} = \begin{pmatrix} 1 & & & \\ & w & & \\ & & \ddots & \\ & & & w^{(n-1)} \end{pmatrix}, \quad P_n = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & w & & w^{(n-1)} \\ \vdots & \vdots & & \vdots \\ 1 & w^{(n-1)} & \cdots & w^{(n-1)^2} \end{pmatrix}.$$

Comme $P_n^T = \sqrt{n} F_n$ (défini en 2.3.3) et d'après la Proposition 2.3.1, $Q_n = \frac{1}{\sqrt{n}} F_n^*$. D'où $Z_1 = F_n^* D_{n,w} F_n$. ■

Théorème 2.4.1 *Nous avons :*

$$Z_1(v) = F_n^* D(\sqrt{n} F_n v) F_n \tag{2.14}$$

Plus généralement, pour tout $f \neq 0$, nous avons :

$$Z_{f^n}(v) = (L_f)^{-1} D(\sqrt{n} L_f v) L_f \tag{2.15}$$

où

$$L_f = F_n D_f^n, \; D_f^n = \mathrm{diag}(1, f, \ldots, f^{n-1})$$

Preuve.(du théorème 2.4.1)

2. Matrice de Fourier

L'équation (2.14) on a :

$$\begin{aligned} Z_1(v) &= \sum_{j=0}^{n-1} v_j Z_1^j \\ &= \sum_{j=0}^{n-1} v_j F_n^* D_{n,w} F_n \quad \text{d'après lemme 2.4.1} \\ &= F_n^* (\sum_{j=0}^{n-1} v_j D_{n,w}) F_n \\ &= F_n^* \operatorname{diag}(\sqrt{n} F_n v) F_n. \end{aligned}$$

L'équation (2.15) on peut voir simplement que :

$$D_f^n Z_{f^n}(v)(D_f^n)^{-1} = \sum_{j=0}^{n-1} (f^j v_j) Z_1^j$$

d'où

$$\begin{aligned} D_f^n Z_{f^n}(v)(D_f^n)^{-1} &= Z_1(\tilde{v}) \\ &= F_n^* D_n(\tilde{v}) F_n \quad \text{d'après l'équation 2.14} \end{aligned}$$

avec $\tilde{v} = (f^j v_j)_{j=0}^{n-1}$. ■

Corollaire 2.4.1 *L'inverse d'une matrice f-circulante est une matrice f-circulante tel que :*

$$Z_1^{-1}(v) = Z_1(\check{v}) \tag{2.16}$$

avec $\check{v} = F_n^* \mathring{v}$. De plus, $\mathring{v} = (\frac{1}{\grave{v}_0}, \ldots, \frac{1}{\grave{v}_{n-1}})$ et $\grave{v} = F_n v$.

$$Z_f^{-1}(v) = Z_f(\bar{v}) \tag{2.17}$$

avec $\bar{v} = L_\varrho^* \ddot{v}$, $\ddot{v} = (\frac{1}{\tilde{v}_0}, \ldots, \frac{1}{\tilde{v}_{n-1}})$ et $\tilde{v} = L_\varrho v$, $\forall \varrho \in \mathbb{K}$ qui vérifie $\varrho^n = f$.

Preuve. On a d'après l'équation (2.14) $Z_1(v) = F_n^* D(\sqrt{n} F_n v) F_n$. D'où,

$$\begin{aligned} Z_1^{-1}(v) &= F_n^* D^{-1}(\sqrt{n} F_n v) F_n \\ &= F_n^* D(\sqrt{n} F_n F_n^* \mathring{v}) F_n \\ &= F^* D(\sqrt{n} F_n \check{v}) F_n \\ &= Z_1(\check{v}) \end{aligned}$$

De même pour l'équation (2.17). ■

Corollaire 2.4.2 *L'inversion d'une matrice f-circulante de taille $n \times n$ nécessite $O(n \log(n))$ flops.*

Corollaire 2.4.3 *La multiplication d'une matrice f-circulante, de taille $n \times n$ par un vecteur coûte $O(n \log n)$ flops.*

2.5 Multiplication et inversion rapides des matrices de Toeplitz

2.5.1 Multiplication rapide

Dans cette partie, nous révélons la corrélation entre les calculs avec des polynômes et des matrices structurées plus précisément les matrices de Toeplitz. Un algorithme ultra-rapide pour la multiplication entre une matrice de Toeplitz et un vecteur colonne va être présenté.

Proposition 2.5.1 *Soit T une matrice de Toeplitz de taille $n \times n$ et v un vecteur colonne de longueur n. Alors, le produit d'un matice-vecteur Tv est en $O(n\log(n))$ flops.*

Preuve. La multiplication d'une matrice de Toeplitz T par un vecteur v peut s'écrire sous la forme d'un produit de deux polynômes $T(x) = \sum_{i=-n+1}^{n-1} t_i x^{i+n-1}$ et $v(x) = \sum_{i=0}^{n-1} v_i x^i$:

$$\left(\sum_{i=-n+1}^{n-1} t_i x^{i+n-1}\right)\left(\sum_{i=0}^{n-1} v_i x^i\right) = \sum_{i=0}^{3n-3} p_i x^i. \tag{2.18}$$

Ainsi L'équation (2.18) est équivalente à l'écriture matricielle suivante :

$$\begin{pmatrix} t_{-n+1} & & & 0 \\ \vdots & \ddots & & \\ t_0 & & \ddots & \\ \vdots & \ddots & & t_{-n+1} \\ t_{n-1} & & \ddots & \vdots \\ & \ddots & & t_0 \\ & & \ddots & \vdots \\ 0 & & & t_{n-1} \end{pmatrix} \begin{pmatrix} v_0 \\ v_1 \\ \vdots \\ \vdots \\ v_{n-2} \\ v_{n-1} \end{pmatrix} = \begin{pmatrix} p_0 \\ \vdots \\ \vdots \\ p_n \\ \vdots \\ \vdots \\ p_{3n-2} \\ p_{3n-3}. \end{pmatrix} \tag{2.19}$$

Donc, pour calculer le vecteur Tv, il suffit de récupérer le vecteur $(p_{n-1}, \ldots, p_{2n-2})^T$ de taille n. D'après la multiplication rapide de deux polynômes via FFT, la complexité du produit Tv est en $O(M(n)) = O(n\log n)$ ops. Pour plus de détails, voir [1, 19]. ■

Remarque 2.5.1 *Une conséquence de cette proposition est que la multiplication d'une matrice de Hankel par un vecteur se fait en $O(n\log(n))$ flops.*

2.5.2 Les méthodes rapides et ultra-rapides pour la résolution d'un système de Toeplitz

Le développement pour résoudre un système de Toeplitz remonte aux travaux de Levinson [85] pour les problèmes de Weiner de filtrage et les travaux ultérieurs par Durin [34] pour la résolution de l'équation de Yule-Walker de la théorie de prédiction, suivi par le travail de

Trench [80] pour la construction de l'inverse de matrice de Toeplitz de la solution d'un ensemble d'équation de Yule-Walker. Il y a plusieurs algorithmes rapides en $O(n^2)$, et ultra-rapide en $O(n\log^2 n)$ direct et/ou des méthodes itératives pour la résolution d'un système d'équations de Toeplitz [63].
Dans cette section, nous détaillons deux types d'algorithmes directs de résolution d'un système de Toeplitz : l'algorithme de type Levinson et de type Schur.

Algorithme de Levinson

L'algorithme de Levinson est un algorithme qui nécessite $O(n^2)$, dont le principe est la décomposition LU de la matrice de Toeplitz. Soit le système $T_n x = b$, avec $b = (b_1, \ldots, b_n)^T$, $x = (x_1, \ldots, x_n)^T$ avec T_n une matrice de Toeplitz définie dans la Table 2.1. Et à chaque étape on cherche x_{k+1} à partir de x_k où x_k est une solution du système $T_k x_k$, $B_k = (b_1, \ldots, b_k)^T$, est T_k est la sous-matrice principale de T_n de taille $k \times k$.

$$T_{k+1} = \begin{pmatrix} T_k & \hat{V}_k \\ \hat{W}_k & t_0 \end{pmatrix} = \begin{pmatrix} I_k & 0 \\ \hat{W}_k^T T_k^{-1} & 1 \end{pmatrix} \begin{pmatrix} T_k & \hat{V}_k \\ 0 & s_k \end{pmatrix}, \tag{2.20}$$

où $V_k = (t_{-1}, \ldots, t_{-k})$, $W_k = (t_1, \ldots, t_k)^T$, $\hat{V}_k = JV_k^T$, $\hat{W}_k = W_k^T J$, $Y_k = V_k^T T_k^{-1}$ et $Z_k = T_k^{-1} W_k$ avec J la matrice définie par (2.3). De plus $s_k = t_0 - W_k^T J T_k^{-1} J V_k = t_0 - W_k^T T_k^{-T} V_k = t_0 - W_k^T (V_k^T T_k^{-1})^T = t_0 - (Y_k W_k)^T$. Soit X_{k+1} comme suit

$$X_{k+1} = \begin{pmatrix} \nu_k \\ x_{k+1} \end{pmatrix}. \tag{2.21}$$

Multiplions (2.20) à gauche par l'inverse du facteur triangulaire et en appliquant l'équation $T_{k+1} X_{K+1} = B_{k+1}$

$$\begin{pmatrix} I_k & 0 \\ \hat{W}_k^T T_k^{-1} & 1 \end{pmatrix} \begin{pmatrix} \nu_k \\ x_{k+1} \end{pmatrix} = \begin{pmatrix} T_k & \hat{V}_k \\ 0 & s_k \end{pmatrix} \tag{2.22}$$

ce qui est équivalent au système

$$\begin{cases} s_k x_{k+1} = -\hat{W}_k^T X_k + b_{k+1} \\ T_k \nu_k + \hat{V}_k x_{k+1} = B_{k+1} \end{cases} \tag{2.23}$$

La résolution du système nous donne la solution X_{k+1} en fonction de X_k

$$X_{k+1} = \begin{pmatrix} X_k - x_{k+1} \hat{Y}_k \\ x_{k+1} \end{pmatrix}, \tag{2.24}$$

où

$$x_{k+1} = \frac{-\hat{W}_k^T X_k + b_{k+1}}{s_k} \text{ et } X_1 = x_1 = \frac{b_1}{t_0}.$$

Il reste à calculer Y_k, Z_k, et s_k.
Soit

$$Y_{k+1}^T = \begin{pmatrix} \mu_k \\ y_{k+1} \end{pmatrix} = V^T T_{k+1}^{-1}, \tag{2.25}$$

En multipliant (2.20) à gauche par (μ_k^T, y_{k+1}), ce qui donne

$$Y_{k+1}^T = \begin{pmatrix} Y_k^T \mu - y_{k+1} \hat{Z}_k^T \\ y_{k+1} \end{pmatrix} \tag{2.26}$$

où

$$y_{k+1} = \frac{-Y_k^T \hat{V}_k - t_{-k-1}}{s_k} \text{ et } Y_1 = y_1 = \frac{t_{-1}}{t_0}.$$

Un calcul similaire nous donne $z_{k+1} = \dfrac{-\hat{W}_k^T Z_k + t_{k+1}}{s_k}$, et $Z_1 = \dfrac{t_1}{t_0}$. De plus, et on peut calculer s_{k+1} récursivement à partir de s_k :

$$s_{k+1} = (1 + z_{k+1} y_{k+1}) s_k.$$

Algorithme de Schur

Soit $Tx = b$ un système de Toeplitz, où T est une matrice de Toeplitz symétrique définie positive. En utilisant l'équation de déplacement d'une matrice de Toeplitz T

$$T - ZTZ^T = R\Sigma R^T$$

où

$$R^T = \frac{1}{\sqrt{t_0}} \begin{pmatrix} t_0 & t_1 & \dots & t_{n-1} \\ 0 & t_1 & \dots & t_{n-1} \end{pmatrix},$$

$$\Sigma = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

Pour toute matrice H vérifie l'équation $H\Sigma H^T = \Sigma$, nous avons

$$R\Sigma R^T = (RH)\Sigma(RH)^T.$$

Des matrices H qui satisfaisant l'équation $H\Sigma H^T = \Sigma$, sont notées Σ-unitaire. Toute matrice Σ-unitaire a la forme

$$H = \frac{1}{\sqrt{1 + |\rho|^2}} \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix} \begin{bmatrix} 1 & \bar{\rho} \\ \rho & 1 \end{bmatrix},$$

où $|a| = |b| = 1$, c-à-d les matrices Σ-unitaires sont le produit d'une matrice de rotation hyperbolique par une matrice diagonale unitaire.
Cet algorithme nécessite $O(n^2)$ flops pour la décomposition de T, et le temps de l'algorithme de Schur ultra-rapides, est de complexités en $O(n \log^2 n)$ et $O(n \log^3 n)$ développés, dans [78] et [3].

Remarque 2.5.2 *(Instabilité)*
La propriété de stabilité de méthodes directes pour les systèmes de Toeplitz symétriques définies positives a été discutée par Sweet (1984), Buch (1985), Cybenko (1987), Sweet (1993) and Bojanczyk et al. (1995).
Ces deux derniers algorithmes sont numériquement instables puisqu'ils ne fonctionnent que lorsque les sous-matrices principales de la matrice T sont non singulières. Pas mal de références dans les quelles mentionnent des solutions pour résoudre ce problème dont l'idée se base sur la vérification du conditionnement dans chaque étape et de dépasser les étapes instables, voir [6, 29].

2.6 Matrices de Toeplitz bandes

Une matrice de Toeplitz bande s'écrit sous la forme suivante :

$$T_n = \begin{pmatrix} t_0 & t_{-1} & \cdots & t_{-m_r} & & & \\ t_1 & t_0 & t_{-1} & & \ddots & & \\ \vdots & \ddots & \ddots & \ddots & & \ddots & \\ t_{m_c} & & t_1 & t_0 & t_{-1} & & t_{-m_r} \\ & \ddots & & \ddots & \ddots & \ddots & \vdots \\ & & \ddots & & \ddots & \ddots & t_{-1} \\ & & & t_{m_c} & \cdots & t_1 & t_0 \end{pmatrix} \tag{2.27}$$

2.6.1 Conditionnement de matrices de Toeplitz bandes

Plusieurs articles ont été consacrés à des matrices de Toeplitz bandes, en particulier à des algorithmes pour leur inversion [2, 81]. Le problème de leur conditionnement a été traité au préalable, par exemple, dans [52,84], alors que les résultats importants concernant la distribution asymptotique des valeurs propres de matrices Toeplitz bandes peuvent être trouvés dans [4, 7]. Nous allons étudier dans quelles conditions la famille de la matrice $\{T_n\}$ est "bien conditionnée" selon la définition suivante :

Définition 2.6.1 *Une matrice T_n est dite bien conditionnée si le nombre de condition $K(T_n)$ est uniformément borné par rapport à n. Elle est dite faiblement bien conditionnée si $K(T_n)$ se développe comme une petite puissance de n.*

Remarque 1 *En observant*

$$K(T_n) = \| T_n \| \| T_n^{-1} \|$$

$\| T_n \|$ est toujours uniformément bornée par rapport à n, alors la famille $\{T_n\}$ sera bien conditionnée (faiblement bien conditionnée) si et seulement si les éléments dans la séquence $\{\| T_n^{-1} \|\}$ sont bien conditionnée (faiblement bien conditionnée).

2.6.2 Notations

Notons $p(z)$ le polynôme de degré $m_r + m_c$ correspondant à T_n définie par (2.27) :

$$p(z) = \sum_{i=-m_r}^{m_c} a_i z^{m_r+i}. \tag{2.28}$$

Soient $z_1, \ldots, z_{m_r+m_c}$ les racines $p(z)$, où

$$0 < z_1 \leq \ldots \leq z_{m_r} \leq \ldots \leq z_{m_r+m_c} < \infty$$

On dit que le polynôme (2.28) est de type (s, u, l) s'il a :

- s zéros avec module plus petit que 1,
- u zéros avec module unitaire,
- l zéros avec module supérieur à 1.

Cette notation a déjà été utilisé dans [4]. Dans cette section, nous allons présenter les résultats concernant le conditionnement de matrice de Toeplitz triangulaire bande, obtenus en utilisant des arguments classiques sur les équations différentielles linéaires.
Soit L_n une matrice de Toeplitz triangulaire bande :

$$\begin{pmatrix} a_0 & & & & \\ \vdots & \ddots & & & \\ a_{m_c} & & \ddots & & \\ & \ddots & & \ddots & \\ & & a_{m_c} & & a_0 \end{pmatrix} \tag{2.29}$$

Dans ce cas, le polynôme (2.28) associé est donnée par

$$p_l(z) = \sum_{i=0}^{m_c} a_i z^i. \tag{2.30}$$

Théorème 2.6.1 *La famille de matrice $\{L_n\}$ définie dans (2.29) est :*

- *bien conditionnée si et seulement si le polynôme (2.30) est de type $(0, 0, m_c)$,*
- *faiblement bien conditionnée si et seulement si le polynôme (2.30) est de type $(0, m_1, m_2)$, avec $m_1 + m_2 = m_c$.*

Preuve. Voir [4]. ■

Théorème 2.6.2 *Soient $\{T_n\}$ une famille de matrices de Toeplitz bande inversible définie dans (2.27) et $p(z)$ le polynôme associé à la matrice (2.27). Alors, la famille de matrices $\{T_n\}$ est :*

- *Bien conditionnée si $p(z)$ est de type $(m_r, 0, m_c)$.*
- *Faiblement bien conditionnée si $p(z)$ est de type (m_1, m_2, m_r), (m_c, k_1, k_2), où $m_1 + m_2 = m_r$ et $k_1 + k_2 = m_c$. Dans ce cas, $K(T_n)$ croit au plus comme $O(n^\mu)$ où μ est la plus élevée parmi la multiplicité des zéros de modules unitaires.*

Preuve. Voir [4]. ■

Exemple 2.6.3 *Soit*

$$T_n^{(1)} = \begin{pmatrix} 1 & 1 & & & \\ -\frac{7}{4} & \ddots & \ddots & & \\ \frac{1}{2} & \ddots & \ddots & \ddots & \\ & \ddots & \ddots & \ddots & 1 \\ & & \frac{1}{2} & -\frac{7}{4} & 1 \end{pmatrix}, \tag{2.31}$$

et

$$p_1(z) = (z-2)^2(2z+1).$$

D'où le polynôme $p_1(z)$ associé à (2.31) est de type $(1, 0, 2)$. D'après le Théorème 2.6.2, nous pouvons conclure que $\{K_\infty(T_n^{(1)})\}$ est uniformément bornée par rapport à n, ceci est illustré dans la Figure 2.31.

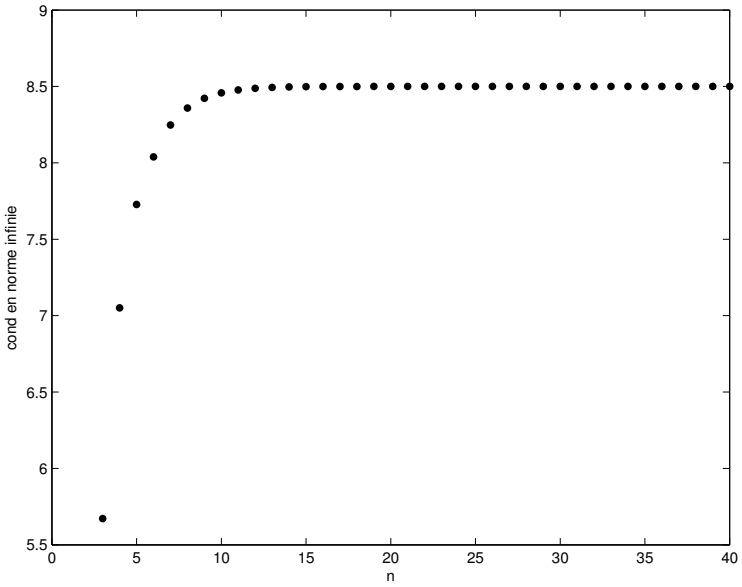


FIGURE 2.1 – Conditionnement de la matrice (2.31)

Exemple 2.6.4 *On considère la matrice*

$$T_n^{(2)} = \begin{pmatrix} 2 & 1 & & & & & \\ -2 & \ddots & \ddots & & & & \\ -4 & \ddots & \ddots & \ddots & & & \\ 1 & \ddots & \ddots & \ddots & \ddots & & \\ 2 & & \ddots & \ddots & \ddots & \ddots & \\ & \ddots & \ddots & \ddots & \ddots & \ddots & 1 \\ & & 2 & 1 & -4 & -2 & 2 \end{pmatrix}, \tag{2.32}$$

et

$$p_2(z) = (z+1)^2(z-1)^2(2z+1).$$

Alors $p_2(z)$ est de type $(1,4,0)$, nous pouvons conclure que $\{K_\infty(T_n^{(2)})\}$, croit plus que $O(n^2)$. Ceci est illustré dans la Figure 2.32

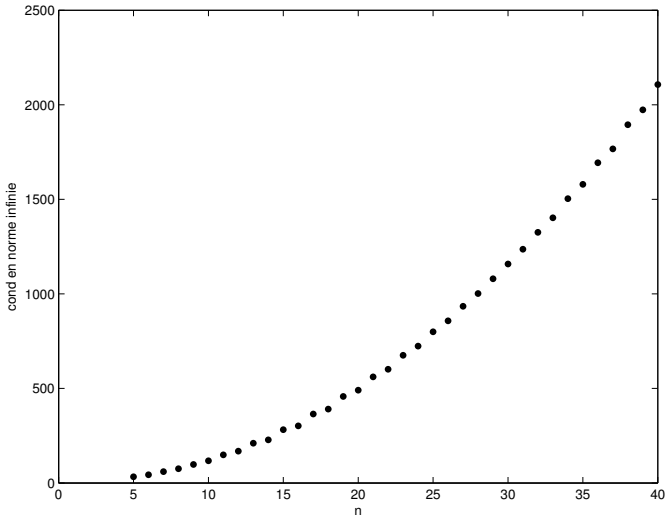


FIGURE 2.2 – Conditionnement de la matrice (2.32)

CHAPITRE 3

MÉTHODES NUMÉRIQUES SUR LES MATRICES TRIANGULAIRES DE TOEPLITZ

3.1 Introduction

Dans ce chapitre, nous allons traiter le problème de calcul de l'inverse d'une matrice triangulaire inférieure de Toeplitz

$$T_n = \begin{pmatrix} t_0 & & & \\ t_1 & t_0 & & \\ \vdots & \ddots & \ddots & \\ t_{n-1} & \cdots & t_1 & t_0 \end{pmatrix},$$

avec $t_j \in \mathbb{K}$ pour $j = 0, 1, ..., n-1$ et $t_0 \neq 0$.

Pour inverser rapidement une matrice triangulaire de Toeplitz, Morf a indiqué dans [61] que la stratégie de Diviser-pour-régner donne un algorithme en $O(n \log n)$ opérations (du même ordre que la convolution en utilisant la transformée de Fourier rapide (FFT)). Commmenges et Monsion [33] ont proposé un algorithme nécessitant $O(n \log n)$ opérations, plus précisément, environ 10 transformées de Fourier rapide (FFT) de n vecteurs (FFT(n)). Après plusieurs années d'investigation intensive, l'approche approximative en utilisant la transformée de Fourier rapide pour inverser une matrice triangulaire de Toeplitz a été étudiée par Bini [10] et Georgiev [39]. Considérant ce problème via une approche polynomiale [19, 20, 66, 70, 70, 75], de nombreuses techniques disponibles pour la division polynomiale peuvent être utilisées, tels que Knuth [56] et Bini & Pan [20]. Récemment, Lin, Ching & Ng [55] ont introduit une méthode d'inversion approchée pour les matrices triangulaires de Toeplitz en se basant sur le polynôme d'interpolation trigonométrique. De plus, ils ont proposé un algorithme révisé de la méthode de Bini.

Dans ce chapitre, nous proposons une nouvelle méthode pour inverser rapidement les matrices triangulaires de Toeplitz.

La nouvelle approche est d'une complexité environ 2FFT$(2n)$, qu'est inspirée de la méthode d'interpolation de Lin, Ching & Ng. Cet algorithme nécessitant uniquement deux FFT$(2n)$ est manifestement efficace par rapport à ces prédécesseurs. Une étude d'erreur théorique a été proposée. Ce travail a donné lieu à une publication intitulée "A note on computing the inverse of a triangular Toeplitz matrix" dans la revue "Applied Mathematics and Computation" [8].

Plusieurs exemples numériques sont proposés pour illustrer l'efficacité et la stabilité des algorithmes proposés.

Dans ce qui suit, nous introduisons quelques propriétés d'une matrice triangulaire inférieure de Toeplitz.

Définition 3.1.1 *Une matrice carrée de taille $n \times n$ est dite une matrice triangulaire inférieure de Toeplitz si elle est une matrice f-circulante avec $f = 0$, c'est à dire $T_n = Z_0(t)$. Ainsi, T_n a la forme suivante :*

$$T_n = \begin{pmatrix} t_0 & & & \\ t_1 & \ddots & & \\ \vdots & \ddots & \ddots & \\ t_{n-1} & \cdots & t_1 & t_0 \end{pmatrix}$$

avec $t_j \in \mathbb{K}$, $j = 0, 1, \ldots, n-1$ et $t_0 \neq 0$.

Proposition 3.1.1 *Chaque matrice triangulaire inférieure est représentée comme une combinaison linéaire des puissances de matrice shift vers le bas :*

$$T_n = t_0 I + t_1 Z_n + \cdots + t_{n-1} Z_n^{n-1}. \tag{3.1}$$

Proposition 3.1.2 *L'inverse d'une matrice triangulaire inférieure de Toeplitz $T_n = Z_0(t)$ est une matrice triangulaire inférieure de Toeplitz $T_n^{-1} = Z_0(b)$ avec $b_0 = 1/t_0$.*

Proposition 3.1.3 *Pour toute matrice triangulaire inférieure de Toeplitz T_n, nous définissons le polynôme :*

$$p_n(z) = \sum_{k=0}^{n-1} t_i z^k = t_0 + t_1 z + ... + t_{n-1} z^{n-1} \tag{3.2}$$

comme polynôme associé à T_n. Le développement en série de Maclaurin de $p_n^{-1}(z)$ est donné par

$$p_n^{-1}(z) = \sum_{k=0}^{\infty} b_k z^k, \tag{3.3}$$

Alors,

$$T_n^{-1} = \begin{pmatrix} b_0 & & & \\ b_1 & b_0 & & \\ \vdots & \ddots & \ddots & \\ b_{n-1} & \cdots & b_1 & b_0 \end{pmatrix}. \tag{3.4}$$

Afin d'obtenir T_n^{-1}, nous avons besoin seulement de calculer les coefficients b_j pour $j = 1, 2, \ldots, n-1$.

Proposition 3.1.4 *Remplaçant z par ρz dans (3.2) et (3.3), nous obtenons*

$$p_{n,\rho}(z) = p_n(\rho z) = \sum_{k=0}^{n-1} \left(t_k \rho^k\right) z^k \quad \textit{et} \quad p_{n,\rho}^{-1}(z) = p_n^{-1}(\rho z) = \sum_{k=0}^{\infty} \left(b_k \rho^k\right) z^k.$$

De manière équivalente, nous avons

$$\begin{pmatrix} t_0 & & & \\ \rho t_1 & t_0 & & \\ \vdots & \ddots & \ddots & \\ \rho^{n-1} t_{n-1} & \cdots & \rho t_1 & t_0 \end{pmatrix}^{-1} = \begin{pmatrix} b_0 & & & \\ \rho b_1 & b_0 & & \\ \vdots & \ddots & \ddots & \\ \rho^{n-1} b_{n-1} & \cdots & \rho b_1 & b_0 \end{pmatrix}.$$

Nous pouvons choisir $\rho \in]0, 1[$ tel que

$$\sum_{k=0}^{\infty} \left|b_k \rho^k\right| < \infty. \tag{3.5}$$

Proposition 3.1.5 *L'inverse de la sous-matrice principale dominante $T_n(1:p, 1:p)$ est égale à la sous matrice principale dominante $T_n^{-1}(1:p, 1:p)$ pour $1 \leq p \leq n$.*

3.2 Inversion exacte

3.2.1 Inversion via substitution

Nous rappelons, dans cette partie, un algorithme d'inversion de la matrice triangulaire de Toeplitz basé sur la substitution, requiert environ $n(n+1)$ opérations arithmétiques [40]. L'algorithme de cette méthode est décrit comme suit :

3.2.2 Inversion via la division polynomiale

Une autre méthode d'inversion exacte d'une matrice triangulaire de Toeplitz s'effectue grâce à un anneau de développements limités [69], ce qui revient à une inversion de 1 par un autre polynôme dans $\mathbb{K}[x]$ selon les puissances croissantes et sans tenir compte des puissances trop grandes. Une telle inversion dans l'anneau des développements limités est nettement plus rapide

Algorithm 3.1 Inversion via substitution

Entrée : $t = (t_0, t_1, \ldots, t_{n-1})$ la première colonne de T_n

Sortie : $b = (b_0, b_1, \ldots, b_{n-1})$ la première colonne de T_n^{-1}

1. n =taille(t) et $b = \text{zeros}(n, 1)$
2. $b(1) = \frac{1}{t(1)}$
3. Pour $k = 2 : n$ alors

$$b(k : n) = b(k : n) + b(k-1)t(2 : n-k+2)$$

$$b(k) = -b(k)/t(1)$$

4. Renvoyer b

qu'une inversion générale de matrice. Et comme pour toute matrice triangulaire inférieure de Toeplitz, on peut définir le polynôme (3.2) alors l'inverse de cette matrice est donnée par ce lemme :

Lemme 3.2.1 *Si* $T = uT(\alpha_0, \ldots, \alpha_{n-1})$ *avec* $\alpha_0 \neq 0$, $T^{-1} = uT(\mu_0, \ldots, \mu_{n-1})$. *En posant*

$$S(x) = \alpha_0 + \alpha_1 x + \cdots + \alpha_{n-1}x^{n-1} = \sum_{k=0}^{n-1} \alpha_k x^k,$$

$$Q(x) = \mu_0 + \mu_1 x + \cdots + \mu_{n-1}x^{n-1} = \sum_{k=0}^{n-1} \mu_k x^k.$$

Alors, $S(x)Q(x) = 1 \mod x^n$.

Ainsi, l'algorithme de cette méthode est décrit par l'Algorithme 3.2.

Pour conclure cette section, nous allons comparer l'Algorithme 3.1 basé sur la substitution et l'Algorithme 3.2 basé sur la division polynomiale par un exemple numérique.
Soit T une matrice inférieure de Toeplitz définie par les coefficients $t_j = \frac{1}{1+j}$, $j = 0, 1, \ldots, n-1$. La Figure 3.1 représente le temps de calcul de l'Algorithme 3.1 et l'Algorithme 3.2.

Algorithm 3.2 Inversion via la division polynomiale

Entrée : $t = (\alpha_0, \alpha_1, \ldots, \alpha_{n-1})$ la première colonne de T_n et aussi les coefficients du polynôme $S(x)$.

Sortie : $S(x)Q(x) = 1 \mod x^n$.

1. $\mu_0 = \frac{1}{\alpha_0}$
2. $r = \lceil \log n \rceil$
3. Pour $i = 0, \ldots, r-1$ alors

$$\mu_{i+1} = \mu_i(2 - S(x)\mu_i) \mod x^{2^i}$$

avec $\mu_i = Q(x) \mod x^{2^i}$

4. Renvoyer $Q(x)$.

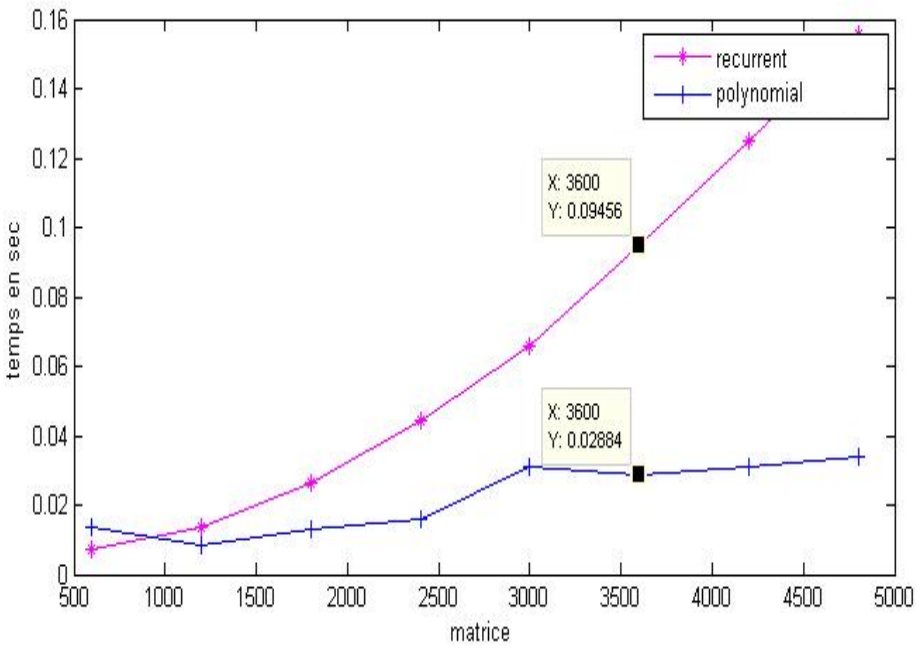


FIGURE 3.1 – Temps de calcul des algorithmes 3.1 et 3.2

On remarque bien d'après la Figure 3.1 que l'inverse d'une matrice triangulaire de Toeplitz via la méthode de l'inversion polynomiale est beaucoup plus avantageuse de point de vue temps de calcul. Malgré l'amélioration importante du coût de calcul de l'inverse exacte d'une matrice triangulaire inférieure de Toeplitz, les méthodes approchées restent souvent moins chères. Nous

allons proposer dans ce qui suit des algorithmes d'inversions approchées plus rapide.

3.3 Inversion approchée

3.3.1 Inversion via interpolation

Nous allons présenter l'algorithme de Fu-Rong Lin, Wai-Ki Ching et M. K. Ng [55] qui permet de calculer l'inverse d'une matrice triangulaire de Toeplitz basé sur l'interpolation polynomiale trigonométrique. Cette méthode génère l'inverse d'une matrice triangulaire à haute précision et à faible coût (nécessite deux FFT et une transformation de cosinus (DCT) de $2n$-vecteurs).

En remplaçant z par e^{-ix}, avec $i^2 = -1$ et $x \in \mathbb{R}$, on désigne aussi par $p_n(e^{-ix})$ un polynôme trigonométrique. Par conséquent, pour récupérer les b_j, nous pensons directement à calculer les coefficients de Fourier de $1/p_n(e^{-ix})$:

$$b_k = \frac{1}{2\pi}\int_0^{2\pi} \frac{1}{p_n(e^{-ix})} e^{ikx} dx \quad \text{pour} \quad k = 0, 1 \ldots, n-1. \tag{3.6}$$

Malheureusement, il est difficile de calculer b_j efficacement en utilisant la formule (3.6) car le calcul explicite de p_n^{-1} est généralement inconnu. Donc, nous considérons d'interpoler p_n^{-1} (3.3) via l'interpolation trigonométrique.

Notons qu'il est difficile de calculer algébriquement b_j via l'interpolation polynomiale car les matrices de Vandemonde d'ordre élevé sont très mal-conditionnées. On pose

$$p_\rho(\theta) = p_{n,\rho}(e^{-i\theta}) = \sum_{k=0}^{n-1} (t_k \rho^k) e^{-ik\theta} = p_\rho^{(r)}(\theta) + \mathrm{i} p_\rho^{(i)}(\theta) \tag{3.7}$$

avec $p_\rho^{(r)}(\theta)$ et $p_\rho^{(i)}(\theta)$ sont la partie réelle et la partie imaginaire de $p_\rho(\theta)$, respectivement. Plus précisément, en utilisant (3.3), nous avons

$$h_\rho(\theta) \equiv p_\rho^{-1}(\theta) = \sum_{k=0}^{\infty} (b_k \rho^k) e^{-ik\theta}.$$

En particulier, la partie réelle de h_ρ est donnée par

$$h_\rho^{(r)}(\theta) = \frac{p_\rho^{(r)}(\theta)}{(p_\rho^{(r)}(\theta))^2 + (p_\rho^{(i)}(\theta))^2} \tag{3.8}$$

Nous avons $h_\rho^{(r)}(\theta)$ une fonction paire et 2π-périodique. Pour des valeurs approchées $\widehat{b}_j$ de b_j, $j = 0, 1, \ldots, n-1$, nous interpolons $h_\rho^{(r)}$ par une fonction de l'ensemble $\prod_{n-1}$, où $\prod_m$ est l'ensemble de tous les polynômes trigonométriques paires de degré $\leq m$. Nous utilisons les points équidistants suivants :

$$\theta_k = \frac{2k+1}{n}\pi, \quad k = 0, 2, \ldots, n-1$$

comme des nœuds d'interpolation. En fait, l'avantage d'utiliser ces points d'interpolation est, d'une part : $h_\rho^{(r)}(\theta)$ peut être calculé efficacement via une transformation rapide de cosinus (DCT) et, d'autre part : le polynôme d'interpolation trigonométrique peut approximer efficacement la fonction de départ.

Soit

$$\tau_{n-1}(\theta) = \sum_{k=0}^{n-1} f_k \cos(k\theta)$$

le polynôme d'interpolation de $h_\rho^{(r)}(\theta)$. En utilisant les conditions suivantes d'interpolation :

$$\tau_{n-1}(\theta_k) = h_\rho^{(r)}(\theta_k),\ k = 0, \ldots, n-1,$$

nous avons

$$\mathcal{C}(f_0, f_1, \ldots, f_{n-1})^T = (h_\rho^{(r)}(\theta_0), h_\rho^{(r)}(\theta_1), \ldots, h_\rho^{(r)}(\theta_{n-1}))^T \tag{3.9}$$

avec $\mathcal{C} = (c_{jk})_{j,k=1,2,\ldots,n}$ tel que $c_{jk} = \cos(\frac{(k-1)(2j-1)\pi}{2n}),\ j, k = 1, 2, \ldots, n.$

Notons que $\mathcal{C} = (\Phi^c)D_n$, avec Φ^c est la matrice de transformation de Fourier discrète de cosinus et

$$D_n = \text{diag}(\sqrt{n}, \sqrt{\frac{n}{2}} I_{n-1}) = \begin{pmatrix} \sqrt{n} & & & \\ & \sqrt{\frac{n}{2}} & & \\ & & \ddots & \\ & & & \sqrt{\frac{n}{2}} \end{pmatrix}.$$

Nous remarquons que si les valeurs $h_\rho^{(r)}(\theta_k)$, $k = 0, 1, \ldots, n-1$ sont données, alors f_k peut être obtenu en utilisant une DCT de n-vecteurs (DCT(n)). Finalement, $\widehat{b}_k$ peut être obtenu en $O(n)$ divisions via

$$\widehat{b}_k = f_k \rho^{-k}.$$

Regardons maintenant l'efficacité du polynôme d'interpolation τ_{n-1}. On a

$$\|\tau_{n-1} - h_\rho^{(r)}\|_\infty \leq (2 + \frac{2}{\pi}\log(n)) E_{n-1}(h_\rho^{(r)}),$$

avec $E_m(h_\rho^{(r)}) = \min_{\tau \in \prod_m} \|\tau - h_\rho^{(r)}\|$ est l'erreur de la meilleure approximation dans $\prod_m$ (voir, [23], [73]).

Théorème 3.3.1 *Soit f de classe C^p alors $E_{m-1}(f) \leq \frac{\pi}{2}(\frac{1}{m})^p \|f^{(p)}\|_\infty$.*

Preuve. La preuve est basée sur le théorème de Jackson (voir [31]). ■

Ainsi, nous proposons le théorème suivant pour déterminer l'efficacité de $\widehat{b}_k$, $k = 0, 1, \ldots, n-1$.

Théorème 3.3.2 *Soient* $\sum_{j=0}^{\infty}(b_j\varepsilon^j)z^j$ *le développement en série de Maclaurin de* $p_n^{-1}(\rho z)$ *et* $\rho \in]0,1[$ *tel que* $\sum_{j=0}^{\infty}|b_j\rho^j| < \infty$. *Soit*

$$\tau_{n-1}(\theta) = \sum_{j=0}^{n-1} f_j \cos(j\theta)$$

le polynôme d'interpolation de $h_\rho^{(r)}(\theta)$ *avec les points d'interpolations sont* $\theta_k = \frac{2k+1}{2n}\pi$, $k = 0,1,\ldots,n-1$ *et* $\widehat{b}_j = f_j\rho^{-j}$, $j = 0,1,\ldots,n-1$. *Alors,*

$$\widehat{b}_0 = b_0 + \rho^{2n}\sum_{m=1}^{\infty}(-1)^m(\rho^{2(m-1)n}b_{2mn}) \tag{3.10}$$

$$\widehat{b}_k = b_k + \rho^{2n}\sum_{m=1}^{\infty}(-1)^m(\rho^{2(m-1)n}b_{2mn+k} + \rho^{2((m-1)n-k))}b_{2mn-k}). \tag{3.11}$$

D'après (3.11), nous remarquons lorsque ρ est petit, $\widehat{b}_k$ est proche de b_k, $k = 0,1,\ldots,n-1$. En effet, mettant dans (3.11) ρ^{2n} en facteur (on peut choisir ρ tel que ρ^{2n} est très proche de 0), nous obtenons

$$\begin{aligned}
&\widehat{b}_0 \approx b_0\\
&\widehat{b}_j \approx b_j - \rho^{2(n-1)}b_{2n-1} = b_j - \rho^{2(n-1)}b_{n+(n-j)}, \quad j = 1,2,\ldots,n-1
\end{aligned}$$

Numériquement, nous ne pouvons pas imposer ρ très petit car f_k/ρ^k ne peut pas être calculé pour k très large. Si ρ est très proche de 1, $\widehat{b}_k$ peut être une bonne approximation de b_k pour k petit car $\rho^{2(n-k)}$ est très proche de zéro. De plus, $\widehat{b}_k$ n'est pas une bonne approximation de b_k pour k proche de n (par exemple, $\widehat{b}_{n-1} - b_{n-1} \approx -\rho^2 b_{n+1}$).

Pour confirmer les résultats du Théorème 3.3.2, la Figure 3.2 représente les erreurs de calcul de l'inverse d'une matrice triangulaire inférieure de Toeplitz définie par :

$$t_k = \frac{1}{(k+1)^2}, \quad k = 0,1,\ldots,511, \ \rho = 1 \text{ et } \rho = 2^{-18/512}.$$

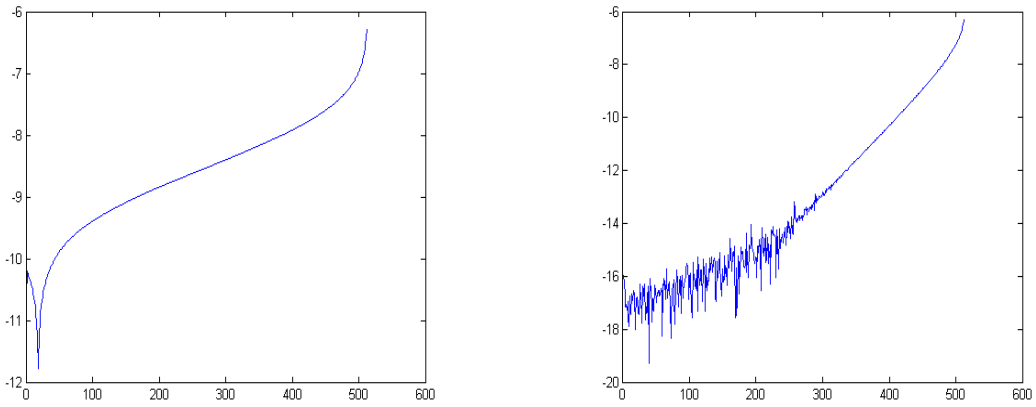


FIGURE 3.2 – $\rho = 1$ (figure gauche), $\rho = 2^{(-18/n)}$ (figure droite).

Figure 3.2 représente la fonction $\log_{10}(\widehat{b}-b)$ avec $\widehat{b}$ la première colonne de l'inversion d'une matrice triangulaire de Toeplitz, $t_k = 1/(k+1)^2,\ k = 0, 1 \ldots, 511$, $\rho = 1$ et $\rho = 2^{(-18/m)}$, respectivement. b représente la première colonne de l'inverse de la matrice T_n obtenue par la méthode de substitution (Algorithme 3.1) et $\widehat{b}$ est la première colonne de l'inverse approchée de la matrice T_n obtenue par la méthode d'interpolation avec la première colonne de T_n sont de coefficients $t_k = 1/(k+1)^2, k = 0, 1 \ldots, 511$, $\rho = 1$ (figure gauche), $\rho = 2^{(-18/n)}$ (figure droite).

Nous observons qu'avec un bon choix de ρ, la solution approchée est efficace, spécifiquement, pour k loin de n. Ce qui vérifie que les résultats numériques coïncident bien avec les résultats théoriques.

Pour améliorer l'efficacité des résultats numériques, nous pouvons interpoler $h_\rho^{(r)}(\theta)$ par un polynôme trigonométrique de degré $n + n_0$, $n_0 > 0$. Dans la suite, nous remplaçons n_0 par n.

Ainsi, l'algorithme de cette méthode est le suivant :

Algorithm 3.3 Inversion via interpolation

Entrer : $[t_j]_{j=0}^{n-1}$ la première colonne de la matrice T_n, $\rho \in]0,1[$.

Sortie : la première colonne de la matrice T_n^{-1},

Étape 1 : choisir ρ et calculer $\tilde{t}_k = \rho^k t_k$ pour $k = 0, 1, \ldots, n-1$

Étape 2 : calculer $P_\rho(\theta_j) = \sum_{k=0}^{n-1} (\tilde{t}_k) e^{-ik\theta_j}$, pour $\theta_j = (2j-1)\pi/4n$ pour $j = 1, \ldots, 2n$.

Étape 3 : calculer $h_j = h_\rho^{(r)}(\theta_j) = \frac{p_{\rho_j}^{(r)}(\theta)}{(p_\rho^{(r)}(\theta_j))^2 + (p_\rho^{(i)}(\theta_j))^2}$ pour $j = 1, \ldots, 2n$

Étape 4 : résoudre $\mathcal{C}(f_0, f_1, \ldots, f_{n-1})^T = (h_\rho(\theta_0), h_\rho(\theta_1), \ldots, h_\rho(\theta_{n-1}))^T$, avec $\mathcal{C} = (c_{jk})_{j,k=1,2,\ldots,n}$ tel que $c_{jk} = \cos(\frac{(k-1)(2j-1)\pi}{4n})$, $j, k = 1, 2, \ldots, 2n$.

Étape 5 : calculer $\widehat{b} = [f_k/\rho^k]_{k=0}^{n-1}$

Pour conclure, nous discutons la complexité de l'algorithme 3.3. En effet, on a

$$P_\rho(\theta_{2j}) \sum_{k=0}^{n-1} (\tilde{t}_k) e^{-\mathrm{i}k\theta_{2j}} = \sum_{k=1}^{n} (\tilde{t}_k e^{-3\pi \mathrm{i}(k-1)/4n}) e^{-2\pi \mathrm{i}(k-1)(l-1)/2n}.$$

Alors, les valeurs de $P_\rho(\theta_{2j})$ pour $j = 1, \ldots, n$ sont calculées via une FFT($2n$). De même, $P_\rho(\theta_{2j-1})$ pour $j = 1, \ldots, n$ sont calculées via une FFT($2n$). La complexité de l'Étape 4 et l'Étape 5 de l'algorithme 3.3 est DCT($2n$). D'où, le coût de l'algorithme 3.3 nécessite environ deux FFT($2n$) et une DCT($2n$).

Les essais numériques montrent que $\rho = 2^{(-18/n)}$ est le meilleur choix.

3.3.2 Inversion via Bini

Pour calculer rapidement et efficacement l'inverse de la matrice triangulaire supérieur de Toeplitz, Bini [10] a proposé une méthode approchée.

Rappelons que

$$T_n = \sum_{i=0}^{n-1} t_j Z_n^j.$$

où Z_n est la matrice shift vers le bas. L'idée de Bini est d'approximer Z_n par $Z_n^{(\varepsilon)} = [z_{jk}^{(\varepsilon)}]_{j,k=1}^n$, où $z_{jk}^{(\varepsilon)} = z_{jk}$ quand $(j,k) \neq (1,n)$ et $z_{1n}^{(\varepsilon)} = \varepsilon^n$ (ε est un petit nombre positif) et pour calculer efficacement l'inverse de

$$T_n^{(\varepsilon)} = \sum_{j=0}^{n-1} t_j (Z_n^{(\varepsilon)})^j$$

en utilisant la transformation de Fourier rapide.

Comme $T_n^{(\varepsilon)}$ est une matrice ε-circulante , alors l'inverse de $T_n^{(\varepsilon)}$ peut être calculée rapidement

en utilisant la diagonalisation de Fourier voir Théorème 2.4.1.

$$(T_n^{(\varepsilon)})^{-1} = (D_n^{(\varepsilon)})^{-1} F_n^* D_n^{-1} F_n D_n^{(\varepsilon)}, \tag{3.12}$$

où $D_n^{(\varepsilon)} = \text{diag}(1, \varepsilon, ..., \epsilon^{n-1})$, $D_n = \text{diag}\,(d)$ avec $d = \sqrt{n} F_n D_n^{(\varepsilon)}\,[t_j]_{j=0}^{n-1}$ et F_n est la matrice de Fourier de taille $n \times n$. Ainsi, l'algorithme de Bini est donné comme suit :

Algorithm 3.4 Inversion de Bini

Entrer : $[t_j]_{j=0}^{n-1}$ première colonne de la matrice T_n, $\varepsilon \in]0, 1[$

Sortie : la première colonne de la matrice approchée T_n^{-1}

Étape 1 : calculer $\tilde{t}_k = \varepsilon^k t_k$ pour $k = 0, 1, \ldots, n-1$

Étape 2 : calculer $d = (\sqrt{n}F)[\tilde{t}_j]_{j=0}^{n-1}$

Étape 3 : calculer $[c_j]_{j=0}^{n-1} = [1/d_j]_{j=0}^{n-1}$

Étape 4 : calculer $f = (F^*/\sqrt{n})c$

Étape 5 : $[b_j^{\varepsilon}]_{j=0}^{n-1} = [f_j/\varepsilon^i]_{j=0}^{n-1}$

Théorème 3.3.3 *Soient* $\sum_{j=0}^{\infty}(b_j\varepsilon^j)z^j$ *le développement en série de Maclaurin de* $p_n^{-1}(\varepsilon z)$ *et* $\varepsilon \in]0, 1[$ *tel que* $\sum_{j=0}^{\infty}|b_j\varepsilon^j| < \infty$. *Soit* $b^{(\varepsilon)}$ *la première colonne de l'inverse approchée de la matrice de Toeplitz obtenue via l'Algorithme 3.4. Alors* $b_j^{(\varepsilon)} - b_j = O(\varepsilon^n)$, $j = 0, 1, \ldots, n-1$. *C'est à dire,*

$$b_j^{(\varepsilon)} = b_j + \varepsilon^n \sum_{k=1}^{\infty} \varepsilon^{(k-1)n} b_{j+kn},\ j = 0, 1, \ldots, n-1 \tag{3.13}$$

Théoriquement, lorsque ε est très petit, l'inverse approchée est plus précise. D'autre part, si ε est proche de zéro, $D_n^{(\varepsilon)}$ sera très mal-conditionnée pour n grand. Par conséquent, nous devons choisir un ε approprié.

Nous illustrons l'effet des erreurs d'arrondis par un simple exemple. Soit T_n la matrice triangulaire inférieure de Toeplitz définie par sa première colonne :

$$t_0 = 1, t_1 = -1, t_j = 0 \text{ pour } j = 2, \ldots, n-1.$$

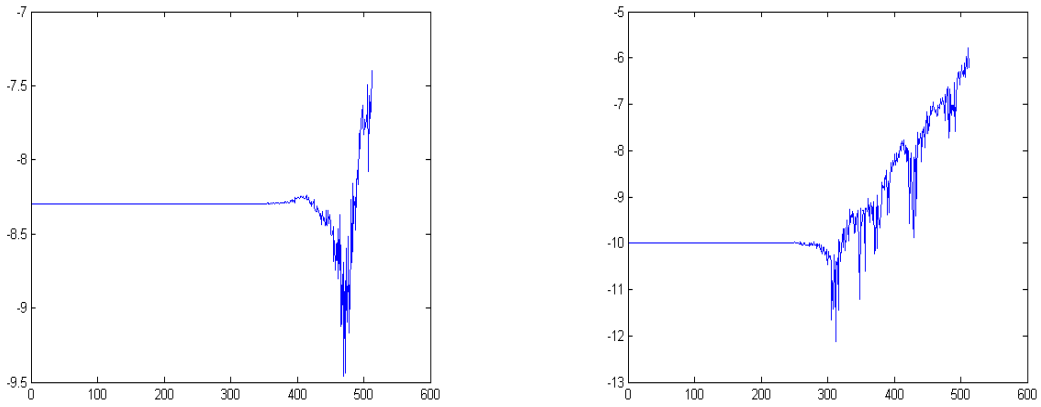


FIGURE 3.3 – La figure à gauche correspond à $\varepsilon = (0.5 \times 10^{-8})^{1/n}$ et la figure à droite pour $\varepsilon = (1.0 \times 10^{-10})^{\frac{1}{n}}$.

Ainsi, la première colonne de $(T_n)^{-1}$ est donnée par $b_j = 1$ pour $j = 1, \ldots, n-1$ et la première colonne de $(T_n^{(\varepsilon)})^{-1}$ est donnée par $b_j^{(\varepsilon)} = \frac{1}{1-\varepsilon^n}$ pour $j = 0, 1, \ldots, n-1$.

C'est-à-dire, $b_j^{(\varepsilon)} - b_j = \frac{\varepsilon^n}{1-\varepsilon^n} \approx \varepsilon^n$ pour tout j.

Remarque 3.3.1 *D'après la Figure 3.3, nous remarquons que $\tilde{b}^{(\varepsilon)}$ est moins précis pour j proche de n.*

3.3.3 Inversion via Bini révisée

En se basant sur les discussions ci-dessus, Lin, Ching et Ng [55] ont révisé l'algorithme de Bini pour obtenir une inversion approchée plus précise en plongeant la matrice triangulaire de Toeplitz d'ordre n dans une autre matrice triangulaire de Toeplitz d'ordre $2n$. Ce qui donne un résultat plus précis par rapport aux inversions approchées et encore plus rapide par rapport aux inversions exactes. L'algorithme peut être énoncé comme suit :

Algorithm 3.5 Inversion de Bini révisée

Entrer : $[t_j]_{j=0}^{n-1}$ la première colonne de la matrice T_n, $\varepsilon \in]0,1[$

Sortie : la première colonne de la matrice approchée T_n^{-1}

Étape 1 : calculer $\tilde{t}_k = \varepsilon^k t_k$ pour $k = 0, 1, \ldots, n-1$

Étape 2 : Poser $\tilde{t}_j = 0$ pour $j = n, n+1, \ldots, 2n-1$

Étape 3 : calculer $d = \sqrt{2n}F_{2n})[\tilde{t}_j]_{j=0}^{2n-1}$

Étape 4 : calculer $[c_j]_{j=0}^{2n-1} = [1/d_j]_{j=0}^{2n-1}$

Étape 5 : calculer $f = (F_{2n}^*/\sqrt{2n})c$

Étape 6 : $[b_j^{\varepsilon}]_{j=0}^{n-1} = [f_j/\varepsilon^i]_{j=0}^{n-1}$

Il est évident que la complexité de l'Algorithme de Bini révisé soit deux FFT($2n$). Les essais numériques montrent que $\varepsilon = (0.5 \times 10^{(-8)})^{\frac{1}{n}}$ et $\varepsilon = 10^{-\frac{5}{n}}$ sont les meilleurs choix pour l'algorithme de Bini et l'algorithme de Bini révisé, respectivement.

3.3.4 Inversion via l'interpolation modifiée

En se basant sur la méthode d'interpolation précédente et plus précisément sur l'équation (3.7), nous pouvons conclure notre deuxième contribution. Ainsi, nous avons

$$h_\rho(\theta) \equiv p_\rho^{-1}(\theta) = \sum_{k=0}^{\infty}(b_k\rho^k)e^{-ik\theta}.$$

De plus,

$$h_\rho(\theta) = \frac{p_\rho^{(r)}(\theta) - ip_\rho^{(i)}(\theta)}{(p_\rho^{(r)}(\theta))^2 + (p_\rho^{(i)}(\theta))^2} = h_\rho^{(r)}(\theta) + ih_\rho^{(i)}(\theta). \tag{3.14}$$

Visiblement, $h_\rho^{(r)}(\theta)$ et $h_\rho^{(i)}(\theta)$ sont des fonctions 2π-périodique paire et impaire, respectivement. Pour obtenir des valeurs approchées $\widehat{b}_j$ de b_j pour $j = 0, 1, \ldots, n-1$, nous interpolons h_ρ par une fonction de l'ensemble $\prod_{n-1}$, où $\prod_m$ est l'ensemble de tous les polynômes trigonométriques paire de degré $\leq m$. Nous utilisons les points équidistants suivants :

$$\theta_k = \frac{2k}{n}\pi, \quad k = 1, 2, \ldots, n-1$$

comme des nœuds d'interpolation. Pour récupérer efficacement $\widehat{b}_k$, on utilise la FFT et on approxime la fonction initiale via une interpolation trigonométrique polynômiale.
Soit

$$\tau_{n-1} = \sum_{k=0}^{n-1} f_k e^{-ik\theta}$$

le polynôme d'interpolation de $h_\rho(\theta)$. En utilisant les conditions d'interpolation

$$\tau_{n-1}(\theta_k) = h_\rho(\theta_k),\ k = 1, \ldots, n,$$

nous pouvons écrire

$$\mathcal{F}(f_0, f_1, \ldots, f_{n-1})^t = (h_\rho(\theta_1), h_\rho(\theta_2), \ldots, h_\rho(\theta_n))^t \tag{3.15}$$

où

$$[\mathcal{F}]_{j,k} = [e^{-2ikj/n}]_{j,k=0}^{n-1}.$$

Notons par $\mathcal{F}$ la matrice de FFT et nous remarquons que si les valeurs de $h_\rho(\theta)$, $k = 1, 2, \ldots, n$ sont connues, alors f_k peut être obtenue en utilisant une FFT de n-vecteurs. Enfin, $\widehat{b}_k$ peut être aussi obtenue en $O(n)$ divisions en passant par

$$\widehat{b}_k = f_k \rho^{-k}.$$

Ainsi, l'algorithme de cette méthode est le suivant :

Algorithm 3.6 Inversion via une interpolation modifiée

Entrer : $[t_j]_{j=0}^{n-1}$ la première colonne de la matrice T_n, $\rho \in]0, 1[$

Sortie : la première colonne de la matrice T_n^{-1}

Étape 1 : choisir $\rho \in]0, 1[$ et calculer $\tilde{t_j} = \rho^j t_j$, pour $j = 0, 1, \ldots, n-1$.

Étape 2 : calculer $p_\rho(\theta_k) = \sum_{l=0}^{n-1} \tilde{t_l} e^{-il\theta_k}$ où $\theta_k = 2k\pi/2n$ pour $k = 1, ..., 2n$.

Étape 3 : calculer $h_k = \frac{p_\rho^{(r)}(\theta_k) - i p_\rho^{(i)}(\theta_k)}{(p_\rho^{(r)}(\theta_k))^2 + (p_\rho^{(i)}(\theta_k))^2}$, pour $k = 1, ..., 2n$.

Étape 4 : calculer $\mathcal{F}(f_0, f_1, \ldots, f_{2n-1})^t = (h_1, h_2, \ldots, h_{2n})^t$, où $[\mathcal{F}]_{j,k} = e^{-i\frac{2jk\pi}{2n}}$, $j, k = 1, 2, \ldots, 2n-1$.

Étape 5 : calculer $[\widehat{b}_k]_{k=0}^{n-1} = [f_k \rho^{-k}]_{k=0}^{n-1}$.

Complexité de l'Algorithm 3.6 Dans l'Étape 1, $p_\rho(\theta_k)$ pour $k = 1, ..., 2n$ sont calculées via une FFT$(2n)$ en passant par la matrice de Fourier $[\mathcal{F}]_{j,k} = e^{-i\frac{2jk\pi}{2n}}$, $j, k = 1, 2, \ldots, 2n$. De plus, nous avons besoin d'une FFT$(2n)$ pour calculer $\widehat{v}_k$, $k = 0, ..., n-1$ dans l'Étape 3.

Faisons maintenant un coup d'œil à la précision théorique de $[\widehat{b}_k]_{k=0}^{n-1}$ calculée par l'algorithme 3.6.

Théorème 3.3.4 *Soit* $\sum_{j=0}^{\infty}(b_j\varepsilon^j)z^j$ *est le développement en série de Maclaurin de* $p_n^{-1}(\rho z)$ *et* $\rho \in]0,1[$ *tel que* $\sum_{j=0}^{\infty}|b_j\rho^j| < \infty$. *Soit*

$$\tau_{n-1}(\theta) = \sum_{j=0}^{n-1} e^{-ij\theta}$$

le polynôme d'interpolation de $h_\rho(\theta)$ *avec les points d'interpolation sont* $\theta_k = \frac{2k}{n}\pi$, $k = 0, 2, \ldots, n-1$ *et* $\widehat{b}_j = f_j\rho^{-j}$. *Alors* $\widehat{b}_k - b_k = O(\rho^n)$, $k = 0, 1, \ldots, n-1$. *Plus précisément,*

$$\widehat{b}_k = b_k + \rho^n \sum_{j=1}^{\infty}(\rho^{(j-1)n} b_{k+jn}), \quad k = 1, 2, \ldots, n-1 \tag{3.16}$$

Preuve. Soit $w = e^{-2i\pi/n}$. Nous avons

$$\tau_{n-1}(\theta_k) = h_\rho(\theta_k) \text{ pour } k = 0, 1 \ldots, n-1.$$

D'où

$$\begin{aligned} f_k &= \frac{1}{n}\sum_{j=0}^{n-1}(\sum_{l=0}^{\infty} b_l\rho^l w^{lj}) \text{ par l'IDFT} \\ &= \sum_{l=0}^{\infty} b_l\rho^l(\frac{1}{n}\sum_{j=0}^{n-1} w^{(l-k)j}) \\ &= \sum_{m=0}^{\infty} b_{k+mn}\rho^{k+mn}. \end{aligned}$$

La dernière égalité est le résultat de la relation d'orthogonalité discrète. En effet, nous pouvons déduire la formule suivante :

$$\sum_{j=0}^{n-1} w^{-lj}w^{kj} = \sum_{j=0}^{n-1} w^{(k-l)j} = \frac{1-w^{(k-l)n}}{1-w^{k-l}}$$

où

$$\frac{1-w^{(k-l)n}}{1-w^{k-l}} = \begin{cases} 0 \text{ si } k \neq l \pmod n \\ n \text{ si } k = l \pmod n \end{cases}$$

■

Remarque 3.3.2 *Selon nos tests numériques préliminaires, on remarque que* $\rho = \exp(-9/n)$ *et* $\rho = (0,5 \times exp(-11))^{1/n}$ *sont de bons choix pour notre approche dans le cas réel et complexe, respectivement.*

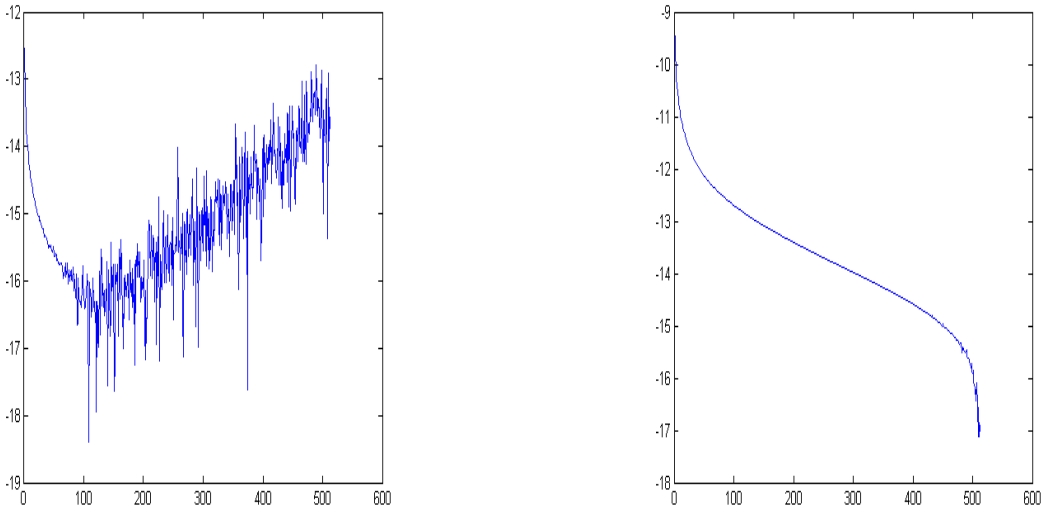


FIGURE 3.4 – $\log_{10}(|\hat{v} - v|)$ pour $t_k = \dfrac{1}{(k+1)^3}, k = 0, 3 \ldots, 511$. Celui de gauche est pour $\rho = e^{-9/n}$ et celui de droite est pour $\rho = e^{-1/n}$.

Nous observons qu'avec un bon choix de ρ, la solution approchée est efficace, spécifiquement, pour k loin de n. Ce qui vérifie que les résultats numériques coïncident bien avec les résultats théoriques.

3.3.4.1 Estimation de l'erreur

Pour effectuer une analyse de l'erreur d'arrondi de l'Algorithme 3.6, en se basant sur la notion de stabilité définie dans le premier chapitre. **Analyse de l'erreur de l'Algorithme 3.6** Soit $l = (t_1, \rho t_2, \rho^2 t_3, \cdots, \rho^{n-1} t_n)^T$ (Étape 1), $P = F_n l$ (Étape 2), $h = (p_1, p_2, \cdots, p_n)^T$ (Étape 3), $f = F_n^{-1} h$ (Étape 4) et $v_k = f_k / \rho^{k-1}$ pour $k = 1, 2, \cdots, n$ (Étape 5).

Théorème 3.3.5 *Le vecteur calculé v de la première colonne de l'inverse de la matrice triangulaire inférieure de Toeplitz satisfait*

$$\frac{\|v - \hat{v}\|_2}{\|v\|_2} \leqslant \frac{\tau \gamma_{n-1}}{\rho^{n-1}} = O(u)$$

où $\tau = nf(n,u) + \xi n f(n,u) + n^2 \xi$, $\xi = \sqrt{n}\left(\dfrac{|\tilde{\delta}| + \beta \|P\|_2 \|h\|_2}{1 - \beta \|P\|_2 \|h\|_2}\right)$ *et* $\beta = n^{1/2} f(n,u) + n^{3/2} \gamma_{n-1} + n^{7/2} f(n,u)$.

Preuve. Dans l'Étape 1, nous avons
$\hat{l} = (t_1, \rho t_2, \rho^2 t_3, \cdots, \rho^{n-1} t_n)^T + \Delta(t_1, \rho t_2, \rho^2 t_3, \cdots, \rho^{n-1} t_n)^T$, avec
$\Delta(t_1, \rho t_2, \rho^2 t_3, \cdots, \rho^{n-1} t_n)^T \in \mathbb{R}^n$ et
$\hat{l} = (t_1(1+\delta_1), \rho t_2(1+\delta_1), \rho^2 t_3(1+\delta_1)(1+\delta_2), \cdots, \rho^{n-1} t_n(1+\delta_1)(1+\delta_2)\cdots(1+\delta_{n-1}))^T$

$|\delta_k| \leqslant u$.
Alors d'après le Lemme 1.3.1,

$$\hat{l} = (t_1(1+\theta_1), \rho t_2(1+\theta_1), \cdots, \rho^{n-1} t_n \rho(1+\theta_{n-1}))^T$$

et

$$\|\Delta l\|_2 \leqslant n\gamma_{n-1}\|l\|_2.$$

En utilisant le Théorème 1.3.1 pour l'Étape 2, nous obtenons

$$\hat{P} = (F_n + \Delta_1)\hat{l}, \qquad \|\Delta_1\| \leqslant f(n,u).$$

Ainsi,

$$\begin{aligned}
\|\Delta(P)\|_2 &\leqslant \|\Delta_1 l\|_2 + \|F_n \Delta l\|_2 + \|\Delta_1 \Delta l\|_2, \text{ puisque } \|F_n\|_2 \leqslant \sqrt{n}\|F_n\|_1 = n^{3/2} \\
&\leqslant (f(n,u) + n^2\gamma_{n-1} + n^3 f(n,u))\|l\|_2 \\
&\leqslant (f(n,u) + n^2\gamma_{n-1} + n^3 f(n,u))\|F_n^{-1}P\|_2 \\
&\leqslant (n^{1/2} f(n,u) + n^{3/2}\gamma_{n-1} + n^{7/2} f(n,u))\|P\|_2 \\
&= \beta\|P\|_2
\end{aligned}$$

où $\beta = n^{1/2} f(n,u) + n^{3/2}\gamma_{n-1} + n^{7/2} f(n,u)$.
En passant par le Lemme 1.3.2 pour l'Étape 3, on peut déduire

$$\begin{aligned}
h_k &= \frac{1}{p_k}, \\
p_k h_k &= 1 \\
\hat{h}_k = fl(1/p_k) &= \frac{1}{\hat{p}_k}(1+\tilde{\delta}), \qquad |\tilde{\delta}| \leqslant \sqrt{2}\gamma_4, \\
\Delta\hat{h}_k &= \frac{\tilde{\delta} - \Delta h_k \Delta p_k - h_k \Delta p_k}{p_k} = \tilde{\delta} - \Delta h_k \Delta p_k - h_k \Delta p_k \\
\|\Delta h\|_2 &\leqslant \sqrt{n}\left(\frac{|\tilde{\delta}| + \beta\|P\|_2\|h\|_2}{1 - \beta\|P\|_2\|h\|_2}\right)\|h\|_2
\end{aligned}$$

alors,

$$\|\Delta h\|_2 \leqslant \xi\|h\|_2$$

avec $\xi = \sqrt{n}\left(\frac{|\tilde{\delta}| + \beta\|P\|_2\|h\|_2}{1 - \beta\|P\|_2\|h\|_2}\right)$.

En utilisant le Théorème 1.3.1 dans l'Étape 4, nous avons

$$\hat{f} = (F_n^{-1} + \Delta_2)\hat{h}.$$

Donc,

$$\|\Delta f\|_2 \leqslant (n^{-1}f(n,u) + \xi n^{-1/2} f(n,u) + n^{1/2}\xi)\|h\|_2 \quad \text{puisque} \|F_n^{-1}\|_2 \leqslant \sqrt{n}\|F_n^{-1}\|_1 = \sqrt{n}$$
$$\|\Delta f\|_2 \leqslant (nf(n,u) + \xi n f(n,u) + n^2\xi)\|f\|_2$$
$$\|\Delta f\|_2 \leqslant \tau\|f\|_2$$

où $\tau = nf(n,u) + \xi n f(n,u) + n^2\xi$.
Finalement, l'Étape 5 génère le résultat suivant :

$$\|\Delta v\|_2 \leqslant \frac{\gamma_{n-1}}{\rho^{n-1}}\|\Delta f\|_2,$$

avec $\zeta = (1, \rho, \rho^2, \ldots, \rho^{n-1})^T$. En effet, soit $\rho \in \mathbb{R}$, pour $k = 1, 2$ alors $\hat{v}_k = \dfrac{\hat{f}_k}{\rho^{k-1}}(1+\acute{\delta})$ et pour $k = 3, \ldots, n$ alors $\hat{v}_k = \dfrac{\hat{f}_k}{\rho^{k-1}(1+\theta_{k-2})}(1+\acute{\delta})$.
Plus précisément, pour $k = 3$, on a $\hat{v}_k = \dfrac{\hat{f}_k}{\rho^{k-1}}(1+\theta_2)$ et pour $k = 4, \ldots, n$, on a $\hat{v}_k = \dfrac{\hat{f}_k}{\rho^{k-1}}(1+\theta_{k-1})$. Ainsi, on conclut le résultat désiré :

$$\|\Delta v\|_2 \leqslant \frac{\tau\gamma_{n-1}}{\rho^{n-1}}\|\zeta\|_2\|v\|_2.$$

■

3.3.4.2 Efficacité

Dans cette section, nous donnons les résultats de quelques expériences numériques simples afin d'illustrer l'efficacité des algorithmes proposés. Tous les tests ont été effectués dans MATLAB R2007a en utilisant l'arithmétique en double précision.

Les tables de 3.1 à 3.6 donnent le comportement de la précision relative

$$\frac{\|\tilde{v} - v\|_1}{\|v\|_1}$$

de l'algorithme 3.4, l'algorithme 3.5, l'algorithme 3.3 et de notre algorithme 3.6 pour quatre différentes séquences de matrices triangulaires de Toeplitz où $\widehat{v}$ est la première colonne de l'inverse approchée et v est la colonne de l'inverse exacte calculée via la méthode diviser-pour-régner.

n	Bini	Interpolation	R-Bini	M-Interpolation
128	1.0128×10^{-008}	2.3219×10^{-011}	6.5218×10^{-012}	2.9564×10^{-012}
256	2.3418×10^{-008}	2.8712×10^{-011}	9.1910×10^{-012}	1.7643×10^{-012}
512	2.5446×10^{-008}	5.4048×10^{-011}	1.3059×10^{-011}	2.3302×10^{-012}
1024	4.4955×10^{-008}	8.8015×10^{-011}	2.2790×10^{-011}	2.7815×10^{-012}
2048	5.2207×10^{-008}	1.0615×10^{-010}	3.3609×10^{-011}	3.7570×10^{-012}
4096	8.8215×10^{-008}	1.5081×10^{-010}	5.1190×10^{-011}	5.6812×10^{-012}
8192	1.3212×10^{-007}	2.1729×10^{-010}	4.7082×10^{-011}	1.4459×10^{-011}
16364	1.9002×10^{-007}	3.1027×10^{-010}	1.0262×10^{-010}	6.6439×10^{-011}
32768	2.6599×10^{-007}	4.3687×10^{-010}	1.5401×10^{-010}	9.3864×10^{-011}

TABLE 3.1 – $t_k = \dfrac{1}{(k+1)}, k = 0, 1, \ldots, n-1.$

n	Bini	Interpolation	R-Bini	M-Interpolation
128	1.5516×10^{-008}	2.5184×10^{-011}	5.3515×10^{-012}	6.8655×10^{-013}
256	1.5727×10^{-008}	3.9087×10^{-011}	1.0074×10^{-011}	1.3487×10^{-012}
512	2.3745×10^{-008}	4.9944×10^{-011}	1.4104×10^{-011}	1.4950×10^{-012}
1024	3.7879×10^{-008}	8.6915×10^{-011}	2.1475×10^{-011}	2.2359×10^{-012}
2048	5.0758×10^{-008}	1.1980×10^{-010}	3.2501×10^{-011}	3.1058×10^{-012}
4096	8.8215×10^{-008}	1.5081×10^{-010}	5.1190×10^{-011}	5.6812×10^{-012}

TABLE 3.2 – $t_k = \dfrac{1}{(k+1)^2}, k = 0, 1, \ldots, n-1.$

n	Bini	Interpolation	R-Bini	M-Interpolation
128	1.1646×10^{-008}	3.0677×10^{-011}	7.0025×10^{-012}	8.2351×10^{-013}
256	1.8453×10^{-008}	3.0922×10^{-011}	8.2538×10^{-012}	1.0167×10^{-012}
512	2.6094×10^{-008}	5.2187×10^{-011}	1.6964×10^{-011}	1.6293×10^{-012}
1024	4.2041×10^{-008}	7.9948×10^{-011}	2.1866×10^{-011}	2.0400×10^{-012}
2048	5.2443×10^{-008}	1.2126×10^{-010}	3.1573×10^{-011}	3.2713×10^{-012}
4096	7.8765×10^{-008}	1.8714×10^{-010}	5.1668×10^{-011}	5.4464×10^{-012}

TABLE 3.3 – $t_k = \dfrac{1}{(k+1)^3}, k = 0, 1, \ldots, n-1.$

n	Bini	Interpolation	R-Bini	M-Interpolation
128	9.4493×10^{-009}	1.4520×10^{-011}	1.0648×10^{-011}	3.9642×10^{-010}
256	2.2179×10^{-008}	3.3477×10^{-011}	1.6156×10^{-011}	3.3788×10^{-010}
512	4.0807×10^{-008}	6.0458×10^{-011}	2.7735×10^{-011}	2.8497×10^{-010}
1024	6.2519×10^{-008}	9.6298×10^{-011}	4.2214×10^{-011}	2.4030×10^{-010}
2048	1.1315×10^{-007}	2.0535×10^{-010}	7.4937×10^{-011}	2.0715×10^{-010}
4096	2.0050×10^{-007}	2.9027×10^{-010}	1.3427×10^{-010}	1.8476×10^{-010}

TABLE 3.4 – $t_k = \dfrac{1}{\log(1+k)}$, $k = 0, 1, \ldots, n-1$.

n	Bini	Interpolation	R-Bini	M-Interpolation
128	1.2595×10^{-008}	2.7473×10^{-011}	4.5118×10^{-012}	6.2685×10^{-013}
256	1.7571×10^{-008}	3.4373×10^{-011}	1.0893×10^{-011}	1.0254×10^{-012}
512	2.6507×10^{-008}	4.8969×10^{-011}	1.0745×10^{-011}	1.4648×10^{-012}
1024	4.6964×10^{-008}	6.7318×10^{-011}	1.8442×10^{-011}	2.3973×10^{-012}
2048	5.3701×10^{-008}	8.4636×10^{-011}	2.9820×10^{-011}	3.5097×10^{-012}
4096	8.4974×10^{-008}	1.2719×10^{-010}	4.4621×10^{-011}	4.6750×10^{-012}

TABLE 3.5 – $t_k = (0.5)^{k+1}$, $k = 0, 1, \ldots, n-1$.

n	Bini	Interpolation	R-Bini	M-Interpolation
128	7.6484×10^{-009}	1.2595×10^{-011}	4.3655×10^{-012}	4.2875×10^{-013}
256	1.2737×10^{-008}	1.9179×10^{-011}	6.5592×10^{-012}	6.4020×10^{-013}
512	1.8297×10^{-008}	2.6409×10^{-011}	9.5046×10^{-012}	1.0169×10^{-012}
1024	2.2617×10^{-008}	4.3277×10^{-011}	1.3324×10^{-011}	1.5802×10^{-012}
2048	3.2648×10^{-008}	6.7644×10^{-011}	2.0438×10^{-011}	2.1189×10^{-012}
4096	5.3984×10^{-008}	9.7592×10^{-011}	2.9285×10^{-011}	3.1395×10^{-012}

TABLE 3.6 – $t_1 = 1,\ t_2 = 1/2,\ t_k = 0, k = 0, 1, \ldots, n-1$.

Bien que les tables de 3.1 à 3.6 sont très favorables pour le calcul de l'inverse d'une matrice triangulaire de Toeplitz, elles démontrent que la méthode proposée (Algorithme 3.6) suggère des résultats similaires voire meilleurs.

De plus, les tables 3.7 et 3.8 montrent aussi la compétitivité de la méthode proposée dans le cas complexe.

n	Bini	R-Bini	M-Interpolation
128	4.7×10^{-9}	1.0×10^{-10}	6.9×10^{-11}
256	5.7×10^{-9}	1.0×10^{-10}	6.9×10^{-11}
512	6.3×10^{-9}	1.0×10^{-10}	7.0×10^{-11}
1024	1.0×10^{-8}	1.0×10^{-10}	7.5×10^{-11}
2048	1.0×10^{-8}	1.0×10^{-10}	7.5×10^{-11}
4096	1.2×10^{-8}	1.4×10^{-10}	7.7×10^{-11}

TABLE 3.7 – $t_0 = t_1 = 1 + i$, $t_k = 0$, $k = 2, 3 \ldots, n-1$.

n	Bini	R-Bini	M-Interpolation
128	1.6×10^{-8}	1.2×10^{-10}	8.6×10^{-11}
256	3.0×10^{-8}	1.2×10^{-10}	9.5×10^{-11}
512	5.6×10^{-8}	1.4×10^{-10}	1.1×10^{-10}
1024	1.0×10^{-7}	1.8×10^{-10}	1.5×10^{-10}
2048	2.0×10^{-7}	2.5×10^{-10}	2.5×10^{-10}
4096	3.8×10^{-7}	4.3×10^{-10}	4.6×10^{-10}

TABLE 3.8 – $t_k = \frac{1}{2} + i\left(\frac{1}{k+1}\right)$, $k = 0, 1 \ldots, n-1$.

3.3.4.3 Temps de calcul

Dans la Figure 3.5, nous présentons le temps de calcul (en secondes) de notre méthode par rapport aux trois approches bien connues dans la littérature via 8 séquences. Soit $n \in \{600, 1200, 1800, 2400, 3000, 3600, 4200, 4800\}$ la taille d'une matrice triangulaire de Toeplitz définie par les coefficients suivants : $t_k = \frac{1}{k+1}$, $k = 0, 1, \ldots, n-1$.

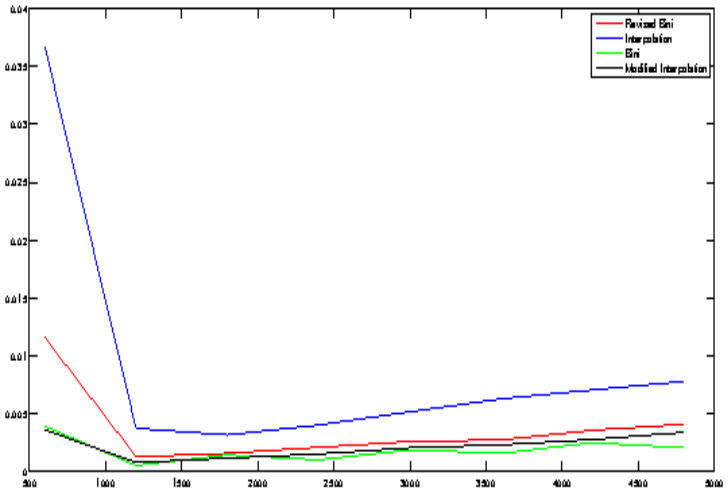


FIGURE 3.5 – Comparaison du temps de calcul pour $t_k = \dfrac{1}{k+1}, k = 0, 1, \ldots, n-1$.

La Figure 3.5 confirme que toutes les approches sont très efficaces puisque le temps de calcul est toujours inférieur à 1 seconde.
Pour améliorer notre approche, Table 3.9 et Table 3.10 illustrent le temps de calcul (en secondes) de toutes les méthodes d'une matrice triangulaire de Toeplitz de taille 32768.

Exemple	Bini	Interpolation	R-Bini	M-Interpolation
Exemple 5.1	0.016847	0.065927	0.024079	0.021729
Exemple 5.2	0.018489	0.072412	0.023844	0.023566
Exemple 5.3	0.018952	0.077513	0.028244	0.023761
Exemple 5.4	0.013866	0.058670	0.019925	0.017903
Exemple 5.5	0.018275	0.067362	0.020341	0.017894

TABLE 3.9 – Temps de calcul (en secondes) dans le cas réel

Exemple	Bini	R-Bini	M-Interpolation
Exemple 5.6	0.030507	0.039170	0.035538
Exemple 5.7	0.029782	0.042097	0.038799
Exemple 5.8	0.029851	0.049248	0.036567

TABLE 3.10 – Temps de calcul (en secondes) dans le cas complexe

Ainsi, l'Algorithme 3.6 propose une légère amélioration de point de vue temps de calcul par rapport à d'autres méthodes bien connues, sauf pour l'algorithme de Bini.

3.3.4.4 Analyse du conditionnement

Nous avons fourni des résultats statistiques sur le comportement des quatre méthodes pour calculer l'inverse d'une matrice triangulaire de Toeplitz avec des coefficients réels ou complexes. Cette étude permettra de mesurer le conditionnement de notre approche par rapport à ces prédécesseurs.

Pour cela, nous perturbons la matrice triangulaire de Toeplitz 3000 fois en rajoutant $k \times 10^{-06}$ à ces coefficients avec $k = k^{(r)} + \mathrm{i}k^{(i)} \in \mathbb{C}$ où $k^{(r)}$, $k^{(i)}$ sont aléatoirement dans $]-1, 1[$ dans le cas complexe et $k^{(i)} = 0$ dans le cas réel.

Les Tables 3.11 et 3.12 présentent le comportement de l'erreur relative par rapport aux 3000 perturbations et une tolérance aléatoire ε pour une matrice triangulaire de Toeplitz de taille 2^{10} avec des coefficients réels et complexes. Gd et Bd représentent le nombre de bonnes et de mauvaises perturbations des 3000 séquences perturbées, respectivement.

ε		$..., 10^{-12}$	10^{-11}	10^{-10}	10^{-9}	$10^{-8}, ...$
M-Interpolation	Gd	0	1030	3000	3000	3000
	Bd	3000	1970	0	0	0
Interpolation	Gd	0	0	3000	3000	3000
	Bd	3000	3000	0	0	0
Revised Bini	Gd	0	331	2309	3000	3000
	Bd	3000	2669	691	0	0
Bini	Gd	0	0	0	0	3000
	Bd	3000	3000	3000	3000	0

TABLE 3.11 – Analyse du conditionnement dans le cas réel

ε		$..., 10^{-11}$	10^{-10}	10^{-9}	10^{-8}	10^{-7}	$10^{-6}, ...$
M-Interpolation	Gd	0	188	1976	2253	2997	3000
	Bd	3000	2812	1024	747	3	0
Revised Bini	Gd	0	153	1546	1876	2948	3000
	Bd	3000	2847	1454	1124	52	0
Bini	Gd	0	0	0	2	2840	3000
	Bd	3000	3000	3000	2888	160	0

TABLE 3.12 – Analyse du conditionnement dans le cas complexe

Ainsi, la conclusion qu'on peut retirer est la suivante : en dehors de 3000 perturbations aléatoires :

- Pour $\epsilon > 10^{-7}$: toutes les méthodes sont souvent bonnes.
- Pour $10^{-11} < \epsilon < 10^{-7}$: notre approche est souvent excellente par rapport à d'autres méthodes bien connues. En particulier, l'analyse de conditionnement de l'algorithme de Bini se détériore.
- Au-delà $\epsilon = 10^{-11}$: toutes les méthodes sont souvent mauvaises.

Cette analyse statistique via la perturbation des entrées appliquées à des séquences de Toeplitz, confirme que notre méthode est bien conditionnée par rapport à des méthodes bien connues et la perturbation ci-dessus ne s'éloigne pas généralement du résultat à partir d'une certaine tolérance fixe ϵ avec $\epsilon > 10^{-11}$.

3.4 Conclusion et perspectives

Dans ce chapitre, des algorithmes numériques pour calculer l'inverse d'une matrice triangulaire de Toeplitz avec des coefficients réels et/ou complexes ont été présentés. Nous avons prouvé que le coût du calcul des nouvelles approches pour trouver l'inverse d'une matrice triangulaire de Toeplitz avec des nombres réels et/ou complexes est inférieure ou égale à celles des algorithmes bien connus. Une analyse de l'erreur théorique est également fournie.

Dans les travaux futurs, nous focalisons notre attention sur l'analyse d'erreur et les aspects de calcul de l'inverse d'une matrice de Toeplitz bande par blocs en utilisant le technique de plongement dans une matrice de Toeplitz triangulaire inférieure.

CHAPITRE 4

RÉSOLUTION RAPIDE DES SYSTÈMES DE TOEPLITZ BANDES

4.1 Le problème et sa pertinence

Certaines méthodes numériques très rapides ont été développées au cours des années pour la résolution d'un système de Toeplitz bande, cette dernière a un rôle important dans de nombreuses applications : solutions numériques des équations différentielles [79], solutions numériques de plusieurs chaînes de Markov provenant de la modélisation des problèmes de fils d'attente [38,62] et le problème du traitement d'images [64], où la matrice Toeplitz bande est utilisée comme un préconditionneur pour accélérer la convergence des techniques de gradient conjugué préconditionné (PGC) [24, 25, 28]. La matrice de Toeplitz bande est une classe intéressante de matrices avec laquelle on peut profiter de deux structures : la structure de Toeplitz et la structure bande de la matrice pour donner des algorithmes plus rapides que ceux utilisant sa structure creuse. Dans ce chapitre, nous sommes intéressés par les méthodes directes pour la solution numérique de systèmes linéaires de Toeplitz bandes de très grandes tailles. Dans la littérature, il existe un certain nombre de méthodes directes rapides pour la résolution des grands systèmes linéaires avec des matrices de Toeplitz bandes [15, 18, 32, 35, 37, 57, 60, 77]. L'approche la plus populaire est la réduction cyclique inventée par Bini et Meini [15, 18] qui semble la plus rapide et probablement stable pour le cas symétrique définie positive ou dans le cas non symétrique à diagonale dominante. Cependant, la méthode de la réduction cyclique échoue parfois à donner des résultats précis dans le cas non symétrique. Malyshev et Sadkane [60] ont récemment proposé une alternative basée sur la factorisation spectrale [53, 54, 88] et la formule de Sherman-Morrison-Woodbury [21, 40] qui peut donner des résultats prometteurs dans le cas symétrique et le cas non-symétrique et surtout quand la bande n'est pas trop grande (c'est à dire, $m = \max(m_c, m_r) \ll n$) Bien que l'algorithme décrit dans [60] semble être la meilleure méthode, elle présente une certaine instabilité. En effet, lors de l'utilisation d'une bande large

$m = \max(m_c, m_r)$ de T, la méthode proposée dans [60] échoue parfois ou nécessite un temps de calcul plus important. Pour résoudre ce problème, nous introduisons une nouvelle approche basée sur l'extension de la matrice de Toeplitz bande (4.2) en plusieurs lignes en dessus et de plusieurs colonnes à droite et d'affecter des zéros et des constantes non nulles dans chacune de ces lignes et colonnes de telle sorte que la matrice augmentée a une structure de Toeplitz triangulaire inférieure. Le coût de calcul de l'algorithme proposé est environ $O(n \log n + m^2)$ opérations arithmétiques. Ce travail a fait l'objet d'une publication intitulée "A fast algorithm for solving banded Toeplitz systems" dans la revue "Computer & Mathematics with Applications [9]. Un système de Toeplitz bande s'écrit de la manière suivante :

$$Tx = f \tag{4.1}$$

où

$$T = \begin{pmatrix} t_0 & t_{-1} & \cdots & t_{-m_r} & & & \\ t_1 & t_0 & t_{-1} & & \ddots & & \\ \vdots & \ddots & \ddots & \ddots & & \ddots & \\ t_{m_c} & & t_1 & t_0 & t_{-1} & & t_{-m_r} \\ & \ddots & & \ddots & \ddots & \ddots & \vdots \\ & & \ddots & & \ddots & \ddots & t_{-1} \\ & & & t_{m_c} & \cdots & t_1 & t_0 \end{pmatrix} \tag{4.2}$$

où T une matrice de Toeplitz bande de taille $n \times n$, $t_{m_c} \neq 0$, $t_{-m_r} \neq 0$ et $m_c, m_r \in \mathbb{N}$.

4.2 Algorithmes rapides

Dans cette section, nous allons examiner brièvement les cas importants pour résoudre un système linéaire de Toeplitz bande $Tx = f$ défini dans l'équation (4.1). Les deux premiers algorithmes sont décrits par Bini et Pan dans [21].

4.2.1 Méthode via la formule de Sherman-Morrison-Woodbury et la matrice circulante

Proposition 4.2.1 *Soit T une matrice de Toeplitz bande de taille $n \times n$ définie par l'équation (4.2), avec une bande de langueur $2k+1$ où $m_r = k$ et $m_c = k$. Alors, nous pouvons décomposer T sous la forme $C + R$ avec C une matrice circulante et R une matrice de rang au plus $2k$, avec*

$$C = \begin{pmatrix} t_0 & t_{-1} & \cdots & t_{-k} & 0 & t_k & \dots & t_1 \\ t_1 & t_0 & t_{-1} & & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & & \ddots & \ddots & t_k \\ t_k & & t_1 & t_0 & t_{-1} & & \ddots & 0 \\ 0 & \ddots & & \ddots & \ddots & \ddots & & t_{-k} \\ t_{-k} & \ddots & \ddots & & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & & \ddots & \ddots & t_{-1} \\ t_{-1} & \dots & t_{-k} & 0 & t_k & \cdots & t_1 & t_0 \end{pmatrix},$$

et

$$R = - \begin{pmatrix} 0 & \dots & 0 & t_k & \dots & t_1 \\ \vdots & \ddots & & \ddots & \ddots & \vdots \\ 0 & & \ddots & & \ddots & t_k \\ t_{-k} & \ddots & & \ddots & & 0 \\ \vdots & \ddots & \ddots & & \ddots & \vdots \\ t_{-1} & \dots & t_{-k} & 0 & \dots & 0 \end{pmatrix}.$$

Corollaire 4.2.1 *Nous supposons que T vérifie la Proposition 4.2.1. Alors, il existe une méthode directe pour résoudre le système $Tx = f$ avec un coût de $O(n \log n) + O(nk \log) + O(k^3)$ flops.*

Preuve. L'idée est d'écrire $R = GH^T$, avec G, $H \in \mathbb{K}^{n \times r}$ où $r = 2k$, et en utilisant Formule de Sherman-Morrison-Woodbury définie par le Théorème 2.2.3 dans le système $(C + R)x = f$. On obtient,

$$x = C^{-1}f - C^{-1}G(T_r + H^T C^{-1} G)^{-1} H^T C^{-1} f.$$

Pour plus de détails voir [51, Corollary 5.3.3]. ■

4.2.2 Augmentation dans une matrice circulante

La notion d'augmentation d'une matrice de Toeplitz bande dans une matrice circulante a été étudiée par Bini

Proposition 4.2.2 *Soit T une matrice de Toeplitz bande de taille $n \times n$. Nous pouvons intégrer T dans une matrice circulante C, avec r sa première colonne donnée par*

$$r = (t_0, \dots, t_k, 0, \dots, 0, t_{-k}, \dots, t_{-1})$$

tel que

$$C = \begin{pmatrix} t_0 & t_{-1} & \cdots & t_{-k} & 0 & t_k & \dots & t_1 \\ t_1 & t_0 & t_{-1} & & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & & \ddots & \ddots & t_k \\ t_k & & t_1 & t_0 & t_{-1} & & \ddots & 0 \\ 0 & \ddots & & \ddots & \ddots & \ddots & & t_{-k} \\ t_{-k} & \ddots & \ddots & & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & & \ddots & \ddots & t_{-1} \\ t_{-1} & \dots & t_{-k} & 0 & t_k & \cdots & t_1 & t_0 \end{pmatrix} \tag{4.3}$$

$$= \begin{pmatrix} T & B \\ D & A \end{pmatrix}. \tag{4.4}$$

Avec A une matrice de taille $k \times k$, D et B^T sont de Toeplitz creuse de taille $k \times n$.

Proposition 4.2.3 *La résolution d'un système de Toeplitz bande en utilisant la Proposition 4.2.2 nécessite $O(n \log n) + O(kn) + O(k \log^2, k)$ flops.*

Preuve. Voir [51, Proposition 5.3.13]. ■

4.2.3 Algorithme de la réduction cyclique

Dans cette section, nous rappelons tout d'abord l'algorithme de la réduction cyclique initialement introduite pour la solution numérique de l'équation de Poisson [40], utilisé dans [14] pour calculer le vecteur invariant pour une matrice stochastique afin de résoudre le système tridiagonale de Toeplitz par blocs. Puis nous introduisons le concept de rang de déplacement et nous rappelons quelques propriétés qui sont nécessaires pour la conception et l'analyse de cette approche. La réduction cyclique est avérée être un algorithme qui est très puissant pour les problèmes structurés. En particulier, pour les matrices de Toeplitz par blocs et tridiagonale par blocs, la méthode est particulièrement utile. L'idée de base est d'éliminer la moitié des inconnues, regrouper les équations et éliminer de nouveau la moitié des inconnues. On considère un système tri-diagonal

$$Ax = v,$$

avec

$$A = \begin{pmatrix} d_1 & f_1 & & & \\ e_1 & d_2 & f_2 & & \\ & \ddots & \ddots & \ddots & \\ & & e_{n-1} & d_{n-1} & f_{n-1} \\ & & & e_n & d_n \end{pmatrix},$$

afin d'expliciter les différentes étapes de la méthode. L'opération fondamentale dans la réduction cyclique est l'élimination simultanée des inconnues impaires indexées. Cette opération peut être aussi décrite comme une factorisation LU par blocs d'une permutation de colonnes et de lignes pour la matrice A. Soit S la matrice de permutation définie tel que :

$$[S(1,2,\ldots,n)]^T = (1,3,\ldots,|2,4,\ldots)^T.$$

La matrice permutée SAS^T devient alors :

$$\left[\begin{array}{ccc|ccc} d_1 & & & f_1 & & \\ & d_3 & & e_3 & f_3 & \\ & & \ddots & & \ddots & \ddots \\ \hline e_2 & f_2 & & d_2 & & \\ & e_4 & \ddots & & d_4 & \\ & & \ddots & & & \ddots \end{array}\right].$$

Ainsi, l'élimination des inconnues impaires indexées est équivalent à calculer une factorisation LU partielle comme suit :

$$SAS^T = \left[\begin{array}{ccc|ccc} 1 & & & 0 & & \\ & \ddots & & & \ddots & \\ & & \ddots & & & 0 \\ \hline l_1 & m_1 & & 1 & & \\ & l_2 & \ddots & & \ddots & \\ & & \ddots & & & 1 \end{array}\right] \left[\begin{array}{ccc|ccc} d_1 & & & f_1 & & \\ & d_3 & & e_3 & f_3 & \\ & & \ddots & & \ddots & \ddots \\ \hline 0 & & & d_1^{(1)} & f_1^{(1)} & \\ & \ddots & & e_2^{(1)} & d_2^{(1)} & \ddots \\ & & \ddots & & \ddots & \ddots \end{array}\right].$$

Pour n paire, cette décomposition est calculée pour $i = 1,\ldots,\frac{n}{2}-1$ avec

$$m_i = f_{2i}/d_{2i+1}, \quad l_i = e_{2i}/d_{2i-1}, \quad l_{n/2} = e_n/d_{n-1}$$

$$f_i^{(1)} = -m_i f_{2i+1}, \quad e_{i+1}^{(1)} = -l_{i+1} e_{2i+1}$$

$$d_i^{(1)} = d_{2i} - l_i f_{2i-1} - m_i e_{2i+i}, \quad d_{n/2}^{(1)} = d_n - l_{n/2} f_{n-1}.$$

La réduction est avérée être un algorithme qui est très puissant pour résoudre les problèmes des matrices structurées. En particulier, pour les matrices qui sont (blocs) Toeplitz et (bloc) tri-diagonal, le méthode est particulièrement utile. L'idée de base est d'éliminer la moitié des inconnues, regrouper les équations et éliminer de nouveau la moitié des inconnues. Cette simple idée initialement introduite pour la solution numérique de l'équation de Poisson [40], utilisé dans [14] pour calculer le vecteur invariant pour une matrice stochastique afin de résoudre le système tridiagonale de Toeplitz par blocs. En particulier, l'algorithme est bien décrit en termes

de complément de Schur 2.2.3 dans l'article de Gander et Golub [83] où une bibliographie approfondie est également donnée. L'idée principale sur laquelle se base la réduction cyclique (CR) est de réorganiser les blocs lignes et colonnes de la matrice T par l'intermédiaire d'une permutation pair-impair et d'éliminer les inconnues indexés impairs au moyen du complément de Schur. La fonctionnalité intéressante est que le nouveau système de la moitié de la taille obtenue de cette manière a le même bloc tridiagonale Toeplitz par blocs, la structure que le système d'origine de sorte que la même procédure peut être répétée jusqu'à ce que cycliquement on arrive à un seul système de taille $m \times m$.

4.2.3.1 Préliminaires

Sans perte de généralité, supposons que $m_r \geq m_c$ et soit n un entier multiple de m ; c'est à dire, $n = mq$, pour q est un entier. La matrice (4.2) peut être partitionnée en $m \times m$ blocs ce qui donne une matrice tridiagonale par blocs de Toeplitz.

$$T_n = \begin{bmatrix} A_0 & A_{-1} & & & \\ A_1 & A_0 & A_{-1} & & \\ & \ddots & \ddots & \ddots & \\ & & A_1 & A_0 & A_{-1} \\ & & & A_1 & A_0 \end{bmatrix} \tag{4.5}$$

avec

$$A_1 = \begin{bmatrix} a_m & a_{m-1} & \cdots & a_1 \\ 0 & a_m & \ddots & \vdots \\ \vdots & \ddots & \ddots & a_{m-1} \\ 0 & \cdots & 0 & a_m \end{bmatrix},$$

$$A_0 = \begin{bmatrix} a_0 & a_{-1} & \cdots & a_1 \\ a_1 & a_0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & a_{-1} \\ a_{m-1} & \cdots & a_1 & a_0 \end{bmatrix},$$

$$A_{-1} = \begin{bmatrix} a_{-m} & 0 & \cdots & 0 \\ a_{-m+1} & a_{-m} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ a_{-1} & \cdots & a_{-m+1} & a_{-m} \end{bmatrix}.$$

A la lumière des blocs ci-dessus, nous pouvons associer à T la matrice polynomiale

$$A(z) = A_{-1} + zA_0 + z^2 A_1$$

.

4.2.3.2 Réduction cyclique

Dans cette section, nous rappelons tout d'abord l'algorithme de la réduction cyclique initialement introduit pour la solution numérique de l'équation de Poisson [40], utilisé dans [14] pour calculer le vecteur invariant pour une matrice stochastique afin de résoudre le système tridiagonale de Toeplitz par blocs. On considère le système suivant :

$$Tx = b \tag{4.6}$$

où T est définie en (4.5), et les vecteurs x et b partitionner en blocs x_k, b_k de dimension m, respectivement. Notons que $x = (x_k)$ et $b = (b_k)$. En effectuant une permutation impair-pair des blocs lignes et des blocs colonnes dans (4.6), nous obtenons ainsi :

$$\begin{pmatrix} D_1^{(0)} & U^{(0)} \\ L^{(0)} & D_2^{(0)} \end{pmatrix} \begin{pmatrix} x_+^{(0)} \\ x_-^{(0)} \end{pmatrix} = \begin{pmatrix} b_+^{(0)} \\ b_-^{(0)} \end{pmatrix} \tag{4.7}$$

où $x_+^{(0)} = (x_{2k}),\quad x_-^{(0)} = (x_{2k-1}),\quad b_+^{(0)} = (b_{2k}),\quad b_-^{(0)} = (b_{2k-1})$

$$D_1^{(0)} = D_2^{(0)} = \begin{pmatrix} A_0 & & & \\ & A_0 & & \\ & & \ddots & \\ & & & \ddots \end{pmatrix}, L^{(0)} = \begin{pmatrix} A_{-1} & & & \\ A_1 & A_{-1} & & \\ & \ddots & \ddots & \\ & & \ddots & \ddots \end{pmatrix},$$

$$U^{(0)} = \begin{pmatrix} A_1 & A_{-1} & & \\ & A_1 & \ddots & \\ & & \ddots & \ddots \\ & & & \ddots \end{pmatrix}.$$

En appliquant l'élimination de Gauss par blocs au système (4.7), nous obtenons alors :

$$\begin{cases} (D_2^{(0)} - L^{(0)} \left(D_1^{(0)}\right)^{-1} U^{(0)}) x_-^{(0)} &= b_-^{(0)} - L^{(0)} \left(D_1^{(0)}\right)^{-1} b_+^{(0)} \\ x_+^{(0)} &= \left(D_1^{(0)}\right)^{-1} (b_+^{(0)} - U^{(0)} x_-^{(0)}) \end{cases} \tag{4.8}$$

Notons par

$$T^{(1)} = D_2^{(0)} - L^{(0)} \left(D_1^{(0)}\right)^{-1} U^{(0)}, \tag{4.9}$$

le complément de Schur pour $D_2^{(0)}$, $x^{(1)} = x_-^{(0)}$ et $b^{(1)} = b_-^{(0)} - L^{(0)} {D_1^{(0)}}^{-1} b_+^{(0)}$ alors le système (4.8) se réduit à la résolution de cette équation :

$$T^{(1)} x^{(1)} = b^{(1)}. \tag{4.10}$$

Il est intéressant d'observer que $T^{(1)}$ est une matrice tridiagonale par blocs qui a la structure de Toeplitz par blocs à l'exception du premier bloc $\widehat{A}^{(1)}$, c'est à dire :

$$T^{(1)} = \begin{pmatrix} \widehat{A}^{(1)} & A_{-1}^{(1)} & & \\ A_1^{(1)} & A_0^{(1)} & A_{-1}^{(1)} & \\ & A_1^{(1)} & A_0^{(1)} & \ddots \\ & & \ddots & \ddots \end{pmatrix}.$$

En d'autre terme, $T^{(1)}$ est déterminé uniquement via les blocs $A_{-1}^{(1)}$, $A_0^{(1)}$, $A_1^{(1)}$, $\widehat{A}_1^{(1)}$. Une fois $x^{(1)}$ a été calculée, la solution du système (4.8) peut être récupérée par la substitution remontante à travers (4.8). Afin de calculer la solution $x^{(1)}$, nous appliquons la même réduction (réduction cyclique) au système (4.10). De cette manière, on obtient une séquence de systèmes $\{T^{(j)}x^{(j)} = b^{(j)}\}$, où $T^{(0)} = T$, et $T^{(j)}$ est une matrice tridiagonale par blocs ayant, à l'exception du bloc de la première diagonale, la structure de Toeplitz par blocs. Chaque matrice $T^{(j)}$ est déterminée uniquement par les blocs $A_{-1}^{(j)}$, $A_0^{(j)}$, $A_1^{(j)}$, $\widehat{A}_1^{(j)}$. Ainsi, pour un système de Toeplitz bande $T_n x = b$, où $n = mq$, $q = 2^k$, p est un entier, la réduction cyclique génère une séquence de systèmes

$$T_{n_j}^{(j)} x^{(j)} = b^{(j)},\ j = 1, 2, \ldots, p,$$

avec $n_j = m2^{p-j}$.

Pour les matrices de taille $n = m2^p$, le procédé de la réduction cyclique est effectué en p étapes, et la solution de (4.6) est calculée par le moyen de substitution descendante, en appliquant de manière récursive (4.8) jusqu'à la $j^{\text{i-ème}}$ étape. Les entrées des blocs $A_i^{(j+1)}$, $i = -1, 0, 1$ et $\widehat{A}^{(j+1)}$ obtenus à la $(j+1)^{\text{i-ème}}$ étape qui définissent la matrice $T_n^{(j+1)}$, sont liées aux entrées des blocs $A_i^{(j)}$, $i = -1, 0, 1$ et $\widehat{A}^{(j)}$ obtenus à la $j^{\text{i-ème}}$ étape, par des relations simples. Par analogie, nous associons à $T^{(j)}$ la matrice polynomiale suivante :

$$A^{(j)}(z) = A_{-1}^{(j)} + zA_0^{(j)} + z^2 A_1^{(j)}. \tag{4.11}$$

En effet, d'après la relation (4.9) nous pouvons facilement déduire que

$$\begin{cases} A^{(j+1)}(z) & = zA_0^{(j)} - (A_{-1}^{j} + zA_1^{(j)}){A_0^{(j)}}^{-1}(A_{-1}^{(j)} + zA_1^{(j)}), \\ \widehat{A}^{(j+1)} & = \widehat{A}^{(j)} - A_{-1}^{(j)}{A_0^{(j)}}^{-1}A_1^{(j)}. \end{cases} \tag{4.12}$$

Les résultats ci-dessus peuvent conduire à un algorithme direct pour la solution du système $T_n x = b$, où T_n est une matrice de taille $n \times n$, avec $n = mq$, $q = 2^p$, p est un entier positif en respectant la mise à jour du second membre $b^{(j)}$ et avec une étape de substitution descendante. Le coût de ce dernier algorithme peut être facilement estimé. En fait, en désignant par $t(m)$ le coût de la résolution d'un système $m \times m$ de type Toeplitz, il se suit que $O(n \log m + (m \log m + t(m)) \log \frac{n}{m})$ opérations arithmétiques sont suffisantes pour faire tourner l'algorithme. La solution d'un système de type Toeplitz de taille $m \times m$ est en fonction des propriétés spécifiques de la matrices impliquées (symétrique, définie positive) des algorithmes comme, rapide, super-rapide, peuvent être utilisé pour le coût $t(m)$, étant $O(m^2)$),

$O(m\log^2 m)$:

$$\begin{cases} A_{-1}^{(j+1)} &= -A_{-1}^{(j)}A_0^{(j)^{-1}}A_{-1}^{(j)} \\ A_0^{(j+1)} &= A_0^{(j)} - A_1^{(j)}A_0^{(j)}A_{-1}^{(j)} - A_{-1}^{(j)}A_0^{(j)^{-1}}A_j^{(j)} \\ A_1^{(j+1)} &= -A_1^{(j)}A_0^{(j)^{-1}}A_1^{(j)} \\ \widehat{A}^{(j+1)} &= \widehat{A}^{(j)} - A_{-1}^{(j)}A_0^{(j)^{-1}}A_1^{(j)} \end{cases} \tag{4.13}$$

Ensuite, il est immédiat de voir par inspection directe que l'équation (4.12) peut aussi être écrite sous la forme fonctionnelle suivante :

$$A^{j+1}(z^2) = A^{(j)^{-1}}(z)A_0^{(j)^{-1}}A^{(j)}(-z) \tag{4.14}$$

4.2.3.3 Propriétés de convergence

Les propriétés de convergence des blocs générés à chaque étape de la réduction cyclique sont bien vérifiées dans le cas où la matrice est diagonale dominante, ses propriétés étaient étudiées par plusieurs auteurs [45, 87].
Aussi, lorsque la matrice est symétrique la convergence de méthode de la réduction cyclique est bien garantie cela est liée à l'emplacement des zéros du polynôme $P(z) = \det(A_{-1} + zA_0 + z^2A_1)$ [16].

4.3 Méthode via la factorisation spectrale et Sherman-Morrison-Woodbury

Dans cette section, nous étudions l'approche proposée dans [60], introduite initialement dans [37] pour la résolution d'un système de Toeplitz bande dans le cas scalaire. Dans [60], Malyshev et Sadkane ont proposé l'utilisation de la technique de la factorisation spectrale de la fonction génératrice associée à la matrice de Toeplitz bande en se basant sur la formule d'inversion de Sherman-Morrison-Woodbury. L'objectif est d'étendre la méthode à partir de [30] pour les systèmes non symétriques de grande taille.

Rappelons que si T est symétrique définie positive ou à diagonale dominante, la réduction cyclique semble être le meilleur choix. Dans le cas asymétrique, cette méthode peut parfois ne pas donner des résultats précis. Cependant, la méthode de Malyshev et Sadkane peut fournir une alternative beaucoup plus meilleure.

L'analyse du conditionnement de la matrice T (4.2) pour n grand à partir de [31] est utilisé en passant par la fonction génératrice correspondante à T, ce qui donne la fonction rationnelle suivante :

$$a(\lambda) = a_{m_c}\lambda^{m_c} + \cdots + a_1\lambda + a_0 + a_{-1}\lambda^{-1} + \cdots + a_{-m_r}\lambda^{-m_r} \tag{4.15}$$

Notons par

$$p(\lambda) \equiv a(\lambda)\lambda^{m_r} = a_{m_c}\lambda^{m_c} + \cdots + a_1\lambda^{m_r+1} + a_0\lambda^{m_r} + a_{-1}\lambda^{m_r-1} + \cdots + a_{-m_r}$$

un polynôme de racines λ_i vérifiant

$$0 < |\lambda_{-1}| \leq |\lambda_1| \leq |\lambda_2| \leq \cdots \leqslant |\lambda_{m_r}| \leq |\lambda_{m_r+1}| \leq \cdots \leq |\lambda_{m_c+m_r}| \tag{4.16}$$

Si $|\lambda_{m_r}| < 1$ et $|\lambda_{m_r+1}| > 1$, alors le nombre de conditionnement de T est borné pour tout n. Si $|\lambda_{m_r}| \leq 1$ et $|\lambda_{m_r+1}| \geq 1$, alors le nombre de conditionnement de T croît de façon polynomiale lorsque $n \to \infty$. Dans d'autres cas, le nombre de conditionnement de T croît de façon exponentielle lorsque $n \to \infty$.

La méthode proposée dans [60] est une combinaison efficace d'une factorisation spectrale stable de la fonction génératrice et de l'utilisation des calculs rapides sur les matrices de Toeplitz, qui concernent essentiellement le système de Toeplitz tridiagonale bande. Les coefficients polynomiaux spectrales sont obtenus à partir des sous-espaces de déflation à droite correspondant à des valeurs propres à l'intérieur et à l'extérieur du disque unité ouvert de la matrice "Pencil" définie en 2.2.4, construite à partir des coefficients de la fonction génératrice. Ces sous-espaces de déflation sont calculées par l'algorithme QZ [1] cité dans [50].

Le coût de calcul de cet algorithme est environ $O(n \log_2 m) + O(m^3)$ opérations arithmétiques pour une matrice de Toeplitz bande de taille $n \times n$ avec m =max(m_c, m_r). Le coût $O(m^3)$ est essentiellement due au calcul de la factorisation spectrale. Lorsque $n >> m$, le coût de l'algorithme devient équivalent à celui de la réduction cyclique [15, 18].

Théorème 4.3.1 *On suppose que $|\lambda_{m_r}| \leqslant 1$ et $|\lambda_{m_r+1}| \geq 1$, alors il existe un scalaire s non nul et deux polynômes*

$$L(\lambda) = 1 + l_1\lambda + \cdots + l_{m_c}\lambda^{m_c} \tag{4.17}$$

$$U(\lambda) = 1 + u_1\lambda + \cdots + u_{m_r}\lambda^{m_r} \tag{4.18}$$

dont les racines se trouvent à l'extérieur du disque unité ouvert et tel que

$$a(\lambda) = L(\lambda).s.U(\lambda^{-1}). \tag{4.19}$$

Par ailleurs, la matrice diagonale $D = sI$ et les matrices triangulaires de Toeplitz bandes

$$L = \begin{pmatrix} 1 & & & & & \\ l_1 & 1 & & & & \\ \vdots & \ddots & & & & \\ l_{m_c} & & l_1 & 1 & & \\ & \ddots & & \ddots & \ddots & \\ & & l_{m_c} & & l_1 & 1 \end{pmatrix}, \quad U = \begin{pmatrix} 1 & u_1 & & u_{m_r} & & \\ & \ddots & \ddots & & \ddots & \\ & & 1 & u_1 & & u_{m_r} \\ & & & \ddots & \ddots & \vdots \\ & & & & \ddots & u_1 \\ & & & & & 1 \end{pmatrix} \tag{4.20}$$

1. Generalized Shur Decomposition

satisfont l'identité suivante :

$$T = LDU + \begin{pmatrix} \breve{L}\breve{D}\breve{U} & & \\ & 0 & \\ & & \ddots \\ & & & 0 \end{pmatrix} \tag{4.21}$$

où

$$\breve{L}\breve{D}\breve{U} = \begin{pmatrix} l_m & \cdots & l_1 \\ & \ddots & \vdots \\ & & l_m \end{pmatrix} .s. \begin{pmatrix} u_m & & \\ \vdots & \ddots & \\ u_1 & \cdots & u_m \end{pmatrix} \tag{4.22}$$

avec la convention $l_k = 0$ *pour* $k > m_c$, *et* $u_k = 0$ *pour* $k > m_r$.

Preuve. Soit $p_{m_r} = (\lambda - \lambda_1)(\lambda - \lambda_2)\cdots(\lambda - \lambda_{m_r})$ et
$p_{m_c} = a_{m_c}(\lambda - \lambda_{m_r+1})\cdots(\lambda - \lambda_{m_c+m_r})$.
Alors, $s = p_{m_c}(0)$, $l(\lambda) = p_{m_c}(\lambda)/s$, $u(\lambda) = \lambda^{m_r} p_{m_r}(1/\lambda)$. Les propriétés (4.21) et (4.22) sont faciles à vérifier par des calculs directs. ■

4.3.1 La factorisation spectrale

Dans le cas où $m_c = m_r$ et $a(\lambda) \geq 0$ pour $|\lambda| = 1$, plusieurs études sont faites, pour avoir des factorisations de type (4.19). Les factorisations sont bien étudiées et comparées dans [58]. Des factorisations spectrales basées sur l'itération de Graeffe et les techniques de l'interpolation ou l'évaluation, ont été proposées dans [12]. Une étude approfondie sur la factorisation spectrale peut être trouvée dans [74]. Nous allons proposer ici l'étude de la factorisation de Miloud & Malyshev [60].

Dans ce qui suit, les polynômes $l(\lambda)$ et $u(\lambda)$ sont représentés par les coefficients l_i et u_i, respectivement. On définit aussi par $A - \lambda B$ la matrice "Pencil" avec

$$A = \begin{pmatrix} 0 & 1 & & & \\ & 0 & 1 & & \\ & & \ddots & \ddots & \\ & & & 0 & 1 \\ -a_{-m_r} & -a_{1-m_r} & \cdots & \cdots & -a_{m_c-1} \end{pmatrix}, \quad B = \begin{pmatrix} 1 & & & & \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & \\ & & & & a_{m_c} \end{pmatrix}$$

dont les valeurs propres coïncident avec les racines du polynôme $p(\lambda)$.

Théorème 4.3.2 *Notons par* $V \in \mathbb{C}^{(m_c+m_r)\times m_c}$ *la matrice dont les colonnes s'étendent aux sous-espaces de déflation à droite pour la matrice "pencil"* $A - \lambda B$ *qui correspond aux valeurs*

propres $\lambda_1, \cdots, \lambda_{m_r}$ *et* $\lambda_{m_r+1}, \cdots, \lambda_{m_r+m-c}$ *respectivement. Les partitions de* V *et* W *sont les suivantes*

$$V = \begin{pmatrix} V_1 \\ v^* \\ V_2 \end{pmatrix}, \; V_1 \in \mathbb{C}^{m_r \times m_r}, \; v^* \in \mathbb{C}^{1 \times m_r}, \; V_2 \in \mathbb{C}^{(m_c-1)\times m_r} \tag{4.23}$$

$$W = \begin{pmatrix} W_1 \\ W^* \\ W_2 \end{pmatrix}, \; W_1 \in \mathbb{C}^{(m_r-1)\times m_C}, \; w^* \in \mathbb{C}^{1 \times m_c}, \; W_2 \in \mathbb{C}^{m_c \times m_c}. \tag{4.24}$$

Si les sous matrices V_1 *et* W_2 *sont inversibles, alors les coefficients de* $l(\lambda)$ *et* $u(\lambda)$ *sont donnés par*

$$(u_{m_r} u_{m_r-1} \cdots u_1) = -v^* V_1^{-1} \tag{4.25}$$

$$(l_1 l_2 \cdots l_{m_c}) = -w^* W_2^{-1}. \tag{4.26}$$

Preuve. Voir [60]. ■

Remarque 4.3.1 *D'après le Théorème 4.3.2, nous pouvons conclure :*

- *Le calcul des matrices* V *et* W *est accompli par l'intermédiaire de la forme de Schur généralisée de la matrice "Pencil"* $A - \lambda B$ *définie dans 2.2.4, suivie par la réorganisation des valeurs propres spécifiées [50].*
- *La non singularité des blocs* V_1 *et* W_2 *est justifiée dans [40]. Le mal conditionnement de* V_1 *et* W_2 *peut conduire à une factorisation spectrale inexacte.*
- *Le calcul direct de* s *par la formule* $s = a_{m_c}/l_{m_c}$ *ou* $s = a_{-m_r}/u_{m_r}$ *peut être soumis à des erreurs si* l_{m_c} *et* u_{m_c} *ne sont pas calculées à une précision relativement élevée. Pour une matrice symétrique* T, $s = a_0/(1+l_1u_1+\ldots+l_mu_m)$ *peut être donné d'une manière stable, où* $l_i = u_i^*, \; i = 1, \ldots, m$.

4.3.2 La résolution d'un système de Toeplitz bande

On suppose que les conditions (4.20) et (4.22) sont bien vérifiées. L'application de la formule de Sherman-Morrison-Woodbury définie dans le Théorème 2.2.3 à

$$T = LDU + E\check{L}\check{D}\check{U}E^T,$$

où $E = \begin{pmatrix} I_m \\ 0 \end{pmatrix}$ génère la représentation suivante :

$$\begin{aligned} T^{-1} =& (LDU)^{-1} - (LDU)^{-1}E\check{L}\check{D}[I + \check{U}E^T(LDU)^{-1}E\check{L}\check{D}]^{-1}\check{U}E^T(LDU)^{-1} \\ =& (LDU)^{-1} - (LDU)^{-1}E\check{L}[I + \check{U}E^T(LU)^{-1}E\check{L}]^{-1}\check{U}E^T(LDU)^{-1}. \end{aligned}$$

Notons par

$$y = (LDU)^{-1} f = \frac{1}{s} U^{-1} L^{-1} f. \tag{4.27}$$

Ce qui donne

$$x = y - u^{-1} L^{-1} \begin{pmatrix} \check{L} \\ 0 \end{pmatrix} Q^{-1} \begin{pmatrix} \check{U} & 0 \end{pmatrix} y, \tag{4.28}$$

où

$$Q = I + \check{U} C \check{L} \tag{4.29}$$

et

$$C = \begin{pmatrix} I_m & 0 \end{pmatrix} U^{-1} L^{-1} \begin{pmatrix} I_m \\ 0 \end{pmatrix}. \tag{4.30}$$

Algorithm 4.1 Algorithme via la factorisation spectrale et Sherman-Morrison-Woodbury

Étape 0 : Factoriser la fonction génératrice (4.15) comme (4.19).

Étape 1 : Calculer la matrice Q en utilisant (4.29)

Étape 2 : Calculer la solution de x en utilisant (4.28).

Remarque 4.3.2 *En résumé, le coût total de l'Algorithme 4.1 s'élève à* $O(m^3) + O(n \log_2 m) + fact(m)$ *opérations arithmétiques, où* $fact(m)$ *est le coût nécessaire pour la factorisation de (4.15), cela requiert* $66m^3$ *opérations arithmétiques [40]. De plus, cette méthode devient plus intéressante dans le cas où* $n \gg m$ *avec un coût total dominé par* $O(n \log_2 m)$.

4.3.3 Étude de la stabilité

Dans cette section, nous présentons l'estimation de l'erreur directe "forward error"

$$\| x - \tilde{x} \| / \| x \|$$

où $\tilde{x}$ est la solution d'un système triangulaire de Toeplitz bande calculée en virgule flottante via l'Algorithme 4.1. Sans perte de généralité, soit $s = 1$ et notons par $A = LU$, la solution exacte calculée par (4.28) est donnée par :

$$x = A^{-1} f - A^{-1} E \check{L} Q^{-1} \check{U} E^T A^{-1} f. \tag{4.31}$$

Supposons que $y = A^{-1} f$ est stable directement. Sa valeur calculée est de la forme $\tilde{y} = (A + \Delta_1)^{-1} f$, où Δ_1 est la perturbation tel que $\| \Delta_1 \| = O(\varepsilon_{machine}) \| A \|$. D'où,

$$\tilde{x} = (A + \Delta_1)^{-1} f - (A + \Delta_2)^{-1} E L Q^{-1} \check{U} E^T (A + \Delta_1)^{-1} f \tag{4.32}$$

où

$$\| \Delta_i \| = O(\varepsilon_{machine}) \| A \|, \quad i = 1, 2$$

si on enlève les termes de perturbation du seconde degré, on obtient les équations asymptotiques comme suit :

$$(A + \Delta_i)^{-1} = A^{-1} - A^{-1}\Delta_i A^{-1}, \quad i = 1, 2 \tag{4.33}$$

Alors (4.32) et (4.31) donnent après un simple calcul, l'inégalité suivante :

$$\frac{\| x - \tilde{x} \|}{\| x \|} \leq O(\varepsilon_{machine})(\| T \| \| T^{-1} \|)^{\frac{3}{2}} \tag{4.34}$$

et pour plus de détails voir [60].

4.4 Méthode via Toeplitz triangulaire

La méthode de la réduction cyclique [15] est une méthode efficace dans le cas où la matrice de Toeplitz soit symétrique ou à diagonale dominante mais elle est coûteuse si la taille de la bande est grande. De plus, la méthode basée sur la factorisation spectrale et Sherman-Morrison-Woodbury introduite dans [60] n'est pas aussi efficace dans le cas où la bande de la matrice est très grande. D'après la Remarque 4.3.1, elle nécessite la non-singularité de V_1, W_2 et la matrice Q. Cela nous a permis d'inventer une alternative plus meilleure quelle que soit la taille de la bande choisie en respectant l'efficacité et la précision.

4.4.1 Augmentation dans une matrice triangulaire inférieure de Toeplitz

Soit T une matrice de Toeplitz bande définie comme (4.2). Nous sommes intéressés par la résolution du problème

$$Tx = f \tag{4.35}$$

Soit T une matrice de Toeplitz bande définie comme (4.2), x le vecteur inconnu, et f est le second membre. Dans ce qui suit, on donne un algorithme qui calcule la solution de (4.35) en utilisant l'idée d'intégrer la matrice de Toeplitz bande dans une matrice de Toeplitz triangulaire bande plus grande. Plus précisément, nous intégrons T dans une matrice de Toeplitz triangulaire inférieure par bande M de taille $(n+p) \times (n+p)$ où la première colonne de M est donnée par :

$$r = (t_{-m_r}, \cdots, t_{-1}, t_0, t_1, \cdots, t_{m_c}, \underbrace{0, \cdots, 0}_{n-(m_c+1)})^T.$$

Plus précisément,

$$
M = \left(\begin{array}{ccc|cccccc|ccc}
t_{-m_r} & & & & & & & & & & \\
\vdots & \ddots & & & & & & & & & \\
t_{-1} & \cdots & t_{-m_r} & & & & & & & & \\
\hline
t_0 & t_{-1} & \cdots & t_{-m_r} & & & & & & & \\
t_1 & t_0 & t_{-1} & & & & & & & & \\
\vdots & \ddots & \ddots & \ddots & & \ddots & & & & & \\
t_{m_c} & & t_1 & & t_{-1} & & t_{-m_r} & & & & \\
 & \ddots & & \ddots & \ddots & \ddots & \vdots & & t_{-m_r} & & \\
 & & & & & \ddots & t_{-1} & & \vdots & \ddots & \\
 & & & t_{m_c} & \cdots & t_1 & t_0 & & t_{-1} & \cdots & t_{-m_r}
\end{array}\right) \tag{4.36}
$$

$$
= \left(\begin{array}{ccc|c}
L & 0 & \cdots & 0 \\
\hline
 & & & \vdots \\
 & T & & 0 \\
 & & & L
\end{array}\right), \quad \text{où } L = \begin{pmatrix} t_{-m_r} & & \\ \vdots & \ddots & \\ t_{-1} & \cdots & t_{-m_r} \end{pmatrix}.
$$

Pour simplifier, nous notons que $p = m_r$, $q = m_c$.

Proposition 4.4.1 *Soit M une matrice triangulaire inférieure de Toeplitz tel que définie dans (4.36). Ainsi, M^{-1} est une matrice triangulaire inférieure de Toeplitz définie par sa première colonne :*

$$
r = (t_{-m_r}, \cdots, t_{-1}, t_0, t_1, \cdots, t_{m_c}, \underbrace{0, \cdots, 0}_{n-(m_c+1)})^T.
$$

En outre, M^{-1} peut être partitionnée comme suit

$$M^{-1} = \left(\begin{array}{cccc|cccccc} v_1 & & & & & & & & & \\ v_2 & v_1 & & & & & & & & \\ \vdots & v_2 & \ddots & & & & & & & \\ v_{n-p+1} & & \ddots & v_1 & & & & & & \\ \vdots & \ddots & & v_2 & v_1 & & & & & \\ v_{n-1} & & \ddots & \vdots & \vdots & \ddots & & & & \\ v_n & v_{n-1} & \cdots & v_{n-p+1} & v_{n-p} & \cdots & v_1 & & & \\ \hline v_{n+1} & v_n & \cdots & v_{n-p+2} & v_{n-p+1} & \cdots & v_2 & v_1 & & \\ v_{n+2} & \ddots & \ddots & \vdots & \vdots & \ddots & & \ddots & \ddots & \\ \vdots & \ddots & \ddots & v_n & v_{n-1} & & \ddots & & \ddots & \ddots \\ v_{n+p} & \cdots & v_{n+2} & v_{n+1} & v_n & v_{n-1} & & \cdots & & v_2 & v_1 \end{array}\right)$$

$$= \begin{pmatrix} A & B \\ C & D \end{pmatrix},$$

où

$$A = \begin{pmatrix} v_1 & & & \\ v_2 & v_1 & & \\ \vdots & v_2 & \ddots & \\ v_{n-p+1} & & \ddots & v_1 \\ \vdots & \ddots & & v_2 \\ v_{n-1} & & \ddots & \vdots \\ v_n & v_{n-1} & \cdots & v_{n-p+1} \end{pmatrix}, \; B = \begin{pmatrix} & & & \\ & & & \\ & & & \\ & & & \\ v_1 & & & \\ \vdots & \ddots & & \\ v_{n-p} & \cdots & v_1 & \end{pmatrix},$$

$$C = \begin{pmatrix} v_{n+1} & v_n & \cdots & v_{n-p+2} \\ v_{n+2} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & v_n \\ v_{n+p} & \cdots & v_{n+2} & v_{n+1} \end{pmatrix}, \; D = \begin{pmatrix} v_{n-p+1} & \cdots & v_2 & v_1 & & \\ \vdots & \ddots & & \ddots & \ddots & \\ v_{n-1} & & \ddots & & \ddots & \ddots \\ v_n & v_{n-1} & & \cdots & & v_2 & v_1 \end{pmatrix}$$

sont des matrices de taille $n \times p$, $n \times n$, $p \times p$, $p \times n$, respectivement.

Théorème 4.4.1 *Soit*

$$M = \begin{pmatrix} L_1 & O_p \\ T & L_2 \end{pmatrix} \tag{4.37}$$

est une matrice non singulière de taille $(n+p) \times (n+p)$, ou O_p est un bloc nul, T est une matrice carrée de Toeplitz bande définie en 4.2 de taille n. De plus, on suppose que M^{-1} admet

la partition suivante :

$$M^{-1} = \begin{pmatrix} A & B \\ C & D \end{pmatrix} \tag{4.38}$$

où A, B, C, et D sont de taille $n \times p$, $n \times n$, $p \times p$, $p \times n$, respectivement. Si T est non singulier, alors C est aussi non singulier, donc, on peut écrire T^{-1} de la façon suivante :

$$T^{-1} = B - AC^{-1}D, \tag{4.39}$$

c'est le complément de Schur du bloc C de la matrice M^{-1}.

Preuve. On a

$$M = \begin{pmatrix} L_1 & O_p \\ T & L_2 \end{pmatrix}$$

est une matrice inversible de taille $(n \times p) \times (n \times p)$, où O_p est matrice non nulle de taille $p \times p$, $L_1 = [L\ 0_p \cdots 0_p]$, et $L_2 = \begin{bmatrix} L \\ 0_p \\ \vdots \\ 0_p \end{bmatrix}$.

Soit

$$P = \begin{pmatrix} O_{n,k} & I_n \\ I_k & O_{k,n} \end{pmatrix}$$

où $O_{n,k}$ notée une matrice nulle de taille $n \times k$, et T est inversible. Ainsi,

$$\begin{aligned} PM &= \begin{pmatrix} T & L_2 \\ L_1 & O_k \end{pmatrix} \\ &= \begin{pmatrix} T & O_{n,k} \\ L_1 & R \end{pmatrix} \begin{pmatrix} I_n & T^{-1}L_2 \\ O_{k,n} & I_k \end{pmatrix} \end{aligned}$$

où $R = -L_1T^{-1}L_2$, si T et R sont inversibles et R, alors

$$\begin{aligned} M^{-1} &= \begin{pmatrix} I_n & -T^{-1}L_2 \\ O_{k,n} & I_k \end{pmatrix} \begin{pmatrix} T^{-1} & O_{n,k} \\ -R^{-1}L_1T^{-1} & R^{-1} \end{pmatrix} P \\ &= \begin{pmatrix} -T^{-1}L_2R^{-1} & T^{-1} + T^{-1}L_2R^{-1}L_1T^{-1} \\ R^{-1} & -R^{-1}L_1T^{-1} \end{pmatrix}. \end{aligned}$$

En utilisant (4.38), nous avons

$$\begin{aligned} A &= -T^{-1}L_2R^{-1} \\ B &= T^{-1} + T^{-1}L_2R^{-1}L_1T^{-1} \\ C &= R^{-1} \\ D &= -R^{-1}L_1T^{-1} \end{aligned}$$

Alors $T^{-1} = B - T^{-1}L_2R^{-1}L_1T^{-1} = B - AC^{-1}D$. ■

4.4.1.1 Algorithme et complexité de la résolution d'un système de Toeplitz bande

Ainsi, l'algorithme de cette méthode est décrit comme suit :

Algorithm 4.2 Algorithme via Toeplitz triangulaire

Step 0 : Récupérer A, B, C et D définies par (4.38).

Step 1 : Calculer $b = Bf$

Step 2 : Résoudre $d = Df$

Step 3 : Chercher a par la résolution de $Ca = -d$

Step 4 : Calculer $a' = Aa$

Step 5 : Calculer la solution x donnée par $x = a' + b$.

Complexité de l'algorithme 4.2.

Pour calculer efficacement M^{-1}, il suffit de calculer la première colonne de M^{-1}. Considérer le calcul de $M^{-1}e_1$, où $e_1 = (1, 0, \ldots, 0)$ est un vecteur de taille $(n+p)$. Supposons que $(n+p)$ est multiple de $k = p + q$ et M partitionnée en $k \times k$ blocs

$$M = \begin{pmatrix} M_0 & & & & \\ M_1 & M_0 & & & \\ & M_1 & M_0 & & \\ & & \ddots & \ddots & \\ & & & M_1 & M_0 \end{pmatrix}. \tag{4.40}$$

Le bloc M_1 est une matrice de Toeplitz M_0 est une matrice de Toeplitz triangulaire inférieure, puis aussi M_0^{-1} est une matrice de Toeplitz triangulaire inférieure. La matrice M_0^{-1} est calculée une seule fois et le calcul de sa première colonne ne coûte pas plus de k^2 flops. Ensuite, le calcul des coûts $L^{-1}f$ k^2, plus $2n/k - 1$ noyaux, qui est $k^2 + O(n \log_2 k)$ opérations arithmétiques (voir [60]). Notons que Bf est calculé en $O(n-p)\log_2(n-p)$, nous pouvons calculer a en $O(p \log_2 p)$ flops, $O(n \log_2 n)$ pour calculer a' flops, et $O(n)$ à la dernière étape. Le système de Toeplitz bande $T_n x = b$ peut être résolu en $O(n \log_2 n + k^2)$. Lorsque $m = \max(p, q)$, on en déduit que la complexité de l'algorithme 4.2 est bornée par $O(n \log n + m^2)$. Maintenant nous comparons le coût de calcul entre la réduction cyclique [15], l'algorithme de Sherman-Morrison-Woodbury [60] et notre algorithme (Algorithme 4.2) dans le tableau 4.1.

	Réduction cyclique [15]	Sherman-Morrison [60]	Notre algorithme (4.2)
m, n	$O(n \log_2 m + m \log_2^2 m \log_2 \frac{n}{m})$	$O(n \log_2 m + m^3)$	$O(n \log_2 n + m^2)$
$m \ll n$	$O(n \log_2 m)$	$O(n \log_2 m)$	$O(n \log_2 n)$
$m \approxeq n$	$O(n \log_2 n + n \log_2^2 n)$	$O(n \log_2 n + n^3)$	$O(n \log_2 n + n^2)$

TABLE 4.1 – Coûts de calcul pour résoudre un système Toeplitz bande de taille $n \times n$.

Lorsque la taille de la bande est grande, nous avons une bonne approche pour résoudre un système Toeplitz bande en terme de coût de calcul et même de point de vue efficacité les autres programmes cessent parfois de fonctionner lorsque on augmente la taille de bande plus de détails dans la partie numérique. De toute évidence, il est clair que notre alternative est plus coûteuse que les autres méthodes connues [15, 18, 60] quand en supposant que la bande n'est pas trop grande.

4.4.2 Étude de la stabilité

Théorème 4.4.2 *La solution calculée x du système (4.6) satisfait*

$$\frac{\|x-\hat{x}\|_p}{\|x\|_p} \leq O(u)\ \mathrm{cond}_p(T),\ (\textit{pour}\ \ p=1\ \ \textit{ou}\ \ \infty) \tag{4.41}$$

où $\mathrm{cond}_p(t) = \| T \|_p \| T^{-1} \|_p$, $O(u) = \dfrac{\eta_3}{1-\gamma_n} + \left(\dfrac{\eta_3(\gamma_n+1)}{(1-\gamma_n)} + \gamma_n\right)\dfrac{\|\hat{B}\|_p}{\|T^{-1}\|_p}$ *où* u, γ_n, *et* η_3 *sont définies dans le Lemme 1.3.1 et l'équation (4.49), respectivement.*

Preuve. Supposons que $X = T^{-1}$, $E = C^{-1}$ où $\hat{A}, \hat{E}, \hat{D}, \hat{B}$ sont calculées par l'algorithme 4.2 et que nous avons

$$-\hat{A}\hat{E}\hat{D} + \hat{B} = \Delta L + X, \qquad |\Delta L| \leq \gamma_n |-\hat{A}\hat{E}\hat{D} + \hat{B}| \leq \gamma_n |\hat{A}\hat{E}\hat{D}| + \gamma_n |\hat{B}| \tag{4.42}$$

$$\begin{aligned}
\hat{x} &= -(\hat{A}+\Delta A)\hat{y} + (\hat{B}+\Delta B)f && |\Delta A| \leq \gamma_n |\hat{A}|,\ |\Delta B| \leq \gamma_n |\hat{B}| \\
&= -(\hat{A}+\Delta A)(\hat{E}+\Delta E)\hat{z} + (\hat{B}+\Delta B)f && |\Delta E| \leq \gamma_n |\hat{E}| \\
&= -(\hat{A}+\Delta A)(\hat{E}+\Delta E)(\hat{D}+\Delta D)f + (\hat{B}+\Delta B)f && |\Delta D| \leq \gamma_n |\hat{D}|\,.
\end{aligned}$$

et

$$\begin{aligned}
|x-\hat{x}| &= |(-\hat{A}\hat{E}\hat{D}+\hat{B})f - (-(\hat{A}+\Delta A)(\hat{E}+\Delta E)(\hat{D}+\Delta D) + (\hat{B}+\Delta B))f| \\
&\leq |(\hat{A}+\Delta A)(\hat{E}+\Delta E)(\hat{D}+\Delta D)f - \hat{A}\hat{E}\hat{D}f| + |\Delta B f|
\end{aligned}$$

En appliquant le lemme 1.3.4, nous obtenons

$$|x-\hat{x}| \leqslant ((1+\gamma_n)^3 - 1)\,|\hat{A}|\,|\hat{E}|\,|\hat{D}|\,|f| + \gamma_n |\hat{B}|\,|f|\,. \tag{4.43}$$

Supposons que $\hat{E}$, $\hat{D}$, $\hat{A}$ sont des matrices non-négatives, ainsi $|\hat{A}\hat{E}\hat{D}| = |\hat{A}|\,|\hat{E}|\,|\hat{D}|$ et

$$|x-\hat{x}| \leqslant ((1+\gamma_n)^3 - 1)\,|\hat{A}\hat{E}\hat{D}|\,|f| + \gamma_n |\hat{B}|\,|f|\,. \tag{4.44}$$

En utilisant (4.42), nous obtenons

$$\begin{aligned}
|\hat{A}\hat{E}\hat{D}| &\leqslant |X| + |\Delta L| + |\hat{B}| \\
&\leqslant |X| + \gamma_n |\hat{A}\hat{E}\hat{D}| + (\gamma_n+1)|\hat{B}|
\end{aligned}$$

Alors,

$$| \hat{A}\hat{E}\hat{D} | \leqslant \frac{1}{1-\gamma_n} | X | + \frac{\gamma_n + 1}{1-\gamma_n} | \hat{B} | \tag{4.45}$$

En se basant sur les équations, (4.44) et (4.45) nous obtenons

$$| x - \hat{x} | \leqslant \frac{((1+\gamma_n)^3 - 1)}{1-\gamma_n} | X | | f | + \left(\frac{((1+\gamma_n)^3 - 1)(\gamma_n + 1)}{(1-\gamma_n)} + \gamma_n \right) | \hat{B} | | f | .$$

Via la norme p (pour $p = 1, \infty$), nous arrivons au résultat suivant :

$$\frac{\|x - \hat{x}\|_p}{\|x\|_p} \leqslant \left[\frac{((1+\gamma_n)^3 - 1)}{1-\gamma_n} + \left(\frac{((1+\gamma_n)^3 - 1)(\gamma_n + 1)}{(1-\gamma_n)} + \gamma_n \right) \frac{\|\hat{B}\|_p}{\|T^{-1}\|_p} \right] \mathrm{cond}_p(T) \tag{4.46}$$

Puisque $1 + x \leq e^x \; \forall \; x \geq 0$, nous avons $(1+\gamma_n)^n < \exp(n\gamma_n)$ et ainsi

$$(1+\gamma_n)^n - 1 < n\gamma_n + \frac{(n\gamma_n)^2}{2!} + \frac{(n\gamma_n)^3}{3!} + \cdots \tag{4.47}$$

$$< n\gamma_n \underbrace{\left(1 + \frac{n\gamma_n}{2} + \left(\frac{n\gamma_n}{2}\right)^2 + \left(\frac{n\gamma_n}{2}\right)^3 + \cdots\right)}_{= \frac{1}{1 - \frac{n\gamma_n}{2}}} \tag{4.48}$$

En fixant

$$\eta_n = \frac{1}{1 - \frac{n\gamma_n}{2}} \tag{4.49}$$

et en utilisant l'équation (4.46), nous concluons

$$\frac{\|x - \hat{x}\|_p}{\|x\|_p} \leqslant \frac{\eta_3}{1-\gamma_n} \|T^{-1}\|_p \, \|T\|_p + \left(\frac{\eta_3(\gamma_n + 1)}{(1-\gamma_n)} + \gamma_n \right) \frac{\|\hat{B}\|_p}{\|T^{-1}\|_p} \|T^{-1}\|_p \|T\|_p. \tag{4.50}$$

■

Pour illustrer les résultats de l'analyse d'erreur, nous traçons dans la Figure 4.1 les erreurs en norme ∞ entre la solution exacte x et la solution de notre algorithme x_C, la solution via la réduction cyclique x_{CR} ainsi que la solution via Sherman-Morrison-Woodbury x_{CMW}, respectivement. Il est clair que les résultats numériques coïncident bien avec les résultats théoriques.

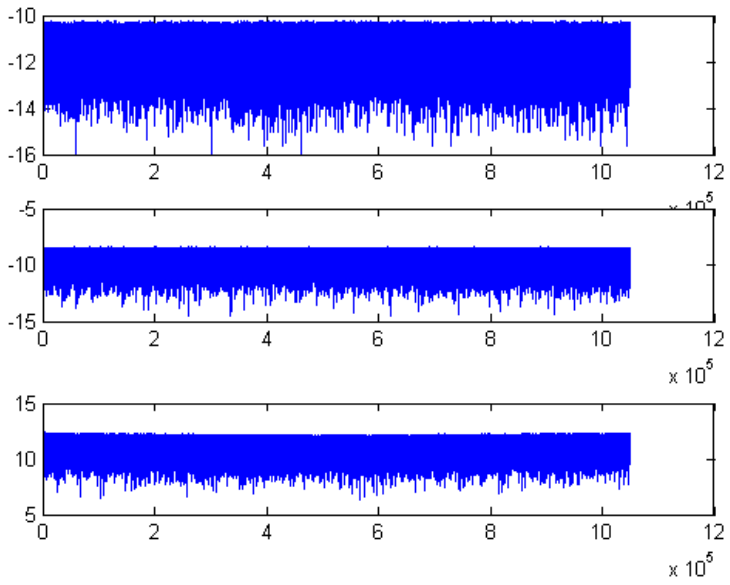


FIGURE 4.1 – $\log_{10}(Error_C^A)$ (en haut), $\log_{10}(Error_{CR}^A)$ (au milieu) et $\log_{10}(Error_{SMW}^A)$ (en bas) pour une matrice de Toeplitz bande de taille 2^{20} et $m_r = m_c = 32$.

4.4.3 Exemples et tests numériques

L'Algorithme 4.2 présenté dans ce chapitre a été implémenté sous Matlab (R2011) sur un ordinateur portable Intel(R) core (TM)i3 CPU M380 avec un processeur de 253 GHZ et RAM de 4 GO. Afin d'entamer la comparaison des algorithmes, nous avons implémenté la réduction cyclique classique [15] et l'algorithme via Sherman-Morrison-Woodbury [60] par nos propres moyens.

Pour l'ensemble des exemples, f est choisi dans le but d'obtenir les solutions exactes égale à 1. Ainsi, l'erreur relative en norme ∞ entre la solution exacte et la solution x_C est égale $\max_{1\leq i\leq n} |(x_C)_i - 1|$.

Les tables de 5.1 à 5.6 montrent la performance de l'algorithme de Toeplitz bande de taille $n \times n$.

Exemple 1 : La matrice de Toeplitz bande T a les coefficients $t_1 = 1000$, $t_0 = 1$, $t_{-1} = 0.001$, $t_{-2} = 1000$. La solution x_C calculée par l'Algorithme 4.2 et la solution x_{SMW} calculée par la formule Sherman-Morrison-Woodbury, possède la précision $\|x - x_C\|_\infty = 2.2.10^{-12}$ et $\|x - x_{SMW}\|_\infty = 1.9.10^{-14}$, respectivement. Dans cet exemple, la réduction cyclique échoue parce que T est loin d'être diagonale dominante.

Exemple 2 : La matrice de Toeplitz bande a le coefficient $t_0 = 0.5$ sur la diagonale principale et 1 ailleurs dans la bande. Le nombre de blocs est fixé tels que $n = 2^{20}$, $m_r = m$ et $m_c = m/2$, où m varie. Les résultats de cet exemple sont donnés dans la Table 4.2. Lorsque m augmente,

l'Algorithme 4.2 donne des résultats beaucoup mieux que l'algorithme de la réduction cyclique et l'algorithme via Sherman-Morrison-Woodbury en termes d'efficacité numérique et temps de calcul. En outre, la précision de la méthode de la réduction cyclique se détériore et la méthode de Sherman-Morrison Woodbury ne fonctionne pas.

Exemple 3 : Dans l'exemple suivant, nous choisissons $t_0 = 1.0001$ sur la diagonale principale et 1 ailleurs dans la bande. Le nombre de blocs de T est fixé tels que $n = 2^{20}$, et $m_r = m$ et $m_c = m/2$, où m varie. Les résultats sont donnés dans la Table 4.2. Lorsque m augmente, l'Algorithme 4.2 fonctionne mieux que la méthode de Sherman-Morrison-Woodbury en terme de temps de calcul. En outre, la méthode de la réduction cyclique échoue.

Exemple 4 : Soit $t_0 = 1 + 10^{-14}$ sur la diagonale principale et 1 ailleurs dans la bande, avec $p = q = m$ et $n = 2^{20}$. Les résultats donnés dans la Table 4.4 montrent que même si $m << n$, l'Algorithme 4.2 est plus performant que l'algorithme de la réduction cyclique et l'algorithme via Sherman-Morrison-Woodbury en termes d'efficacité et temps de calcul.

Exemple 5 : La matrice T a comme $t_0 = 1$ dans la diagonale principale et 2 ailleurs dans la bande. La taille de T est fixée par $n = 2^{20}$. Nous choisissons $p = m$ et $q = m$, où m varie. La Table 4.5 renforce l'efficacité de notre approche à l'égard de l'algorithme de la réduction cyclique et l'algorithme via Sherman-Morrison-Woodbury en termes de précision et temps de calcul.

Exemple 6 : T est une matrice de Toeplitz bande de coefficients $t_3 = 1, t_2 = 3, t_1 = 2, t_0 = 3/5, t_{-1} = 4$. La Table 4.6 assure aussi l'efficacité de notre approche à l'égard de l'algorithme de la réduction cyclique et l'algorithme via Sherman-Morrison-Woodbury en termes de précision et temps de calcul.

m	$\|x - x_C\|_\infty$	Temps	$\|x - x_{SMW}\|_\infty$	Temps$_{SMW}$ (s)	$\|x - x_{CR}\|_\infty$	Temps$_{CR}$ (s)
4	4.60×10^{-12}	0.53	1.82×10^{-11}	0.81	3.75×10^{-12}	0.22
8	8.50×10^{-11}	0.53	1.84×10^{-7}	0.90	2.40×10^{-12}	0.21
16	3.53×10^{-10}	0.8	1.27×10^{-4}	1.80	2.94×10^{-11}	0.34
32	6.30×10^{-11}	0.9	$2.31 \times 10^{+12}$	2.26	4.37×10^{-9}	0.36
64	1.18×10^{-10}	0.98	Ne marche pas		1.89×10^{-9}	0.52
128	8.18×10^{-10}	0.72	Ne marche pas		7.89×10^{-9}	0.64
256	8.18×10^{-10}	0.57	Ne marche pas		1.35×10^{-7}	1.23
512	9.83×10^{-8}	1.15	Ne marche pas		2.24×10^{-7}	8.83
1024	6.18×10^{-8}	0.58	Ne marche pas		2.85×10^{-6}	35.41

TABLE 4.2 – Résultats numériques pour l'exemple 2

m	$\|x - x_C\|_\infty$	Temps	$\|x - x_{SMW}\|_\infty$	Temps$_{SMW}$ (s)	$\|x - x_{CR}\|_\infty$	Temps$_{CR}$ (s)
32	2.64×10^{-10}	0.54	3.77×10^{-12}	1.18	Echoue	0.62
64	1.59×10^{-10}	0.54	7.48×10^{-12}	1.2	Echoue	0.75
128	3.35×10^{-10}	0.55	2.01×10^{-11}	1.4	Echoue	1.51
256	8.73×10^{-7}	0.57	4.68×10^{-8}	7.344	Echoue	2.76
512	2.72×10^{-6}	0.69	7.82×10^{-7}	38.18	Echoue	2.34
1024	1.04×10^{-8}	0.52	8.30×10^{-8}	315.789	Echoue	8.2614

TABLE 4.3 – Résultats numériques pour l'exemple 3.

m	$\|x - x_C\|_\infty$	Temps	$\|x - x_{SMW}\|_\infty$	Temps$_{SMW}$ (s)	$\|x - x_{CR}\|_\infty$	Temps$_{CR}$ (s)
4	$4,27 \times 10^{-5}$	0.86	$6,3 \times 10^{-3}$	0.62	3.5×10^{20}	0.52
8	$2,7145 \times 10^{-11}$	0.73	$5,2 \times 10^{-2}$	0.7	3×10^{15}	0.53

TABLE 4.4 – Résultats numériques pour l'exemple 4.

m	$\|x - x_C\|_\infty$	Temps	$\|x - x_{SMW}\|_\infty$	Temps$_{SMW}$ (s)	$\|x - x_{CR}\|_\infty$	Temps$_{CR}$ (s)
8	8.50×10^{-11}	0.62	5.11×10^{-11}	0.85	2.40×10^{-12}	0.22
16	3.53×10^{-10}	0.8	1.34×10^{-6}	1.46	2.94×10^{-11}	0.29
32	6.3×10^{-11}	0.73	0.003	1.57	4.37×10^{-9}	0.29
64	1.18×10^{-10}	0.51	3.48×10^{7}	2.13	1.89×10^{-9}	0.25
128	1.97×10^{-10}	0.6	Ne marche pas		7.89×10^{-9}	0.52
256	8.18×10^{-10}	0.69	Ne marche pas		1.35×10^{-7}	1.58
512	9.83×10^{-8}	0.7	Ne marche pas		2.24×10^{-7}	6.09
1024	6.18×10^{-8}	0.58	Ne marche pas		2.85×10^{-6}	31.31

TABLE 4.5 – Résultats numériques pour l'exemple 5.

n	$\|x - x_C\|_\infty$	Temps	$\|x - x_{SMW}\|_\infty$	Temps$_{SMW}$ (s)	$\|x - x_{CR}\|_\infty$	Temps$_{CR}$ (s)
2^8	1.665×10^{-14}	0.001	2.977×10^{138}	0.25	1.0667	0.0082
2^9	9.226×10^{-14}	0.002	Ne marche pas		1.0667	0.0085
2^{10}	3.619×10^{-14}	0.0041	Ne marche pas		1.0667	0.01
2^{11}	1.654×10^{-13}	0.0043	Ne marche pas		1.0667	0.086
2^{12}	1.722×10^{-11}	0.005	Ne marche pas		1.0667	0.055

TABLE 4.6 – Résultats numériques pour l'exemple 6.

4.4.4 Cas des matrices mal-conditionnées

Nous avons prolongé la matrice de Toeplitz bande T dans une matrice triangulaire de Toeplitz M. Cela a des inconvénients et des avantages, comme on va le montrer par la suite. Parmi les avantages de cette nouvelle approche, nous n'avons pas besoin de passer par la factorisation spectrale. Malheureusement, cette méthode ne peut pas toujours assurer de bons résultats quand la matrice T est une matrice symétrique ou à diagonale dominante. Dans ce qui suit, nous allons discuter les inconvénients de l'approche proposée.

4.4.4.1 M mal-conditionnée

Il peut arriver que même pour une matrice T bien conditionnée, la matrice M définie dans (4.36) serait mal-conditionnée ou non inversible. Ceci est illustré par l'exemple suivant : considérons le cas où T est tridiagonale, symétrique définie positive, fortement diagonale dominante, avec $a > 0$ sur la diagonale principale et ε sur sa première supérieure diagonale et sa première inférieure diagonale, avec $0 < \varepsilon << a$.

Nous obtenons une matrice M de taille $(n+1)\times(n+1)$ a $\varepsilon = 1$ sur la diagonale principale, $a = 10$ sur la première sous-diagonale, $\varepsilon = 1$ sur la deuxième sous-diagonale et 0 ailleurs. Le nombre de conditionnement de la matrice T et de la matrice triangulaire M, en fonction de n, est le suivant :

n	$\text{cond}_\infty(T)$	$\text{cond}_\infty(M)$
2^7	1.500	3.676×10^{128}
2^8	1.500	1.02×10^{256}
2^9	1.500	inf
2^{10}	1.500	inf
2^{11}	1.500	inf

TABLE 4.7 – Conditionnement des matrices T et M pour M mal-conditionnée.

4.4.4.2 T mal conditionnée

Dans ce cas, le prolongement de la matrice T dans une matrice triangulaire inférieure de Toeplitz bande M a un bon effet de la régularisation de la solution. Par exemple, on considère le cas où T est une matrice tridiagonale, avec $a > 0$ sur la diagonale principale et b ($0 < a \ll b$) sur ses premières diagonales supérieures et inférieures. Nous avons considéré les valeurs suivantes : $a = 10^{-30}$, $b = 1$.

Le nombre de conditionnement des matrices T et M, en fonction de n, est le suivant :

n	$\mathrm{cond}_\infty(T)$	$\mathrm{cond}_\infty(M)$
2^7	2×10^{30}	130
2^8	2×10^{30}	258
2^9	2×10^{30}	514
2^{10}	2×10^{30}	1026
2^{11}	2×10^{30}	2050
2^{12}	2×10^{30}	4098

TABLE 4.8 – Conditionnement des matrices T et M, pour T mal conditionnée.

4.5 Conclusions et perspectives

Dans ce chapitre, nous avons proposé une nouvelle méthode pour la résolution des systèmes d'équations linéaires avec des matrices de Toeplitz bandes. Des exemples numériques montrent que l'algorithme proposé est une des bonnes alternatives en termes d'efficacité et temps de calcul lorsque la bande est modérément grande. Malheureusement, le nouvel algorithme pourrait souffrir de certaines instabilités numériques. En fait, même si la matrice de Toeplitz bande T est bien conditionnée, la matrice triangulaire inférieure de Toeplitz M pourrait être très mal conditionnée. De plus, les propriétés comme la positivité ou la dominance diagonale de la matrice T sont parfois perdues.

La principale perspective recherche dans le Chapitre 5 est de généraliser cette approche dans le cas d'une matrice de Toeplitz bande par blocs. Il s'avère intéressant aussi de viser la connexion de ces algorithmes conçus pour des applications phares des mathématiques comme la restauration d'images.

CHAPITRE 5

RÉSOLUTION D'UN SYSTÈME DE TOEPLITZ BANDE PAR BLOCS DE TOEPLITZ BANDES

5.1 Introduction

La modélisation mathématique de problèmes du monde réel conduit souvent à la résolution de systèmes d'équations linéaires avec certaines structures intrinsèques.
Ce type de matrices, très souvent des matrices de Toeplitz bandes par blocs Toeplitz, sont rencontrés en particulier dans les domaines tels que le traitement d'images et dans la solution numérique des équations différentielles.
Une large étude des problèmes où les matrices sont de Toeplitz bandes par blocs de Toeplitz bandes (TBBTB) sont données dans [17].
En raison de la grande taille de blocs de matrices, il est obligatoire d'exploiter à la fois la structure externe de Toeplitz bande, et la structure de Toeplitz extérieure, de concevoir des algorithmes efficaces pour la résolution de ses systèmes. Plusieurs techniques ont été introduites dans la littérature pour la résolution des systèmes TBBTB, comme les méthodes itératives en particulier la méthode du gradient conjugué pré-conditionné [27, 76] qui coûte $O(nm \log m)$ pour chaque itération, les techniques multigrilles [36] et des algorithmes basés sur la réduction cyclique [16, 17]. Il y a aussi quelques méthodes directes, comme des généralisations pour le rang de déplacement [72] et la convolution approchée [86].
Un algorithme d'une complexité de $O(nm^3) + O(k^3m^2)$ pour résoudre un système de Toeplitz bande par blocs dont les blocs sont de Toeplitz non nécessairement bandes a été introduit dans [32]. L'idée de cet algorithme est basée sur la construction d'une matrice circulante et avec la formule de Sherman Morrison Woodbury on peut transformer son inverse à un inverse de la matrice d'origine.

L'algorithme que nous proposons ici transforme la matrice TBBTB en une matrice de Toeplitz triangulaire inférieure par blocs de Toeplitz bandes, avec un coût intéressant.
Soit T une matrice de Toeplitz bande par blocs de Toeplitz bandes, définie comme suit :

$$T = \begin{pmatrix} T_0 & T_{-1} & \dots & \dots & T_{-k} & 0 & \dots & 0 \\ T_1 & T_0 & \ddots & \ddots & \ddots & T_{-k} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ T_k & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & T_{-k} \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & T_{-1} \\ 0 & \dots & 0 & T_k & \dots & \dots & T_1 & T_0 \end{pmatrix} \tag{5.1}$$

et chaque T_i avec $i = -k, \dots, 0, \dots, k$ s'écrit sous la forme

$$T_i = \begin{pmatrix} T_{0,i} & T_{-1,i} & \dots & \dots & T_{-k,i} & 0 & \dots & 0 \\ T_{1,i} & T_{0,i} & \ddots & \ddots & \ddots & T_{-k,i} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ T_{k,i} & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & T_{-k,i} \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & T_{-1,i} \\ 0 & \dots & 0 & T_{k,i} & \dots & \dots & T_{1,i} & T_{0,i} \end{pmatrix}$$

avec T est formée de n blocs de taille $m \times m$, et supposons que T est de taille $N \times N$ où $N = nm$ et m représente la taille de chaque bloc.
Concernant le conditionnement d'une matrice TBBTB, l'auteur dans [51] a fait des tests numériques dans lesquels il a appliqué le logarithme à base 10 sur les nombres de conditionnements des matrices TBBTB avec les mêmes tailles et à chaque fois on augmente les largeurs de bandes, ce qui lui permet de remarquer que le conditionnement dépend de la largeurs de la bande. C-à-d les matrices TBBTB avec de petites bandes sont mal conditionnées par rapport aux matrices de bandes plus grandes.

5.2 Matrices triangulaires de Toeplitz par blocs

On définit dans cette partie les matrices de Toeplitz triangulaires par blocs et leurs propriétés. On note L une matrice de Toeplitz triangulaire inférieure par blocs avec X son premier bloc colonne et U une matrice de Toeplitz triangulaire supérieure par blocs avec Y^T est son premier

bloc ligne, définies comme suit :

$$L = \begin{pmatrix} L_1 & & & \\ \vdots & \ddots & & \\ \vdots & & \ddots & \\ L_n & \dots & \dots & L_1 \end{pmatrix}, \qquad X = \begin{pmatrix} L_1 \\ \vdots \\ \vdots \\ L_n \end{pmatrix} \tag{5.2}$$

$$U = \begin{pmatrix} U_1 & \dots & \dots & U_n \\ & \ddots & & \vdots \\ & & \ddots & \vdots \\ & & & U_1 \end{pmatrix}, \qquad Y^T = (U_1, \dots\dots, U_n) \tag{5.3}$$

Proposition 5.2.1

- *Le produit d'une matrice de Toeplitz triangulaire par blocs par un vecteur V par blocs avec $V = (V_i)_{i=1,n}$, où (V_i) sont de tailles $m \times p$, coûte $O((m+p)mn\log n + nm^2p)$ flops, voir [13].*
- *Si $p = m = 1$, le coût de calcul se réduit en $O(n\log n)$, par rapport à $O(n^2)$ pour la multiplication d'une matrice par un vecteur.*
- *Si m a une largeur en respectant $\log n$, alors le coût de calcul de la FFT est négligeable par rapport au coût de calcul d'un produit d'une matrice de Toeplitz par blocs par un vecteur par blocs, voir [13, Algorithm 2.3].*

Remarque 5.2.1 *Si les blocs d'une matrice triangulaire de Toeplitz par blocs ne sont pas de matrices triangulaires de Toeplitz, l'inverse est une matrice triangulaire de Toeplitz par blocs mais les blocs ne sont pas de matrices de Toeplitz.*

Proposition 5.2.2 *L'inversion d'une matrice triangulaire de Toeplitz par blocs coûte $O(m^2n\log n + m^3n)$.*

Preuve. Pour plus de détails, voir [13, Algorithm 25]. ■

5.3 Matrices de Toeplitz bande par blocs Toeplitz bandes

En fait le cas bloc n'est qu'une généralisation du cas scalaire par contre nous avons rencontré quelque difficultés parmi lesquelles le nombre d'opérations arithmétiques effectués.

5.3.1 Le cas blocs de la réduction cyclique

L'idée est la même que le cas scalaire. En fait, il suffit de repartitionner la matrice de TBBTB en une matrice tridiagonale de Toeplitz par blocs, définie par

$$Q_n = \begin{bmatrix} A_0 & A_{-1} & & & & \\ A_1 & A_0 & A_{-1} & & & \\ & \ddots & \ddots & \ddots & & \\ & & A_1 & A_0 & A_{-1} & \\ & & & A_1 & A_0 & \end{bmatrix} \tag{5.4}$$

(voir Chapitre 4 Section 4.2.3). Cet algorithm coûte $O(k^2 m \log^2 m \log n + Knm \log m)$ en appliquant la factorisation LU pour la matrice Q_n avec n le nombre de blocs, m la taille de chaque bloc et K le rang de déplacement de L_n, la matrice de Toeplitz triangulaire inférieure bande par blocs.

5.3.2 Notre contribution

Proposition 5.3.1 *On peut prolonger une matrice TBBTB T définie par (5.1) de taille $N \times N$ avec $N = nm$ où n est le nombre de blocs et m est la taille de chaque bloc, dans une matrice triangulaire inférieure de Toeplitz par blocs L (TITB)*

$$L = \begin{pmatrix} T_{-k} & & & & & & & & \\ \vdots & \ddots & & & & & & & \\ T_{-1} & & T_{-k} & & & & & & \\ T_0 & T_{-1} & & T_{-k} & & & & & \\ \vdots & \ddots & \ddots & & \ddots & & & & \\ T_k & & T_0 & T_{-1} & & T_{-k} & & & \\ 0 & \ddots & & \ddots & \ddots & & \ddots & & \\ \vdots & \ddots & \ddots & & \ddots & \ddots & & T_{-k} & \\ 0 & \dots & 0 & T_k & \dots & T_0 & T_{-1} & \dots & T_{-k} \end{pmatrix} \tag{5.5}$$

$$= \begin{pmatrix} L_1 & 0 \\ T & L_1 \end{pmatrix} \tag{5.6}$$

avec $L_1 = \begin{pmatrix} T_{-k} & & \\ \vdots & \ddots & \\ T_{-1} & \dots & T_{-k} \end{pmatrix}$, *définie par (5.2) de taille $M \times M$, où $M = m(n+k)$ et R représente sa première bloc colonne :*

$$R = (T_{-k}, \dots, T_{-1}, T_0, \dots, T_k)\,.$$

$$
L^{-1} = \left(\begin{array}{cccc|cccccc}
V_1 & & & & & & & & & \\
V_2 & V_1 & & & & & & & & \\
\vdots & V_2 & \ddots & & & & & & & \\
V_{n-p+1} & & \ddots & V_1 & & & & & & \\
\vdots & \ddots & & v_2 & v_1 & & & & & \\
V_{n-1} & & \ddots & \vdots & \vdots & \ddots & & & & \\
V_n & V_{n-1} & \cdots & V_{n-p+1} & V_{n-p} & \cdots & V_1 & & & \\
\hline
V_{n+1} & V_n & \cdots & V_{n-p+2} & V_{n-p+1} & \cdots & V_2 & V_1 & & \\
V_{n+2} & \ddots & \ddots & \vdots & \vdots & \ddots & & \ddots & \ddots & \\
\vdots & \ddots & \ddots & V_n & V_{n-1} & & \ddots & & \ddots & \ddots \\
V_{n+p} & \cdots & V_{n+2} & V_{n+1} & V_n & V_{n-1} & & \cdots & & V_2 \quad V_1
\end{array}\right)
$$

$$
= \begin{pmatrix} A & B \\ C & D \end{pmatrix}.
$$

Théorème 5.3.1 *Soit T une matrice de Toeplitz bande par blocs inversible de taille $(n \times m) \times (n \times m)$ et L une matrice triangulaire inférieure de Toeplitz par blocs associée à T . Supposons que L^{-1} est réparti comme suit :*

$$
L^{-1} = \begin{pmatrix} A & B \\ C & D \end{pmatrix}
$$

où A, B, C et D sont des matrices blocs de taille $(n \times m) \times (k \times m)$, $(n \times m) \times (n \times m)$, $(k \times m) \times (n \times m)$, $(k \times m) \times (n \times m)$ respectivement. Ensuite, si C est inversible alors la résolution d'un système de Toeplitz bande par blocs $Tx = f$ est donnée par :

$$
x = Bf - AC^{-1}Df. \tag{5.7}
$$

Preuve. On peut réduire la résolution du système $Tx = f$ à la résolution d'un système

$$
L \begin{pmatrix} x \\ 0 \end{pmatrix} = \begin{pmatrix} e \\ f \end{pmatrix} \tag{5.8}
$$

dont e est un vecteur inconnu de longueur $k \times m$. Or, d'après la Remarque 5.2.1 on a

$$
L^{-1} = \begin{pmatrix} L_1 & & \\ \vdots & \ddots & \\ L_{n+k} & \dots & L_1 \end{pmatrix} \tag{5.9}
$$

$$
= \begin{pmatrix} A & B \\ C & D \end{pmatrix} \tag{5.10}
$$

où A, B, C, D sont des blocs matrices de tailles $(n \times m) \times (k \times m)$, $(n \times m) \times (n \times m)$, $(k \times m) \times (k \times m)$ et $(k \times m) \times (n \times m)$, respectivement. Alors, d'après (5.8) et (5.10) le vecteur e peut être déterminer comme suit

$$e = -C^{-1}Df. \tag{5.11}$$

D'où, la résolution d'un système de Toeplitz bande par blocs $Tx = b$ est comme suit :

$$x = Bf - AC^{-1}Df. \tag{5.12}$$

■

Proposition 5.3.2 *La résolution d'un système de TBBTB $Tx = b$, coûte environ $O(m^2 n \log n + m^3 n + M^3)$.*

Preuve. Pour calculer la solution x, on applique l'algorithme suivant :

Algorithm 5.1 Algorithme par blocs

Étape 0 : Calculer l'inverse de L.

Étape 1 : Calculer $e_1 = Db$.

Étape 2 : Calculer $Ce_2 = b$.

Étape 4 : Calculer $x_1 = Ae_2$.

Étape 5 : Calculer $x_2 = Bf$.

Étape 6 : $x = x_1 - x_2$.

L'inverse de L coûte $O(m^2 n \log n + m^3 n)$ avec k est la taille de bloc bande de L, e_2 coûte $O(M^3)$ où $M = k * m$, et x_2 $O(m^2 n \log n + m^3 n)$, alors la solution du système $Tx = f$ nécessite $O(m^2 n \log n + m^3 n + M^3)$ opérations arithmétiques. ■

5.4 La structure en traitement d'images

5.5 Intoduction

Une question fondamentale en traitement du signal et de l'image est la suppression de flou. Le signal ou l'image obtenu à partir d'une source ponctuelle dans le cadre du processus de brouillage est appelé PSF ("Point spread function"). Le signal ou l'image g observé est juste la convolution de cette fonction p (PSF) avec le signal ou l'image initiale f. Le problème de

restauration ou de déflouage est de récupérer f de la fonction $g = p * f$ compte tenue de la fonction p. Ce problème de base apparaît sous plusieurs formes dans le traitement du signal et de l'image [5, 26, 41, 47].
Ainsi, dans la résolution de f à partir d'une longueur g finie, nous avons besoin de certaines hypothèses sur les valeurs de f en dehors du domaine où g est définie. Ces hypothèses sont appelées conditions aux limites. L'approche naturelle et classique est d'utilisée la condition aux limites (de Dirichlet) égale à zéros [5, p $211 - 220$]. Il suffit de supposer que les valeurs de f en dehors du domaine sont nuls. Il en résulte une matrice de floue qui est une matrice de Toeplitz bande dans le cas unidimensionnel et d'une matrice de Toeplitz bande par blocs avec des blocs de Toeplitz bandes (TBB/BTB) dans le cas bidimensionnel, voir [5, p 71].

5.6 Le problème de restauration (ou déflouage)

Définition 5.6.1 *(Image floue) Une matrice floue A est déterminée par deux ingrédients, le PSF qui définit la manière dont chaque pixel est floue, et les conditions aux bords qui spécifient nos hypothèses sur la scène juste à l'extérieure de notre image.*

Définition 5.6.2 *(PSF "Point spread function")*
La matrice P de PSF est l'image d'un pixel blanc, et ses dimensions sont généralement beaucoup plus petites que ceux de l'image floue B et l'image originale X. Si le floue local et spécialement invariant, alors P contient toutes les informations sur toute l'image.

5.6.1 Problème 1-D

Pour plus de simplicité, nous commençons par le cas unidimensionnel. Nous considérons l'image originale

$$\tilde{\mathbf{f}} = (\ldots, f_{-m+1}, \ldots, f_0, f_1, \ldots, f_n, f_{n+1}, \ldots, f_{n+m}, \ldots)$$

et la fonction (PSF) de floue donnée par

$$\mathbf{p} = (\ldots, 0, 0, p_{-m}, p_{-m+1}, \ldots, p_0, \ldots, p_{m-1}, p_m, 0, 0, \ldots). \tag{5.13}$$

L'image floue est la convolution de $\mathbf{p}$ et $\tilde{\mathbf{f}}$, soit g_i le $i^{\text{ème}}$ coefficient de l'image floue donnée par

$$g_i = \sum_{j=-\infty}^{+\infty} p_{i-j} f_j. \tag{5.14}$$

Le problème de restauration est de récupérer le vecteur $\mathbf{f} = (f_1, \ldots, f_n)$ étant donné la fonction floue p et l'image floue $\mathbf{g} = (g_1, \ldots, g_n)$ de longueur fini. A partir de (5.14), nous pouvons

écrire :

$$\begin{pmatrix} p_m & \dots & p_0 & \dots & p_{-m} & & 0 \\ & \ddots & & & & \ddots & \\ & & \ddots & & & & \ddots \\ 0 & & & p_m & \dots & p_0 & \dots & p_{-m} \end{pmatrix} \begin{pmatrix} f_{-m+1} \\ f_{-m+2} \\ \vdots \\ f_0 \\ f_1 \\ \vdots \\ f_n \\ f_{n+1} \\ \vdots \\ f_{n+m-1} \\ f_{n+m} \end{pmatrix} = \begin{pmatrix} g_1 \\ \vdots \\ \vdots \\ g_n \end{pmatrix}. \tag{5.15}$$

Ainsi l'image floue $\mathbf{g} = (g_1, \dots, g_n)$ est déterminée non seulement par $\mathbf{f}$ mais aussi par

$$(f_{-m+1}, \dots, f_0, f_1, \dots, f_n, f_{n+1}, \dots, f_{n+m}).$$

Le système (5.15) est sous-déterminé. Pour surmonter cela, nous faisons certaines hypothèses (dite condition aux limites) sur des données inconnues $f_{-m+1}, \dots, f_0$ et $f_{n+1}, \dots, f_{n+m}$ afin de réduire les conditions d'inconnues.

Avant de discuter les conditions aux limites, tout d'abord nous écrivons (5.15) comme

$$T_l \mathbf{f}_l + T\mathbf{f} + T_r \mathbf{f}_r = \mathbf{g}, \tag{5.16}$$

où

$$T_l = \begin{pmatrix} p_m & \dots & p_1 \\ & \ddots & \vdots \\ & & p_m \\ & & \\ 0 & & \end{pmatrix}, \qquad \mathbf{f}_l = \begin{pmatrix} f_{-m+1} \\ f_{-m+2} \\ \vdots \\ f_{-1} \\ f_0 \end{pmatrix}, \tag{5.17}$$

$$T = \begin{pmatrix} p_0 & \dots & p_{-m} & & 0 \\ \vdots & & & \ddots & \\ p_m & & & & p_{-m} \\ & \ddots & & & \vdots \\ 0 & & p_m & \dots & p_0 \end{pmatrix}, \qquad \mathbf{f} = \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_{n-1} \\ f_n \end{pmatrix}, \tag{5.18}$$

$$T_r = \begin{pmatrix} & & 0 \\ & & \\ p_{-m} & & \\ \vdots & \ddots & \\ p_{-1} & \cdots & p_{-m} \end{pmatrix}, \qquad \mathbf{f}_r = \begin{pmatrix} f_{n+1} \\ f_{n+2} \\ \vdots \\ f_{n+m-1} \\ f_{n+m} \end{pmatrix}. \tag{5.19}$$

5.6.1.1 Condition aux limites de Dirichlet

La condition aux limites (de Dirichlet) égale à zéro assure que l'image observée à l'extérieure du domaine de **g** est zéro [5, p.211-220] c-à-d

$$\mathbf{f}_l = \mathbf{f}_r = 0.$$

D'où, le système (5.16) devient

$$T\mathbf{f} = \mathbf{g}.$$

Alors, d'après (5.18) la matrice T dans le cas unidimensionnel est bien une matrice de Toeplitz bande.

5.6.2 Problème 2-D

Les résultats de la section 5.6.1 peut être prolongés d'une manière naturelle à des problèmes de restauration d'images bidimensionnels. Dans ce cas, on est concerné par la résolution d'un problème similaire à (5.15), sauf que maintenant la matrice est une matrice par blocs. Avec la condition aux limites qui égale à zéro, la matrice obtenue dans le cas bidimensionnel est une matrice (TBB/BTB).

5.6.2.1 Exemple de calcul

Pour des problèmes de restauration d'images spatialement invariantes, la structure spécifique de la matrice T dépend des conditions aux limites imposées, et peuvent impliquer les matrices de Toeplitz, circulantes et de Hankel.
Rappelons que la convolution de PSF avec une image originale nous permet d'obtenir une image floue. En effet, l'opération de convolution pour des images bidimensionnelles est très similaire au cas unidimensionnel.

Soit $X = \begin{bmatrix} x_{1,1} & x_{1,2} & x_{1,3} \\ x_{2,1} & x_{2,2} & x_{2,3} \\ x_{3,1} & x_{3,2} & x_{3,3} \end{bmatrix}, P = \begin{bmatrix} p_{1,1} & p_{1,2} & p_{1,3} \\ p_{2,1} & p_{2,2} & p_{2,3} \\ p_{3,1} & p_{3,2} & p_{3,3} \end{bmatrix}, B = \begin{bmatrix} b_{1,1} & b_{1,2} & b_{1,3} \\ b_{2,1} & b_{2,2} & b_{2,3} \\ b_{3,1} & b_{3,2} & b_{3,3} \end{bmatrix}$

avec $p_{2,2}$ est le centre de la matrice PSF, alors l'élément $b_{2,2}$ dans le centre de B est donnée par

$$\begin{aligned} b_{2,2} = & p_{3,3}.x_{1,1} + p_{3,2}.x_{1,2} + p_{3,1}.x_{1,3} \\ & + p_{2,3}.x_{2,1} + p_{2,2}.x_{2,2} + p_{2,1}.x_{2,1} \\ & + p_{1,3}.x_{3,1} + p_{1,2}.x_{3,2} + p_{1,1}.x_{3,3}. \end{aligned}$$

Pour les autres coefficients dans cet exemple de taille 3×3, nous devrons faire attention à la condition aux limites. En particulier, si nous supposons que la condition aux limites égale à zéro, alors le coefficient b_{21} à la frontière de B est égale à :

$$\begin{aligned} b_{2,1} = & p_{3,3}.0 + p_{3,2}.x_{1,1} + p_{3,1}.x_{1,2} \\ & + p_{2,3}.0 + p_{2,2}.x_{2,1} + p_{2,1}.x_{2,2} \\ & + p_{1,3}.0 + p_{1,2}.x_{3,1} + p_{1,1}.x_{3,2}. \end{aligned}$$

On a $b = vec(B)$ et $x = vec(X)$ sont liés par

$$\begin{bmatrix} b_{1,1} \\ b_{2,1} \\ b_{3,1} \\ b_{1,2} \\ b_{2,2} \\ b_{3,2} \\ b_{1,3} \\ b_{2,3} \\ b_{3,3} \end{bmatrix} = \left[\begin{array}{ccc|ccc|ccc} p_{2,2} & p_{1,2} & & p_{2,1} & p_{1,1} & & & & \\ p_{3,2} & p_{2,2} & p_{1,2} & p_{3,1} & p_{2,1} & p_{1,1} & & & \\ & p_{3,2} & p_{2,2} & & p_{3,1} & p_{2,1} & & & \\ \hline p_{2,3} & p_{1,3} & & p_{2,2} & p_{1,2} & 0 & p_{2,1} & p_{1,1} & \\ p_{3,3} & p_{2,3} & p_{1,3} & p_{3,2} & p_{2,2} & p_{1,2} & p_{3,1} & p_{2,1} & p_{1,1} \\ & p_{3,3} & p_{2,3} & & p_{3,2} & p_{2,2} & & p_{3,1} & p_{2,1} \\ \hline & & & p_{2,3} & p_{1,3} & & p_{2,2} & p_{1,2} & \\ & & & p_{3,3} & p_{2,3} & p_{1,3} & p_{3,2} & p_{2,2} & p_{1,2} \\ & & & & p_{3,3} & p_{2,3} & & p_{3,2} & p_{2,2} \end{array}\right] \begin{bmatrix} x_{1,1} \\ x_{2,1} \\ x_{3,1} \\ x_{1,2} \\ x_{2,2} \\ x_{3,2} \\ x_{1,3} \\ x_{2,3} \\ x_{3,3} \end{bmatrix}. \quad (5.20)$$

La matrice définie dans (5.20) est une matrice de Toeplitz bande par blocs avec les blocs sont de Toeplitz bandes (TBB/BTB). Elle prend en compte la condition aux limites égale à zéro.
D'autres structures par blocs peuvent être introduites dans le problème de restauration d'images comme :
CB/BC : Circulante par blocs avec les blocs sont circulantes,
HB/BH : Hankel par blocs avec les blocs sont de Hankel,
TB/BH : Toeplitz par blocs avec les blocs sont de Hankel,
HB/BT : Hankel par blocs avec les blocs sont de Toeplitz.
Avec ces notions, nous pouvons décrire précisément la structure de la matrice T pour diverses conditions aux limites :

- Conditions aux limites égales à zéros, dans ce cas la matrice T est une matrice TBB/BTB,
- Conditions aux limites périodiques, dans ce cas où T est une matrice CB/BC,

- Conditions aux limites (de Neumann) réflexive, dans ce cas T est la somme de matrices $BT/TB, TB/BH, HB/BT$, et HB/BH.

Pour plus de détails, voir [42].

5.6.3 Tests numériques

Les tests numériques qui sont présentés dans cette section ont été implémentés sous Matlab (R2013a) sur un ordinateur portable Intel(R) core (TM)i7 CPU-4510u avec un RAM de 8 GO. L'algorithme 5.1 a été implémente et testé pour un problème de restauration d'images, avec une image de taille $N \times N$ pixels, et un bruit PSF défini comme suit :

$$PSF = \frac{1}{25}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}. \tag{5.21}$$

Avec ce PSF, nous avons un système de Toeplitz bande par blocs bandes de Toeplitz, avec 5 bandes et $N \times N$ blocs où chaque bloc est aussi de taille N, avec 5 bandes. Donc le système obtenu est un système de taille $N^2 \times N^2$.

Exemple 1 **:** Dans la Figure 5.1, la Figure 5.1(a) est l'image originale de Cameraman de taille 512×512 récupérée de Matlab. Figure 5.1(b) est l'image bruitée obtenue par convolution entre l'image originale et le PSF de taille 5×5 (5.21). La Figures 5.1(c) est l'image reconstruite par notre algorithme 5.1.

Exemple 2 **:** Dans la figure 5.2, la Figure 5.2(a) est l'image originale de taille 256×256 [42]. Figure 5.2(b) est l'image bruitée obtenue par convolution entre l'image originale et le PSF de taille 5×5 (5.21). La Figure 5.2(c) est l'image reconstruite par notre algorithme 5.1.

Similarité et PSNR

Comme indicateur de qualité d'image, on va utiliser le PSNR et le SSIM, définies comme suit :

PSNR : est une mesure de la qualité de l'image reconstruite par rapport à l'image originale, en décibels (db). Le PSNR d'une image u reconstruite de taille $M \times N$ pixels est défini comme suit :

$$PSNR = 10 \log_{10}\left(\frac{D^2}{MSE}\right) \tag{5.22}$$

où $MSE = \frac{1}{MN}\sum_{m,n}(u_{m,n} - v_{m,n})^2$

- v est l'image de référence,

(a) Image originale

(b) Image bruitée

(c) Image reconstruite par notre algorithme

FIGURE 5.1 – Exemple 1

- D est l'écart entre la plus grande et la plus petite valeur d'intensité de l'image,
- MSE est l'erreur quadratique moyenne entre u et v.

Concernant la **SSIM (Structural Similarity)**, c'est une mesure de similarité entre deux images numériques. La SSIM est calculée sur plusieurs fenêtres d'images, la mesure entre deux fenêtres u et v de taille $N \times N$ est

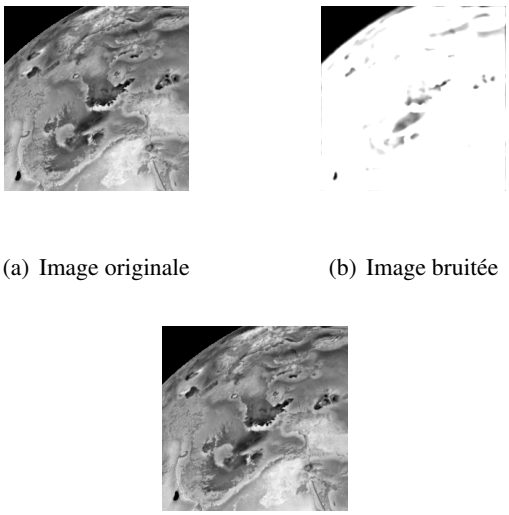
$$SSIM(u,v) = \frac{(2\mu_u\mu_v + d_1)(2cov_{uv} + d_2)}{(\mu_u^2 + \mu_v^2 + d_1)(\sigma_u^2 + \sigma_v^2 + d_2)} \tag{5.23}$$

où

- μ_u (resp μ_v) la moyenne de u (resp v),
- σ_u^2 (resp σ_v^2) la variance de u (resp v),
- cov_{uv} la covariance de u et v,

$d_1 = (k_1 L)^2$ et $d_2 = (k_2 L)^2$ deux variables destinées à stabiliser la division quand le dénominateur est très faible,

- L est la dynamique des valeurs des pixels, soit 100 pour les images codées sur 8 bits,



(a) Image originale (b) Image bruitée

(c) Image reconstruite par notre algorithme

FIGURE 5.2 – Exemple 2

- $k_1 = 0.001$ et $k_2 = 0.03$ par défaut.

Si $u = v$, $SSIM(u, v) = 1$, et si $SSIM(u, v)$ est proche de 1 alors u et v ont presque la même qualité visuelle.
Dans la Table 5.1, nous avons comparé la similarité et le PSNR entre l'image originale et l'image reconstruite. Nous notons que les PSNR et SSIM calculées par notre méthode est un succès à la fois de point de vue du critère PSNR et de point de vue visuel. De même le SSIM de notre méthode est égale à 1.

	Exemple 1		Exemple 2	
	SSIM	PSNR	SSIM	PSNR
Notre méthode	1	+25.84 db	1	+25.51 db

TABLE 5.1 – Qualité d'images

5.7 Conclusion

Dans ce chapitre, nous avons essayé de résoudre un système de Toeplitz bande par blocs avec les blocs sont de Toeplitz bandes, en utilisant la technique de prolongement proposées dans le Chapitre 4. Nous avons abouti à un résultat intéressant mais ça nous n'empêche pas d'avouer d'avoir rencontrer de problèmes, tel que le coût de calcul qui est un peu coûteux et aussi au

niveau de l'inversion de la matrice C définie dans (5.10). Nous pensons que ça est dûe au mal-conditionnent de la matrice proposée par cette application. Ceci nous a encouragé d'appliquer des techniques de pré-conditionnement au niveau de la matrice C, qui restera une question à rependre dans le futur.

CHAPITRE 6

CONCLUSIONS ET PERSPECTIVES

6.1 Conclusions

Le but de cette thèse est la conception de nouveaux algorithmes rapides en calcul numérique via les matrices Toeplitz. Notre contribution consiste essentiellement à concevoir différents types d'algorithmes rapides.

Nous avons introduit un nouvel algorithme pour le calcul rapide de l'inverse d'une matrice triangulaire de Toeplitz. L'objectif recherché est de concevoir des algorithmes rapides pour inverser une matrice triangulaire de Toeplitz et de dégager le meilleur possible, c'est à dire le plus rapide théoriquement et pratiquement, occupant le moins d'espace mémoire possible. Dans cette direction, nous avons focalisé notre attention sur l'amélioration de l'algorithme de Bini [10] pour le calcul de l'inverse d'une matrice triangulaire de Toeplitz. L'idée est inspirée de l'article de Lin, Ching et Ng [55] pour une inversion approchée d'une matrice de Toeplitz triangulaire.

Un nouvel algorithme rapide a été proposé en se basant sur des notions d'interpolation polynomiale. Cet algorithme nécessitant uniquement deux FFT$(2n)$ est manifestement efficace par rapport à ces prédécesseurs. Des expérimentations numériques illustrent l'efficacité, le temps de calcul et la stabilité numérique de cette approche. Une étude d'erreur théorique a été aussi introduite. Ce travail nous a mené à une publication d'un article intitulé "A note on computing the inverse of a triangular Toeplitz matrix" dans la revue "Applied Mathematics and Computation" (2014) [8].

Nous avons aussi réussi à développer un algorithme de résolution d'un système linéaire de Toeplitz bande qui nous a mené à un article publié "A Fast algorithm for solving banded Toeplitz" en "Computers & Mathematics with Application" [9]. Cette nouvelle approche est basée sur l'extension de la matrice donnée dans une matrice augmentée a la structure d'une matrice triangulaire inférieure de Toeplitz. La stabilité de l'algorithme est discutée et son rendement est justifié par des expériences numériques.

Nous avons ensuite abordé le cas de la résolution d'un système de Toeplitz bande par blocs bandes de Toeplitz. Il s'est avéré que ce problème a des difficultés car la généralisation de notre

contribution (dans le cas de matrices de Toeplitz bande) semble un peu coûteuse. Et pour motiver le sujet, nous avons appliquer nos algorithmes à quelques problèmes en traitement d'image un domaine phare en mathématiques appliquées.

6.2 Perspectives

Durant une longue étude concernant ce sujet, nous avons remarqué que les matrices de Toeplitz ont été largement étudiées pour presque environ un siècle. La littérature est vraiment immense. Malgré cela la résolution d'un système de Toeplitz par blocs $TX = F$, où

$$T = \begin{pmatrix} T_0 & T_1 & T_2 & \dots & T_{n-1} \\ T_{-1} & \ddots & \ddots & \ddots & \vdots \\ T_{-2} & \ddots & \ddots & \ddots & T_2 \\ \vdots & \ddots & \ddots & \ddots & T_1 \\ T_{-n+1} & \dots & T_{-2} & T_{-1} & T_0 \end{pmatrix}$$

avec T_i, $i = -n+1, \dots, n-1$ une matrice de taille $m \times m$ et F est le second membre de taille n^2, reste un sujet ouvert vu qu'il est peu étudié. Les matrices multi-niveaux se présentent fréquemment dans les applications multidimensionnelles où les tailles de matrices peuvent être très grandes et des algorithmes rapides deviennent cruciaux. Et malheureusement, les algorithmes de cas scalaires ne sont pas faciles pour s'adapter au cas multi-niveaux. Ce qui nous explique le nombre de publications limitées. Il existe une publication [65], dans la quelle les auteurs ont essayé de résoudre le problème en utilisant la somme de produit Kronecker de matrices de Toeplitz. Et jusqu'à nos jours, il n'existe pas des algorithmes rapides qui résolu un système de Toeplitz par blocs général.

De plus, le problème de restauration d'images peut se transformer en un système linéaire $Tx = b$, avec T est une matrice structurée (matrice circulante par blocs avec les blocs sont des matrices circulantes, voir [59], matrice de Toeplitz par blocs plus une matrice de Hankel par blocs, ...) de taille $n^2 \times n^2$, d'une image de taille $n \times n$ pixels. En général, les matrices obtenues sont des matrices mal-conditionnées et l'étude de pré-conditionnement par les méthodes itératives avec une certaine régularisation est très important.

Bibliographie

[1] J. Abdeljaoued and H. Lombardi. *Méthodes Matricielles. Introduction à la Complexité Algébrique*. Mathématiques et Applications. Springer, 2003.

[2] E. L. Allgower. Exact inverses of certain band matrices. *Numer. Math.*, 21(4) :279–284, 1973.

[3] G.-S. Ammar and W.-B. Gragg. The generalized Schur algorithm for the superfast solution of Toeplitz systems. In *Rational Approximation and its Applications in Mathematics and Physics*, pages 315–330. Springer, 1987.

[4] P. Amodio and L. Brugnano. The conditioning of Toeplitz band matrices. *Mathematical and Computer Modelling*, 23(10) :29–42, 1996.

[5] H. C. Andrews and B. R. Hunt. *Digital Image Restoration*. Prentice Hall Professional Technical Reference, 1977.

[6] M. V. Barel and A. Bultheel. A lookahead algorithm for the solution of block Toeplitz systems. *Linear Algebra and its Applications*, 266 :291–335, 1997.

[7] M. Beam and F. Warming. The asymptotic spectra of banded Toeplitz and quasi-Toeplitz matrices. *SIAM J. Sci. Comput*, 14(4) :971–1006, 1993.

[8] S. Belhaj and M. Dridi. A note on computing the inverse of a triangular Toeplitz matrix. *Applied Mathematics and Computation*, 236 :512–523, 2014.

[9] S. Belhaj, M. Dridi, and A. Salam. A fast algorithm for solving banded Toeplitz systems. *Computers & Mathematics with Applications*, 70(12) :2958–2967, 2015.

[10] D. A. Bini. Parallel solution of certain Toeplitz linear systems. *SIAM Journal on Computing*, 13(2) :268–276, 1984.

[11] D. A. Bini and M. Capovani. Tensor rank and border rank of band Toeplitz matrices. *SIAM J. Comput*, 16(2) :252–258, 1987.

[12] D. A. Bini, G. Fiorentino, L. Gemignani, and B. Meini. Effective fast algorithms for polynomial spectral factorization. *Numerical Algorithms*, 34(2-4) :217–227, 2003.

[13] D. A. Bini, G. Latouche, and B. Meini. *Numerical Methods for Structured Markov Chains (Numerical Mathematics and Scientific Computation)*. Oxford University Press, Inc., New York, NY, USA, 2005.

[14] D. A. Bini and B. Meini. On cyclic reduction applied to a class of Toeplitz-like matrices arising in queueing problems. *In Computations with Markov Chains*, pages 21–38, 1995.

[15] D. A. Bini and B. Meini. Effective methods for solving banded Toeplitz systems. *SIAM J. Matrix Anal. Appl*, 20(3) :700–719, 1999.

[16] D. A. Bini and B. Meini. Solving block-banded block Toeplitz systems with banded Toeplitz blocks. In *Advanced Signal Processing Algorithms, Architectures, and Implementations IX, 300*, volume 3807, pages 300–311, 1999.

[17] D. A. Bini and B. Meini. Solving block banded block Toeplitz systems with structured blocks : New algorithms and open problems. volume 73 of *Notes on Numerical Fluid Mechanics*, pages 15–24. Vieweg, 1999.

[18] D. A. Bini and B. Meini. The cyclic reduction algorithm : from poisson equation to stochastic processes and beyond. *Numerical Algorithms*, 51(1) :23–60, 2009.

[19] D. A. Bini and V. Pan. Polynomial division and its computational complexity. *Journal of Complexity*, 2(3) :179 – 203, 1986.

[20] D. A. Bini and V. Pan. Improved parallel polynomial division and its extensions. In *Foundations of Computer Science, 1992. Proceedings, 33rd Annual Symposium on*, pages 131–136, 1992.

[21] D. A. Bini and V. Pan. *Polynomial and Matrix Computations*. Fundamental Algorithms, 1994.

[22] R. Bitmead and O. Anderson. Asymptotically fast solution of Toeplitz and related systems of linear equations. *Linear Algebra and its Applications*, 34 :103–116, 1980.

[23] L. Brutman. Lebesgue functions for polynomial interpolation - a survey. *Annals of Numerical Mathematics*, 4 :111–128.

[24] S. Capizzano. Optimal, quasi-optimal and superlinear band-Toeplitz preconditioners for asymptotically ill-conditioned positive definite Toeplitz systems. *Math. Comput*, 66(218) :651–665, 1997.

[25] S. Capizzano. Toeplitz preconditioners constructed from linear approximation processes. *SIAM Journal on Matrix Analysis and Applications*, 20(2) :446–465, 1998.

[26] R. Castleman. *Digital Image Processing*. Prentice Hall Press, Upper Saddle River, NJ, USA, 1996.

[27] R. H. Chan and M. K. Ng. Conjugate gradient methods for Toeplitz systems. *SIAM Review*, 38(3) :427–482, 1996.

[28] R. H. Chan and P. Tang. Fast band-Toeplitz preconditioners for hermitian Toeplitz systems. *SIAM J. Scientific Computing*, 15(1) :164–171, 1994.

[29] T. F. Chan and P. C. Hansen. A look-ahead levinson algorithm for general Toeplitz systems. *IEEE Transactions on Signal Processing*, 40(5) :1079–1090, 1992.

[30] W. Chen, C. H. Smith, and S. C. Fralick. A Fast Computational Algorithm for the Discrete Cosine Transform. *IEEE Transactions on Communications*, 25 :1004–1009, 1977.

[31] E. W. Cheney. *Introduction to Approximation Theory*. AMS Chelsea Publishing, New York, 1982.

[32] A. Chesnokov and V. M. Barel. A direct method to solve block banded block Toeplitz systems with non-banded Toeplitz blocks. *J. Comput. Appl. Math.*, 234(5) :1485–1491, 2010.

[33] D. Commenges and M. Monsion. Fast inversion of triangular Toeplitz matrices. *Automatic Control, IEEE Transactions on*, 29(3) :250–251, 1984.

[34] J. Durbin. The fitting of time-series models. *Revue de l'Institut International de Statistique / Review of the International Statistical Institute*, 28(3) :233–244, 1960.

[35] P. Favati, G. Lotti, and O. Menchi. Recursive algorithms for unbalanced banded Toeplitz systems. *Numer Linear Algebra*, 16(7) :561–587, 2009.

[36] G. Fiorentino and S. Serra. Multigrid methods for indefinite Toeplitz matrices. *CALCOLO*, 33(3-4) :223–236.

[37] D. Fischer, G. Golub, O. Hald, C. Leiva, and O. Widlund. On Fourier-Toeplitz methods for separable elliptic problems. *Mathematics of Computation*, 28(126) :349–368, 1974.

[38] H. R. Gail, S. L. Hantler, and B. A. Taylor. Non-skip-free M/G/1 and G/M/1 type markov chains. *Advances in Applied Probability*, 29(3) :733–758, 1997.

[39] R. E. Georgiev. Inversion of triangular Toeplitz matrices by using the fast Fourier transform. *J. New Gener. Comput. Sys*, 2 :247–256, 1989.

[40] G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, 3rd edition, 1996.

[41] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Addison-Wesley, 1992.

[42] P. C. Hansen, J. G. Nagy, and D. P. O'Leary. *Deblurring Images : Matrices, Spectra, and Filtering*. SIAM, 1 edition, 2006.

[43] M. T. Heideman, D. H. Johnson, and C. S. Burrus. Gauss and the history of the fast Fourier transform. *Archive for History of Exact Sciences*, 34(3) :265–277, 1985.

[44] G. Heinig and K. Rost. *Algebraic methods for Toeplitz-like matrices and operators*. Operator Theory, Advances and Applications. Birkhäuser Verlag, 1984.

[45] D. Heller. Some aspects of the cyclic reduction algorithm for block tridiagonal linear systems. *SIAM Journal on Numerical Analysis*, 13(4) :484–496, 1976.

[46] N. J. Higham. *Accuracy and Stability of Numerical Algorithms*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2nd edition, 2002.

[47] A. K. Jain. *Fundamentals of Digital Image Processing*. Prentice-Hall, Englewood Cliffs, 1989.

[48] T. Kailath, S-Y. Kung, and M. Morf. Displacement ranks of matrices and linear equations. *Journal of Mathematical Analysis and Applications*, 68(2) :395–407, 1979.

[49] T. Kailath, A. Vieira, and M. Morf. Inverses of Toeplitz operators, innovations, and orthogonal polynomials. *SIAM Review*, 20(1) :106–119, 1978.

[50] B. Kägström and P. Poromaa. Computing eigenspaces with specified eigenvalues of a regular matrix pair (a ; b) and condition estimation : Theory, algorithms and software. *Numerical Algorithms*, 22(1) :369–407, 1994.

[51] H. Khalil. *Matrices structurées et matrices de Toeplitz par blocs de Toeplitz en calcul numérique et formel*. Thesis, Université Claude Bernard - Lyon I, 2008.

[52] I. Koltracht and P. Lancaster. Condition numbers of Toeplitz and block Toeplitz matrices. In I. Gohberg, editor, *I. Schur Methods in Operator Theory and Signal Processing*, volume 18 of *Operator Theory : Advances and Applications*, pages 271–300. Birkhäuser Basel, 1986.

[53] V. Kucera. Factorization of rational spectral matrices : a survey of methods. In *Control 1991. Control '91, International Conference on, Edinburgh*, pages 1074–1078 vol.2, 1991.

[54] D. P. Laurie. Efficient implementation of Wilson's algorithm for factorizing a self-reciprocal polynomial. *BIT Numerical Mathematics*, 20(2) :257–259, 1980.

[55] F.-R. Lin, W.-K. Ching, and M. K. Ng. Fast inversion of triangular Toeplitz matrices. *Theoretical Computer Science*, 315(2-3) :511 – 523, 2004.

[56] E. K. Lloyd. The art of computer programming. *Software : Practice and Experience*, 12(9) :883–884, 1982.

[57] G. Lotti. A note on the solution of not balanced banded Toeplitz systems. *Numer Linear Algebra*, 14(8) :645–657, 2007.

[58] G. Lotti. A note on the solution of not balanced banded Toeplitz systems. *Numerical Linear Algebra with Applications*, 14(8) :645–657, 2007.

[59] Kin-Wai Mak and R. H. Chan. Parallel implementation of 2-dimensional Toeplitz solver on MasPar with applications to image restoration. In *High Performance Computing on the Information Superhighway, 1997. HPC Asia '97*, pages 389–394, 1997.

[60] A. N. Malyshev and M. Sadkane. Fast solution of unsymmetric banded Toeplitz systems by means of spectral factorizations and Woodbury's formula. *Numerical Linear Algebra with Applications*, 21(1) :13–23, 2014.

[61] M. Morf. Doubling algorithms for Toeplitz and related equations. *Proc. IEEE Int. Conf. on Acoust. Speech and Singual Process*, 3 :954–959, 1980.

[62] M. F. Neuts. *Structured stochastic matrices of M/G/1 type and their applications*. CRC Press, 1989.

[63] M. K. Ng. *Iterative Methods for Toeplitz Systems (Numerical Mathematics and Scientific Computation)*. Oxford University Press, Inc, New York, NY, USA, 2004.

[64] M. K. Ng and R. H. Chan. Scientific applications of iterative Toeplitz solvers. *CALCOLO*, 33(3-4) :249–267.

[65] V. Olshevsky, I. Oseledets, and E. Tyrtyshnikov. Tensor properties of multilevel Toeplitz and related matrices. *Linear Algebra and its Applications*, 412(1) :1–21, 2006.

[66] V. Pan. Complexity of computations with matrices and polynomials. *SIAM Review*, 34(2) :225–262, 1992.

[67] V. Pan. Parallel solution of Toeplitz-like linear systems. *J. Complexity*, 8(1) :1–21, 1992.

[68] V. Pan. Decreasing the displacement rank of a matrix. *SIAM Journal on Matrix Analysis and Applications*, 14(1) :118–121, 1993.

[69] V. Pan. *Structured Matrices and Polynomials : Unified Superfast Algorithms*. Springer Verlag, 2001.

[70] V. Pan, A. Sadikou, and E. Landowne. Polynomial division with a remainder by means of evaluation and interpolation. *Information Processing Letters*, 44(3) :149 – 153, 1992.

[71] V. Pan, R. Youssef, and W. Xinmao. Structured matrices and newton's iteration : unified approach. *Linear Algebra and its Applications*, 343–344 :233–265, 2002. Special Issue on Structured and Infinite Systems of Linear equations.

[72] S. J. Reeves. Fast algorithm for solving block banded Toeplitz systems with banded Toeplitz blocks. In *ICASSP*, pages 3325–3328. IEEE, 2002.

[73] T. J. Rivlin. The Lebesgue constants for polynomial interpolation. In H.G. Garnir, K.R. Unni, and J.H. Williamson, editors, *Functional Analysis and its Applications*, volume 399, pages 422–437. Springer Berlin Heidelberg, 1974.

[74] A. H. Sayed and T. Kailath. A survey of spectral factorization methods. *Numerical Linear Algebra with Applications*, 8(6-7) :467–496, 2001.

[75] A. Schönhage. Asymptotically fast algorithms for the numerical muitiplication and division of polynomials with complex coefficients. In J. Calmet, editor, *Computer Algebra*, volume 144, pages 3–15. Springer Berlin Heidelberg, 1982.

[76] S. Serra. Asymptotic results on the spectra of block Toeplitz preconditioned matrices. *SIAM Journal on Matrix Analysis and Applications*, 20(1) :31–44, 1998.

[77] G. W. Stewart. On the solution of block hessenberg systems. *Numer Linear Algebra*, 2(3) :287–296, 1995.

[78] M. Stewart. A superfast Toeplitz solver with improved numerical stability. *SIAM J. Matrix Analysis Applications*, 25(3) :669–693, 2003.

[79] J. Stoer and R. Bulirsch. *Introduction to Numerical Analysis*. Springer-Verlag, New York, 1992.

[80] W. F. Trench. An algorithm for the inversion of finite Toeplitz matrices. *Journal of the Society for Industrial and Applied Mathematics*, 12(3) :515–522, 1964.

[81] W. F. Trench. Explicit inversion formulas for Toeplitz band matrices. *SIAM Journal on Algebraic Discrete Methods*, 6(4) :546–554, 1985.

[82] C. Van Loan. *Computational Frameworks for the Fast Fourier Transform*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1992.

[83] G. Walter and G. H. Golub. Cyclic reduction - history and applications, 1997.

[84] H. Widom. Review : I. C. Gohberg and , I. A. Fel'dman, Convolution equations and projection methods for their solution. *Bull. Amer. Math. Soc.*, 81 :546–551, 1975.

[85] N. Wiener. *The Wiener RMS (Root Mean Square) Error Criterion in Filter Design and Prediction*, pages 129–148. MIT Press, 1949.

[86] A. E. Yagle. A fast algorithm for Toeplitz-block-Toeplitz linear systems. In *ICASSP*, pages 1929–1932. IEEE, 2001.

[87] P. Yalamov and V. Pavlov. Stability of the block cyclic reduction. *Linear Algebra and its Applications*, 249(1–3) :341–358, 1996.

[88] D. C. Youla and N. Kazanjian. Bauer-type factorization of positive matrices and the theory of matrix polynomials orthogonal on the unit circle. *Circuits and Systems, IEEE Transactions on*, 25(2) :57–69, 1978.

Les méthodes rapides pour la résolution de systèmes de Toeplitz bandes

Résumé

Cette thèse vise à la conception de nouveaux algorithmes rapides en calcul numérique via les matrices de Toeplitz. Tout d'abord, nous avons introduit un algorithme rapide sur le calcul de l'inverse d'une matrice triangulaire de Toeplitz en se basant sur des notions d'interpolation polynomiale. Cet algorithme nécessitant uniquement deux FFT(2n) est manifestement efficace par rapport à ses prédécesseurs. Ensuite, nous avons introduit un algorithme rapide pour la résolution d'un système linéaire de Toeplitz bande. Cette approche est basée sur l'extension de la matrice donnée par plusieurs lignes en dessus, de plusieurs colonnes à droite et d'attribuer des zéros et des constantes non nulles dans chacune de ces lignes et de ces colonnes de telle façon que la matrice augmentée a la structure d'une matrice triangulaire inférieure de Toeplitz. La stabilité de l'algorithme a été discutée et son efficacité a été aussi justifiée. Finalement, nous avons abordé la résolution d'un système de Toeplitz bandes par blocs bandes de Toeplitz. Ceci étant primordiale pour établir la connexion de nos algorithmes à des applications en restauration d'images, un domaine phare en mathématiques appliquées.

Mots-clés : Matrices de Toeplitz, matrices bandes, matrice triangulaire inférieure, interpolation polynomiale trigonométrique, transformation de fourrier rapide (FFT), étude d'erreur, stabilité numérique, factorisation polynomiale, réduction cyclique, restauration d'images.

Fast methods for solving banded Toeplitz systems

Abstract

This thesis aims to design new fast algorithms for numerical computation via the Toeplitz matrices. First, we introduced a fast algorithm to compute the inverse of a triangular Toeplitz matrix with real and/or complex numbers based on polynomial interpolation techniques. This algorithm requires only two FFT (2n) is clearly effective compared to predecessors. A numerical accuracy and error analysis is also considered. Numerical exemples are given to illustrate the effectiveness of our method. In addition, we introduced a fast algorithm for solving a linear banded Toeplitz system. This new approach is based on extending the given matrix with several rows on the top and several columns on the right and to assign zeros and some nonzero constants in each of these rows and columns in such a way that the augmented matrix has a lower triangular Toeplitz structure. Stability of the algorithm is discussed and its performance is showed by numerical experiments. This is essential to connect our algorithms to applications such as image restoration applications, a key area in applied mathematics.

Key-words : Toeplitz matrices, Lower triangular Toeplitz matrices, banded matrices, Fast Fourier transforms (FFT), Trigonometric polynomial interpolation, error study, numerical stability, polynomial factorization, cyclic reduction, image restauration.