République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

UNIVERSITE MOHAMED KHIDER BISKRA
FACULTE DES SCIENCES
DEPARTEMENT DE MATHEMATIQUES

Mémoire

Présenté en vue de l'obtention du diplôme de MAGISTERE
En Mathématiques

Etude numérique des matrices structurées

Option
Analyse & Modèles aléatoires

Présenté par:
Rajah Faouzia

Devant le jury composé de :

Dr. Brahim Mezerdi	**Pr: président**	**(U.M.K. Biskra)**
Dr. Rachid Benacer	**Pr: Examinateur**	**(U.CHL. Batna)**
Dr. Abdehakim Necir	**M. C: Examinateur**	**(U.M.K. Biskra)**
Dr. Lamine Melkemi	**M. C: Rapporteur**	**(U.M.K. Biskra)**

Soutenue le : 09 /11 /2003

Remerciements

Je suis sensible à l'honneur que me fait Monsieur B. Mezerdi, Professeur à l'Université de Biskra, en présidant le jury de ce mémoire. Je rends hommage à ses qualités humaines et scientifiques tout au long de mon cursus universitaire. Je lui adresse mes vifs remerciements.

Mes remerciements vont aussi à Monsieur R. Benacer, Professeur à l'Université de Batna, pour l'honneur qu'il me fait en acceptant d'examiner le mémoire et de faire partie du jury.

Je suis heureuse également de remercier, Monsieur A. Necir, Maître de conférence à l'Université de Biskra d'avoir accepté d'examiner le mémoire et de participer au jury. Je sensible à sa générosité qu'il m'a témoignée et son aide qu'il m'a prodiguée. Je lui exprime ma gratitude pour ses conseils précieux.

Ce mémoire a été proposé et dirigé par Monsieur L. Melkemi, Maître de conférence à l'Université de Biskra, Je tiens à lui exprimer ici ma reconnaissance et qu'il accepte mes remerciements.

Je voudrais adresser mes remerciements à tous ceux qui m'ont aidé à mener ce travail à son terme. En particulier, les étudiants de la première et la deuxième année post graduation électronique et mathématique.

ملخص

في الاطروحة التي بحوزتكم عرضنا دراسة كاملة حول المصفوفات البنيوية، حيث ركزنا اهتمامنا على مصفوفات توبليتز و فاندارموند، كما قدمنا بنيةانتقالية من نوع سيلفاستر لمصفوفات فاندارموند المرفدية، التي هي عبارة عن تعميم للبنية المعروفة في حالة مصفوفة فاندارموند العادية. النتائج الجديدة في هذه الاطروحة تهدف الى حل جملة معادلات بنيوية خطية لفاندارموند المرفدية باستعمال طريقة غوص السريعة والتي هي عبارة عن تكرارات منتهية لمتممة شور.

RESUME

Dans ce mémoire, nous donnons une étude complète sur les matrices structurées, nous nous intéressons aux matrices Toeplitz et Vandermonde et nous présentons une structure de déplacement de type Sylvester pour les matrices de Vandermonde confluentes qui apparaissent comme une généralisation naturelle connue dans le cas d'un système de Vandermonde simple. Notre nouveau résultat dans ce mémoire a comme objectif la résolution d'un système linéaire structuré de Vandermonde confluent par l'utilisation de la méthode d'élimination de Gauss rapide par blocs, qui n'est rien d'autre qu'une répétition finie du complément de Schur.

ABSTRACT

In This memory, we give a complete study on the structured matrices. We are interested in Toeplitz and Vandermonde matrices and we present displacement structure of the sylvester type for the confluent Vandermonde matrices which appear as a natural generalisation well known in the case of the simple Vandermonde system. Our new result has as objective the resolution of a structure Vandermond's linear system which is confluent, by the use of the rapid elimination Gauss's method by blocks, which is a finished repetition of Schur's complement.

Table des matières

1 Introduction générale. **3**

2 Transformation Discrète de Fourier. **7**

2.1 **TDF et inverse :** 8

2.2 **TDF et polynômes :** 9

2.3 **TDF et matrices circulantes :** 10

2.4 **FFT itérative :** 11

2.5 **Analyse de l'erreur :** 15

3 Matrices Toeplitz. **18**

3.1 **Structure de déplacement pour les matrices Toeplitz :** 19

3.2 **matrices Toeplitz spéciales :** 22

3.2.1 Matrices Toeplitz circulantes : 22

3.2.2 Matrices Toeplitz triangulaires : 23

3.3 **Produit d'une matrice Toeplitz par un vecteur :** 28

3.4 **Transformation en une matrice Toeplitz :** 30

4 Structures de déplacement des matrices de Vandermonde. **32**

4.1 **matrices de Vandermonde :** 32

4.2 **matrices de Vandermonde par blocs :** 35

4.3 **Matrices de Vandermonde confluentes :** 36

4.4 **Structure de déplacement pour les matrices de Vandermonde confluentes :** 37

5 Elimination de Gauss pour les matrices de Vandermonde confluentes. 42

5.1 **Stabilité de la structure de déplacement par le complément de Schur :** 43

5.2 **Structure de déplacement de W :** 45

5.3 **Structures de déplacement pour les éléments en blocs :** 49

5.3.1 Stockage de la première ligne de W : 50

5.3.2 Stockage de la première colonne de W : 51

5.3.3 Stockage de la première ligne de $S^k(W)$: 51

5.3.4 Stockage de la première colonne de $S^k(W)$: 53

5.4 **Opérations rapides sur les éléments en blocs.** 55

5.5 **L'algorithme d'élimination de Gauss par blocs et complexité :** 59

6 Annexe des travaux pratiques. 64

Chapitre 1

Introduction générale.

L'analyse numérique est une discipline carrefour, ce qui en fait l'intérêt et la difficulté. L'étude et la résolution d'un problème d'analyse numérique en vraie grandeur passe par plusieurs phases.

Depuis plusieurs années, l'analyse numérique connaît un sort considérable et la plupart des facultés de sciences et de génie offrent au moins un cours d'introduction à cette discipline. Les méthodes numériques fournissent un outil extrêmement performant qui permet d'aborder et de résoudre des problèmes dont la solution est inimaginable par les méthodes analytiques classiques. La maîtrise de cet outil est devenue indispensable dans la formation scientifique en général et en particulier dans celle des ingénieurs.

Parmi les buts importants de l'analyse numérique et sur lesquels nous nous orientons dans ce mémoire, on peut citer la résolution d'un système linéaire à cause de leurs utilités pratiques et théoriques. Le principe de ce but est toujours le même : on se donne une matrice A et un vecteur b et on cherche a trouvé des algorithmes performants permettant de déterminer le vecteur x solution de l'équation $Ax = b$. Dans la plupart des cas, les méthodes standards de la résolution des systèmes linéaires du type $(n \times n)$ coûtant $O(n^3)$ opérations arithmétique est inapplicable lorsque n est grand. Heureusement, plusieurs équations linéaires prennent certaines structures dans l'application, ces structures sont exploitées pour la rapidité du calcul. Par exemple les systèmes de Vandermonde linéaires peuvent être résolus rapidement par l'utilisation de $O(n^2)$ opérations seulement.

Les systèmes d'équations linéaires de Vandermonde (en l'honneur du mathématicien français Alexandre Vandermonde (1735-1796)) apparaissent naturellement lorsqu'on veut trouver un polynôme de degré $n-1$ qui prend

des valeurs données q_i en n points donnés distincts x_i (polynôme d'interpolation de Lagrange). On est alors amené à résoudre le système d'équations linéaires.

Il est connu que les matrices du type Vandermonde deviennent très mal conditionnées lorsque n augmente. Ceci entraîne qu'on devra éviter d'interpoler par des polynômes de degré élevé.

Dans notre mémoire dont l'objet est la résolution d'un système linéaire structuré, nous sommes intéressés particulièrement aux matrices de Vandermonde confluentes. Ce sont des matrices bien connues dans la littérature dotées de structure de déplacement convenable dans les applications.

Pour aboutir à une telle structure pour les matrices de Vandermonde confluentes simplement, il faut aboutir une généralisation naturelle de celle connue dans le cas d'un système de Vandermonde simple. La démonstration de ce résultat tire profit d'une propriété intéressante disant, dans un sens que nous préciserons plus loin, qu'une telle matrice est en fait plonger dans une matrice de Vandermonde par blocs. Ce plongement n'est rien d'autre qu'une matrice de permutation qui nous permet d'obtenir l'équation de déplacement vérifiée par les matrices de Vandermonde confluentes à partir de l'équation de déplacement vérifié par les matrices de Vandermonde par blocs.

La plupart des opérations qu'on a effectuées dans notre étude tournent autour des opérations appliquées sur les matrices Toeplitz, parmi lesquelles, on peut citer les deux opérations suivantes : 1) le calcul du produit matrice-vecteur Tx, où T est Toeplitz triangulaire, et 2) le calcul du produit x^TT, où T est Toeplitz triangulaire aussi, d'où le calcul de Tx et x^TT est fortement lié aux opérations sur les polynômes à une variable. Alors, ces opérations de base sont réalisées par des algorithmes rapides qu'effectuent $O(d \log d)$ opérations. Ces algorithmes sont numériquement stables et connues dans la littérature sous le nom de FFT (Fast Fourier Transform). C'est pour cette raison, nous avons présentés ce mémoire par la transformation discrète de Fourier dans le but d'introduire les matrices Toeplitz.

Ce mémoire est réparti en cinq chapitres.

Au deuxième chapitre, nous nous intéressons à la pertinence visuelle des caractéristiques de la transformation discrète de Fourier dans les calculs, plus particulièrement dans le calcul des polynômes. Nous présentons quelques techniques et résultats fondamentaux satisfaits par la transformation discrète

de Fourier afin de réduire la complexité de notre algorithmes présenter dans ce mémoire.

Nous avons consacrés le troisième chapitre à l'exposé d'une définition formelle des matrices Toeplitz, qui prenne une forme particulière nous permet de trouver facilement les structures de déplacement qui les caractérisent, puis, nous réexaminons cette classe des matrices Toeplitz d'un point de vue algébrique et montrons leurs corrélations avec les polynômes en une variable, lesquelles jouent un rôle très important dans la réalisation du produit d'une matrice Toeplitz par un vecteur. Dans ce cas FFT (Fast Fourier Transform) devient un outil fondamental pour les calculs des matrices dans cette classe.

Au quatrième chapitre, nous construisons des structures de déplacement utiles pour les matrices de Vandermonde confluentes. L'idée principale dans cette construction repose sur un résultat récent établi dans [34] affirmant que les matrices de Vandermonde confluentes font partie, tout comme les matrices de Vandermonde, de la classe des matrices structurées.

Au début, on donne l'équation de déplacement vérifiée par les matrices de Vandermonde, de sorte qu'on peut déduire la structure de déplacement qui caractérise les matrices de Vandermonde par blocs à partir de cette dernière. Les résultats (4.2) et (4.3) affirment qu'une matrice de Vandermonde confluente est plongée dans une matrice de Vandermonde par blocs. L'application de ce plongement à l'équation de déplacement des matrices de Vandermonde par blocs donne directement l'équation de déplacement caractérisant les matrices de Vandermonde confluentes.

Dans le dernier chapitre, on présente les principaux nouveaux résultats concernant la résolution d'un système linéaire structuré de Vandermonde confluent par l'application de la méthode d'élimination de Gauss par blocs rapide, qu'il n'est rien d'autre qu'une répétition finie de l'opération du complément de Schur. Afin de faciliter l'apprentissage, on présente à la première section de ce chapitre un rappel sur ce dernier. On démontre la stabilité du complément de Schur par des structures de déplacement convenables de la forme (5.1). On voit qu'il est obligatoire de transformer la structure de déplacement (4.11) qui caractérisant les matrices de Vandermonde confluentes en une structure de déplacement prenne la forme (5.1) pour le but d'appliquer le complément de Schur sur ces matrices. Dans ce qui suit, on utilise la stabilité de ce dernier pour obtenir étape par étape le processus d'élimination de Gauss par blocs, qu'on trouve après qu'il est effectué d'une façon itérative basé sur des opérations par blocs des matrices structurées de complément de Schur ayant des structures de déplacement convenables, c'est pour cela, il

est nécessaire de présenter les deux algorithmes (5.1) et (5.2) pour effectuer ces opérations de base, lesquels sont rapides. On termine ce chapitre par la présentation de notre résultat principale sous forme d'un algorithme et on étudie sa complexité.

Chapitre 2

Transformation Discrète de Fourier.

Il s'agit d'une opération numérique de base qui est fameuse et bien documentée de par sa grande utilité pratique et théorique.

Considérons la matrice du type $(n \times n)$ suivante :

$$F(w,n) = (w^{(i-1)(j-1)})_{1 \leq i,j \leq n}, \tag{2.1}$$

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

$$w_n = e^{i\frac{2\pi}{n}}, \quad w_n^{-1} = \overline{w_n} = e^{-i\frac{2\pi}{n}} \qquad i^2 = -1 \tag{2.2}$$

sont des racines $n^{ième}$ principales de l'unité. Ceci étant, le vecteur

$$y = F(w,n).x \tag{2.3}$$

résultat du produit de $F(w,n)$ par le vecteur x, s'appelle la transformation discrète de Fourier (en abrégé TDF) de x.

La popularité de la transformation discrète de Fourier découle notament de l'existence d'algorithmes réalisant l'opération (2.3) en utilisant $O(n \log n)$ opérations élémentaires seulement. Ces algorithmes rapides sont plus connus sous le nom de FFT (pour Fast Fourier Transform). Ce qui nous a motivé à considérer la transformation discrète de Fourier dans ce chapitre réside

d'un côté dans le fait que les algorithmes que nous proposons dans cette thèse font appel à la FFT afin d'améliorer leur complexité, et d'un autre côté dans l'intention de bien préciser que la FFT, en plus de sa rapidité, est une opération numériquement stable. Ce point étant ; jusque là ; rarement discuté dans la littérature.

Avant de détailler cet aspect, nous jugeons utile de rappeler ; ne serait ce que brièvement ; quelques résultats fondamentaux satisfaits par la TDF.

2.1 TDF et inverse :

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

$$F(w,n)^{-1} = \frac{1}{n}F(w^{-1},n), \tag{2.4}$$

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

Posons par ailleurs

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

on peut alors vérifier que

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

de sorte qu'on puisse déduire que Q est unitaire. Notons par $\|.\|$ la norme euclidienne dans $\mathbb{C}^n$, et rappelons que si $A \in \mathbb{C}^{n\times n}$

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

On peut énoncer le résultat suivant :

Théorème 2.1 : Supposons que $y = F(w,n).x$. Alors

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

(ii) $\|y\| = \sqrt{n}\,\|x\|$. □

2.2 TDF et polynômes :

Considérons le problème consistant à calculer le produit $C(x) = A(x) \cdot B(x)$ de deux polynômes $A(x)$ et $B(x)$ de degré $< n$. En posant $A(x) = \sum_{i=0}^{n-1} a_i x^i$ et $B(x) = \sum_{i=0}^{n-1} b_i x^i$, il est immédiat que

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

avec

$$c_i = \sum_{k+j=i} a_k b_j. \tag{2.5}$$

Aussi il est clair que l'approche directe pour calculer $C(x)$ requiert au pire des cas $O(n^2)$ opérations élémentaires. Pour améliorer cette complexité, il convient d'abord d'interpréter le produit polynômial ainsi considéré en termes de vecteurs. Dans ce sens, on pose $a = \left[\begin{array}{cccccc} a_0 & a_1 & \dots & a_{n-1} & 0 & \dots & 0 \end{array}\right]^T$, $b = \left[\begin{array}{cccccc} b_0 & b_1 & \dots & b_{n-1} & 0 & \dots & 0 \end{array}\right]^T$ et $c = \left[\begin{array}{cccc} c_0 & c_1 & \dots & c_{2n-2} & 0 \end{array}\right]^T$ de sorte que $a, b, c \in \mathbb{C}^{2n}$.

Ensuite, on fait intervenir la TDF en se basant sur le résultat suivant :

Théorème 2.2 : Soit w une racine $(2n)^{ième}$ principale de l'unité et posons $F = F(w, 2n)$. Alors

$$F.c = (F.a) \odot (F.b) \tag{2.6}$$

où $\odot$ désigne le produit ponctuel :

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

En faisant appel à la FFT, on en déduit qu'il est possible de calculer $c^* = F.c$ en utilisant $O(n \log n)$ opérations élémentaires. Or d'après (2.4) on a $F(w, 2n)^{-1} = \frac{1}{2n} F(w^{-1}, 2n)$. Par conséquent

$$c = \frac{1}{2n} F(w^{-1}, 2n).c^*$$

et peut ainsi être déterminé en utilisant $O(n \log n)$ opérations élémentaires. Ci après un algorithme réalisant le produit polynômial.

Algorithme 2.1. **prod** $(A(x), B(x))$.

Données : $a_0, a_1, \ldots, a_{n-1}$ et $b_0, b_1, \ldots, b_{n-1}$.
Résultats : $c_0, c_1, \ldots, c_{2n-2}$.
1. $a^* = F(w, 2n).a \quad \left(a = \left[\begin{array}{ccccccc} a_0 & a_1 & \ldots & a_{n-1} & 0 & \ldots & 0 \end{array}\right]^T\right)$
2. $b^* = F(w, 2n).b \quad \left(b = \left[\begin{array}{ccccccc} b_0 & b_1 & \ldots & b_{n-1} & 0 & \ldots & 0 \end{array}\right]^T\right)$
3. $c^* = a^* \odot b^*$ ($\odot$ est le produit ponctuel)
4. $c = \frac{1}{2n} F(w^{-1}, 2n).c^*$.
Complexité : $O(n \log n)$ opérations.

2.3 TDF et matrices circulantes :

Soit $A \in \mathbb{C}^{n \times n}$ une matrice complexe du type $(n \times n)$. On dit que A est circulante si :

$$A_{ij} = f\left[(j-i)\,[\mathrm{mod}\, n]\right] \quad \forall i,\ j;\ 1 \leq i,\ j \leq n \tag{2.7}$$

où $k\,[\mathrm{mod}\, n]$ désigne le reste de la division de k sur n et est compris entre 0 et $n-1$. La relation (2.7) signifie que A_{ij} est fonction de $(j-i)\,[\mathrm{mod}\, n]$.

Exemple de matrices circulantes :

$$\left[\begin{array}{cccc} a_0 & a_3 & a_2 & a_1 \\ a_1 & a_0 & a_3 & a_2 \\ a_2 & a_1 & a_0 & a_3 \\ a_3 & a_2 & a_1 & a_0 \end{array}\right] \qquad \left[\begin{array}{ccc} a_0 & a_2 & a_1 \\ a_1 & a_0 & a_2 \\ a_2 & a_1 & a_0 \end{array}\right].$$

Il est clair qu'une fois la première colonne d'une matrice circulante est connue, les autres colonnes peuvent être directement déduites. Cette observation justifie bien l'écriture $C(a)$ pour désigner la matrice circulante dont la première colonne est le vecteur a. Par exemple si $a = \left[\begin{array}{cccc} 1 & 0 & -1 & 2 \end{array}\right]^T$, alors on a :

$$C(a) = \left[\begin{array}{cccc} 1 & 2 & -1 & 0 \\ 0 & 1 & 2 & -1 \\ -1 & 0 & 1 & 2 \\ 2 & -1 & 0 & 1 \end{array}\right].$$

On sait que le produit de deux matrices circulantes est une matrice circulante. De même, l'inverse d'une matrice circulante est une matrice circulante. En fait nous pouvons, grâce à la TDF, caractériser d'une façon complète les matrices circulantes.

Avant d'établir ce fait on adopte la notation suivante :

Notation : Soit $v = \begin{bmatrix} v_1 & \dots & v_n \end{bmatrix}^T \in \mathbb{C}^n$, notons alors

$$D_v = diag\,(v) = \begin{bmatrix} v_1 & 0 & \cdots & 0 \\ 0 & v_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & v_n \end{bmatrix}.$$

Ceci étant, on peut énoncer le résultat suivant :

Théorème 2.3 : Soit $a \in \mathbb{C}^n$ un vecteur d'ordre n et soit $a^* = F(w,n)a$ sa transformée discrète de Fourier. Posons $F = F(w,n)$. Alors :

(i) $F.C(a).F^{-1} = D_{a^*}$

(ii) $F^{-1}.D_{a^*}.F = C(a)$. $\square$

Ce résultat affirme que toute matrice circulante est semblable à une matrice diagonale assurant ainsi la caractérisation entière des matrices circulantes.En effet, on peut dire que l'ensemble des matrices circulantes est formé de $F^{-1}.D_v.F$ où v parcourt $\mathbb{C}^n$.

2.4 FFT itérative :

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

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

et

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

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

$$\Delta_w = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & w & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & w^{\frac{n}{2}-1} \end{bmatrix}. \tag{2.8}$$

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

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

$$F(w,n).a = \begin{bmatrix} I_{\frac{n}{2}} & \Delta_w \\ I_{\frac{n}{2}} & -\Delta_w \end{bmatrix} \cdot \begin{bmatrix} F(w^2, \frac{n}{2}).a^{[0]} \\ F(w^2, \frac{n}{2}).a^{[1]} \end{bmatrix}. \tag{2.9}$$

En développant davantage, on obtient :

$$F(w,n).a =$$

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

Pour simplifier, posons :

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

et

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

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

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

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

$$A_j = I_{2^{j-1}} \otimes \begin{bmatrix} I_{\frac{n}{2^j}} & \Delta_{w^{2^{j-1}}} \\ I_{\frac{n}{2^j}} & -\Delta_{w^{2^{j-1}}} \end{bmatrix}. \tag{2.10}$$

On peut alors énoncer le résultat suivant :

Théorème 2.4 : Il existe une matrice de permutation P qu'on peut déterminer telle que

$$F(w, n) = A_1.A_2.A_3 \cdots A_q P. \quad \square \tag{2.11}$$

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

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

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

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

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

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

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

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

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

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

Le résultat suivant détermine la matrice de permutation en question.

Théorème 2.5 : On a $\forall j = 1 : n$

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

(e_j) désignant la base canonique □.

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

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

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

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

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

$$\begin{aligned} y_{t+km} &= x_{t+km} + w^{t \cdot 2^{j-1}} x_{t+km+\frac{m}{2}} \\ y_{t+km+\frac{m}{2}} &= x_{t+km} - w^{t \cdot 2^{j-1}} x_{t+km+\frac{m}{2}}. \end{aligned} \tag{2.12}$$

Enfin, on peut présenter l'algorithme suivant qu'on peut nommer la méthode FFT itérative basée sur la formule (2.11).

Algorithme 2.2 : FFT itérative. Dans cet algorithme, on suppose qu'on dispose des n racines de l'unité : 1, w, w^2, ..., w^{n-1} stockées dans un tableau nommé R. Par conséquent $R[k] = w^{k-1}$. L'objet de cet algorithme consiste à réaliser l'opération $b = Fa = A_1 A_2 \cdots A_q P a$, où $a = \begin{pmatrix} a_0 & a_1 & \cdots & a_{n-1} \end{pmatrix}^T$ et $b = \begin{pmatrix} b_0 & b_1 & \cdots & b_{n-1} \end{pmatrix}^T$.

Commentaire : calcul de $Pa = c$.

1. **pour** $s = 0 : n-1$
2. $\quad c_s = a_{rev(s)+1}$;
3. **pour** $j = q : 1$
4. $\quad$ **pour** $k = 0 : 2^{j-1} - 1$
5. $\quad\quad m = \frac{n}{2^{j-1}}$;
6. $\quad\quad$ **pour** $t = 0 : \frac{m}{2} - 1$
7. $\quad\quad\quad d[t + km] = c[t + km] + R[t \cdot 2^{j-1}]\, c\left[t + km + \frac{m}{2}\right]$;
8. $\quad\quad\quad d\left[t + km + \frac{m}{2}\right] = c[t + km] - R[t \cdot 2^{j-1}]\, c\left[t + km + \frac{m}{2}\right]$;
9. $\quad\quad$ fin pour
10. $\quad$ fin pour
11. **pour** $s = 0 : n-1$
12. $\quad c[s] = d[s]$;
13. **pour** $s = 0 : n-1$
14. $\quad b[s] = c[s]$.

Pour analyser la complexité de cet algorithme, on va compter le nombre de multiplications et négliger celui de l'addition. Notons par $*_{\mathbb{C}}$ une mutiplication complexe. On observe qu'au niveau des lignes 7 et 8, on a $2 \cdot *_{\mathbb{C}}$. Avec la boucle 6-9, on compte $2 \cdot \frac{m}{2} \cdot *_{\mathbb{C}} = \frac{n}{2^{j-1}} \cdot *_{\mathbb{C}}$. On utilise donc $n \cdot *_{\mathbb{C}}$ dans le segment 4-10. En conséquence, on totalise $qn \cdot *_{\mathbb{C}} = n(\log_2 n) *_{\mathbb{C}}$.

2.5 Analyse de l'erreur :

On suppose qu'on travaille dans un système de nombres en virgule flottante d'unité d'erreur d'arrondi u et basé sur le modèle standard suivant :

$$fl\ (x\ op\ y) = (x\ op\ y)\ (1+\varepsilon)^{\pm 1} \quad |\varepsilon| \leq u,$$

où $op = +, -, *, /, \ldots$. Dans un tel système les opérations (2.12) deviennent :

$$\begin{aligned} \widehat{y}_{t+km} &= x_{t+km}(1+\varepsilon_1) + w^{t\cdot 2^{j-1}} x_{t+km+\frac{m}{2}}(1+\alpha) \\ \widehat{y}_{t+km+\frac{m}{2}} &= x_{t+km}(1+\varepsilon_2) - w^{t\cdot 2^{j-1}} x_{t+km+\frac{m}{2}}(1+\beta). \end{aligned}$$

On peut supposer que $|\alpha|, |\beta|, |\varepsilon_1|, |\varepsilon_2| \leq 4u$. En conséquence le calcul de $y = A_j x$ dans ce système donne $\widehat{y}$ tel que :

$$\widehat{y} = (A_j + \Delta A_j)x$$

avec

$$|\Delta A_j| \leq 4uM_j$$

$$M_j = |A_j| = I_{2^{j-1}} \otimes \begin{bmatrix} I_{\frac{n}{2^j}} & I_{\frac{n}{2^j}} \\ I_{\frac{n}{2^j}} & I_{\frac{n}{2^j}} \end{bmatrix}.$$

On en déduit le théorème suivant :

Théorème 2.6 : La réalisation de $b = Fa$ par la FFT présentée par l'algorithme précédent en utilisant le système adopté donne le vecteur approché $\widehat{b}$ tel que :

$$\widehat{b} = (A_1 + E_1)(A_2 + E_2)\ldots(A_q + E_q)Pa, \quad |E_j| \leq 4uM_j,$$

ou bien

$$\widehat{b} = (A_1 \cdot A_2 \ \ldots \ A_q + E)\, Pa, \quad |E| \leq 4quM_1M_2\ldots M_q. \quad \square$$

Il vient

$$\widehat{b} = b + EPa,$$

donc

$$\left|\widehat{b} - b\right| \leq 4(\log_2 n)uM_1M_2\ldots M_qPa.$$

Par ailleurs, il n'est pas difficile de montrer que :

$$\|M_j\| = 2 \quad (\forall j = 1:q).$$

En conséquence

$$\left\|\widehat{b} - b\right\| \leq 4uq\ 2^q \|a\| = (4un\log_2 n)\|a\|.$$

Or

$$\left\| \frac{1}{\sqrt{n}} Fa \right\| = \|a\|,$$

ou

$$\|a\| = \frac{1}{\sqrt{n}} \|b\|,$$

on en déduit que :

$$\left\| \widehat{b} - b \right\| \leq 4u\sqrt{n}\,(\log_2 n)\,\|b\|. \tag{2.13}$$

Ce résultat permet de conclure que la FFT itérative est en norme numériquement stable. Il est bien quand même de signaler à ce propos que nous nous sommes accordés deux facilités. D'un côté, on a supposé que les w^j sont calculés au départ et d'une manière exacte. C'est une hypothèse tout à fait valable puisque la plupart des ordinateurs sont dotés de logiciels évaluant les fonctions élémentaires d'une manière très précise. D'un autre côté, on a supposé que si $\prod_{i=1}^{n}(1+\delta_i) = 1+\delta$ avec $|\delta_i| \leq u$ $(\forall i = 1:n)$, on a $|\delta| \leq nu$, ce qui est mathématiquement faux. Rigoureusement, on a l'implication suivante :

$$|\delta_i| \leq u\ (\forall i = 1:n) \Longrightarrow |\delta| \leq \frac{nu}{1-nu} \tag{2.14}$$

à condition que $nu < 1$. Heureusement, il a été établi que les résultats établis sur la base de notre hypothèse à priori erronée sont modulo $O(u^2)$ identiques à ceux obtenus sur la base de l'implication (2.14). Désormais, on suppose que :

$$\left\| \widehat{b} - b \right\| \leq \varepsilon u \|b\|$$

et

$$\widehat{b} = (F + \Delta F)\,a, \quad |\Delta F| \leq \varepsilon. \tag{2.15}$$

Chapitre 3

Matrices Toeplitz.

Dans ce chapitre, on a comme but la présentation des matrices Toeplitz et ses propriétés avec quelques techniques de base dans les calculs matriciels, qu'on aura remarqué qu'elles sont fortement liées aux opérations sur des polynômes à une variable et qu'on peut réaliser d'une façon rapide par l'utilisation de FFT.

Définition : Une matrice $T \in \mathbb{R}^{m \times n}$ du type $(m \times n)$ est de Toeplitz, si ses éléments sont de la forme :

$$T_{i,j} = T_{i-j}.$$

D'une autre manière, une matrice Toeplitz est caractérisée par le fait que ses éléments le long de chacune de ses diagonales parallèles à la diagonale principale sont égaux.

Exemple :

$$T = \begin{bmatrix} 1 & 4 & 0 & 2 \\ 3 & 1 & 4 & 0 \\ -2 & 3 & 1 & 4 \\ -5 & -2 & 3 & 1 \\ 7 & -5 & -2 & 3 \\ 5 & 7 & -5 & -2 \end{bmatrix},$$

T est une matrice Toeplitz du type (6×4). On observe bien que chaque matrice Toeplitz est définie à partir de sa première ligne et sa première colonne, c'est pour cela dans ce qui suit, on peut noter $L(a, b)$ la matrice Toeplitz dont

a représente sa première ligne et b sa première colonne, dans notre exemple on a :

$$a = \begin{bmatrix} 1 & 3 & -2 & -5 & 7 & 5 \end{bmatrix}^T$$

et

$$b = \begin{bmatrix} 1 & 4 & 0 & 2 \end{bmatrix}.$$

S'il est vrai que la somme de deux matrices Toeplitz est une matrice Toeplitz, il n'en est pas forcément de même pour les autres opérations. Ainsi l'inverse d'une matrice Toeplitz régulière qui est souvent demandé dans les applications n'est pas en générale Toeplitz. C'est cet état de fait qui a conduit à la construction de structures de déplacement adéquates. La section suivante sera consacrée à cet aspect.

3.1 Structure de déplacement pour les matrices Toeplitz :

La structure de déplacement associée aux matrices Toeplitz est donnée sous la forme :

$$Z_t T - T Z_s = e_1 u^T + v e_n^T \tag{3.1}$$

où $T \in \mathbb{R}^{m \times n}$ est une matrice Toeplitz du type $(m \times n)$ et $u \in \mathbb{R}^m$, $v \in \mathbb{R}^n$, $Z_t \in \mathbb{R}^{m \times m}$ et $Z_s \in \mathbb{R}^{n \times n}$ sont des matrices de déplacement élémentaires de la forme :

$$Z_t = \begin{bmatrix} 0 & 0 & 0 & \cdots & t \\ 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & & 0 \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix}. \tag{3.2}$$

On remarque bien que chaque matrice Toeplitz est carctérisée par sa structure de déplacement (3.1) au lieu de ses $m \times n$ éléments (T_{ij}); $1 \leq i \leq m$, $1 \leq j \leq n$ dans la représentation usuelle. Alors la nouvelle représentation permet de reduire la complexité des opérations sur ces matrices, ainsi que l'espace mémoire utilisé.

Pour se convaincre que les matrices Toeplitz vérifient (3.1) il importe de le prouver. Soit T une matrice Toeplitz du type $(m \times n)$, dont on peut la représenter sous la forme :

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

Grâce à la définition d'une matrice Toeplitz, il est clair que ses éléments sont caractérisés par la relation suivante :

$$T_{i,j} = T_{i+1,j+1} \quad \textit{pour} \quad i = 0, m-2 \textit{ et } j = 0, n-2. \tag{3.3}$$

La prémultiplication de T par la matrice de permutation Z_t définie dans (3.2) donne la matrice suivante :

$$Z_t T = \begin{bmatrix} tt_{m-1} & tt_{m-2} & \cdots & \cdots & tt_0 \\ t_0 & t_{-1} & \cdots & \cdots & t_{-n+1} \\ \vdots & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \vdots \\ t_{m-2} & \cdots & \cdots & t_0 & t_{-1} \end{bmatrix}.$$

Si on pose $A = Z_t T$, on remarque que ses éléments sont donnés par :

$$\left.\begin{array}{ll} A_{0,j} & = tT_{m-1,j} \\ A_{i,j} & = T_{i-1,j} \quad \text{pour} \quad i = \overline{1, m-1} \end{array}\right\} \text{pour} \quad j = \overline{0, n-1}.$$

Ainsi que la postmultiplication de T par Z_s définie dans (3.2), en remplaçant t par s, donne :

$$TZs = \begin{bmatrix} t_{-1} & t_{-2} & \cdots & t_{-n+1} & st_0 \\ t_0 & t_{-1} & \cdots & \cdots & st_1 \\ \vdots & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & t_{-1} & \vdots \\ t_{m-2} & \cdots & \cdots & t_0 & st_{m-1} \end{bmatrix}.$$

Si on pose maintenant : $B = TZ_s$, les éléments de B sont :

$$\left.\begin{array}{ll} B_{i,j} & = T_{i,j+1} \quad \text{pour} \quad j = \overline{0, n-2} \\ B_{i,n-1} & = sT_{i,0} \end{array}\right\} \text{pour} \quad i = \overline{0, m-1}.$$

Alors, les éléments de la matrice $C = A - B$ sont caractérisés par :

- Pour $i = 0$ $\quad C_{0,j} = A_{0,j} - B_{0,j}$
  $= tT_{m-1,j} - tT_{0,j+1} \quad \text{pour} \quad j = \overline{0, n-2}$
- Pour $i = 0$ et $j = n-1$ $\quad C_{0,n-1} = A_{0,n-1} - B_{0,n-1}$
  $= tT_{m-1,n-1} - sT_{0,0}$
- Pour $j = n-1$ $\quad C_{i,n-1} = A_{i,n-1} - B_{i,n-1}$
  $= T_{i-1,n-1} - sT_{i,0} \quad \text{pour} \quad i = \overline{1, m-1}$

les éléments qui restent de C, $\left(i = \overline{1, m-1}\right)$ et $\left(j = \overline{0, n-2}\right)$ sont nuls, c'est -à-dire :

$$C_{ij} = T_{i-1,j} - T_{i,j+1} = 0 \quad \text{(d'aprés la formule (3.3))}.$$

Finalement la matrice C, telle que $C = Z_t T - T Z_s$ prend la forme suivante :

$$\begin{bmatrix} tt_{m-1} - t_{-1} & tt_{m-2} - t_{-2} & \cdots & \cdots & tt_0 - st_0 \\ 0 & \cdots & \cdots & 0 & t_{-n+1} - st_1 \\ \vdots & \ddots & & \vdots & \vdots \\ \vdots & & \ddots & \vdots & \vdots \\ 0 & \cdots & \cdots & 0 & t_{-1} - st_{m-1} \end{bmatrix}.$$

Ce qui peut s'écrire encore $e_1 u^T + v e_n^T$ exactement comme dans (3.1),
où $u = \left[e_m^T (tT)\right]^T$ et $v = (sT)\, e_1$. ■

L'opérateur de Toeplitz : $\Phi(T) = Z_t T - T Z_s$ est injectif pour peu que $t \neq s$, en conséquence, on peut définir la matrice Toeplitz directement à partir de l'équation de déplacement (3.1).

Pour les raisons évoquées à la fin de la section précédente, il convient de définir une matrice Toeplitz comme étant toute matrice T qui vérifie :

$$\Phi(T) = Z_t T - T Z_s = GH, \tag{3.4}$$

où G et H sont du types $(m \times 2)$ et $(2 \times n)$ respectivement.

Ceci étant, les matrices Toeplitz sont stables par quelques opérations usuelles. Par exemple l'inverse d'une matrice Toeplitz, carrée et régulière (inversible) est une matrice Toeplitz.

En effet, soit T une matrice Toeplitz, carrée et régulière qui vérifie la structure de déplacement (3.1)

$$Z_t T - T Z_s = e_1 u^T + v e_n^T = \begin{pmatrix} e_1 & v \end{pmatrix} \begin{pmatrix} u^T \\ e_n^T \end{pmatrix},$$

par la prémultiplication et la postmultiplication de cette dernière structure par T^{-1}, on obtient la nouvelle structure de déplacement suivante :

$$Z_s T^{-1} - T^{-1} Z_t = -T^{-1} \begin{pmatrix} e_1 & v \end{pmatrix} \begin{pmatrix} u^T \\ e_n^T \end{pmatrix} T^{-1}.$$

On observe que la matrice T^{-1} vérifie l'opérateur de Toeplitz (3.4), où

$$G = \left[-T^{-1} \begin{pmatrix} e_1 & v \end{pmatrix}\right] \text{ et } H = \left[\begin{pmatrix} u^T \\ e_n^T \end{pmatrix} T^{-1}\right].$$

Alors T^{-1} est une matrice Toeplitz. ■

3.2 matrices Toeplitz spéciales :

On a mentionné précédemment que l'inverse d'une matrice Toeplitz et le produit de deux matrices Toeplitz ne sont pas en général Toeplitz. Heureusement, lorsqu'on se restreint à des matrices Toeplitz spéciales, il s'avère que ces deux opérations deviennent stables. Dans cette section nous allons présenter deux types de matrices spéciales.

3.2.1 Matrices Toeplitz circulantes :

Une classe trés importante de matrices Toeplitz est la classe des matrices circulantes, laquelle on l'a déja citée dans le chapitre précédent. On peut définir les matrices circulantes à l'aide de la matrice de permutation Z_1 de la manière suivante :

$$C(v) = \left[v,\ Z_1 v,\ Z_1^2 v,\ \ldots,\ Z_1^{n-1} v\right], \tag{3.5}$$

où Z_1 est la matrice Z_t en remplaçant t par 1 et $v = \begin{bmatrix} v_0 & v_1 & \ldots & v_{n-1} \end{bmatrix}^T$. La formule (3.5) justifie le fait que la connaissance de la première colonne

d'une matrice circulante implique la connaissance des autres colonnes de cette matrice.

A partir de la relation (ii) du théorème (2.3), on conclut immédiatement que le produit d'une matrice circulante par un vecteur b; $C(a).b$ peut être réalisé par l'application de TDF trois fois et la multiplication des vecteurs. Alors pour calculer le produit de la forme $C(a).b$, on suit les étapes suivantes :

1. $b^* = F(w, 2n).b.$
2. $a^* = F(w, 2n).a.$
3. $y^* = b^* \odot a^*.$
4. $y = \frac{1}{2n} F(w^{-1}, 2n)y^*.$

Il est nécessaire de rappeler que pour réaliser l'opération $C(a).b$, on utilise $O(n \log n)$ opérations élémentaires.

3.2.2 Matrices Toeplitz triangulaires :

La deuxième classe importante des matrices Toeplitz est la classe des matrices triangulaires.

Exemples :

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

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

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

$$a^* = \left[\begin{array}{cccccccc} a_0 & a_1 & \dots & a_{n-1} & 0 & \dots & 0 \end{array} \right]^T$$

et

$$b^* = \left[\begin{array}{cccccccc} b_0 & b_1 & \dots & b_{n-1} & 0 & \dots & 0 \end{array} \right]^T,$$

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

$$\tilde{a} = \left[\begin{array}{cccc} a_{n-1} & a_{n-2} & \dots & a_0 \end{array} \right]^T.$$

On cherche maintenant de trouver le résultat du produit d'un vecteur ligne par une matrice Toeplitz triangulaire inférieure, $b^T.L(a) = c^T$ ce qui est équivalent au produit matriciel $L(\tilde{b}).L(a) = L(\tilde{c})$, cette formule nous permet de dire que notre vecteur c est les n premiers éléments du retournement vertical du vecteur associé au produit des deux représentations polynômiales $r(x)$ et $p(x)$ du vecteur $\tilde{b}$ et a respectivement, telle que $r(x)$ est donné par :

$$r(x) = b_{n-1} + b_{n-2}x + b_{n-3}x^2 + \cdots + b_0 x^{n-1},$$

c'est-à-dire que c dans ce cas peut s'écrire de la manière suivante :

$$\tilde{c} = z\left[1 : n\right], \text{ telle que}$$

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

où

$$\left(\tilde{b}\right)^* = \left[\begin{array}{cccccccc} b_{n-1} & b_{n-2} & \dots & b_0 & 0 & \dots & 0 \end{array} \right]^T.$$

avec le même principe, on répéte les mêmes opérations sur les matrices Toeplitz triangulaires supérieures.

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

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

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

$$\tilde{c} = z\left[1 : n\right], \text{où}$$

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

La dernière opération est la prémultiplication de $R(a)$ par un vecteur ligne b^T, $b^T.R(a)$ qui donne un vecteur ligne c^T, tel que c est composé par les n premiers coéfficients du polynôme $q(x).p(x)$ est équivalent au vecteur suivant :

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

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

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

Algorithme 3.1. **mult**(a,b).

données : $a_0, a_1, \ldots, a_{n-1}$ et $b_0, b_1, \ldots, b_{n-1}$.
Résultats : $d_0, d_1, \ldots, d_{n-1}$.
1. $a^* = F(w, 2n)a$.
2. $b^* = F(w, 2n)b$.
3. $c^* = a^* \odot b^*$.
4. $c = \frac{1}{2n}F(w^{-1}, 2n).c^*$.
5. $d = c[1:n]$.

données		Résultats
a	b	d
$\begin{bmatrix} 1 & 2 & 3 \end{bmatrix}^T$	$\begin{bmatrix} -1 & 1 & 1 \end{bmatrix}^T$	$\begin{bmatrix} -1 & -1 & 0 \end{bmatrix}^T$
$\begin{bmatrix} 1 & -2 & 4 & 5 \end{bmatrix}^T$	$\begin{bmatrix} 2 & 1 & -3 & 0 \end{bmatrix}^T$	$\begin{bmatrix} 2 & -3 & 3 & 20 \end{bmatrix}^T$
$\begin{bmatrix} 1 & 1 & 2 & -2 & 3 \end{bmatrix}^T$	$\begin{bmatrix} 2 & 2 & -1 & 1 & 1 \end{bmatrix}^T$	$\begin{bmatrix} 2 & 4 & 5 & 0 & 2 \end{bmatrix}^T$

Tableau 1. d'exécution du programme **mult.**

Enfin, on se propose aussi un algorithme pour calculer l'inverse d'une matrice Toeplitz triangulaire inférieure qui est basé sur une idée très simple.

Soit $L(a)$ une matrice Toeplitz triangulaire inférieure carrée de dimension $(n \times n)$ générée par sa première colonne $a = \begin{bmatrix} a_1 & a_2 & \ldots & a_n \end{bmatrix}^T$, telle

que n ici est une puissance de 2, c'est-à-dire : $n = 2^q$. On commence maintenant notre problème d'inverser la matrice $L(a)$ par la représentation de cette dernière par blocs de la façon suivante :

$$L(a) = \begin{bmatrix} L(b) & 0 \\ T & L(b) \end{bmatrix}, \tag{3.6}$$

telle que $b = \begin{bmatrix} a_1 & a_2 & \dots & a_{\frac{n}{2}} \end{bmatrix}^T$, où $\frac{n}{2} = 2^{q-1}$ aussi est une puissance de 2 et $T = L(c,d)$ est une matrice Toeplitz carrée de dimension $\left(\frac{n}{2} \times \frac{n}{2}\right)$ définie par sa première colonne et sa première ligne c et d , qui sont les suivantes :

$$c = \begin{bmatrix} a_{2^{q-1}+1} & a_{2^{q-1}+2} & \dots & a_{2^q} \end{bmatrix}$$

et

$$d = \begin{bmatrix} a_{2^{q-1}+1} & a_{2^{q-1}} & \dots & a_2 \end{bmatrix}.$$

Il est clair que l'inverse de la matrice $L(a)$ représentée sous la forme (3.6) est donné immédiatement par :

$$L(a)^{-1} = \begin{bmatrix} L(b)^{-1} & 0 \\ -L(b)^{-1}TL(b)^{-1} & L(b)^{-1} \end{bmatrix}. \tag{3.7}$$

Alors, on remarque bien que l'inverse $L(a)^{-1}$ peut être obtenu immédiatement par une répétition finie de la formule (3.7) comme l'indique l'algorithme suivant :

Algorithme 3.2 l'algorithme **PVMIT** calcul l'inverse d'une matrice Toeplitz triangulaire inférieure a.

Données : l'entier q et la matrice a.
Résultat : le vecteur v qui caractérise la matrice inverse, $a^{-1} = L(v)$.

1. $f = a(:,1)$
2. $W = 1/a_{11}$.

pour $i = 0 : q-1$

3. $b = \begin{bmatrix} f_1 & f_2 & \dots & f_{2^i} \end{bmatrix}$.
4. $c = \begin{bmatrix} f_{2^i+1} & f_{2^i+2} & \dots & f_{2^{i+1}} \end{bmatrix}$.

5. $d = \left[\begin{array}{cccc} f_{2^i+1} & f_{2^i} & \ldots & f_2 \end{array} \right]$.
6. $T = L(c, d)$.
7. $D = -W * T * W$.
8. $t = \left[\begin{array}{ccccccc} W_{1,1} & \ldots & W_{2^i,1} & D_{1,1} & \ldots & D_{2^i,1} \end{array} \right]$.
9. $W = L(t)$.
10. $v = W(:, 1)$.

Données		Résultats
q	a	v
2	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 \\ -1 & 2 & 1 & 0 \\ 3 & -1 & 2 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & -2 & 5 & -15 \end{bmatrix}^T$
3	$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T$

Tableau 2. d'exécution du programme **PVMIT**.

3.3 Produit d'une matrice Toeplitz par un vecteur :

Dans cette section on va essayer de présenter quelques méthodes qui nous permettent d'effectuer le produit matrice Toeplitz par vecteur. La première méthode que nous allons présenter est la méthode de prolongement d'une matrice Toeplitz en une matrice circulante et la deuxième est la méthode de décomposition d'une matrice Toeplitz en deux matrices Toeplitz triangulaires.

On sait qu'il existe une relation très importante entre les matrices circulantes, les matrices Toeplitz et le TDF et on consiste d'évaluer le produit matrice par vecteur de la forme $T.b$, où T est une matrice Toeplitz.

Premièrement, il est facile de résoudre notre problème si on se base sur l'idée clée suivante : toute matrice Toeplitz peut être prolongée en une matrice circulante. Par exemple, soit la matrice de Toeplitz

$$T = \begin{bmatrix} 1 & 4 & 5 \\ 2 & 1 & 4 \\ 3 & 2 & 1 \end{bmatrix},$$

une sous matrice du type 3×3 d'une matrice circulante C

$$C = \begin{bmatrix} 1 & 4 & 5 & 3 & 2 \\ 2 & 1 & 4 & 5 & 3 \\ 3 & 2 & 1 & 4 & 5 \\ 5 & 3 & 2 & 1 & 4 \\ 4 & 5 & 3 & 2 & 1 \end{bmatrix}.$$

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

$$C(:,1) = \begin{bmatrix} T(1:n,1) \\ \\ T(1,n:-1:2) \end{bmatrix}. \tag{3.8}$$

On ajoute, qu'une fois le problème produit matrice circulante par vecteur de la forme $Cb = y$, tel que : $b(n+1:2n-1) = 0$ est résolu comme on a vu précédement, notre problème de la forme $y(1:n) = Tb(1:n)$ l'est aussi.

On peut représenter toute matrice Toeplitz comme somme de deux matrices Toeplitz triangulaires, la première inférieure et la deuxième supérieure. Par exemple ; Soit la matrice Toeplitz T :

$$T = \begin{bmatrix} 1 & 2 & 5 & 7 \\ 4 & 1 & 2 & 5 \\ 8 & 4 & 1 & 2 \\ 9 & 8 & 4 & 1 \end{bmatrix},$$

qu'elle peut se représenter encore sous la forme :

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 4 & 1 & 0 & 0 \\ 8 & 4 & 1 & 0 \\ 9 & 8 & 4 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 2 & 5 & 7 \\ 0 & 0 & 2 & 5 \\ 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

D'une façon générale, si $T = (t_{ij})$ est une matrice Toeplitz du type $(n \times n)$, alors :

$$T = L(t_1) + R(t_2), \tag{3.9}$$

telle que : les vecteur t_1 et t_2 sont donnés par les relations :

$$t_1 = T(:, 1) \tag{3.10}$$

$$t_2 = (0, T(1, 2 : n)). \tag{3.11}$$

Si on se base sur cette idée, le produit d'une matrice Toeplitz par vecteur b ; Tb devient un problème de multiplication d'une matrice Toeplitz triangulaire inférieure et triangulaire supérieure par le vecteur b, $L(t_1).b$ et $R(t_2).b$ respectivement lequel on a déja énoncé dans la section précédente.

On peut dire exactement la même chose dans le cas de la prémultiplication d'un vecteur ligne b^T par une matrice Toeplitz. C'est-à-dire : le calcul de la quantité $b^T \cdot T$ équivalent au calcul des deux quantités $b^T \cdot L(t_1)$; la prémultiplication d'un vecteur ligne par une matrice Toeplitz triangulaire inférieure et $b^T \cdot R(t_2)$; la prémultiplication d'un vecteur ligne par une matrice Toeplitz triangulaire supérieure $R(t_2)$, qui sont faciles à calculer.

3.4 Transformation en une matrice Toeplitz :

Dans cette section on s'interessé de transformer toute matrice structurée en une matrice Toeplitz.

Théorème 3.1 : Soit A une matrice structurée caractérisée par la structure de déplacement suivante :

$$Z_1 A - AZ = u \cdot v^T + w \cdot g^T \tag{3.12}$$

où Z_1 et Z sont les matrices de déplacement de la même forme que Z_t en remplaçant t par 1 et 0 respectivement. Alors la matrice :

$$C(u)^{-1} A L(\breve{g})^{-1}$$

est une matrice Toeplitz, où $C(u)$ est une matrice circulante définie par sa première colonne u et $L(\tilde{g})$ est une matrice Toeplitz triangulaire inférieure caractérisée par le vecteur $\tilde{g} = [g_n, g_{n-1}, \ldots, g_1)]^T$. □

Démonstration : Soit A une matrice structurée ayant la structure de déplacement (3.12). Grâce à la définition des matrices circulantes et les matrices Toeplitz triangulaires inférieueres, on peut écrire :

$$u = C(u) \cdot e_1 \quad \text{et} \quad g^T = e_n^T L(\tilde{g}).$$

Par substitution de u et g^T dans l'équation (3.12) on trouve :

$$Z_1 A - AZ = C(u) \cdot e_1 \cdot v^T + w \cdot e_n^T L(\widetilde{g}).$$

En prémultipliant cette équation par $C(u)^{-1}$ et postmultipliant par $L(\check{g})^{-1}$, on obtient :

$$C(u)^{-1} Z_1 A L(\widetilde{g})^{-1} - C(u)^{-1} AZ L(\widetilde{g})^{-1} = e_1 \cdot v^T + w \cdot e_n^T.$$

On rappelle que l'inverse d'une matrice circulante (resp Toeplitz triangulaire inférieure) est une matrice circulante (resp Toeplitz triangulaire inférieure) et le produit matriciel dans la classe des matrices circulantes (resp Toeplitz triangulaires inférieures) est commutatif. A partir de ces propriétés, on peut écrire aussi :

$$Z_1 C(u)^{-1} A L(\widetilde{g})^{-1} - C(u)^{-1} A L(\widetilde{g})^{-1} Z = e_1 \cdot v^T + w \cdot e_n^T.$$

On remarque que la matrice $C(u)^{-1} A L(\tilde{g})^{-1}$ vérifie l'opérateur de Toeplitz, alors c'est une matrice Toeplitz. ■

Chapitre 4

Structures de déplacement des matrices de Vandermonde.

L'une des structures les plus utiles dans la résolution des systèmes polynômiaux est la structure de Vandermonde. Alors, on étudie dans ce chapitre la structure de déplacement caractérisant les matrices de Vandermonde par blocs dans le but d'aboutir la structure de déplacement caractérisant les matrices de Vandermonde confluentes grâce au plongement de cette dernière dans les matrices de Vandermonde par blocs.

4.1 matrices de Vandermonde :

On commence cette section par une définition générale d'une matrice de Vandermonde. Considérons des bases P= $(P_0(x), ..., P_{m-1}(x))$ de l'espace $\mathbb{C}_m[X]$ des polynômes de degré $\leq m-1$, telle que : deg $P_i(x) = i$.

On appelle une matrice P-Vandermonde V_p toute matrice du type $(m \times n)$ de la forme :

$$V_p = [f(x_1)....f(x_n)], \tag{4.1}$$

où $f(x) = (P_0(x)....P_{m-1}(x))^T$ et $x_1, ..., x_n$ sont n nombres deux à deux distincts.

Dans notre étude on s'intéresse au type des matrices P-Vandermonde lorsqu'on prend à la place de p, la base canonique :

$$h = (1,\ x,\ x^2, ...,\ x^{m-1}). \tag{4.2}$$

La matrice V_h est appellée la matrice de Vandermonde et s'écrit sous la forme :

$$V_h = \begin{bmatrix} 1 & 1 & \cdots & \cdots & 1 \\ x_1 & x_2 & \cdots & \cdots & x_n \\ x_1^2 & x_2^2 & \cdots & \cdots & x_n^2 \\ \vdots & \vdots & & & \vdots \\ x_1^{m-1} & x_2^{m-1} & \cdots & \cdots & x_n^{m-1} \end{bmatrix}. \tag{4.3}$$

On observe que les éléments de V_h sont donnés par :

$$(V_h)_{k,j} = x_j^k \quad \forall\, 0 \leq k \leq m-1 \text{ et } 1 \leq j \leq n,$$

qu'on peut carctériser aussi par la relation :

$$(V_h)_{k+1,j} - x_j(V_h)_{k,j} = 0 \quad \forall k < m-1. \tag{4.4}$$

Alors, on déduit que la matrice de Vandermonde V_h ayant la structure de déplacement :

$$Z_s^T V_h - V_h D_x = e_m.u^T, \tag{4.5}$$

telle que : $D_x = diag(x_1, ..., x_n)$ et $u^T = (s - x_1^m, ..., s - x_n^m)$.

Il est clair que la structure (4.5) est déduite directement à partir de la propiété vérifiée par la matrice de permutation Z_s^T et la formule qui caractérise les éléments d'une matrice de Vandermonde, de la manière suivante :

On a :

$$Z_s^T V_h = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ x_1^2 & x_2^2 & \cdots & x_n^2 \\ \vdots & \vdots & & \vdots \\ x_1^{m-1} & x_2^{m-1} & \cdots & x_n^{m-1} \\ s & s & \cdots & s \end{bmatrix}$$

et

$$V_h D_x = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ x_1^2 & x_2^2 & \cdots & x_n^2 \\ \vdots & \vdots & & \vdots \\ x_1^{m-1} & x_2^{m-1} & \cdots & x_n^{m-1} \\ x_1^m & x_2^m & \cdots & x_n^m \end{bmatrix},$$

par conséquent

$$Z_s^T V_h - V_h D_x = \begin{bmatrix} 0 & 0 & \cdots & \cdots & 0 \\ \vdots & \vdots & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & \cdots & 0 \\ s - x_1^m & s - x_2^m & \cdots & \cdots & s - x_n^m \end{bmatrix},$$

ce qui peut s'écrire encore $e_m.u^T$, où $u^T = (s - x_1^m, ..., s - x_n^m)$ comme exactement dans (4.5). ■

Dans le cas où les x_j sont tous non nuls, la structure de déplacement de V_h , on peut l'écrire d'une autre maniére pour éviter les calculs des puissances x_k^m des nombres représentés en virgule flottante, on voit que si $k \geq 1$:

$$(V_h)_{k-1,j} - x_j^{-1}(V_h)_{k,j} = 0 \tag{4.6}$$

de sorte que la structure (4.5) sera écrite de la forme suivante :

$$Z_s V_h - V_h D_x^{-1} = e_1.v^T \tag{4.7}$$

telle que : $D_x^{-1} = diag(x_1^{-1}, ..., x_n^{-1})$ et $v^T = (sx_1^{m-1} - x_1^{-1}, ..., sx_1^{m-1} - x_n^{-1})$.

Dans la nouvelle structure, il est clair que les puissances x_j^{m-1} seront éliminées si $s = 0$.

C'est exactement de la même manière de la structure (4.5) qu'on peut trouver la nouvelle structure de déplacement (4.7) d'une matrice de Vandermonde.

Il est clair que :

$$Z_s V_h = \begin{bmatrix} sx_1^{m-1} & sx_2^{m-1} & \cdots & \cdots & sx_n^{m-1} \\ 1 & 1 & \cdots & \cdots & 1 \\ x_1 & x_2 & & & x_n \\ \vdots & \vdots & & & \vdots \\ x_1^{m-2} & x_2^{m-2} & \cdots & \cdots & x_n^{m-2} \end{bmatrix}$$

et

$$V_h D_x^{-1} = \begin{bmatrix} x_1^{-1} & x_2^{-1} & \cdots & \cdots & x_n^{-1} \\ 1 & 1 & \cdots & \cdots & 1 \\ x_1 & x_2 & & & x_n \\ \vdots & \vdots & & & \vdots \\ x_1^{m-2} & x_2^{m-2} & \cdots & \cdots & x_n^{m-2} \end{bmatrix},$$

alors, on conclut que :

$$Z_s V_h - V_h D_x^{-1} = \begin{bmatrix} sx_1^{m-1} - x_1^{-1} & sx_2^{m-1} - x_2^{-1} & \cdots & \cdots & sx_n^{m-1} - x_n^{-1} \\ 0 & 0 & \cdots & \cdots & 0 \\ \vdots & \vdots & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & \cdots & 0 \end{bmatrix}$$

ce qui peut représenter aussi sous la formule $e_1.v^T$, ce qui à démontrer. ■

Ici, on peut dire aussi que la matrice de Vandermonde V_h est définie à partir de ses structures de déplacement (4.5) et (4.7), qui sont restées valables même si : $Z_s = Z_0 = Z$, en remplaçant s par 0.

4.2 matrices de Vandermonde par blocs :

Une matrice de Vandermonde par blocs est exactement la même chose qu'une matrice de Vandermonde ; mais la différence c'est au lieu des nombres x_k, on considére n matrices régulières $B_1, B_2, ..., B_n$ du type $(d \times d)$ deux à deux distinctes. Alors la matrice de Vandermonde par blocs est une matrice du type $(md \times nd)$, telle que :

$$V_{b,h} = \begin{bmatrix} I_d & I_d & \cdots & \cdots & I_d \\ B_1 & B_2 & \cdots & \cdots & B_n \\ B_1^2 & B_2^2 & \cdots & \cdots & B_n^2 \\ \vdots & \vdots & & & \vdots \\ B_1^{m-1} & B_2^{m-1} & \cdots & \cdots & B_n^{m-1} \end{bmatrix}, \tag{4.8}$$

où le symbole b dans $V_{b,h}$ désigne l'écriture par blocs et h désigne la base canonique.

Les matrices de Vandermonde par blocs sont caractérisées par les structures de déplacement suivantes :

$$(Z^d)^T V_{b,h} - V_{b,h} D_B = -E_m.U^T \tag{4.9}$$

$$Z^d V_{b,h} - V_{b,h} D_B^{-1} = -E_1.U'^T \tag{4.10}$$

avec $E_m = \begin{bmatrix} 0 & \cdots & 0 & I_d \end{bmatrix}^T$, $E_1 = \begin{bmatrix} I_d & 0 & \cdots & 0 \end{bmatrix}^T$,

$$U^T = \left[\begin{array}{cccc} B_1^m & B_2^m & \cdots & B_n^m \end{array}\right]^T, \acute{U}^T = \left[\begin{array}{cccc} B_1^{-1} & B_2^{-1} & \cdots & B_n^{-1} \end{array}\right],$$
$$D_B = diag\,(B_1, \ldots, B_n) \text{ et } D_B^{-1} = diag(B_1^{-1}, \ldots, B_n^{-1}).$$

Les matrices de déplacement Z^d et $(Z^d)^T$ effectuent une permutation sur d lignes semblable à la permutation effectuée par Z et $(Z)^T$ sur une seule ligne. Alors, on construit les structures (4.9) et (4.10) directement à partir des structures (4.5) et (4.7) respectivement, en remplaçant les éléments x_k dans ces deux dernières équations par les matrices B_k et le nombre s par 0.

4.3 Matrices de Vandermonde confluentes :

Dans cette section aussi on considère des bases $P = (P_0(x), ..., P_{m-1}(x))$ de l'espace $\mathbb{C}_m[x]$ (des polynômes de degré $\leq m-1$, telle que $\deg P_i(x) = i$). Parmi lesquelles en particulier la base canonique pour définir la matrice de Vandermonde confluente. D'une façon générale on dit qu'une matrice du type $(m \times dn)$ est de Vandermonde P-confluente et on la note W_p, si elle est de la forme :

$$W_p = [f(x_1)f^{(1)}(x_1) \cdots f^{(d-1)}(x_1) \cdots f(x_n)f^{(1)}(x_n) \cdots f^{(d-1)}(x_n)] \quad (4.11)$$

avec $f(x) = (p_0(x), \ldots, p_{m-1}(x))^T$ et $f^{(1)}(x_n) \ldots f^{(d-1)}(x_n)$ sont les dérivées successives de $f(x)$.

Remarque : En principe, le nombre des dérivées d devrait dépendre de x_k, c'est-à-dire $d = d_k$; mais en guise de simplicité, on suppose que c'est la même pour tout $k = 1, \ldots, n$.

Dans cette thèse, on considère seulement la matrice de Vandermonde confluente W_h, obtenue en remplaçant p par h.

Exemple : La matrice suivante :

$$W_h = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ x_1 & 1 & 0 & x_2 & 1 & 0 \\ x_1^2 & 2x_1 & 2 & x_2^2 & 2x_2 & 2 \\ x_1^3 & 3x_1^2 & 6x_1 & x_2^3 & 3x_2^2 & 6x_2 \\ x_1^4 & 4x_1^3 & 12x_1^2 & x_2^4 & 4x_2^3 & 12x_2^2 \\ x_1^5 & 5x_1^4 & 20x_1^3 & x_2^5 & 5x_2^4 & 20x_2^3 \end{bmatrix}$$

c'est une matrice de Vandermonde confluente du type $6 \times 6 = 6 \times 3\,(2)$, c'est-à-dire $m = 6$ et $n = 2$.

Dans la prochaine section, on donnera la structure de déplacement des matrices de Vandermonde confluentes.

4.4 Structure de déplacement pour les matrices de Vandermonde confluentes :

Maintenant, on annonce le résultat principal dans ce chapitre, qui donne immédiatement la structure de déplacement pour les matrices de Vandermonde confluentes.

Théorème 4.1 : La matrice W_h satisfait la structure de déplacemet suivante :

$$Z^T W_h - W_h D_B = -e_m y^T \tag{4.12}$$

telle que :

$$D_B = diag(B_1, ..., B_n)$$

et

$$y^T = (x_1^m (m)_1 x_1^{m-1} .. (m)_d x_1^{m-d} ..... x_n^m (m)_1 x_n^{m-1} .. (m)_d x_n^{m-d}). \quad \square$$

La démonstration de ce résultat n'est pas triviale ; alors on aura besoin d'annoncer quelques résultats préliminaires et nécessaires, qui joueraient un rôle intermédiaire dans cette démonstration.

Premièrement, on introduit le cas particulier suivant :

$$B(x) = \begin{bmatrix} x & 1 & 0 & \cdots & 0 \\ 0 & x & 2 & & \vdots \\ \vdots & 0 & \ddots & \ddots & 0 \\ \vdots & & \ddots & \ddots & d-1 \\ 0 & \cdots & \cdots & 0 & x \end{bmatrix}, \tag{4.13}$$

avec $B_k = B(x_k)$ sont les éléments de base de la matrice de Vandermonde par blocs. Ensuite, il est nécessaire de définir le comportement des puissances de ces dernières $B(x)^k$.

On note par $\alpha_{i,j}^k$, où $0 \leq i \leq j \leq d-1$ l'élément en position (i,j) de $B(x)^k$.

Pour $k \leq d-2$, $B(x)^k$ est bande, de largeure-bande $k+1$, c'est-à-dire : les éléments $\alpha_{i,j}^k$ sont définis par :

$$\begin{aligned} \alpha_{i,j}^k &= 0 \qquad \text{si } j-i \geq k+1 \\ \alpha_{i,j}^k &= \frac{j!}{i!}\binom{k}{j-i}x^{k-j+1} \\ &= \frac{j!k!}{i!(j-i)!(k-j+1)!}x^{k-j+1} \qquad 0 \leq i \leq j \leq i+k \end{aligned} \tag{4.14}$$

où, $\binom{k}{j} = \frac{k!}{j!(k-j)!}$, $(k)_j = k(k-1)\ldots(k-j+1) \qquad k \geq j$ et par convention $0! = 1$.

Démonstration : On essayera de démontrer la relation (4.14) par récurrence. On commence par le cas initial :

• Pour $k = 1$, on a remarqué que les éléments de $B(x)$ sont tous nuls sauf les éléments $\alpha_{i,j}$ telle que : $(i = j)$ et $(j = i+1)$, alors :

$$\begin{aligned} \alpha_{i,j} &= \frac{j!k!}{i!(j-i)!(k-j+1)!}x^{k-j+1} \qquad \text{pour } i = j \\ &= \frac{j!}{i!(j-i)!(k-j+1)!}x^{k-j+1} \\ &= x, \end{aligned}$$

les éléments dans le cas, où $j = i+1$ sont donnés par :

$$\begin{aligned} \alpha_{i,j} &= \frac{(i+1)!1!}{i!1!0!}x^0 \\ &= i+1 \qquad \text{pour } j = i+1, \end{aligned}$$

alors la relation (4.14) est vraie pour $k = 1$.

• Maintenant, on suppose que cette formule est vraie jusqu'à $k-1$ et on démontre qu'elle est vraie pour k.

$$\begin{aligned} (\alpha_{i,j}^k)_{0\leq i\leq j\leq i+k} &= B(x)^k \\ &= B(x)B(x)^{k-1} \\ &= \begin{bmatrix} x & 1 & 0 & . & . & 0 \\ 0 & x & 2 & . & . & 0 \\ . & 0 & x & 3 & . & . \\ . & . & 0 & . & . & . \\ 0 & . & . & . & . & d-1 \\ 0 & 0 & . & . & 0 & x \end{bmatrix} \cdot (\alpha_{i,j}^{k-1})_{0\leq i\leq j\leq i+k-1}, \end{aligned}$$

c'est-à-dire on a :

$$\begin{aligned}
\alpha_{i,j}^{k} &= x \cdot \alpha_{i,j}^{k-1} + (i+1)\alpha_{i+1,j}^{k} \\
&= x\frac{j!}{i!}\binom{k-1}{j-i}x^{k-1-j+1} + (i+1)\frac{j!}{(i+1)!}\binom{k-1}{j-i-1}x^{k-1-j+i+1} \\
&= \frac{j!}{i!}\left(\binom{k-1}{j-i} + \binom{k-1}{j-i+1}\right)x^{k-j+i} \\
&= \frac{j!}{i!}\binom{k}{j-i}x^{k-j+1} \qquad \forall\, 0 \leq i \leq j \leq i+k. \quad \blacksquare
\end{aligned}$$

Le résultat suivant joue un rôle important dans la démonstration du théorème (4.1) parcequ'il forme l'idée clée dans cette dernière.

Lemme 4.2 : La première ligne de $B(x)^k$est formée des éléments suivants : $x^k, kx^{k-1}, k(k-1)x^{k-2}\ldots$ c'est-à-dire x_k et ses dérivées successives. □

Démonstration : La première ligne est obtenue à partir de la relation (4.14), on remplace i par 0 de la manière suivante :

$$\begin{aligned}
\alpha_{0,j}^{k} &= \frac{j!}{0!}\binom{k}{j-0}x^{k-j+0} \\
&= \frac{j!k!}{0!(j-0)!(k-j)!}x^{k-j} \\
&= \frac{j!k!}{0!(j-0)!(k-j)!}x^{k-j} \\
&= (k)_j x^{k-j} \\
&= k(k-1)(k-2)\ldots(k-j+1)x^{k-j}. \quad \blacksquare
\end{aligned}$$

Comme on a vu dans (4.8) la matrice de Vandermonde par blocs est constituée par les puissances $B(x)^k$, c'est pour cela le lemme (4.2) nous permet de déduire que la $(kd+1)^{iéme}$ ligne de $V_{b,h}$ est identique à la $(k+1)^{iéme}$ ligne de la matrice de Vandermonde confluente W_h. Cette remarque on peut l'interpréter mathématiqument sous forme du résultat suivante.

Théorème 4.3 : Soit $P \in \mathbb{R}^{md\times md}$ une matrice de permutation telle que sa colonne $kd+1$ soit égale à e_{k+1} ; c'est -à-dire que :

$$P \cdot e_{kd+1} = e_{k+1}.$$

Alors

$$P \cdot V_{b,h} = \begin{bmatrix} W_h \\ X \end{bmatrix}, \tag{4.15}$$

où X est une matrice dont la détermination n'est pas en question dans ce cadre. □

Démonstration : Puisque on a : la $(kd+1)^{iéme}$ ligne de $V_{b,h}$ est identique à la $(k+1)^{iéme}$ ligne de W_h et d'aprés la relation vérifiée par la matrice de permutation P, on remarque que la $(k+1)^{iéme}$ ligne de $P \cdot V_{b,h}$ est identique à la $(kd+1)^{iéme}$ ligne de $V_{b,h}$, ce qui nous permet de déduire immédiatement le résultat. ■

Il est possible maintenant de démontrer notre théorème principal dans ce chapitre.

Démonstration du théorème 4.1 : L'idée de notre démonsrtation est la déduction de la structure de déplacement de W_h à partir de la structure de déplacement de $V_{b,h}$ et en s'aidant des résultats annoncés précedemment. Tout d'abord, on considère la structure de déplacement d'une matrice de Vandermonde par blocs (4.9), puis on la multiplie par la matrice de permutation P, on obtient :

$$P(Z^d)^T \cdot V_{b,h} - P \cdot V_{b,h} D_B = -PE_m U^T$$

si en tenant compte les calculs suivants :

$$P(Z^d)^T \cdot V_{b,h} = P \cdot \begin{bmatrix} B_1 & B_2 & \cdots & B_n \\ B_1^2 & B_2^2 & \cdots & B_n^2 \\ \vdots & \vdots & & \vdots \\ B_1^{m-1} & B_2^{m-1} & \cdots & B_n^{m-1} \end{bmatrix} = \begin{bmatrix} Z^T \cdot w_h \\ * \end{bmatrix},$$

$$P \cdot V_{b,h} = \begin{bmatrix} w_h \\ x \end{bmatrix}$$

et

$$PE_m = \begin{bmatrix} \begin{bmatrix} 0 & \cdots & \cdots & 0 \\ \vdots & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & \cdots & 0 \end{bmatrix} \\ \vdots \\ \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & \ddots & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \cdots & 0 & 1 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} 0 & 0 & \ldots & 0 \\ 0 & . & \ldots & 0 \\ . & . & \ldots & . \\ 1 & 0 & \ldots & 0 \end{bmatrix} \\ * \end{bmatrix} = \begin{bmatrix} e_m \cdot e_1^T \\ * \end{bmatrix}$$

on trouve la structure de déplacement suivante :

$$Z^T W_h - W_h D_B = -e_m \cdot e_1^T \cdot U^T,$$

où $U^T = [B_1^m, ..., B_n^m]$, alors $e_1^T \cdot U^T$ est la $1^{ére}$ ligne de U^T, c'est -à-dire les $1^{ére}$ lignes des matrices $B(x_1)^m, \ldots, B(x_n)^m$, qui sont formées d'aprés le lemme (4.2) par les éléments $\left(x_j^m\right)_{j=\overline{1,n}}$ et ses dérivées successives jusqu'à la dérivée d'ordre $(d-1)$, cela permet de conclure que :

$$\begin{aligned} e_1^T \cdot U^T &= y^T \\ &= (x_1^m (m)_1 x_1^{m-1} .. (m)_d x_1^{m-d} .... x_n^m (m)_1 x_n^{m-1} .. (m)_d x_n^{m-d}), \end{aligned}$$

où, $(m)_j = \frac{m!}{j!}$. Enfin on peut dire que l'équation suivante :

$$Z^T W_h - W_h D_B = -e_m \cdot y^T,$$

caractérisant les matrices de Vandermonde confluentes. ■

Chapitre 5

Elimination de Gauss pour les matrices de Vandermonde confluentes.

Ce chapitre a pour but de présenter nos activités autour de la résolution d'un système linéaire structuré de Vandermonde confluent par l'application de la méthode d'élimination de Gauss rapide, qu'on remarquera que c'est une répétition finie du complément de Schur.

Au début, on commence par un petit rappel sur ce dernier.

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

$$A = \begin{bmatrix} M & F \\ E & D \end{bmatrix}, \tag{5.1}$$

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

$$S(A) = D - EM^{-1}F. \tag{5.2}$$

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

$$J_1 A = \begin{bmatrix} M & F \\ 0 & S(A) \end{bmatrix}, \tag{5.3}$$

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

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

$$S^0(A) = A \qquad S^{k+1}(A) = S\left(S^k(A)\right) \quad k = \overline{0, n-2}. \tag{5.4}$$

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

5.1 Stabilité de la structure de déplacement par le complément de Schur :

Soit la matrice structurée A définie dans (5.1) ayant la structure de déplacement suivante :

$$FA - AU = \begin{bmatrix} u & w \end{bmatrix} \begin{bmatrix} v & z \end{bmatrix}^H, \tag{5.5}$$

où $F \in \mathbb{C}^{(n+s)\times(n+s)}$ est bidiagonale inférieure, $B \in \mathbb{C}^{(n+s)\times(n+s)}$ est bidiagonale supérieure et u, v, w, $z \in \mathbb{C}^{(n+s)}$, en concordance avec (5.1), il est nécessaire de représenter les matrices F, B et $\begin{bmatrix} u & w \end{bmatrix} \begin{bmatrix} v & z \end{bmatrix}^H$ dans (5.5) par blocs comme suit :

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

et

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

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

Théorème 5.1 : Supposons que A vérifie l'équation (5.5). Alors :

$$F_2 S(A) - S(A) B_2 = \begin{bmatrix} u' & w' \end{bmatrix} \begin{bmatrix} v' & z' \end{bmatrix}^H \tag{5.6}$$

avec

$$\begin{bmatrix} 0 & 0 \\ u' & w' \end{bmatrix} = \begin{bmatrix} u & w \end{bmatrix} - \begin{bmatrix} I_s \\ EM^{-1} \end{bmatrix} \begin{bmatrix} u_1 & w_1 \end{bmatrix} \tag{5.7}$$

et

$$\begin{bmatrix} 0 & 0 \\ v' & z' \end{bmatrix} = \begin{bmatrix} v & z \end{bmatrix}^H - \begin{bmatrix} v_1 & z_1 \end{bmatrix}^H \begin{bmatrix} I_s & M^{-1}F \end{bmatrix}. \quad \square \tag{5.8}$$

Démonstration (voir [15]) : Soit A la matrice structurée caractérisée par sa structure de déplacement (5.5). Si on prémultiplie et postmultiplie cette structure par J_1 et K_1 respectivement, où J_1 est la première matrice d'élimination de Gauss et K_1 est la suivante :

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

on obtient l'équation :

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

En tenant compte que :

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

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

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

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

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

et

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

où

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

Ce qui nous permet d'écrire :

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

et

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

Par des calculs simples, on trouve :

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

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

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

Avant d'attaquer notre but principal d'appliquer la méthode d'élimination de Gauss basée sur le complément de Schur sur une matrice de Vandermonde confluente, on suit quelques étapes importantes. Premièrement, on cherche à trouver une nouvelle structure de déplacement d'une matrice de Vandermonde confluente plus éfficace que la structure (4.11), dans ce qui suit, on note par W la matrice de Vandermonde confluente.

5.2 Structure de déplacement de W :

On considère la matrice W définie dans (4.11), en remplaçant m par nd et la base P par la base canonique. C'est-à-dire W est du type $(nd \times nd)$ définie comme suit :

$$W = \left[f(x_1)f^{(1)}(x_1)\cdots f^{(d-1)}(x_1)\cdots f(x_n)f^{(1)}(x_n)\cdots f^{(d-1)}(x_n)\right]$$

telle que :

$$f(x) = \begin{bmatrix} 1 & x & x^2 & \cdots & x^{nd-1} \end{bmatrix},$$

et $x_1, \ldots, x_n$ sont n nombres deux à deux distincts. W est caractérisée par la structure de déplacement suivante :

$$Z^T W - W D_B = -e_{nd} y^T, \tag{5.9}$$

où

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

est une matrice diagonale par blocs, telle que les matrices $B(x_k)$; $k = \overline{1,n}$ dans (4.13), qu'on peut les représenter encore sous la forme suivante :

$$B(x) = xI_d + A;\ A = \left[\begin{array}{ccccc} 0 & e_1 & 2e_2 & 3e_3 & \cdots & (d-1)\,e_{d-1} \end{array}\right], \tag{5.10}$$

et

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

Si on prémultiplie et postmultiplie la structure (5.9) par Z et D_B^{-1} respectivement et en tenant compte que :

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

et

$$Ze_{nd} = 0,$$

on trouve immédiatement que cette structure devienne :

$$ZW - WD_B^{-1} = e_1 v^T, \tag{5.11}$$

où :

$$v^T = -\left(\begin{array}{ccc} r(x_1) & \cdots & r(x_n) \end{array}\right); \tag{5.12}$$

$$r(x) = \left(x^{-1}, -x^{-2}, 2x^{-3}, \ldots, (-1)^{d-1}\,(d-1)!x^{-d}\right),$$

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

Dans l'équation de déplacement (5.11), on observe que Z est triangulaire inférieure et D_B^{-1} est triangulaire supérieure, alors le théorème (5.1) indique que le complément de Schur de W prend le même type de structure. C'est pour cette raison, on préfère la structure (5.11).

Définition (5.1) : Soit A une matrice du type $(rd \times rd)$; $r = \overline{1,n}$. On dit que A est matrice r structurée Vandermonde confluente, s'il existe deux vecteurs g et h de dimension rd telle que :

$$ZA - AD_{r,B}^{-1} = gh^T, \tag{5.13}$$

où :

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

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

On a bien observé que la matrice r Vandermonde confluente A définie par l'équation de déplacement (5.13) est caractérisée par r et les vecteurs g et h, alors leur identification nécessite $O(nd)$ stockages et par convension, on peut l'écrire :

$$A \longleftrightarrow [g, h]^r\,, \tag{5.14}$$

laquelle on peut la représenter par blocs de la manière suivante :

$$A = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1n} \\ A_{21} & A_{22} & \cdots & A_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n1} & A_{n2} & \cdots & A_{nn} \end{bmatrix}, \tag{5.15}$$

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

$$V = [V_1, V_2, \ldots, V_r]^T\,, \tag{5.16}$$

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

En concordance avec la représentation (5.1), A peut être écrit comme suit :

$$A = \begin{bmatrix} A_{11} & G \\ H & K \end{bmatrix}, \tag{5.17}$$

où

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

et

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

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

$$S(A) = K - HA_{11}^{-1}G. \tag{5.18}$$

Par une application directe du théorème (5.1) sur la matrice A, on présente le résultat suivant :

Théorème 5.2 : Soit A la matrice de Vandermonde confluente donnée dans (5.17) et notée par (5.14) ; $A \longleftrightarrow [g, h]^r$ $(r \geq 2)$. Le complément de Schur $S(A)$ définie par (5.18) est lui même une matrice de Vandermonde confluente ; $S(A) \longleftrightarrow [g', h']^{r-1}$ avec :

$$\begin{aligned} g'_k &= g_{k+1} - A_{k+1,1}A_{11}^{-1}g_1 \quad k = \overline{1, r-1} \\ h'^T_k &= h^T_{k+1} - h_1^T A_{11}^{-1} A_{1,k+1} \quad k = \overline{1, r-1}. \quad \square \end{aligned} \tag{5.19}$$

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

$$W = [e_1, v]^n$$

où v est le vecteur définie dans (5.12) et on la représente par blocs de la façon suivante :

$$W = \begin{bmatrix} W_{11} & W_{12} & \cdots & W_{1n} \\ W_{21} & W_{22} & \cdots & W_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ W_{n1} & W_{n2} & \cdots & W_{nn} \end{bmatrix}. \tag{5.20}$$

Comme On a dit précédemment que le processus d'élimination de Gauss est une répétition finie du complément de Schur, alors on peut l'obtenir par l'application de (5.4) sur W, c'est-à-dire :

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

à l'aide de cette notation, on conclut d'aprés le théorème (5.2) que $S^k(W)$; $S^k(W) \longleftrightarrow [g[k], h[k]]^{n-k}$ sont des matrices structurées de Vandermonde confluentes, telle que :

$$\begin{aligned} g[0] &= e_1, \quad h[0]^T = v \\ g_i[k+1] &= g_{i+1}[k] - \left(S^k(W)\right)_{i+1,1} \left(S^k(W)\right)_{11}^{-1} g_1[k] \\ h_i[k+1]^T &= h_{i+1}[k]^T - h_1[k] \left(S^k(W)\right)_{11}^{-1} \left(S^k(W)\right)_{1,i+1} \end{aligned} \tag{5.21}$$

pour $i = \overline{1, n-k+1}$.

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

$$W = LU$$

lesquelles sont définies par :

$$\begin{aligned} L_{i+k,k+1} &= \left(S^k(W)\right)_{i,1} \left(S^k(W)\right)_{11}^{-1} \quad i = \overline{1, n-k} \\ U_{k+1,j+k} &= \left(S^k(W)\right)_{1,j} \quad j = \overline{1, n-k} \end{aligned} \tag{5.22}$$

pour $k = \overline{0, n-1}$.

On remarque bien que les relations de récurence (5.21) qui forment les éléments de base dans le processus d'élimination de Gauss et les formules (5.22) sont données en fonction des éléments de la première ligne et la première colonne de complément de Schur par blocs ; $\left(S^k(W)\right)_{i1}$ et $(S^k(W))_{1j}$, donc il est obligatoire de les stocker à partir de ses structures de déplacement.

5.3 Structures de déplacement pour les éléments en blocs :

En premier lieu, on voit qu'il est obligatoire de donner quelques propriétés sur les matrices Toeplitz qui sont nécessaires dans les démonstrations de cette section.

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

1. $u, v \in \mathbb{C}^d \Longrightarrow \ni z_1 \in \mathbb{C}^d$ telle que $L(u) \cdot L(v) = L(v) \cdot L(u) = L(z_1)$ (resp $\ni z_2 \in \mathbb{C}^d$ telle que $R(u) \cdot R(v) = R(v) \cdot R(u) = R(z_2)$).
2. $u \in \mathbb{C}^d$ et $u_1 \neq 0 \Longrightarrow \ni v_1 \in \mathbb{C}^d$ telle que $L(u)^{-1} = L(v_1)$ (resp $\ni v_2 \in \mathbb{C}^d$ telle que $R(u)^{-1} = R(v_2)$).

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

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

Ensuite, on essaye de trouver les structures de déplacement associées aux matrices de la première ligne et la première colonne de $S^k(W)$; $k = \overline{0, n-1}$.

5.3.1 Stockage de la première ligne de W :

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

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

telle que :

$$V(x) = -\left[\ \rho(x) \quad \rho^{(1)}(x) \quad \rho^{(2)}(x) \quad \cdots \quad \rho^{(d-1)}(x)\ \right] \tag{5.23}$$

$$\rho(x) = \left[\ 1 \quad x \quad x^2 \quad \cdots \quad x^{d-1}\ \right].$$

Ces matrices sont structurées et on peut les transformer en matrices Toeplitz comme l'indique le lemme suivant :

Lemme 5.3 : Soit D la matrice diagonale du type $(d \times d)$ suivante :

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

donc :

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

où

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

Dautre part, si C est une matrice carrée du type $(d \times d)$, alors $C = V(x)$ si et seulement si :

$$ZC - CB(x)^{-1} = e_1 v(x)^T, \tag{5.24}$$

telle que :

$$v(x)^T = -\left[\ \Phi(x) \quad \Phi^{(1)}(x) \quad \Phi^{(2)}(x) \quad \cdots \quad \Phi^{(d-1)}(x)\ \right];\ \Phi(x) = x^{-1}.$$

où $B(x)$ est définie dans (5.10). □

Démonstration : Le premier résultat est un résultat immédiat de calcul matriciel $D^{-1} \cdot v(x)$. Pour le deuxième résultat, il est facile de démontrer que la matrice $V(x)$ est caractérisée par la structure (5.24) à partir de la représentation par blocs de la structure (5.11) ; la structure qui caractérise la matrice de Vandermonde confluente W, qui est représentée par blocs dans (5.20), où :

$$Z = \begin{bmatrix} Z & 0 & 0 & \cdots & 0 \\ e_1 e_d^T & Z & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & e_1 e_d^T & Z \end{bmatrix}$$

et

$$B(x)^{-1} = \begin{bmatrix} B(x_1)^{-1} & 0 & \cdots & 0 \\ 0 & B(x_2)^{-1} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & B(x_n)^{-1} \end{bmatrix}$$

sont les représentations par blocs de Z et $B(x)^{-1}$ dans cette structure et par des calculs simples, on peut conclure que $V(x)$ vérifie vraiment la structure (5.24). Inversement, si C vérifie (5.24), alors :

$$Z(C - V(x)) - (C - V(x))B(x)^{-1} = 0$$

ce qui nous permet de déduire que $C = V(x)$. ■

5.3.2 Stockage de la première colonne de W :

On observe que les éléments de la première colonne de W ; $(W_{i1})_{i=\overline{2,n}}$, on peut les écrire en fonction de W_{11} de la manière suivante :

$$W_{i1} = W_{11} B(x)^{(i-1)d} \quad \text{pour } i = \overline{2, n}.$$

5.3.3 Stockage de la première ligne de $S^k(W)$:

On considère la première ligne par blocs $\left(S^k(W)_{1j}\right)_{j=\overline{1,n-k}}$ de $S^k(W)$. Dans la section précédente, on a dit que $S^k(W) \longleftrightarrow [g[k], h[k]]^{n-k}$ est une matrice $n-k$ structurée de Vandermonde confluente.

La matrice $S^k(W)_{1j} = F(x)$ est caractérisée par la structure de déplacement suivante :

$$ZF(x) - F(x)B(x)^{-1} = zu^T, \tag{5.25}$$

où

$$x = x_{j+k} \quad \text{et} \quad zu^T = g_1[k]\, h_j[k]^T .$$

Laquelle, on peut l'obtenir à partir de la représentation par blocs de la structure de déplacement associée au matrice $S^k(W)$.

Il est clair que les structures (5.24) et (5.25) sont semblables, on peut les identifier par la transformation de z à e_1 et u à v. C'est-à-dire les matrices de la première ligne de $S^k(W)$, on peut les transformer sous forme des matrices de la première ligne de W.

Théorème 5.4 : Soit $F(x)$ la matrice définie par la structure de déplacement (5.25). Alors :

$$F(x) = L(z)V(x)D^{-1}R(q)^{-1}D \tag{5.26}$$

telle que :

$$u' = D^{-1}u \quad \text{et} \quad L(q)u' = D^{-1}v(x),$$

où $q = (q_1, q_2, \ldots, q_d)^T$ est un vecteur de dimension d. □

Démonstration : Soit $F(x)$ la matrice qui vérifie la structure (5.25), telle que le vecteur z dans cette structure vérifie la relation suivante :

$$z = L(z)e_1.$$

Si on prémultiplie l'équation (5.25) par $L(z)^{-1}$ et on tenant compte que $L(z)^{-1}Z = ZL(z)^{-1}$, on trouve que la structure (5.25) devienne :

$$Z\left[L(z)^{-1}F(x)\right] - \left[L(z)^{-1}F(x)\right]B(x)^{-1} = e_1u^T, \tag{5.27}$$

alors, on a transformé le vecteur z au vecteur e_1 et il reste à transformer u au v. Avant de continuer notre démonstration, on a besoin de dire que $DB(x)D^{-1}$ est une matrice Toeplitz triangulaire supérieure défnie comme suit :

$$DB(x)D^{-1} = xI_d + Z^T = R(xe_1 + e_2) =: T(x) \tag{5.28}$$

où $D = diag(1, 1!, 2!, 3!, \ldots, (d-1)!)$ et $B(x)$ est donnée dans (5.10).

Maintenant, on peut continuer notre démonstration par une postmultiplication de (5.27) par D^{-1}, on obtient la structure suivante :

$$Z\left[L(z)^{-1}F(x)D^{-1}\right] - \left[L(z)^{-1}F(x)D^{-1}\right]T(x)^{-1} = e_1 u'^T, \quad u' = D^{-1}u. \tag{5.29}$$

On introduit les vecteurs suivants :

$$\begin{aligned}\tilde{v}(x) &= \left[\ \tfrac{1}{(d-1)!}\Phi^{(d-1)}(x) \quad \tfrac{1}{(d-2)!}\Phi^{(d-2)}(x) \quad \tfrac{1}{(d-3)!}\Phi^{(d-3)}(x) \quad \cdots \quad \tfrac{1}{1}\Phi(x)\ \right]^T \\ &= \left[(-1)^{i-1}x^{-i}\right]_{i=\overline{1,d}}, \quad \Phi(x) = x^{-1}\end{aligned}$$

$$q = (q_1, q_2, \ldots, q_d)^T, \quad \tilde{q} = (q_d, q_{d-1}, \ldots, q_1)^T,$$

où $\tilde{q}$ est la solution du système linéaire triangulaire supérieure suivant :

$$R(u')\tilde{q} = \tilde{w}(x), \ w = D^{-1}v$$

ce qui équivalent à dire que :

$$R(u')R(q) = R(w)$$

c'est-à-dire on a :

$$u'^T R(q) = v^T D^{-1}.$$

Par la postmultiplication de (5.29) par $R(q)$, on obtient :

$$Z\left[L(z)^{-1}F(x)D^{-1}R(q)\right] - \left[L(z)^{-1}F(x)D^{-1}R(q)\right]T(x)^{-1} = e_1 v^T D^{-1},$$

il reste a postmultiplié cette dernière structure par D pour obtenir immédiatement le résultat, de cette façon, on déduire que :

$$L(z)^{-1}F(x)D^{-1}R(q)D = V(x). \quad \blacksquare$$

5.3.4 Stockage de la première colonne de $S^k(W)$:

A partir de la structure de déplacement par blocs vérifiée par les matrices $(n-k)$ structurée de Vandermonde cofluente $S^k(W) \longleftrightarrow [g[k], h[k]]^{n-k}$ pour tout k, on observe que les matrices de sa première colonne $\left(S^k(W)_{i1}\right)_{i=\overline{2,n-k}}$ sont carctérisées par les structures de déplacement de la forme suivante :

$$ZS^k(W)_{i1} - S^k(W)_{i1}B(x)^{-1} = g_i\,[k]\,h_1\,[k]^T - e_1 e_d^T S^k(W)_{i-1,1},\ x = x_{k+1}, \tag{5.30}$$

qu'on peut l'écrire somme de deux matrices structurées, comme on va voir dans le théorème suivant :

Théorème 5.5 : $S^k(W)_{i1} = F_1 + F_2$, telle que :

$$ZF_1 - F_1B(x)^{-1} = g_i\,[k]\,h_1\,[k]^T$$

et

$$ZF_2 - F_2B(x)^{-1} = -e_1 e_d^T S^k(W)_{i-1,1}. \quad \square$$

En dernier lieu, on peut dire que par l'application directe du théorème (5.4), les matrices $S^k(W)_{i1}$ peut-être représentées par la somme de deux matrices structurées qui prennent la même forme que les matrices de la première ligne de W. C'est-à-dire :

$$S^k(W)_{i1} = F_1 + F_2$$

telle que :

$$F_1 = L(g_i\,[k])V(x)D^{-1}R(q_1)^{-1}D,$$

où le vecteur q_1 est défini comme suit :

$$v_1' = D^{-1}h_1\,[k]^T, \quad L(q_1)v_1' = D^{-1}v(x)$$

et

$$F_2 = V(x)D^{-1}R(q_2)^{-1}D,$$

où q_2 est le vecteur défini de la façon suivante :

$$v_2' = D^{-1}e_d^T S^k(W)_{i-1,1}, \quad L(q_2)v_2' = D^{-1}v(x).$$

Alors :

$$S^k(W)_{i1} = L(g_i\,[k])V(x)D^{-1}R(q_1)^{-1}D + V(x)D^{-1}R(q_2)^{-1}D$$

pour $i = 2, n-k$.

5.4 Opérations rapides sur les éléments en blocs.

Dans cette section, on présente des opérations de base qui jouent des rôles extrêmement importants dans le processus d'élimination de Gauss par blocs du matrice W, ces opérations sont basées sur $F(x)$, la représentation (5.26), comme la prémultiplication matrice par vecteur de la forme $F(x)\cdot b$ et la post-multiplication d'un vecteur ligne par matrice de forme $b^T \cdot F(x)$, lesquelles, on peut les réaliser par l'utilisation de $O(d \log d)$ opérations élémentaires et les interpréter sous forme des deux algorithmes suivants, mais avant de les présenter, on sait que la matrice inverse $R(q)^{-1}$ dans la représentation (5.26) est une matrice Toeplitz triangulaire supérieure $R(a)$, où le vecteur $a = (a_1, a_2, \ldots, a_d)^T$ peut être obtenu directement par la résolution du système suivant :

$$R(v')\tilde{a} = \tilde{u}'$$

telle que :

$$v' = D^{-1}v(x)$$

et

$$\tilde{a} = (a_d, \ldots, a_1)^T\,,\ \tilde{u}' = (u'_d, \ldots, u'_1)^T\,.$$

Algorithme 5.1. **MVP1** $(z,\ u, x\ , b \longmapsto \acute{b})$ cet algorithme réalise la multiplication $F(x)\cdot b = \acute{b}$, où $F(x)$ définie par (5.26) et représentée par les paramètres z, u et x.

Données : z, u, x, b.
Résultats : le vecteur $\acute{b}$.

1. calcul $\acute{u}^T := u^T D^{-1}$, où $\tilde{u}'^T = (\acute{u}_d, \ldots, \acute{u}_1)^T$.
2. calcul $\acute{v} := D^{-1}v(x)$.
3. résolution du système $R(\acute{v})\tilde{a} = \tilde{u}'$; $\tilde{a} = (a_d, \ldots, a_1)^T$ et $a = (a_1, \ldots, a_d)^T$.
4. calcul $\acute{b}_1 := Db$.
5. calcul $\acute{b}_2 := R(s)\acute{b}_1$.
6. calcul $\acute{b}_3 := D^{-1}\acute{b}_2$.
7. calcul $\acute{b}_4 := L(\eta(x))\acute{b}_3$, où $\eta(x) = (1, x, \frac{x^2}{2!}, \frac{x^3}{3!}, \ldots, \frac{x^{d-1}}{(d-1)!})$.
8. calcul $\acute{b}_5 := D\acute{b}_4$.
9. retour de $\acute{b} := L(z)\acute{b}_5$.

Algorithme 5.2. **MVP2** $(z, u, x, p \longmapsto c)$ cet algorithme est similaire au précédent, il réalise la multiplication d'un vecteur ligne b^T par $F(x)$ définie par (5.26) et représenée toujours par les paramètres z, u et x.

Données : z, u, x, b.
Résultats : le vecteur c.

1. calcul $\acute{u}^T := u^T D^{-1}$, où $\tilde{u}' = (\acute{u}_d, \ldots, \acute{u}_1)^T$.
2. calcul $\acute{v} := D^{-1} v(x)$.
3. résolution du système $R(\acute{v})\tilde{a} = \tilde{u}'$; $\tilde{a} = (a_d, \ldots, a_1)^T$ et $a = (a_1, \ldots, a_d)^T$.
4. calcul $b_1^T := b^T L(z)$.
5. calcul $b_2^T := b_1^T D$.
6. calcul $b_3^T := b_2^T L(\eta(x))$, où $\eta(x) = (1, x, \frac{x^2}{2!}, \frac{x^3}{3!}, \ldots, \frac{x^{d-1}}{(d-1)!})$.
7. calcul $b_4^T := b_3^T D^{-1}$.
8. calcul $b_5^T := b_4^T R(s)$.
9. retour de $c^T := b_5^T D$.

Données			Résultats
d	x	u	a
3	2	$\begin{bmatrix} 1 & 1 & 1 \end{bmatrix}^T$	$\begin{bmatrix} -2 & -2 & -2 \end{bmatrix}^T$
4	2	$\begin{bmatrix} -1 & 1 & -2 & 2 \end{bmatrix}^T$	$\begin{bmatrix} -2 & -2 & 4 & -4 \end{bmatrix}^T$
5	2	$\begin{bmatrix} 1 & 3 & 0 & 2 & 2 \end{bmatrix}^T$	$\begin{bmatrix} -2 & -6 & 0 & -4 & -4 \end{bmatrix}^T$

Tableau 3. d'exécution du programme **Vect**.

Données					Résultats	
x	d	z	u	b	$\acute{b}$	
2	3	$\begin{bmatrix}1\\1\\2\end{bmatrix}$	$\begin{bmatrix}-1\\0\\-2\end{bmatrix}$	$\begin{bmatrix}1\\1\\1\end{bmatrix}$	$\begin{bmatrix}10\\32\\94\end{bmatrix}$	
2	4	$\begin{bmatrix}1\\2\\4\\-1\end{bmatrix}$	$\begin{bmatrix}4\\2\\1\\3\end{bmatrix}$	$\begin{bmatrix}-2\\-1\\0\\1\end{bmatrix}$	$\begin{bmatrix}-16\\-68\\-240\\-704\end{bmatrix}$	
2	5	$\begin{bmatrix}1\\1\\1\\1\\1\end{bmatrix}$	$\begin{bmatrix}2\\2\\2\\2\\2\end{bmatrix}$	$\begin{bmatrix}3\\3\\3\\3\\3\end{bmatrix}$	$1.0e+004$	$\begin{bmatrix}-0.0408\\-0.1620\\-0.5220\\-1.5900\\-4.7484\end{bmatrix}$

Tableau 4. d'exécution du programme **MVP1**.

Données					Résultats	
x	d	z	u	b	c	
2	3	$\begin{bmatrix}1\\-1\\2\end{bmatrix}$	$\begin{bmatrix}-1\\-2\\-3\end{bmatrix}$	$\begin{bmatrix}0\\1\\4\end{bmatrix}$	$\begin{bmatrix}34\\22\\20\end{bmatrix}$	
2	4	$\begin{bmatrix}1\\1\\2\\2\end{bmatrix}$	$\begin{bmatrix}-2\\-2\\1\\1\end{bmatrix}$	$\begin{bmatrix}1\\0\\0\\-1\end{bmatrix}$	$\begin{bmatrix}-102\\-134\\-280\\-408\end{bmatrix}$	
2	5	$\begin{bmatrix}1\\1\\2\\3\\2\end{bmatrix}$	$\begin{bmatrix}-1\\-3\\-1\\0\\0\end{bmatrix}$	$\begin{bmatrix}1\\4\\2\\1\\0\end{bmatrix}$	$1.0e+003$	$\begin{bmatrix}0.0240\\0.1200\\0.6120\\2.1120\\2.3040\end{bmatrix}$

Tableau 5. d'exécution du programme **MVP2**.

Dans la section précédente, l'algorithme d'élimination de Gauss par blocs dépend itérativement par les opérations clées suivantes :

(a) $h^T S^k(W)_{1j}$
(b) $S^k(W)_{i1} g$
(c) $e_d^T S^k(W)_{i1}$

c'est pour cette raison, on présente le résultat suivant pour donner la complexité de ces opérations.

Théorème 5.6 : le produit matrice-vecteur $c = L(a) \cdot b$ est réalisé par $C(m)d\log d$ opérations et l'inversion d'une matrice $L(b) = L(a)^{-1}$ est réalisée par l'utilisation de $2C(m)d\log d$ opérations. Alors les opérations par blocs (a), (b) et (c) précédentes peut être effectuer par l'utilisation de $26C(m)d\log d+O(d)$ opérations. Plus précisément, pour effectuer l'opération (a), on a besoin de $6C(m)d\log d+O(d)$ opérations, pour (b), on a besoin de $11C(m)d\log d+O(d)$ opérations et pour (c), $9C(m)d\log d+O(d)$ opérations.

Démonstration : pour effectuer les opérations (a), (b) et (c), on fait l'appel aux algorithmes (5.1) et (5.2) comme on va voir prochainement.

($\star$) Pour l'opération (a) : on peut trouver facilement que l'opération (a) est effectuée par l'utilisation de $O(d\log d)+O(d)$ opérations dû a l'appel de l'algorithme **MVP2** et au nombre d'opérations $3C(m)d\log d$ effectué à la troisième ligne.

($\star\star$) Pour l'opération (b) : on a vu précédemment, éxactement dans le théorème (5.5) que $S^k(W)_{i1}g$ peut s'écrire de la manière suivante :

$$S^k(W)_{i1}g = F_1g + F_2g,$$

alors, le calcul de la quantité (b) ; $S^k(W)_{i1}g$ nécessite le calcul des deux quantités F_1g et F_2g. Donc il est obligatoire de faire l'appel à l'algorithme **MVP1**, sur lequel, on peut dire que sa troisième ligne peut être exécutée par l'utilisation de $3C(m)d\log d$ opérations, c'est-à-dire l'opération F_1g est effectuée par $6C(m)d\log d+O(d)$ opérations et ce qui concerne le calcul de F_2g, il est clair que l'instruction $\acute{b} := L(z)\acute{b}_5$ au niveau de la neuvième ligne de l'algorithme (5.1) est négligeable si $z = e_1$, comme une conséquence, la quantité F_2g est exécutée par l'utilisation de $5C(m)d\log d+O(d)$ opérations, ce qui nous permet de dire que pour effectuer l'opération (b), on a besoin de $11C(m)d\log d+O(d)$ opérations.

($\star\star\star$) Pour l'opération (c) : c'est exactement de la même manière que (b), on peut compter le nombre d'opérations nécessaire dans l'effectation de (c). Premièrement, l'opération par blocs $e_d^TS^k(W)_{i1}$ est représentée comme suit :

$$e_d^TS^k(W)_{i1} = e_d^TF_1 + e_d^TF_2,$$

c'est-à-dire le calcul de (c) est équivalent aux calculs de $e_d^T F_1$ et $e_d^T F_2$.

En outre, il est clair que l'instruction $b_1^T := b^T L(z)$ dans la quatrième ligne de l'algorithme (5.2) est négligeable si $b = e_d$. Donc le calcul de $e_d^T F_1$ peut être effectué par l'utilisation de $5C(m)d\log d + O(d)$ opérations. De plus, on a remarqué que $4C(m)d\log d + O(d)$ est le nombre d'opérations nécessaire dans la réalisation de $e_d^T F_2$ tout simplement parce que les lignes 4 et 6 de l'algorithme (5.2) sont négligeables lorsque $b = e_d$ et $z = e_1$.

Enfin, on conclut que le nombre d'opérations nécessaire dans ce cas est vraiment $9C(m)d\log d + O(d)$ opérations.

5.5 L'algorithme d'élimination de Gauss par blocs et complexité :

Dans cette section, nous sommes prés de présenter l'algorithme d'élimination de Gauss par blocs ayant comme but, la résolution du système linéaire structuré $Wa = f$ et il est basé sur les structures des matrices, qui sont invariantes par le complément de Schur. Lorsque, on a la matrice structurée par blocs W du type $(nd \times nd)$ de sorte que ces éléments sont des matrices du type $(d \times d)$, on trouve que l'élimination de Gauss par blocs peut être réalisé par l'utilisation de $O(n^2\tau(d))$ opérations, où $\tau(d)$ est le nombre d'opérations nécessaire d'effectuer une opération par blocs dans cette matrice. On sait que les méthodes standards pour résoudre un système linéaire du type $(d \times d)$ utilisent $O(d^2)$ opérations, c'est-à-dire $\tau(d) = d^2$, dans le cas où, on utilise l'algorithme FFT rapide, $\tau(d)$ est réduit à $O(d\log d)$ opérations seulement. L'algorithme d'élimination de Gauss par blocs suivant calculera d'une façon implicite la factorisation LU de W ; $W = LU$ par la transformation du système $Wa = f$ au système équivalent triangulaire supérieur suivant :

$$Ua = \overline{f},$$

où

$$\overline{f} = L^{-1}f,$$

comme on va voir prochainement dans l'algorithme suivant.

Algorithme 5.3. L'élimination de Gauss par blocs de $Wa = f$.

Le processus d'élimination de Gauss par blocs, on peut le déviser en trois étapes, la première est l'étape initiale qui reçoit les données initiales, la deuxième étape est la principale dans ce processus, elle est effectuée d'une

façon itérative pour transformer le système $Wa = f$ au système $Ua = \overline{f} = L^{-1}f$ à partir d'une fonction **BGE**$(k := 0 : (n-2))$, dont elle reçoit le $k^{ème}$ complément de Schur $S^k(W) \longleftrightarrow [g[k], h[k]]^{n-k}$ de W et résulte la $(k+1)^{ème}$ complément de Schur $S^{k+1}(W) \longleftrightarrow [g[k+1], h[k+1]]^{n-k-1}$ de W, de plus la même fonction **BGE**$(k := 0 : (n-2))$ transforme $(f_i[k])_{i=1:n}$ au $(f_i[k+1])_{i=1:n}$. Dans la troisième étape, on calcule la matrice triangulaire supérieure U et le vecteur $\overline{f} = (f_1[k-1])_{k=1:n}^T$ pour résoudre le système $Ua = f$.

Données : les n nombres non nuls $x_1, x_2, \ldots, x_n$ et le vecteur f.
Résultats : le vecteur a, telle que $Wa = f$.

Etape 1 : étape initiale.

1. $g_1[0] := (1, 0, \ldots, 0)^T$.
2. **pour** $i := 1 : n$.
3. $\quad h_i[0] := (x_i^{-1}, -x_i^{-2}, (-1)^2(2!)x_i^{-3}, \ldots, (-1)^{d-1}(d-1)!x_i^{-d})^T$.
4. $\quad f_i[0] := f_i$.
5. $\quad$ **si** $(i \neq 1)$ $g_i[0] = 0$.

Etape 2 : étape principale.

1. **pour** $k := 0 : (n-1)$
2. $\quad$ **BGE**(k).

fonction BGE$(k = 0 : (n-2))$ cette fonction est s'intéresse au calcul de $S^{k+1}(W) \longleftrightarrow [g[k+1], h[k+1]]^{n-k-1}$ et $(f_i[k+1])_{i=1:n}$ à partir de $g_i[k], h_i[k]$; $i = 1 : n-k$ et $(f_i[k])_{i=1:n}$.

1. $S^k(w)_{11} := U_{k+1,k+1}$ utiliser sa structure de déplacement

$$ZS^k(w)_{11} - S^k(w)_{11}B(x_{k+1}) = g_1[k]\, h_1[k]^T$$

2. $v[k]^T := h_1[k]^T S^k(w)_{11}^{-1}$.
3. $lr_1[k]^T := e_d^T S^k(w)_{11}$.
4. $z[k] := S^k(w)_{11}^{-1} g_1[k]$.
5. $f^{-1}[k] := S^k(w)_{11}^{-1} f_1[k]$.
6. **pour** $i := 1 : (n-k-1)$

7. calcul $S^k(w)_{i+1,1}$ utiliser sa structure de déplacement

$$ZS^k(w)_{i+1,1} - S^k(w)_{i+1,1}B(x_{k+1}) = g_{i+1}[k]\, h_1[k]^T - e_1 lr_i[k]^T$$

8. $lr_{i+1}[k]^T := e_d^T S^k(w)_{i+1,1}$.
9. $g_i[k+1] := g_{i+1}[k] - S^k(w)_{i+1,1} z[k]$.
10. $f_i[k+1] := f_{i+1}[k] - S^k(w)_{i+1,1} f^{-1}[k]$.
11. calcul $S^k(w)_{1,i+1} := U_{k+1,i+k+1}$ utiliser sa structure de déplacement

$$ZS^k(w)_{1,i+1} - S^k(w)_{1,i+1}B(x_{k+i+1}) = g_1[k]\, h_{i+1}[k]^T$$

12. $h_i[k+1]^T := h_{i+1}[k]^T - v_1[k]^T U_{k+1,i+k+1}$
13. retour de $S^{k+1}(W) \longleftrightarrow [g[k+1], h[k+1]]^{n-k-1}$ et $(f_i[k+1])_{i=1:n}$

Etape 3 : étape de substitution.

1. $a_n := U_{nn}^{-1} f_1[n]$.
2. **pour** $k = n-1 : 1$
3. $sum := f_1[k]$
4. **pour** $j = (k+1) : n$
5. $sum := sum - U_{kj} a_j$
6. $a_k := U_{kk}^{-1} sum$.

Données				Résultats
x	n	d	f	a
$\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$	3	4	$\begin{bmatrix} 1 \\ -1 \\ 2 \\ -2 \\ 3 \\ 4 \\ 5 \\ -3 \\ 8 \\ 7 \\ -5 \\ -5 \end{bmatrix}$	$\begin{bmatrix} 18950 \\ -15960 \\ -1539.9 \\ -854.27 \\ -1.1861 \\ -5.3877 \\ 3.2325 \\ -0.64564 \\ 0.0027404 \\ -0.00032033 \\ 0.00015134 \\ -2.5429e-005 \end{bmatrix}$

$\begin{bmatrix} 2 \\ -3 \\ 4 \\ 5 \end{bmatrix}$	4	4	$\begin{bmatrix} 2 \\ 1 \\ -3 \\ -5 \\ 8 \\ -7 \\ 9 \\ 11 \\ 13 \\ -2 \\ -1 \\ 1 \\ 1 \\ 4 \\ 5 \\ 6 \end{bmatrix}$	$\begin{bmatrix} 61.42 \\ -41.545 \\ 11.469 \\ -16.473 \\ 7.0024e+013 \\ -1.2177e+014 \\ 6.3325e+013 \\ -1.1909e+013 \\ -4.340.8 \\ 3513.2 \\ 1909.5 \\ 200.85 \\ 1.8787e-013 \\ 1.1767e-013 \\ 2.6397e-014 \\ 2.1113e-015 \end{bmatrix}$
$\begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{bmatrix}$	5	4	$\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 2 \\ 2 \\ 2 \\ 2 \\ 3 \\ 3 \\ 3 \\ 3 \\ 4 \\ 4 \\ 4 \\ 4 \\ 5 \\ 5 \\ 5 \\ 5 \end{bmatrix}$	$\begin{bmatrix} -9.9946e+030 \\ 3.3004e+030 \\ -6.11e+029 \\ 3.076e+029 \\ 0.00015456 \\ -0.00011164 \\ 3.2725e-005 \\ -3.2925e-005 \\ 3.3197e+007 \\ -4.4608e+007 \\ 2.6434e+007 \\ -4.2172e+006 \\ -2.8578e-005 \\ -1.0917e-005 \\ 1.555e-006 \\ 1.0907e-006 \\ 2.1514e-025 \\ -2.066e-025 \\ 6.6525e-026 \\ -6.928e-027 \end{bmatrix}$

Tableau 6. d'exécution du programme **programme principal.**

Avant d'achever ce chapitre, il est nécessaire d'analyser la complexité en temps de cet algorithme. On se base sur les résultats énoncés dans la section précédente.

Théorème 5.7 : Le produit matrice par vecteur $c = L(u)v$ peut être réalisé avec $C(m)d\log d$ opérations et l'inversion de la matrice $L(v) = L(u)^{-1}$ peut être réalisée par l'utilisation de $2C(m)d\log d$ opérations. Alors la complexité en temps de l'algorithme (5.3) est :

$$21.5C(m)n^2d\log d + O(n^2d) + O(nd\log d). \quad \square$$

Démonstration : Par l'application du théorème (5.6), les opérations utilisées dans les lignes 8,9,10 et 12 de la fonction **BGE** sont effectuées par l'utilisation respective de $9C(m)d\log d$, $11C(m)d\log d$, $11C(m)d\log d$ et $6C(m)d\log d$ opérations, donc la fonction **BGE** utilise :

$$T_{BGE}(n,d,k) = 37C(m)(n-k-1)d\log d + O((n-k-1)d) + O(d\log d)$$

opérations, on ajoute que les lignes 2,3 et 4 de la même fonction **BGE** sont réalisées par l'utilisation de $O(d\log d)$ opérations, ce qui nous permet de déduire que la deuxième étape dans l'algorithme (5.3) utilise :

$$T_{\text{étape }2}(n,d) = 18.5C(m)(n-k-1)d\log d + O(n^2d) + O(nd\log d).$$

Pour effectuer la troisième étape, on a besoin d'utiliser le nombre d'opérations suivant :

$$T_{\text{étape }3}(n,d) = 3C(m)n^2d\log d + O(n^2d) + O(nd\log d).$$

Finalement, on conclut que la complexité en temps de l'algorithme (5.3) est

$$\begin{aligned} T &= T_{\text{étape }1} + T_{\text{étape }2} + T_{\text{étape }3} \\ &= 21.5C(m)n^2d\log d + O(n^2d) + O(nd\log d) \end{aligned}$$

c'est exactement le résultat chercher. ■

Chapitre 6

Annexe des travaux pratiques.

Les algorithmes qu'on a présentés précédemment sont disponibles sous forme des programmes en language **Matlab.**

```
%La multiplication de deux polynômes basée sur la transformation
%de Fourier et son inverse.
function[c]=Mult(p1,p2);
%d=input('donner le dimension des vecteurs d=' );
%p1=input('donner le vecteur p1=' );
%p2=input('donner le vecteur p2=' );
d=length(p1);
%On terminera chaque vecteur par les zéros jusqua l'ordre 2*d.
for i=1 :2*d
if i<=d
pu(i)=p1(i);
pd(i)=p2(i);
else
pu(i)=0;
pd(i)=0;
end
end
%La représentation des vecteurs sous forme des polynômes qui sont
%représentés comme vecteurs lignes dont les composantes sont
%ordonnées par ordre des puissances décroissantes.
%Un polynôme de degré n est représenté par un vecteur ligne de
%taille (n+1).
%Le retournement horizontal d'un vecteur ligne ce fait grâce a la
%fonction "fliplr".
pv=fliplr(pu);
```

```
pw=fliplr(pd);
ps=ifft(fft(pv).*fft(pw));
for i=1 :d
c1(i)=ps(2*d-i);
end
c=c1';
end


%Un sous programme qui donne le premier vecteur d'une
%matrice inverse Toeplitz triangulaire inférieure.
clear
clc
function [bb]=PVMITTI(ad,nn);
%ad=input('donner la matrice ad= ');
%Le dimension de la matrice ad est de la forme n=2^q.
%nn=input('donner le nombre nn= ');
%La fonction size indique le dimension d'une matrice.
%[nn nn]=size(ad);
q=input('donner le nombre q= ');
c=ad( :,1);
b0=1/ad(1,1);
i=0;
while i<=q-1
for j=1 :2^(i)
c1(j)=c(j);
c2(j)=c(2^(i)+j);
end;
for j=1 :2^(i)
if j==1
c3(j)=c2(j);
else
c3(j)=c1(2^(i)-j+2);
end;
end;
b1=toeplitz(c2,c3);
b2=-b0*b1*b0;
for j=1 :2^(i)
b3(j)=b0(j,1);
b3(2^(i)+j)=b2(j,1);
end;
```

```
for k=1 :2^(i+1)
if k==1
b4(k)=b3(k);
else
b4(k)=0;
end;
end;
b0=toeplitz(b3,b4);
i=i+1;
end;
%Le premier vecteur de la matrice inverse.
bb=b0( :,1);
end;


%Le sous programme "vect" qui recherche un vecteur aa.
function [aa]=Vect(x,u);
%d=input('donner la dimension des vecteurs d=');
%x=input('donner la valeur de x=');
%u=input ('donner le vecteur u=');
d=length(x);
%La fonction "factorial" ;factorial(n) c'est le factorielle d'un nombre n.
for i=1 :d
vu(i)=factorial(i-1);
end;
%L'écriture d'une matrice diagonale a l'aide le la fonction "diag".
v=diag(vu);
%La multiplication de l'inverse d'une matrice avec un vecteur.
vd=inv(v)*u;
for i=1 :d
vq(i)=((-1)^(i))*factorial(i-1)*x^(-i);
end;
vc=inv(v)*vq';
%L'écriture d'une matrice Toeplitz triangulaire supérieure nécessite la
%connaissance de la première ligne et la première colonne de cette matrice.
for i=1 :d
if i==1
vs(i)=vc(i);
else
vs(i)=0;
```

```
end;
end;
r=[toeplitz(vc,vs)]';
%La fonction flipud fait un retourenement vertical d'un vecteur.
vt=inv(r)*flipud(vd);
aa=flipud(vt);
end;


%Le sous programme "premult" fait la postmultiplication d'une
%matrice représenteé par Toeplitz avec un vecteur b.
function [p]=premult(x,z,u,b,d1);
%On introduit les données x z u b a l'aide de la fonction 'input'.
%x=input('la valeur x est : ');
%d1=input('donner le dimension des vecteur d1=');
%z=input('le vecteur z est :');
%u=input('le vecteur u est donner par :');
%b=input('donner le vecteur b=');
%La fonction 'factorial' calcul le factorielle d'un nombre.
for i=1 :d1
wu(i)=factorial(i-1);
end
wu;
%Le produit ponctuel de deux vecteurs.
wd=wu'.*b;
%On fait l'appel du sous programme "Vect".
wt=Vect(x,u);
%Le retournement vertical d'un vecteur ce fait en Matlab grâce a la
%fonction flipud.
wq=flipud(wd);
p1=wt;
p2=wq;
%Le premier appel du sous programme "Mult".
wc= Mult(wt,wq);
ws=flipud(wc);
for i=1 :d1
w0(i)=factorial(i-1)^(-1);
end
w0;
w1=w0'.*ws;
for i=1 :d1
```

```
w2(i)=x^(i-1)/factorial(i-1);
end
p1=w2;
p2=w1;
%Le deuxième appel du sous programme "Mult" .
w3= Mult(w2,w1);
w4=wu'.*w3;
p1=z;
p2=w4;
%Le troisième appel du sous programme "Mult" .
p=Mult(z,w4);
end


%Le sous programme "MVP2" fait la prémultiplication d'une matrice
%représenteé par Toeplitz avec un vecteur ligne.
function [ff]=postmult(x,z,u,b,d1);
%On introduit les données.
%x=input('la valeur x est : ');
%d=input('donner le dimension des vecteur d=');
%z=input('le vecteur z est :');
%u=input('le vecteur u est donner par :');
%b=input('donner le vecteur b=');
b1=flipud(b);
p1=b1;
p2=z;
%L'appel du sous programme 'Mult'.
l1=Mult(b1,z);
b2=flipud(l1);
for i=1 :d1
f1(i)=factorial(i-1);
end
%Le produit ponctuel de deux vecteurs.
b3=b2.*f1';
for i=1 :d1
f2(i)=x^(i-1)/factorial(i-1);
end
f3=flipud(b3);
p1=f3;
p2=f2;
%Deuxième appel du sous programme 'Mult'.
```

```
l2=Mult(f3,f2);
b4=flipud(l2);
for i=1 :d1
f4(i)=1/factorial(i-1);
end
b5=b4.*f4';
%L'appel du sous programme 'Vect'.
f5=Vect(x,u);
p1=b5;
p2=f5;
%Troisième appel du sous programme 'Mult'.
l3=Mult(b5,f5);
b6=flipud(l3);
ff=b6.*f1';
end;


%Un Proqramme principal de résolution d'un système
%linéaire structuré de de Vandermonde confluent Wa=f.
clear
clc
X=input('donner le vecteur X=');
n=input('donner la valeur n=');
n=length(X);
%d est une puissance de deux.
d=input('donner la valeur d=');
f=input('donner le vecteur f=');
%Présentation des vecteurs g et h sous forme de
%matrices du type (n*d).
for i=1 :d
for j=1 :n
if (i==1)&(j==1)
g(1,1)=1;
else
g(i,j)=0;
end;
h(i,j)=(-1)^(i-1)*factorial(i-1)*X(j)^(-i);
end;
end;
%Présentation du vecteur f sous forme d'une matrice F.
for i=1 :n
```

```
F( :,i)=f((i-1)*d+1 :i*d) ;
end ;
%Les premières colonnes de F, on les stockées dans une
%matrice G.
G( :,1)=F( :,1) ;
s=0 ;
for k=0 :n-1
g1=g( :,1) ;
h1=h( :,1) ;
for i=1 :d
v1(i)=factorial(i-1) ;
end ;
%L'écriture d'une matrice diagonale ce fait à partir de
%la fonction diag.
D=diag(v1) ;
%On fait l'appel du sous programme Vect.
x=X(k+1) ;
u=h1 ;
au=Vect(X(k+1),h1) ;
%L'écriture d'une matrice Toeplitz triangulaire inférieure
%définie par sa première colonne au.
for i=1 :d
if i==1
a1(i)=au(i) ;
else
a1(i)=0 ;
end ;
end ;
M1=(Toeplitz(au,a1))' ;
su1=M1*D ;
for i=1 :d
v2(i)=1/factorial(i-1) ;
end ;
D1=diag(v2) ;
su2=D1*su1 ;
for i=1 :d
v3(i)=X(k+1)^(i-1)/factorial(i-1) ;
end ;
for i=1 :d
if i==1
v4(i)=v3(i) ;
```

```
else
v4(i)=0;
end;
end;
M2=Toeplitz(v3,v4);
su3=M2*su2;
su4=D*su3;
for i=1 :d
if i==1
g2(i)=g1(i);
else
g2(i)=0;
end;
end;
M3=Toeplitz(g1,g2);
su=M3*su4;
U(k*d+1 :(k+1)*d, :)=su;
k1=D1*g1;
for i=1 :d
if i==1
k2(i)=k1(i);
else
k2(i)=0;
end;
end;
r1=(Toeplitz(k1,k2))';
for i=1 :d
v5(i)=(-1)^(i)*factorial(i-1)*X(k+1)^(-i);
end;
q0=inv(r1)*flipud(v5');
q=flipud(q0);
for i=1 :d
if i==1
q1(i)=q(i);
else
q1(i)=0;
end;
end;
R=(Toeplitz(q,q1))';
ss1=D1*R;
ss2=ss1*D;
```

```
ad=M2;
nn=d;
b1=PVMITTI(M2,d);
for i=1 :d
if i==1
b2(i)=b1(i);
else
b2(i)=0;
end;
end;
M4=Toeplitz(b1,b2);
ss3=ss2*M4;
ss4=ss3*D1;
for i=1 :d
if i==1
g2(i)=g1(i);
else
g2(i)=0;
end;
end;
g3=toeplitz(g1,g2);
ad=g3;
nn=d;
b3=PVMITTI(g3,d);
for i=1 :d
if i==1
b4(i)=b3(i);
else
b4(i)=0;
end;
end;
M4=Toeplitz(b3,b4);
ss=ss4*M4;
V(k*d+1 :(k+1)*d, :)=ss;
v=h1'*ss;
idd=eye(d);
i1=idd( :,1);
id=idd( :,d);
x=X(k+1);
z=g1;
u=h1;
```

```
b=id;
d1=d;
lr( :,1)=postmult(X(k+1),g1,h1,id,d);
zu=ss*g1;
F1=ss*F( :,1);
h1=h( :,1);
for i=1 :(n-k-1)
x1=X(k+i+1);
T=h( :,i+1);
gi=g( :,i+1);
fi=F( :,i+1);
lri=lr( :,i);
x=X(k+1);
z=gi;
u=h1;
b=id;
d1=d;
lriu=postmult(X(k+1),gi,h1,id,d);
b=zu;
giu=premult(X(k+1),gi,h1,zu,d);
b=F1;
fiu=premult(X(k+1),gi,h1,F1,d);
z=i1;
u=lri;
b=id;
lrid=postmult(X(k+1),i1,lri,id,d);
b=zu;
gid=premult(X(k+1),i1,lri,zu,d);
b=F1;
fid=premult(X(k+1),i1,lri,F1,d);
lr( :,i+1)=lriu+lrid;
gii=giu+gid;
g( :,i)=gi-gii;
fii=fiu+fid;
%La i^ième colonne de la matrice F.
F( :,i)=fi-fii;
G( :,k+2)=F( :,1);
st1=M3*D;
for j=1 :d
v5(j)=x1^(j-1)*(factorial(j-1))^(-1);
end;
```

```
for j=1 :d
if j==1
v6(j)=v5(j);
else
v6(j)=0;
end;
end;
M5=Toeplitz(v5,v6);
st2=st1*M5;
st3=st2*D1;
v7=Vect(X(k+1),u);
for j=1 :d
if j==1
v8(j)=v7(j);
else
v8(j)=0;
end;
end;
M6=(Toeplitz(v7,v8))';
st4=st3*M6;
st=st4*D;
if k>0
s=s+(n-k)*d;
end;
if k<=n-2
W1((i-1)*d+1+s :i*d+s, :)=st;
end;
h( :,i)=T-(v*st)';
end;
end;
fsh((n-1)*d+1 :n*d)=V((n-1)*d+1 :n*d, :)*G( :,n);
for k=n-1 :-1 :1
s=s+d-(n-k)*d;
sum=G( :,k);
for j=(k+1) :n
sum=sum-W1((j-k-1)*d+1+s :(j-k)*d+s, :)*fsh((j-1)*d+1 :j*d)';
end;
fsh((k-1)*d+1 :k*d)=V((k-1)*d+1 :k*d, :)*sum;
end;
```

```
a=fsh';
%L'affichage du résultat finale.
disp(a);
end;
```

Conclusion

Dans ce mémoire de thèse, nous avons mené une étude numérique sur les matrices structurées. De fait nous avons considérer en particulier les matrices Toeplitz, les matrices de Vandermonde et les matrices de Vandermonde confluentes. La structure de déplacement conçue (et investie) Pour les matrices Toeplitz a été découverte par Kailath, Kung et Morf [27] en 1978. Depuit, il s'est avéré que la construction d'une équation de déplacement caractérisant une classe de matrices conduit directement à des implémentations numériques performantes sur de telles matrices. En 1984, Heining et Rost [25] ont fait remarquer que les matrices de Vandermonde satisfont une équation de déplacement analogue à celle proposée dans [27]. Dans ce contexte de motivation, Melkemi [34] a, plus récement, démontré que les matrices de Vandermonde confluentes font également partie de la classe des matrices structurées.

Etant donnée une matrice de Vandermonde confluente du type $(md \times md)$ vue comme une matrice du type $(m \times m)$ dont les éléments sont des matrices du type $(d \times d)$. Nous avons montré ici que ces éléments sont des matrices structurées et admettent, en conséquance, une décomposition canonique en termes de matrices Toeplitz triangulaires. Ce qui permet une manipulation numérique rapide de ces éléments grâce aux FFT. C'est donc pour nous une aubaine heureuse de parler de la transformation discrète de Fourier puisqu'il est question de s'assurer que l'algorithme que nous proposons est entre autres numériquement stable.

La stabilité numérique de la FFT que nous avons proposé au premier chapitre a été emprunté de l'analyse effectuée par Higham [26]. Concernant le produit de Kronecker auquel on a fait appel au cours de cette analyse, on est invité a voir le livre de Golub et Van Loan [19].

Pour la programmation des algorithmes, nous avons utilisé MATLAB [37], [38] qui est à notre avis un code en adéquation avec notre souci d'introduire des opérations matricielles auxiliaires.

Bibliographie

[1] **A. V. Aho. J. E. Hopcroft et J. D. Ullman.** The design and analysis of comuter algorithms Addition Wesly, Reading, MA. 1973.

[2] **G. S. Ammar et W. B. Gragg.** Superfast solution of real positive definite Toeplitz systems SIAM J. Matrix Anal. 9(1988) 61-76.

[3] **A. W. Bojanckzyk, R. P. Brent,F. R. de Hoog et D.R. Sweet.** On the stability of the Bareiss and related Toeplitz factorisation algorithms.SIAM J. Matrix anal. appl. 16(1995) 40-57.

[4] **A. Bjork et T. Elfving.** Algorithms for confluent Vandermonde systems, numer. Math. 21(1973) 130-137.

[5] **T. Boros, T. Kailath et V. Olshevsky.** Predictif pivoting and backward stability of fast Cauchy solvers. Preprint, 1995.

[6] **E. Boros et C. Di Fiore.** On the use of certain matrix agebras associated with discret trigonomitric transforms in matrix displacement decomposition SIAM J. Matrix Anal. Appl. 16(1995) 312-326.

[7] **S. Chandrasekaran et A. H. Sayed.** Stabilizing the fast ganeralized Shur algorithme. SIAM J. Matrix Anal. Appl. 17(1996) 950-983.

[8] **S. Chandrasekaran et A. H. Sayed.** A fast stable solver for non symmetric Toeplitz and quasi-Toeplitz systems of linear equation, SIAM J. Matrix Anal. Appl. 19(1998) 107-139.

[9] **J. Chun and T Kailath.**Generalized displacement structure for block Toplitz , Toeplitz-block and Toeplitz-derived matrices SIAM J. Matrix Anal. Appl. 15(1994) 114-128.

[10] **J. Chun and T. Kailath.** Divide and conquer solutions of least squares problems for matrices with displacementstructures. SIAM J. Matrix Anal. Appl.12(1991) 128-145.

[11] **C. de Boor et pinkus.** Backward error analysis for totally positive linear systems, Linear Algebra and Appl. 27(1977) 485-490.

[12] **C. J. Demeur.** QR factorisation of confluent Vandermonde matrices IEEE Trans. Acoust. Speech, Signal Processig 38(1990) 1799-1802.

[13] **M. Fielder.** Hankel and Loewer matrices Lin. Alg. Appm. 58(1984) 75-95.

[14] **F. R. Gantmacher et M. G. Krein.** Oscillatory matrices and Kernels, and small vibrations of machanical systemes, 2^e édition. GITTL, Moscou. 1950.

[15] **I. Gohberg, T. Kailath et V. Olshevsky.** Fast Gaussian elimination with partial pivoting for matrices with displacement structure Math. Comput. 64(1995) 1557-1576.

[16] **I. Gohberg et V. Olshevsky.** Fast algorithms with preprocessing for matrix-vector multiplication problems J. Complexity 10(1994) 1557-1576.

[17] **I. Gohberg et I. Koltracht.** on the inversion of Cauchy matrices, dans signal processing, Scatteing and Operator theory, and numarical methods. Proc. of the MTNS-89 (M.A.Kaashoek.J.H.van Schuppen et A.C.M.Ran) Birkhauser. Boston. MA. (1990) 381-392.

[18] **G.H.Golub et V. Olshevsky.** Pivoting for structured matrices preprint 1997.

[19] **G. H. Golub et C. F. VanLoan.** Mtrix computations, John Hopkins. Univ. Press, Baltimore, Third edition, 1996.

[20] **M.Gu.** Stable and efficient algorithms for structured systems of linear equations, SIAM J. Matrix Anal. Appl. 19(1998) 279-306.

[21] **G. Heining.** Inversion of generalized Cauchy matrices and other clacess of structured matrices linear Algrbra in Signal Processing, IMA VOL. MATH. APPL. 69(1994) 95-114.

[22] **G. Heining.** Generalized Cauchy- Vandermonde matrices, Lin. Alg. Appl. 270(1998) 45-77.

[23] **G. Heining, W. Hoppe et K. Rost.** Structured matrices interpolation and approximation problems Wissenchaftl. Zeitsschrift der TU Karl-marx-Stadt 31(1989) 196-202.

[24] **G. Heining, P. Jankowski et K. Rost.** Fast inversion of toeplitz-plus-Hankel matrices Numer. Math. 52(1988)665-682.

[25] **G. Heining et K. Rost.** Algebraic methods for Toeplitz-like matrices and operators Oper. Theory, 13(1984) 109-127.

[26] **N.J. Higham.** Accuracy and stability of Numerical algorithmes, SIAM. Philadelphia. 1996.

[27] **T. Kailath. S. Y. Kung et M. Morf.** Displacement ranks of matrices and linear equation. J. Math. Anal. 68(1979) 395-407.

[28] **T. Kailath et V. Olshevsky.** Displacement structure approach to Chebychev-Vandermonde and related matrices Integ. Rq. and Oper. Th. 22(1995) 65-92.

[29] **T. Kailath and A. H. Sayed.** Displacement structure : Theroy and applications SIAM review 37(1995) 297-386.

[30] **J. Makhoul.** Linear prediction : A tutorial review, Proq. IEEE 63(1975) 561-580.

[31] **L. Melkemi.** Confluent Vandermonde matrices using Sylvester's structure RR 98-16, ENS-Lyon Laboratoire LIP 1998.

[32] **L. Melkemi.** Contribution à l'étude des matrices structurées. Thèse de doctorat en Mathématique (2000).

[33] **L. Melkemi.** Fast QR factorization for confluent Vandermonde-like matrices and Chebycheb-Vandermonde-like matrices RR 98-20, ENS-Lyon Laboratoire LIP 1998.

[34] **L. Melkemi.** Structures de déplacement pour les matrices de vandermonde p-confluente C. R. Acad. Sci. Paris Série I(1999) 621-926.

[35] **L. Melkemi. A. Benaissa et R. Benacer.** Factorisation QR des matrices de Tchebychev-Vandermonde confluentes C. R. Acad. Sci. Paris Série I(2000) 147-152.

[36] **L.Melkemi et F.Rajah.** Block Gaussian eliminations for confluent Vandermonde matrices. Séminaire "Algorithmes numérique appliqués à l'industrie", Calais. Mai 2003.

[37] **M. Mokhtari et A. Mesbah.** Apprendre et maîtrise MATLAB versions 4 & 5 et SIMULINK.

[38] **M. Mokhtari et A. Mesbah.** MATLAB versions 5.2 & 5.3 et SIMULINK.

[39] **A. V. Oppenhein.** Applications of digital signal processing, Prentice-Hall. Inc Englewood cliffs, 1978.

[40] **V. Plan.** On computing with structured matrices, Math. comp.55(1990) 179-190.

[41] **Sylvie Fabre, Jean-Michel Morel, Yann Gousseau.** Notes de cours d'analyse, ENS Cachan premiére année 9 juillet 2002.