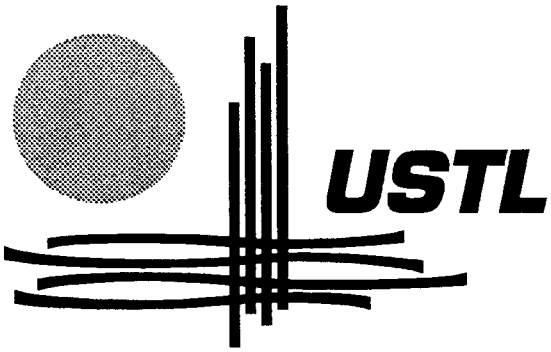
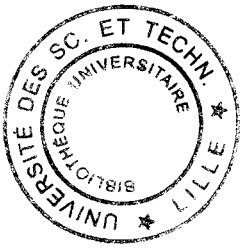




N° d'ordre : 2314

THESE

Nouveau régime

présentée à
L'UNIVERSITE DES SCIENCES ET TECHNOLOGIES DE LILLE

pour obtenir le titre de
DOCTEUR en MATHEMATIQUES

par
El Hassan AYACHOUR

APPLICATION DE LA BIORTHOGONALITE AUX METHODES DE PROJECTION

soutenue le premier octobre 1998 devant la commission d'examen

JURY

Président :	C. BREZINSKI, Professeur, UFF IEEA, USTL, Lille
Rapporteurs :	A. DRAUX, Professeur, INSA, Rouen
	A. BULTHEEL, Professeur, KUL, Leuven, Belgique
Examinateurs :	J. VAN ISEGHEM, Professeur, UFR Maths, USTL, Lille
	H. SADOK, Professeur, Université du Littoral, Calais

Laboratoire d' Analyse Numérique et d' Optimisation UFR IEEA - M3, USTL,
59655 Villeneuve d' Ascq - Cedex, France.

J'exprime toute ma gratitude au professeur Claude BREZINSKI, de l'Université des Sciences et Technologies de Lille et directeur du laboratoire d'Analyse Numérique et d'Optimisation, qui a bien voulu diriger mes travaux de recherche, m'a aidé de ses conseils et qui me fait l'honneur de présider ce jury.

Je suis tout particulièrement reconnaissant à Jeannette Van ISEGHEM, professeur à l'Université des Sciences et Technologies de Lille, de m'avoir suivi avec attention dans mes travaux et de m'avoir prodigué conseils et encouragements tout au long de la réalisation de ce travail.

Je remercie vivement le professeur Hassane SADOK, de l'Université du Littoral, tant pour son aide que pour les précieuses discussions que j'ai eues avec lui.

Mes remerciements vont également au professeur Bernard GERMAIN-BONNE, de l'Université des Sciences et Technologies de Lille, qui a bien voulu participer à la correction de ce travail, et qui ne m'a ménagé ni sa disponibilité ni sa patience.

Je suis sincèrement honoré de la présence au jury de Monsieur le Professeur André DRAUX de l'INSA de Rouen, ainsi que celle de Monsieur le Professeur Adhémar BULTHEEL de l'Université Catholique de Leuven. Qu'ils soient assurés de ma profonde reconnaissance pour avoir accepté d'être rapporteurs de cette thèse.

Je tiens à remercier tous les membres du laboratoire d'Analyse Numérique et d'Optimisation, notamment B. Beckermann, A. Messaoudi, M. Prevost, A.C. Matos, F. Van Iseghem, J.C. Fiorot, K. Jbilou, A. Salam, N. Revol, J.P. Chehab et particulièrement M. Heyouni et A. Essai.

Mes remerciements vont aussi à mes anciens enseignants M. Hamzaoui, M. Bouhlal, M. Boutalb, M. Haouari, M.A. Mesnaoui, T. Ghemires, D. Kharchaf, N. Mikou, ... qui ont contribué à ma formation.

Je remercie mes amis M. Charif, A. Yachou, A. Maati, F. Driouch, Y. Karzazi, Y. Ruichek, M. Mezrouai, M. Karrat, Y. Saidi, A. El Moussati, K. Dahbouri, J.B. Yogo, ... pour leur aide à la réalisation de l'après-thèse.

Que mes amis Nourdine, Yassin, Zaoui, Fayssal, Abdelrahman, Abdelrahim, Meziane, Fath, Rachid, Abdellah, Mostafa, Sindy, Tatiana, Vivianne, Oussama, Houssin ... soient assurés de ma grande reconnaissance pour leur présence.

Que mes amis nadoris Nouaman Aissami, Rachid Ouja, Mohammed Rochdi, Mimoun El Marssi, Hakim Chemlal, Oussama Chemlal, Saïd Naharat, Fouad Aissami, Abdelouahad Amraoui, Mohammed Azougar, Lahbib Zaalouk, Jamal Dehmej, Hassan Ajaji, ... ne soient pas oubliés.

Finalement, je remercie tous les membres de ma famille, surtout ma plus grande soeur, son mari et ma mère, qui ont contribué à leur façon à l'aide que j'ai reçue pour l'aboutissement de ce travail.

Résumé

Dans cette thèse, nous avons défini une méthode générale qui construit une paire de bases biorthogonales par rapport à une forme sesquilinéaire, non nécessairement hermitienne définie positive. La construction de bases biorthogonales nous a permi de déterminer de manière récursive une projection particulière bien définie, que nous avons appelée *projection biorthogonale*, d'un vecteur arbitraire sur un sous-espace sesquilinéaire régulier. Pour la résolution des systèmes d'équations linéaires, nous avons montré que de nombreuses méthodes connues comme GMRES et MINRES sont basées sur la projection biorthogonale de la solution, par rapport à une forme hermitienne définie positive, sur un sous-espace de Krylov.

Nous avons étudié le problème de la division par zéro dû à la singularité de certains polynômes orthogonaux formels. Afin d'éviter ce problème et au lieu d'utiliser une stratégie de "look-ahead", nous avons proposé d'employer une nouvelle méthode ALA qui consiste à remplacer ceux qui n'existent pas par des polynômes biorthogonaux. Nous avons donné trois mises en oeuvre principales de ALA que nous avons appliquées à la méthode de Lanczos, à l'approximation de Padé et à l'epsilon algorithme.

A la fin de cette thèse, pour la résolution d'un système d'équations linéaires, nous avons montré que l'utilisation d'un système augmenté hermitien avec un préconditionnement est recommandé en cas de stagnation des méthodes qui projettent la solution sur les sous-espaces de Krylov $K_i(A, r_0)$ pour $i = 1, 2, \ldots, n$, comme c'est le cas pour GMRES.

Mots clefs

Biorthogonalité, bases biorthogonales, forme sesquilinéaire, projection, espace de Krylov, polynômes orthogonaux, méthode de Lanczos, approximation de Padé, epsilon algorithme, système augmenté, préconditionnement, stagnation.

Abstract

In this thesis, we defined a general method which build a pair of biorthogonal bases with respect to a sesquilinear form, non necessarily Hermitian positive definite. The construction of biorthogonal bases allowed to determine recursively a particular well defined projection, that we called *biorthogonal projection*, of an arbitrary vector onto a regular sesquilinear subspace. For the solution of systems of linear equations, we proved that many known methods as GMRES and MINRES are based on the biorthogonal projection of the solution, with respect to a Hermitian positive definite form, onto a Krylov subspace.

We studied the breakdown problem due to the singularity of certain formal orthogonal polynomials. In order to avoid this problem and instead of using a look-ahead strategy, we proposed to use a new method ALA which consists of replacing those which do not exist by biorthogonal polynomials. We gave three principal implementations of ALA which we applied to the Lanczos method, to Padé approximation and to the epsilon algorithm.

At the end of this thesis, for the solution of systems of linear equations, we proved that using an augmented Hermitian system with a preconditioner is recommended when the methods which project the solution onto the Krylov subspaces $K_i(A, r_0)$ for $i = 1, 2, \ldots, n$ stagnate, as for GMRES.

Key words

Biorthogonality, biorthogonal bases, sesquilinear form, projection, Krylov space, orthogonal polynomials, Lanczos method, Padé approximation, epsilon algorithm, augmented system, preconditioner, stagnation.

Table des matières

Liste des notations

Nous donnons ici la liste des notations utilisées dans cette thèse.

$\mathbb{N}$	Espace des entiers naturels
$\mathbb{Z}$	Espace des entiers relatifs
$\mathbb{R}$	Espace des nombres réels
$\mathbb{C}$	Espace des nombres complexes
$\lfloor n \rfloor$	Partie entière de n
DPZ	Division Par Zéro
$PDPZ$	Presque Division Par Zéro
g^*	Conjuguée-transposée de la forme g
A^T	Transposée de A
A^*	Adjoint de A
$cond(A)$	conditionnement de la matrice A
$\overline{a}$	Conjugué du nombre complexe a
$\perp_g$	g-orthogonalité à droite
${}_g\perp$	g-orthogonalité à gauche
$< .,. >_k$	Produit hermitien de $\mathbb{C}^k$
$\|.\|_k$	Norme euclidienne de $\mathbb{C}^k$
$\|.\|_A$	A-norme ($\|z\|_A =< Az, z >_k$)
g_L	Restriction de g au sous-espace L
$\widehat{V}$	Espace dual de V
$\overline{\widehat{V}}$	Espace dual conjugué de V
$dim\ L$	Dimension de L
$ker(f)$	Noyau de f
$Im(f)$	Image de f
$L' \oplus L$	Somme directe de L' et L
$Vect(u_1, u_2, \ldots, u_k)$	Espace engendré par les vecteurs $u_1, u_2, \ldots, u_k$
$\mathbb{C}[X]$	Espace des polynômes à variable x

Introduction générale

Rien n'est dû au hasard.
Toute chose a une histoire,
et toute histoire a un passé.
Suis-je passé pour être présent dans votre futur?

La résolution des systèmes linéaires est un sujet classique de l'algèbre linéaire qui intervient dans de nombreux domaines et qui reste néanmoins toujours d'actualité. La modélisation des problèmes que nous rencontrons en physique, en mécanique, en électrotechnique et d'une façon générale dans l'ingénierie, conduit, éventuellement après une discrétisation, à la résolution des systèmes d'équations linéaires en dimension finie. Citons par exemple la discrétisation par la méthode des différences finies ou par la méthode des éléments finis d'un problème d'équations aux dérivées partielles.

Soit à résoudre le système d'équations linéaires

$$Ax = b, \tag{0.1}$$

où A est une matrice et b un vecteur de $\mathbb{C}^n$.

Pour une matrice A carrée régulière de dimension n, le calcul numérique d'une valeur approchée de la solution exacte $x = A^{-1}b$ se fait de deux manières différentes, l'une directe et l'autre itérative.

- ♣ Les méthodes directes consistent à factoriser la matrice A en un produit de matrices faciles à inverser. En pratique, la stabilité numérique d'une telle factorisation a une importance capitale. Les deux factorisations LU avec pivotage et QR de la matrice A sont les plus utilisées en raison de leur stabilité numérique. Cependant, dans le cas d'une matrice A de grande dimension n, ces deux factorisations nécessitent un stockage de l'ordre de $\mathcal{O}(n^2)$ et un coût algorithmique élevé de l'ordre de $\mathcal{O}(n^3)$.

♣ Contrairement aux méthodes directes, les méthodes itératives sont le plus souvent utilisées lorsque la matrice A est d'une grande dimension ($n > 1000$) et surtout si A est creuse. La programmation de ces méthodes ne modifie pas la structure de A et nécessite, en général, un stockage et un coût algorithmique convenables. En plus, ces méthodes itératives sont les mieux adaptées au calcul parallèle. Une classe de ces méthodes itératives est basée sur la minimisation d'une certaine norme de l'erreur sur un sous-espace de Krylov. La convergence des méthodes de cette classe vers une valeur approchée de la solution exacte est obtenue, en général, en un petit nombre d'itérations par rapport, bien sûr, à la taille globale de la matrice A. Parmi les éléments de cette classe, citons la méthode du gradient conjugué (CG : Conjugate Gradient method) due à Hestenes et Stiefel [70] qui est utilisée lorsque la matrice A est hermitienne définie positive ainsi que les méthodes de type CG suivantes :

- MINRES (MINimum RESidual method) et SYMMLQ (SYMMetric LQ) dues à Paige et Saunders [88] qui supposent seulement que la matrice A soit hermitienne. Notons que MINRES est mathématiquement équivalente à la variante CR (Conjugate Residual) de la méthode CG sauf qu'elle résout un problème aux moindres carrés obtenu grâce au processus hermitien de Lanczos [76] pour minimiser le résidu.

- La méthode CG appliquée respectivement aux équations normales $AA^*y = b$ avec $x = A^*y$ et $A^*Ax = A^*b$ nous donne respectivement les deux méthodes CGNE (Conjugate Gradient on Normal equations to minimize the Error) due à Craig [41] et CGNR (Conjugate Gradient on Normal equations to minimize the Residual) due à Hestenes et Stiefel [70]. Ces deux méthodes CGNE et CGNR ne supposent aucune condition sur A et minimisent respectivement la norme euclidienne de l'erreur et du résidu. Cependant, il est connu que la matrice AA^* (ou A^*A) peut être mal conditionnée, ce qui entraîne une convergence lente de ces méthodes.

- LSQR (Least Squares QR), due à Paige et Saunders [89], est mathématiquement équivalente à CGNR. Mais LSQR peut être considérée comme une bonne manière de mettre en oeuvre CGNR.

- USYMLQ (UnSYMmetric LQ) et USYMQR (UnSYMmetric QR) sont dues à Saunders, Simon et Yip [107]. Ces deux méthodes ont une caractéristique particulière qui consiste à projeter la solution du système sur une somme de deux sous-espaces de Krylov. Notons cependant que ces deux méthodes présentent un problème de division par zéro auquel nous ne pouvons remédier que par

un changement des vecteurs qui sont choisis initialement. Par la suite, nous adaptons l'abréviation DPZ qui signifie *division par zéro.*

Il est important de noter que CGNE, CGNR, LSQR, USYMLQ et USYMQR sont utilisées pour une matrice A quelconque. En général, sans préconditionnement, ces méthodes ont une convergence lente, ce qui a poussé les chercheurs à développer d'autres méthodes plus compétitives. Nous en citons ci-dessous quelques unes.

- ORTHOMIN est due à Vinsome [118]. Les résultats théoriques sur la convergence de cette méthode ont été analysés par Axelsson [3] et les auteurs de [48, 49]. Cette méthode est aussi connue sous le nom de GCR (Generalized Conjugate Residual method) dans [49].
- ORTHORES et ORTHODIR sont dues à Young et Jea [127]. ORTHOMIN, ORTHORES et ORTHODIR diffèrent seulement dans la façon de programmer le résidu. Mais il s'avère que ORTHOMIN est la plus stable numériquement.
- FOM (Full Orthogonalization Method), due à Saad [97], utilise le processus d'Arnoldi [1, 96, 97] et la condition d'orthogonalité de Petrov-Galerkin pour se ramener, à chaque itération, à un système linéaire dont la matrice est de Hessenberg. Cette matrice de Hessenberg peut être singulière, ce qui est considéré comme un handicap. Nous disons aussi que FOM souffre d'une DPZ causée par la condition de Petrov-Galerkin. FOM est aussi dite *méthode d'Arnoldi.*
- GMRES (Generalized Minimal RESidual method), due à Saad et Schultz [100], minimise le résidu et utilise le processus d'Arnoldi, ce qui lui permet d'échapper à la DPZ de la méthode FOM. GMRES consiste à se ramener grâce au processus d'Arnoldi à un problème de minimisation au sens des moindres carrés, qui se résout à l'aide d'une factorisation QR obtenue par les rotations de Givens. GMRES utilise une orthogonalisation complète du processus d'Arnoldi. Il est bien connu qu'une telle orthogonalisation nécessite un coût algorithmique élevé. C'est pourquoi Wu et Saad ont utilisé la technique d'orthogonalisation incomplète [98, 96] pour définir la méthode DQGMRES [103]. Une autre technique souvent utilisée est celle du redémarrage.
- BCG (Bi-Conjugate Gradient method), due à Lanczos [77] et rendue populaire par Fletcher [53], utilise le processus non hermitien de Lanczos et la condition de biorthogonalité de Petrov-Galerkin. BCG est une extension de la méthode du gradient conjugué au cas non hermitien. Malheureusement, sa convergence peut être irrégulière et comme FOM, BCG souffre d'une DPZ qui correspond à la singularité de la matrice de Hessenberg. BCG souffre d'une autre DPZ qui

est celle du processus non hermitien de Lanczos. Cette dernière DPZ peut être évitée grâce aux stratégies de *"look-ahead"* connues [90, 56, 83]. La convergence irrégulière de BCG peut être rendue régulière grâce à la méthode MRS (Minimal Residual Smoothing) due à Schönauer [110] et qui a été étudiée en détail dans [121, 122, 123].

- QMR (Quasi Minimal Residual method) est due à Freund dans le cas symétrique [55] et à Freund et Nachtigal dans le cas non symétrique [54]. QMR utilise le processus non hermitien de Lanczos pour se ramener à un problème de "quasi-minimisation" qui évite une éventuelle DPZ due à la singularité de la matrice de Hessenberg de la méthode BCG. La DPZ due au processus non hermitien de Lanczos peut être évitée de la même façon que dans BCG par les stratégies de "look-ahead".
- CMRH (Changing Minimal Residual method based on the Hessenberg process), due à Sadok [105], utilise la technique de " quasi-minimisation" du résidu comme pour la méthode QMR et le processus de Hessenberg [73, 124].
- CGM (Conjugate Gradient Multiplied) est une classe de méthodes, introduite simultanément par Brezinski [14] et Gutknecht [64]. Ce type de méthodes est aussi connu sous le nom de *"Lanczos-type product methods"*(LTPM). Les deux méthodes CGS (Conjugate Gradient Squared) due à Sonneveld [112] et Bi-CGSTAB due à Van Der Vorst [116] sont de type CGM. Cette classe CGM est basée sur la méthode de Lanczos et donc ce type de méthodes a aussi un problème de DPZ.
- MRSe (Minimal Residual Seminorm method), due à Heyouni et Sadok [71], est basée sur une semi-minimisation du résidu. Les résultats théoriques et numériques de cette méthode ont été étudiés en détail dans [72].

La plupart des travaux qui ont été fait dans le domaine de la résolution des systèmes linéaires utilisent la notion de projection orthogonale par rapport à une forme hermitienne. Ici, nous n'allons pas nous restreindre au cas hermitien, mais nous allons considérer un cas plus général, celui d'une forme sesquilinéaire qui n'est pas nécessairement non dégénérée, et définir ainsi la notion de la projection biorthogonale. Ce cas général va nous permettre de retrouver les méthodes directes comme des méthodes de projection biorthogonale par rapport à une forme sesquilinéaire, en général non hermitienne. Par exemple, nous allons voir que la méthode de Gauss est une méthode de projection biorthogonale, c'est-à-dire basée sur la construction de deux bases biorthogonales et qui consiste à projeter le vecteur b de $Ax = b$ sur des sous-espaces engendrés par des éléments de ces deux bases. Tout ceci

sera le sujet du **Chapitre 1.**

En 1892, le mathématicien français Henri Padé (1863-1953) a étudié en détail dans sa thèse [86], sous la direction de Charles Hermite (1821-1901), des approximants qui portent maintenant son nom : Approximants de Padé. Mais ces approximants avaient déjà été obtenus pour des cas particuliers en 1758 par Johann Heinrich Lambert (1728-1777) et aussi en 1776 par Joseph Louis Lagrange (1736-1813) suivant l'approche des fractions continues. Une étude historique bien détaillée a été faite sur ce sujet dans [13]. L'utilisation de ces approximants de Padé dans divers domaines leur a donné une importance capitale. Il y a plusieurs manières pour calculer un approximant de Padé. Nous pouvons directement résoudre un système linéaire pour trouver l'approximant de Padé que nous cherchons. Cette résolution de système linéaire peut se faire par une méthode itérative qui minimise l'erreur suivant une certaine norme. Dans ce travail, nous allons plutôt nous intéresser aux relations qui nous permettent de calculer un approximant de Padé quelconque à partir d'autres approximants de Padé. C'est-à-dire, en d'autres termes, avoir des relations de récurrence qui nous permettent de suivre un chemin arbitraire dans la table de Padé. Plusieurs auteurs ont travaillé sur ce sujet. Baker a donné dans [6] un algorithme pour se déplacer dans la table de Padé le long de deux antidiagonales adjacentes en escaliers montants. L'algorithme de Pindor proposé dans [93] utilise le même chemin que celui de Baker mais dans les deux sens : en montant et en descendant. Watson a donné dans [120] un algorithme pour se déplacer sur deux diagonales adjacentes en escaliers descendants. Brezinski a généralisé la méthode de Gragg [61] dans [9] en utilisant une méthode de bordage de la résolution des systèmes d'équations, afin de se déplacer le long d'une diagonale en descendant. D'autres méthodes basées sur la théorie des fractions continues peuvent être trouvées dans [125], malheureusement ces méthodes nécessitent la connaissance de tout le tableau *qd* qui s'obtient à partir de l'algorithme *qd*. L'algorithme *qd* (quotient-difference algorithm) est dû à Rutishauser [95]. Henrici a montré dans [69] que la forme progressive de *qd* est la plus stable. Toutes ces relations de récurrence sont retrouvées dans le cadre unifié de la théorie des polynômes orthogonaux formels [10].

Dans le cas général d'une table de Padé non normale (c.à.d qui présente des blocs), tous les algorithmes que nous avons cité ci-dessus peuvent heurter des blocs et avoir ainsi une défaillance. Cette défaillance est liée au problème de la DPZ dans la programmation des polynômes orthogonaux formels. Elle n'est pas due à l'algorithme utilisé mais plutôt à la série étudiée. Il y a plusieurs méthodes pour éviter une telle défaillance, par exemple celle qui consiste à se déplacer sur la frontière des blocs, étudiée par Graves-Morris dans

[63], ou celles que nous allons citer au deuxième chapitre. Mais ces méthodes nécessitent le calcul de plusieurs éléments d'une certaine table, par exemple celle de *qd* ou celle des déterminants de Hankel. L'algorithme donné par Draux et Van Ingelandt dans [45] est le seul qui ne nécessite le calcul global d'aucune table et qui permet de se déplacer dans la table de Padé suivant un chemin arbitraire.

Si nous regardons soigneusement ce qui a été fait jusqu'à présent sur l'étude des blocs d'une table de Padé, notamment celle des polynômes orthogonaux, alors nous allons constater que pour calculer les approximants de Padé nous passons directement d'un côté à l'autre d'un bloc, ce qui équivaut à faire un saut. Nous pouvons dire alors que chaque technique qui consiste à sauter les blocs est un "look-ahead", comme dans la méthode de Lanczos. C'est pourquoi dans le **Chapitre 2**, nous allons introduire la méthode ALA dont l'idée originale est donnée dans [4]. ALA apporte un remède au problème de la DPZ et remplace les stratégies de "look-ahead". Pour les approximants de Padé, ALA va nous permettre de construire une sorte de pont pour chaque bloc et d'éviter ainsi de sauter en se servant des relations de récurrence seulement à trois termes. Plus précisément nous allons introduire, à l'intérieur des blocs, des approximants générés par des polynômes biorthogonaux qui vérifient des relations de récurrence à trois termes et dont les degrés ne se succèdent pas. Les propriétés de ces polynômes biorthogonaux permettent de rejoindre par une suite de récurrence à trois termes les points intéressants de la table de Padé. Ce qui nous pousse à dire que la présence d'un bloc dans la table de Padé n'est plus guère un problème ou une difficulté. Au contraire, elle favorise les approximants qui se trouvent au Nord et à l'Ouest d'un bloc. Le chapitre 2 traite, en plus de l'approximation de Padé, de la méthode de Lanczos et de l'epsilon algorithme.

Dans le **Chapitre 3**, nous introduisons la méthode MINERES(λ) (MINimum Expanded RESidual method). Cette méthode est une généralisation des méthodes MINRES et LSQR. De plus, elle a l'avantage d'avoir une bonne convergence lorsqu'elle est combinée avec un bon préconditionnement et elle est facilement parallélisable. MINERES(λ) consiste à appliquer MINRES à un système hermitien étendu qui dépend de la constante λ. La raison principale pour laquelle nous avons introduit cette méthode est le problème de la stagnation ; en effet, les méthodes qui projettent la solution de $Ax = b$ sur les sous-espaces de Krylov $K_i(A, r_0)$, engendrés par $r_0, Ar_0, \ldots, A^{i-1}r_0$, peuvent stagner lorsque la matrice A est non hermitienne. En revanche, nous allons montrer que MINERES(λ) n'a aucun problème de stagnation. Nous allons aussi donner des exemples numériques qui montrent cet avantage de MINERES(λ).

Chapitre 1

Projection biorthogonale

1.1 Introduction

Diverses méthodes directes et itératives pour résoudre un système d'équations linéaires sont des méthodes de projection. Dans ce chapitre, nous allons définir une méthode générale de biorthogonalisation par rapport à une forme sesquilinéaire quelconque g, que nous appelons GBO (General BiOrthogonalization). Cette méthode GBO donne lieu à deux processus GBOR et GBOD. Le processus GBOR généralise plusieurs processus connus, en particulier, le processus de Gram-Schmidt, le processus de Lanczos [77] et celui de Hessenberg généralisé [124]. GBO va nous permettre de définir par la suite une méthode générale de projection biorthogonale qui dépend du choix de certains paramètres. Selon le choix de ces paramètres, nous allons retrouver plusieurs méthodes connues de résolution des systèmes linéaires.

Dans les sections suivantes, nous désignons par E et F deux espaces vectoriels sur le corps des nombres complexes $\mathbb{C}$ de dimensions respectives m et m'. Dans le cas où E et F sont de dimensions infinies, nous posons par convention $m = m' = \infty$. Dans ce travail, nous nous intéressons aux espaces vectoriels qui admettent des bases finies ou dénombrables.

1.2 Formes sesquilinéaires

Pour résoudre n'importe quel système linéaire, nous utilisons souvent des méthodes pour orthogonaliser ou orthonormaliser une base donnée comme par exemple le processus de Gram-Schmidt [97] ou celui de Lanczos dans le cas hermitien. Cette orthonormalisation dépend d'un produit scalaire hermitien. Ici, au lieu d'un produit scalaire, nous utilisons la notion d'une forme sesquilinéaire quelconque, c'est-à-dire éventuellement dégénérée et

non hermitienne.

Nous commençons tout d'abord par étudier les espaces munis de formes sesquilinéaires. Signalons que ceux munis de formes bilinéaires hermitiennes ont été étudiés par plusieurs auteurs, et particulièrement par Bognár [8] et Scharlau [109].

Définition 1.1 *Nous appelons forme sesquilinéaire (ou application sesquilinéaire) sur* $E \times F$ *toute application* $g : E \times F \longmapsto \mathbb{C}$ *vérifiant les conditions suivantes :*

- $\forall z \in F$ *l'application* $y \in E \longmapsto g(y,z)$ *est linéaire,*
- $\forall y \in E$ *l'application* $z \in F \longmapsto g^*(z,y)$ *est linéaire,*

où la conjuguée-transposée g^* *de la forme sesquilinéaire* g *est définie sur* $F \times E$ *par*

$$g^*(y,z) = \overline{g(z,y)}, \tag{1.1}$$

la barre désignant le conjugué d'un nombre complexe.

La paire $(E \times F, g)$ est appelée *espace sesquilinéaire.* Lorsque $E = F$, cet espace sesquilinéaire est désigné par la paire (E,g).

Remarque 1.1 Dans la définition d'une forme sesquilinéaire, nous pouvons remplacer l'application $z \longmapsto \overline{z}$ par un automorphisme involutif quelconque de $\mathbb{C}$ et tous les résultats que nous allons énoncer restent valables. Ainsi, comme la coïncidence $z \longmapsto z$ est un automorphisme involutif de $\mathbb{C}$, une forme bilinéaire sera considérée, par la suite, comme un cas particulier d'une forme sesquilinéaire.

D'après la définition, il est clair que g^* de la forme sesquilinéaire g est aussi sesquilinéaire. Si la condition $g = g^*$ est vérifiée, alors la forme g est dite *hermitienne* [109]. Une forme hermitienne est *définie positive* (respectivement *semi-définie positive*) si pour tout $y \in E, g(y,y) > 0$ (respectivement $g(y,y) \geq 0$).

Définition 1.2 *Un vecteur* $y \in E$ *est g-orthogonal à gauche au vecteur* $z \in F$ *si* $g(y,z) = 0$. *Nous disons aussi dans ce cas que* z *est g-orthogonal à droite au vecteur* y *et nous notons* $y \ {}_g\!\perp z$, $z \perp_g y$.

Soient L et L' deux sous-espaces respectifs de E et F. Nous désignons par

$$L^{\perp_g} = \{z \in F / \ \forall y \in L, \ g(y,z) = 0\}$$

le sous-espace g-orthogonal à droite à L et par

$$L'^{g\perp} = \{y \in E/ \ \forall z \in L', \ g(y,z) = 0\}$$

le sous-espace g-orthogonal à gauche à L'.
Grâce à (1.1), nous avons les deux égalités suivantes

$$L'^{g\perp} = L'^{\perp_{g^*}} \quad \text{et} \quad L^{\perp_g} = L^{g^*\perp}. \tag{1.2}$$

1.2.1 Notations matricielles

Pour tout k fini ou infini, le produit hermitien $< .,. >_k$ (sous réserve qu'il existe) de deux vecteurs $y = (y_i)_{i=1,\ldots,k}$ et $z = (z_i)_{i=1,\ldots,k}$ de $\mathbb{C}^k$ est défini par

$$< y, z >_k = \sum_{i=1}^{k} y_i \overline{z_i},$$

que nous écrivons souvent sous forme d'un produit de matrices $< y, z >_k = z^* y$. Si k est fini ($k < \infty$), alors $< .,. >_k$ est une forme hermitienne définie positive associée à la norme euclidienne de $\mathbb{C}^k$ définie par

$$\|y\|_k = < y, y >_k^{1/2}.$$

L'indice k est omis s'il n'y a pas d'ambiguité.
Considérons une base finie ou dénombrable $ß = \{u_1, u_2, \ldots\}$ de l'espace vectoriel E et une autre base $ß' = \{u'_1, u'_2, \ldots\}$ de F. Si g est une forme sesquilinéaire sur $E \times F$, alors

$$G = G_{g,ß,ß'} = (g(u_i, u'_j))_{i,j} = (G_{i,j})_{i,j}$$

est appelée la matrice de g par rapport à $ß$ et $ß'$. Ainsi, il est clair que la matrice de g^* est la conjuguée-transposée G^* de G. Par conséquent, si g est hermitienne, alors G est aussi hermitienne.
Pour chaque $y = \sum_{i=1}^{m} y_i u_i$ de E et $z = \sum_{i=1}^{m'} z_i u'_i$ de F nous avons

$$\begin{aligned} g(y,z) &= \begin{pmatrix} y_1 \\ y_2 \\ \vdots \end{pmatrix}^t \begin{pmatrix} g(u_1,u'_1) & g(u_1,u'_2) & \ldots \\ g(u_2,u'_1) & g(u_2,u'_2) & \ldots \\ \vdots & \vdots & \ddots \end{pmatrix} \begin{pmatrix} \overline{z_1} \\ \overline{z_2} \\ \vdots \end{pmatrix} \\ &= y^T G \overline{z} = < G^T y, z >_{m'} = < y, \overline{G} z >_m \end{aligned} \tag{1.3}$$

où nous identifions y et z avec les vecteurs colonnes appropriés qui contiennent leurs composantes respectivement par rapport aux deux bases $ß$ et $ß'$.
Notons que l'espace $Sq(E \times F, \mathbb{C})$ de toutes les formes sesquilinéaires est isomorphe à l'espace $\mathbb{C}^{m \times m'}$ de toutes les matrices rectangles ou infinies $m \times m'$. Cet isomorphisme nous permet d'étudier $Sq(E \times F, \mathbb{C})$ à travers $\mathbb{C}^{m \times m'}$.

1.2.2 Espaces réguliers

Commençons par rappeler la définition d'un espace régulier.

Définition 1.3 *Un espace sesquilinéaire $(E \times F, g)$ est régulier (ou non dégénéré, ou non singulier) si $F^{g\perp} = \{0\}$ ou $E^{\perp g} = \{0\}$.*

Un espace sesquilinéaire non régulier est appelé *singulier*. La relation (1.3) nous permet de déduire le théorème suivant

Théorème 1.1 *Les propriétés suivantes sont équivalentes.*

1. *L'espace sesquilinéaire $(E \times F, g)$ est régulier.*
2. *Les colonnes ou les lignes de la matrice G de g (par rapport à deux bases arbitraires de E et F) sont linéairement indépendantes.*
3. *L'espace sesquilinéaire $(F \times E, g^*)$ est régulier.*

Si de plus $m < \infty$ et $m' < \infty$, alors les trois propriétés précédentes sont équivalentes aux deux propriétés suivantes.

4. *Le sous-espace $E^{\perp g}$ est de dimension* $\max\{m' - m, 0\}$.
5. *Le sous-espace $F^{g\perp}$ est de dimension* $\max\{m - m', 0\}$.

Si $m = m' < \infty$, alors nous pouvons facilement voir que les deux sous-espaces $F^{g\perp}$ et $E^{\perp g}$ sont réduits à $\{0\}$ si au moins l'un des deux l'est. Aussi, dire que $(E \times F, g)$ est régulier revient à dire que G est inversible.
Pour deux sous-espaces L de E et L' de F, nous notons $g_{L\times L'}$ la restriction de g à $L \times L'$. $(L \times L', g_{L\times L'})$ est appelé sous-espace sesquilinéaire de $(E \times F, g)$. Ce sous-espace est souvent noté (L, g_L) dans le cas où $E = F$ et $L = L'$ avec $g_L = g_{L\times L}$. Si $(L \times L', g_{L\times L'})$ est régulier, alors $L \times L'$ est appelé sous-espace régulier de E.

Donnons maintenant quelques notations et définitions relatives à un espace vectoriel V qui admet une base finie ou dénombrable. Nous désignons par *dim* V la dimension de V. Une application $s : V \longmapsto \mathbb{C}$ est appelée *semi-linéaire*, si $\overline{s}$ est linéaire [34]. Par exemple, la forme g est semi-linéaire par rapport à la seconde variable. Notons $\widehat{V}$ l'espace *dual* de V (l'espace vectoriel de toutes les applications linéaires $V \longmapsto \mathbb{C}$). L'ensemble de toutes les applications semi-linéaires est un espace vectoriel appelé le dual conjugué de V, noté

$\overline{\widehat{V}}$. Pour chaque base $\aleph = \{v_1, v_2, \ldots\}$ de V, nous désignons par $\widehat{\aleph} = \{\widehat{v}_1, \widehat{v}_2, \ldots\}$ la base duale de $\widehat{V}$ associée à $\aleph$ et définie par

$$\widehat{v}_i(v_j) = \delta_{ij} \text{ (symbole de Kronecker).}$$

Alors, il est facile de voir que $\overline{\widehat{\aleph}} = \{\overline{\widehat{v}}_1, \overline{\widehat{v}}_2, \ldots\}$ est une base de $\overline{\widehat{V}}$, appelée base duale conjuguée.

Soit l'application $h : V \times \overline{\widehat{V}} \longmapsto \mathbb{C}$ qui fait correspondre le nombre complexe $\psi(x)$ à tout couple (x, ψ) de $V \times \overline{\widehat{V}}$. Nous pouvons facilement voir que h est une forme sesquilinéaire, ce qui nous permet de définir le sous-espace $L^{\perp_h}$ de $\overline{\widehat{V}}$ h-orthogonal au sous-espace L de V et le sous-espace $L'^{\perp_h}$ de V h-orthogonal au sous-espace L' de $\overline{\widehat{V}}$. D'une manière analogue, considérons l'application

$$\begin{array}{rccc} f: & \overline{\widehat{V}} \times \overline{\overline{\widehat{V}}} & \longmapsto & \mathbb{C} \\ & (\psi, \phi) & \longmapsto & \phi(\psi) \end{array}$$

qui fait correspondre le nombre complexe $\phi(\psi)$ à tout couple (ψ, ϕ) de $\overline{\widehat{V}} \times \overline{\overline{\widehat{V}}}$.

Lemme 1.1 *Si V est de dimension finie, alors l'application linéaire canonique*

$$\begin{array}{rccc} I_V: & V & \longmapsto & \overline{\overline{\widehat{V}}} \\ & y & \longmapsto & \phi_y : \ \psi \in \overline{\widehat{V}} \longmapsto \psi(y) \in \mathbb{C} \end{array}$$

est un isomorphisme et nous avons $I_V(L^{h\perp}) = L^{\perp_f}$ pour tout sous-espace L de $\overline{\widehat{V}}$.

Preuve :

La propriété $I_V(L^{h\perp}) = L^{\perp_f}$ est une conséquence du fait que I_V est un isomorphisme. Il est simple de prouver que I_V est un isomorphisme. En effet, elle est similaire à celle donnée par exemple dans [30] où il suffit de remplacer l'espace dual $\widehat{V}$ par l'espace dual conjugué $\overline{\widehat{V}}$. ∎

Si nous identifions V et $\overline{\overline{\widehat{V}}}$ à l'aide de I_V, alors $L^{h\perp}$ et $L^{\perp_f}$ deviennent un même sous-espace de V.

Théorème 1.2 *Supposons que V est de dimension finie k. Pour tout sous-espace L de V nous avons*

$$dim\ L + dim\ L^{\perp_h} = k.$$

Pour tout sous-espace L' de $\widehat{V}$, nous avons

$$dim\ L' + dim\ L'^{h\perp} = k.$$

Preuve :

Tout d'abord, notons que la seconde propriété de ce théorème se déduit de la première en utilisant le Lemme 1.1. Nous allons donc montrer uniquement la première propriété. Si $L = V$, il est est évident que $L^{\perp h} = \{0\}$. De même, si $L = \{0\}$, il est évident que $L^{\perp h} = \widehat{V}$. Il reste à examiner le cas où $dim\ L = i \in \{1, 2, \ldots, k-1\}$ avec $k \geq 2$. Soit $\{y_1, y_2, \ldots, y_i\}$ une base de L que nous complétons en une base $\aleph = \{y_1, y_2, \ldots, y_k\}$ de V. La base duale conjuguée de $\widehat{V}$ associée à $\aleph$ est $\widehat{\aleph} = \{\widehat{y}_1, \widehat{y}_2, \ldots, \widehat{y}_k\}$.
Une forme linéaire ψ, écrite dans la base duale conjuguée sous la forme $\psi = \sum_{j=1}^{k} \alpha_j \widehat{y}_j$, est un élément de $L^{\perp h}$ si et seulement si $h(y_j, \psi) = \psi(y_j) = 0$ pour $j = 1, 2, \ldots, i$, c'est-à-dire tout simplement que $\alpha_j = 0$ pour $j = 1, 2, \ldots, i$. D'où $L^{\perp h} = Vect(\widehat{y}_{i+1}, \widehat{y}_{i+2}, \ldots, \widehat{y}_k)$ est de dimension $k - i$. ∎

Soient ß et ß′ deux bases respectives de E et F. Considérons maintenant trois transformations $\widehat{g}\colon F \longmapsto \widehat{E}$, $\overline{\widehat{g}}\colon E \longmapsto \overline{\widehat{F}}$ et $\overline{\widehat{g}}^*\colon F \longmapsto \overline{\widehat{E}}$ définies par $\widehat{g}(z)\colon y \longmapsto g(y,z)$, $\overline{\widehat{g}}(y)\colon z \longmapsto g(y,z)$ et $\overline{\widehat{g}}^*(z)\colon y \longmapsto \overline{g(y,z)}$ pour tout $y \in E$ et $z \in F$. Il est évident que $\overline{\widehat{g}}$ et $\overline{\widehat{g}}^*$ ainsi définies sont linéaires. L'une des propriétés de cette construction est l'égalité $\overline{\widehat{g}}^* = \overline{\widehat{g^*}}$. La matrice de $\overline{\widehat{g}}$ par rapport aux bases ß, $\widehat{ß}'$ est exactement la matrice G de g par rapport à ß et ß′. Et comme nous venons de voir que $\overline{\widehat{g}}^* = \overline{\widehat{g^*}}$, alors la matrice de $\overline{\widehat{g}}^*$ est G^*.
Ceci montre que $(E \times F, g)$ est régulier précisément lorsque au moins l'une des deux transformations linéaires $\overline{\widehat{g}}$ et $\overline{\widehat{g}}^*$ est injective.

Théorème 1.3 *Soient deux sous-espaces L de E et L' de F de dimensions finies.*

i/ Si $m' < \infty$, alors $dim\ L + dim\ L^{\perp_g} = m' + dim\ F^{g\perp} \cap L$.

ii/ Si $m < \infty$, alors $dim\ L' + dim\ L'^{g\perp} = m + dim\ E^{\perp_g} \cap L'$.

iii/ $L' \subset (L'^{g\perp})^{\perp_g}$ et $L \subset (L^{\perp_g})^{g\perp}$.

Preuve :

i/ est tout simplement une conséquence de la propriété très connue d'une application linéaire $f: L \longmapsto \widehat{\overline{F}}$ suivante

$$dim\ Ker(f) + dim\ Im(f) = dim\ L,$$

où $Im(f)$ et $Ker(f)$ sont respectivement l'image et le noyau de f. Il suffit de remplacer f par la restriction de $\overline{\overline{g}}$ à L. Ce qui nous donne $Ker(f) = F^{\perp_g} \cap L$. Le Théorème 1.2 implique que $dim\ Im(f) = dim\ F - dim\ L^{\perp_g}$ car $L^{\perp_g} = Im(f)^{\perp_h}$. D'où le résultat de i/. D'après les égalités de (1.2), il est clair que *ii/* n'est qu'une conséquence de *i/*. Tandis que la preuve de *iii/* est évidente. ∎

Corollaire 1.1 *Soient deux sous-espaces L de E et L' de F. Supposons que $(E \times F, g)$ soit un espace régulier avec $m' < \infty$ et $m < \infty$.*

i/ Si $m \leq m'$, alors $dim\ L + dim\ L^{\perp_g} = m'$.

ii/ Si $m' \leq m$, alors $dim\ L' + dim\ L'^{\perp_g} = m$.

iii/ Si $m = m'$, alors $L' = (L'^{\perp_g})^{\perp_g}$ et $L = (L^{\perp_g})^{\perp_g}$.

Preuve:

En utilisant le théorème précédent sous l'hypothèse que $(E \times F, g)$ est régulier, le résultat de ce corollaire est immédiat. Nous pouvons aussi proposer une preuve similaire à celle donnée dans [34]. ∎

Théorème 1.4 *Tout sous-espace régulier $L \times L'$ de $(E \times F, g)$ avec $dim\ L = l < \infty$ et $dim\ L' = l' < \infty$ a les propriétés suivantes :*

i/ $dim\ L \cap L'^{\perp_g} = \max\{l - l', 0\}$ et $dim\ L' \cap L^{\perp_g} = \max\{l' - l, 0\}$.

ii/ Si $l' \leq l$, alors $L + L'^{\perp_g} = E$.

iii/ Si $l \leq l'$, alors $L' + L^{\perp_g} = F$.

Preuve:

$L \times L'$ est supposé régulier, donc d'après le Théorème 1.1 nous avons le résultat de *i/*. Soit $y \in E$, posons $f = \overline{\overline{g}}(y)|_{L'}$ qui désigne la restriction de $\overline{\overline{g}}(y)$ à L'. Si nous supposons

que $l' \leq l$, alors $\widehat{g_{L\times L'}}$ est surjective puisque $L \times L'$ est régulier. Par conséquent, il existe $z \in L$ tel que

$$f = \widehat{g_{L\times L'}}(z) = \overline{\widehat{g}}(z)|_{L'}.$$

Ainsi pour tout $t \in L'$,

$$g(y,t) = \overline{\widehat{g}}(y)(t) = f(t) = g(z,t).$$

Alors, nous pouvons écrire $y = z + (y - z)$ où $z \in L$ et $y - z \in L'^{g\perp}$. Ce qui prouve que $L + L'^{g\perp} = E$. La propriété $\overline{\widehat{g}}^* = \widehat{\overline{g^*}}$ que nous avons vu précédemment nous permet de conclure pour la seconde somme $L' + L^{\perp g} = F$ de *iii/*. ∎

En supposant, dans le théorème précédent, que les deux sous-espaces L et L' ont la même dimension $l = l' < \infty$, nous obtenons comme conséquence le corollaire suivant.

Corollaire 1.2 *Pour tout sous-espace régulier $L \times L'$ de $(E \times F, g)$ vérifiant dim L = dim L', nous avons*

$$L' \oplus L^{\perp g} = F \text{ et } L \oplus L'^{g\perp} = E.$$

1.2.3 Base orthogonale et biorthogonale

Soit $D = \{p_1, p_2, \ldots, p_l\}$ un ensemble de vecteurs de E avec l un entier qui peut prendre une valeur infinie (c.à.d $l = \infty$) pour désigner l'ensemble dénombrable $\{p_1, p_2, \ldots\}$. De même soit $Q = \{q_1, q_2, \ldots, q_{l'}\}$ un ensemble de vecteurs de F.

Définition 1.4 *Si $g(p_i, q_j) = 0$ pour $i \neq j$, alors nous disons que D est g-biorthogonal à Q à gauche et que Q est g-biorthogonal à D à droite. Si de plus $g(p_i, q_i) = 1$ pour $i = 1, \ldots, \min\{l, l'\}$, alors D est g-biorthonormal à Q à gauche et Q est g-biorthonormal à D à droite.*

En l'absence de toute confusion, nous dirons tout simplement que les deux familles D et Q sont g-biorthogonales ou g-biorthonormales.

Lemme 1.2 *i/ Si D et Q sont g-biorthogonales et vérifient la condition*

$$g(p_i, q_i) \neq 0 \quad \text{pour} \quad i = 1, \ldots, \min\{l, l'\}, \tag{1.4}$$

alors $\{p_1, p_2, \ldots, p_k\}$ et $\{q_1, q_2, \ldots, q_k\}$ sont deux familles libres avec $k = \min\{l, l'\}$.

ii/ Si $L \times L'$ est un sous-espace régulier de $(E \times F, g)$ et si D et Q sont deux bases g-biorthogonales respectives de L et L', alors D et Q vérifient la condition (1.4).

Preuve :

i/ Supposons que la famille $\{p_1, p_2, \ldots, p_k\}$ soit liée ; il existe alors un élément p_{i_0} qui s'écrit comme combinaison linéaire des autres. En utilisant le fait que D et Q sont g-biorthogonales, nous aurons $g(p_{i_0}, q_{i_0}) = 0$. Or ceci est en contradiction avec la condition (1.4).
D'une manière analogue, nous montrons que $\{q_1, q_2, \ldots, q_k\}$ est libre.

ii/ est une conséquence de la définition d'un espace régulier. En effet, s'il existe un indice $i_0 \in \{1, \ldots, \min\{l, l'\}\}$ tel que $g(p_{i_0}, q_{i_0}) = 0$, alors $p_{i_0} \in L'^{g\perp}$ et $q_{i_0} \in L^{\perp g}$, ce qui est en contradiction avec l'hypothèse que $L \times L'$ est un sous-espace régulier. ■

Si D et Q sont deux bases g-biorthogonales respectives de L et L' vérifiant la condition (1.4), alors la normalisation suivante de p_i et q_i

$$\begin{cases} g(p_i, q_i) = z^2 \\ p_i' = z^{-1} p_i \\ q_i' = \overline{z}^{-1} q_i \end{cases} \tag{1.5}$$

associe à D et Q deux bases g-biorthonormales $D' = \{p_1', p_2', \ldots, p_l'\}$ et $Q' = \{q_1', q_2', \ldots, q_{l'}'\}$.

Théorème 1.5 *Si $L \times L'$ est un sous-espace régulier de $(E \times F, g)$ avec L ou L' de dimension finie, alors il existe deux bases, l'une de L et l'autre de L', qui sont g-biorthogonales.*

Preuve :

Nous allons donner une démonstration constructive. Soient $N = \{v_1, v_2, \ldots, v_l\}$ et $D = \{w_1, w_2, \ldots, w_{l'}\}$ deux bases respectives de L et L'. Nous pouvons obtenir, à partir de N et D, deux bases g-biorthogonales $M = \{p_1, p_2, \ldots, p_l\}$ de L et $Q = \{q_1, q_2, \ldots, q_{l'}\}$ de L' en biorthogonalisant par rapport à g les deux bases N et D. D'après le Lemme 1.2, M et Q doivent vérifier la condition (1.4). La méthode qui consiste à construire les deux bases M et Q sera appelée la méthode générale de biorthogonalisation par rapport à g que nous noterons brièvement GBO. Cette méthode donne naissance à deux processus nouveaux GBOR et GBOD. La lettre D dans GBOD signifie le mot direct car le processus GBOD que nous allons donner plus tard utilise à chaque étape tous les éléments des deux bases N et D. La lettre R dans GBOR signifie le mot récursif car, dans le processus GBOR que nous allons décrire ci-dessous, les éléments des deux bases N et D sont donnés

récursivement, c'est-à-dire au cours de l'application de GBOR. Commençons maintenant par décrire ce nouveau processus GBOR.

Etape $k = 1$: (Initialisation)

$L \times L'$ est supposé régulier, donc il existe deux vecteurs v_{i_1} et w_{j_1} pour lesquels $g(v_{i_1}, w_{j_1}) \neq 0$. Nous définissons récursivement deux permutations σ de $\{1, 2, \ldots, l\}$ et θ de $\{1, 2, \ldots, l'\}$, c'est-à-dire qu'à chaque étape k nous donnons les valeurs de $\theta(k)$ et $\sigma(k)$. Pour commencer, nous posons $\sigma(1) = i_1$ et $\theta(1) = j_1$. Notons que i_1 et j_1 peuvent être choisis de différentes manières suivant l'espace $E \times F$ sur lequel nous travaillons, par exemple voir [4] où σ est prise égale à l'identité et $\theta(1)$ est le plus petit indice j_1 tel que $g(v_1, w_{j_1}) \neq 0$. Nous normalisons les deux vecteurs $v_{\sigma(1)}$ et $w_{\theta(1)}$ pour obtenir respectivement les vecteurs normalisés p_1 et q_1.

Etape $k = 2$:

Posons $L_1 = Vect(v_{\sigma(1)})$ (le sous-espace de L engendré par $v_{\sigma(1)}$), $L'_1 = Vect(w_{\theta(1)})$ (le sous-espace de L' engendré par $w_{\theta(1)}$) et $L_0 = L'_0 = \{0\}$. Soit $j_2 \in \{1, 2, \ldots, l'\} \backslash \{\theta(1)\}$, le vecteur w_{j_2} est biorthogonalisé à droite par rapport à g à un vecteur $\tilde{p}_1$ que nous choisissons dans $L_1 \backslash L_0$, en retranchant de w_{j_2} un multiple de q_1, pour que le vecteur résultant q soit g-biorthogonal à $\tilde{p}_1$ à droite, c'est-à-dire

$$q = w_{j_2} - \overline{g(\tilde{p}_1, w_{j_2})/g(\tilde{p}_1, q_1)} q_1.$$

Soit $i_2 \in \{1, 2, \ldots, l\} \backslash \{\sigma(1)\}$, le vecteur v_{i_2} est biorthogonalisé à gauche par rapport à g à un vecteur $\tilde{q}_1$ que nous choisissons dans $L'_1 \backslash L'_0$, en retranchant de v_{i_2} un multiple de p_1, pour que le vecteur résultant p soit g-biorthogonal à $\tilde{q}_1$ à gauche, c'est-à-dire

$$p = v_{i_2} - g(v_{i_2}, \tilde{q}_1)/g(p_1, \tilde{q}_1) p_1.$$

Les choix de i_2 et j_2 se font dans le but d'avoir $g(p, q) \neq 0$, ce qui est tout à fait possible puisque $L \times L'$ est régulier. La normalisation des deux vecteurs résultants p et q nous donne alors deux vecteurs normalisés p_2 et q_2. Nous posons $\sigma(2) = i_2$ et $\theta(2) = j_2$.

Etape $k = 3, 4, \ldots, \min\{l, l'\}$:

Posons $L_{k-1} = Vect(v_{\sigma(1)}, v_{\sigma(2)}, \ldots, v_{\sigma(k-1)})$ et $L'_{k-1} = Vect(w_{\theta(1)}, w_{\theta(2)}, \ldots, w_{\theta(k-1)})$. Soit $j_k \in \{1, 2, \ldots, l'\} \backslash \theta(\{1, 2, \ldots, k-1\})$, le vecteur w_{j_k} est biorthogonalisé à droite par rapport à g à des vecteurs $\tilde{p}_i$ que nous choisissons dans $L_i \backslash L_{i-1}$ pour $i = 1, 2, \ldots, k-1$, en retranchant de w_{j_k} une combinaison linéaire des q_i qui sont déjà calculés, pour que le

vecteur résultant q soit g-biorthogonal aux vecteurs $\widetilde{p}_i$ à droite, c'est-à-dire

$$q = w_{j_k} - \sum_{i=1}^{i=k-1} \overline{g(\widetilde{p}_i, w_{j_k})/g(\widetilde{p}_i, q_i)} q_i.$$

Soit $i_k \in \{1,2,\ldots,l\}\backslash\sigma(\{1,2,\ldots,k-1\})$, le vecteur v_{i_k} est biorthogonalisé à gauche par rapport à g à des vecteurs $\widetilde{q}_j$ que nous choisissons dans $L'_j\backslash L'_{j-1}$ pour $j = 1,2,\ldots,k-1$, en retranchant de v_{i_k} une combinaison linéaire des p_j qui sont déjà calculés, pour que le vecteur résultant p soit g-biorthogonal aux vecteurs $\widetilde{q}_j$ à gauche, c'est-à-dire

$$p = v_{i_k} - \sum_{j=1}^{j=k-1} g(v_{i_k}, \widetilde{q}_j)/g(p_j, \widetilde{q}_j) p_j.$$

Le choix de i_k et j_k se fait dans le but d'avoir $g(p,q) \neq 0$, ce qui est tout à fait possible puisque $L \times L'$ est régulier. La normalisation des deux vecteurs résultants p et q nous donne alors deux vecteurs normalisés p_k et q_k. Nous posons $\sigma(k) = i_k$ et $\theta(k) = j_k$.

Si $l' > l$, pour $k = l+1, l+2, \ldots, l'$:

Nous choisissons $\theta(k) = j_k \in \{1,2,\ldots,l'\}\backslash\theta(\{1,2,\ldots,k-1\})$. Comme ce que nous avons fait pour une étape $k \leq \min\{l,l'\} = l$, nous obtenons un vecteur $q \in L'_k$ vérifiant $g(\widetilde{p}_i, q) = 0$, pour $i = 1,2,\ldots,l$. Nous normalisons q pour avoir le vecteur normalisé q_k.

Si $l' < l$, pour $k = l'+1, l'+2, \ldots, l$:

Nous choisissons $\sigma(k) = i_k \in \{1,2,\ldots,l\}\backslash\sigma(\{1,2,\ldots,k-1\})$. Comme ce que nous avons fait pour une étape $k \leq \min\{l,l'\} = l'$, nous obtenons un vecteur $p \in L_k$ vérifiant $g(p, \widetilde{q}_j) = 0$, pour $j = 1,2,\ldots,l'$. Nous normalisons p pour avoir le vecteur normalisé p_k.

Ainsi, à la fin du processus GBOR nous aurons une paire de bases g-biorthogonales $M = \{p_1, p_2, \ldots, p_l\}$ de L et $Q = \{q_1, q_2, \ldots, q_{l'}\}$ de L'. ∎

Théorème 1.6 *Soient $N = \{v_1, v_2, \ldots, v_l\}$ et $D = \{w_1, w_2, \ldots, w_{l'}\}$ deux bases respectives de L et L' avec $l < \infty$ ou $l' < \infty$. $L \times L'$ est un sous-espace régulier de $(E \times F, g)$ si et seulement s'il existe deux permutations σ de $\{1,2,\ldots,l\}$ et θ de $\{1,2,\ldots,l'\}$ telles que $L_i \oplus L_i'^{g\perp} = E$ et $L_i^{\perp g} \oplus L'_i = F$ pour $i = 1,2,\ldots,\min\{l,l'\}$.*

Preuve :

Si nous supposons que $L \times L'$ est un sous-espace régulier de $(E \times F, g)$, alors d'après la preuve du Théorème 1.5, il existe deux permutations σ de $\{1,2,\ldots,l\}$ et θ de $\{1,2,\ldots,l'\}$

telles que $L_i \oplus L_i^{'g\perp} = E$ et $L_i^{\perp g} \oplus L_i' = F$ pour $i = 1, 2, \ldots, \min\{l, l'\}$. La réciproque de ce résultat est aussi vraie ; pour le montrer, il suffit de donner à i la valeur $\min\{l, l'\}$ et d'utiliser ensuite la propriété qui dit : si $B \subset C \subset E$, alors $C^{\perp g} \subset B^{\perp g}$ et si $B \subset C \subset F$, alors $C^{g\perp} \subset B^{g\perp}$. ∎

Remarque 1.2 Lorsque $L \times L'$ est un sous-espace régulier avec L et L' étant tous les deux de dimension infinie, nous pouvons obtenir, d'une manière similaire au cas où l'un des deux est de dimension finie, deux familles g-biorthogonales dont l'une est une base soit de L soit de L' selon que $L^{\perp g} = \{0\}$ ou $L'^{g\perp} = \{0\}$. Mais, si nous avons $L^{\perp g} = L'^{g\perp} = \{0\}$, ces deux familles deviennent deux bases g-biorthogonales et les résultats des deux théorèmes précédents 1.5,1.6 restent valables. Notons aussi que dans le cas d'une forme sesquilinéaire hermitienne, la condition $L^{\perp g} = L'^{g\perp} = \{0\}$ est satisfaite si $L' = L$.

Le processus GBOR que nous allons énoncer maintenant construit à partir de deux bases quelconques $N = \{v_1, v_2, \ldots, v_l\}$ de L et $D = \{w_1, w_2, \ldots, w_{l'}\}$ de L' deux autres bases g-biorthogonales $M = \{p_1, p_2, \ldots, p_l\}$ de L et $Q = \{q_1, q_2, \ldots, q_{l'}\}$ de L'. Dans cet algorithme, nous supposons que $L \times L'$ est régulier et si L et L' sont tous les deux de dimension infinie (c.à.d $l = \infty$ et $l' = \infty$) nous ajoutons la condition $L^{\perp g} = L'^{g\perp} = \{0\}$. Considérons deux sous-espaces vectoriels L_i et L_k' engendrés respectivement par les familles libres $N_i = \{v_{\sigma(1)}, v_{\sigma(2)}, \ldots, v_{\sigma(i)}\}$ et $D_k = \{w_{\theta(1)}, w_{\theta(2)}, \ldots, w_{\theta(k)}\}$, pour $i = 1, \ldots, l$ et pour $k = 1, \ldots, l'$. Les vecteurs $v_{\sigma(j)}$ de L et $w_{\theta(j)}$ de L' sont choisis récursivement afin que les sous-espaces $L_j \times L'_j$ soient réguliers pour $j = 1, \ldots, \min\{l, l'\}$. La j-ième étape du processus GBOR consiste à compléter la paire de bases g-biorthogonales $\{p_1, p_2, \ldots, p_{j-1}\}$ et $\{q_1, q_2, \ldots, q_{j-1}\}$ de l'espace régulier $L_{j-1} \times L'_{j-1}$ en une paire de bases g-biorthogonales $\{p_1, p_2, \ldots, p_j\}$ et $\{q_1, q_2, \ldots, q_j\}$ de l'espace régulier $L_j \times L'_j$ avec $p_j \in L_j \backslash L_{j-1}$ et $q_j \in L'_j \backslash L'_{j-1}$.

Processus GBOR

1. **Initialisation** : Choisir $\sigma(1)$ et $\theta(1)$ pour lesquels $g(v_{\sigma(1)}, w_{\theta(1)}) \neq 0$. Normaliser $p = v_{\sigma(1)}$ et $q = w_{\theta(1)}$ par c_{11} et c'_{11} pour obtenir respectivement $p_1 = p/c_{11}$ et $q_1 = q/c'_{11}$.

2. **Boucle principale** : Pour $j = 2, \ldots, \min\{l, l'\}$,

   (*a*) choisir $v_{\sigma(j)}$, $w_{\theta(j)}$, $\tilde{p}_{j-1} \in L_{j-1} \backslash L_{j-2}$ et $\tilde{q}_{j-1} \in L'_{j-1} \backslash L'_{j-2}$,

   (*b*) calculer $q = w_{\theta(j)}$,

   (*c*) $c'_{ij} = \overline{g(\tilde{p}_i, q)/g(\tilde{p}_i, q_i)}$ et $q \leftarrow q - c'_{ij} q_i$ pour $i = 1, 2, \ldots, j-1$,

   (*d*) $p = v_{\sigma(j)}$,

   (*e*) $c_{ij} = g(p, \tilde{q}_i)/g(p_i, \tilde{q}_i)$ et $p \leftarrow p - c_{ij} p_i$ pour $i = 1, 2, \ldots, j-1$,

   (*f*) si $g(p, q) \neq 0$, alors
   normaliser p et q par c_{jj} et c'_{jj} pour obtenir respectivement
   $p_j = p/c_{jj}$ et $q_j = q/c'_{jj}$,
   sinon aller à (*a*),

   fin (si),

   fin de la boucle principale.

3. Boucles complémentaires:

Si $l < l'$ alors
choisir $\widetilde{p}_l \in L_l \backslash L_{l-1}$,
pour $j = l+1, \ldots, l'$:

choisir $w_{\theta(j)}$,

calculer $q = w_{\theta(j)}$,

$c'_{ij} = \overline{g(\widetilde{p}_i, q)/g(\widetilde{p}_i, q_i)}$ et $q \leftarrow q - c'_{ij} q_i$ pour $i = 1, 2, \ldots, l$,

normaliser q par c'_{lj} pour obtenir $q_j = q/c'_{lj}$,

fin (pour),
fin (si).

Si $l > l'$ alors
choisir $\widetilde{q}_{l'} \in L'_{l'} \backslash L'_{l'-1}$,
pour $j = l'+1, \ldots, l$:

choisir $v_{\sigma(j)}$,

calculer $p = v_{\sigma(j)}$,

$c_{ij} = g(p, \widetilde{q}_i)/g(p_i, \widetilde{q}_i)$ et $p \leftarrow p - c_{ij} p_i$ pour $i = 1, 2, \ldots, l'$,

normaliser p par $c_{jl'}$ pour obtenir $p_j = p/c_{jl'}$,

fin (pour),
fin (si).

Fin des boucles complémentaires.

$r \leftarrow s$ signifie que nous attribuons la valeur de s à r.

Notons que ce processus GBOR généralise celui de Gram-Schmidt décrit dans [60, 124] ainsi que le processus de Gram-Schmidt double (Two-sided Gram-Schmidt process) donné dans [90] pour le cas d'une forme hermitienne définie positive.

Supposons que $l \leq l'$. A l'aide de (d) et (e) dans GBOR nous avons

$$v_{\sigma(j)} = c_{1j} p_1 + c_{2j} p_2 + \ldots + c_{jj} p_j, \quad j = 1, 2, \ldots, l. \tag{1.6}$$

De même, à l'aide de (b) et (c) dans GBOR, nous avons

$$w_{\sigma(j)} = c'_{1j}q_1 + c'_{2j}q_2 + \ldots + c'_{jj}q_j, \quad j = 1, 2, \ldots, l', \tag{1.7}$$

avec $c'_{ij} = 0$ si $l < i < j$. En utilisant la notation $(d_1 d_2 \ldots d_j)$ pour désigner la matrice dont les colonnes sont les vecteurs $d_1, d_2, \ldots, d_j$, (1.6) et (1.7) impliquent respectivement les deux factorisations suivantes

$$\begin{pmatrix} v_{\sigma(1)} & v_{\sigma(2)} & \ldots & v_{\sigma(l)} \end{pmatrix} = \begin{pmatrix} p_1 & p_2 & \ldots & p_l \end{pmatrix} \begin{pmatrix} c_{11} & c_{12} & \ldots & c_{1l} \\ & c_{22} & \ldots & c_{2l} \\ & & \ddots & \vdots \\ & & & c_{ll} \end{pmatrix} \tag{1.8}$$

et

$$\begin{pmatrix} w_{\theta(1)} & w_{\theta(2)} & \ldots & w_{\theta(l')} \end{pmatrix} = \begin{pmatrix} q_1 & q_2 & \ldots & q_{l'} \end{pmatrix} \begin{pmatrix} c'_{11} & c'_{12} & \ldots & c'_{1l} & c'_{1l+1} & \ldots & c'_{1l'} \\ & c'_{22} & \ldots & c'_{2l} & c'_{2l+1} & \ldots & c'_{2l'} \\ & & \ddots & \vdots & \vdots & & \vdots \\ & & & c'_{ll} & c'_{ll+1} & \ldots & c'_{ll'} \\ & & & & c'_{l+1l+1} & & \\ & & & & & \ddots & \\ & & & & & & c'_{l'l'} \end{pmatrix}. \tag{1.9}$$

D'une manière analogue, si $l \geq l'$ nous obtenons, toujours grâce à GBOR, les deux factorisations suivantes

$$\begin{pmatrix} w_{\theta(1)} & w_{\theta(2)} & \ldots & w_{\theta(l')} \end{pmatrix} = \begin{pmatrix} q_1 & q_2 & \ldots & q_{l'} \end{pmatrix} \begin{pmatrix} c'_{11} & c'_{12} & \ldots & c'_{1l'} \\ & c'_{22} & \ldots & c'_{2l'} \\ & & \ddots & \vdots \\ & & & c'_{l'l'} \end{pmatrix} \tag{1.10}$$

et

$$\begin{pmatrix} v_{\sigma(1)} & v_{\sigma(2)} & \ldots & v_{\sigma(l)} \end{pmatrix} = \begin{pmatrix} p_1 & p_2 & \ldots & p_l \end{pmatrix} \begin{pmatrix} c_{11} & c_{12} & \ldots & c_{1l'} & c_{1l'+1} & \ldots & c_{1l} \\ & c_{22} & \ldots & c_{2l'} & c_{2l'+1} & \ldots & c_{2l} \\ & & \ddots & \vdots & \vdots & & \vdots \\ & & & c_{l'l'} & c_{l'l'+1} & \ldots & c_{l'l} \\ & & & & c_{l'+1l'+1} & & \\ & & & & & \ddots & \\ & & & & & & c_{ll} \end{pmatrix}. \tag{1.11}$$

Remarque 1.3 1. Une caractéristique fondamentale du processus GBOR est qu'il nous donne, grâce aux deux boucles complémentaires, les deux sous-espaces $L \cap L'^{g\perp}$ et $L' \cap L^{\perp g}$. Si $l < l'$, $L \cap L'^{g\perp} = \{0\}$ et $L' \cap L^{\perp g}$ est engendré par $q_{l+1}, q_{l+2}, \ldots, q_{l'}$. Si $l' < l$, alors $L' \cap L^{\perp g} = \{0\}$ et $L \cap L'^{g\perp}$ est engendré par $p_{l'+1}, p_{l'+2}, \ldots, p_l$.

2. Les coefficients c_{ij}, c'_{ij} sont théoriquement les mêmes pour n'importe quel choix des vecteurs $\tilde{p}_k$ et $\tilde{q}_k$. Ceci veut dire en d'autres termes, que la manière de calculer ces coefficients dépend du choix des vecteurs $\tilde{p}_k$ et $\tilde{q}_k$. C'est ce choix qui donne, par exemple, les divers algorithmes Lanczos/Orthodir, Lanczos/Orthores, Lanczos/Orthomin [18].
   Dans le cas particulier où $\tilde{p}_k = p_k$ et $\tilde{q}_k = q_k$, nous avons

$$c'_{ij} = \overline{g(p_i, w_{\theta(j)})/g(p_i, q_i)} \text{ et } c_{ij} = g(v_{\sigma(j)}, q_i)/g(p_i, q_i).$$

A la j-ième étape de la boucle principale de GBOR, nous choisissons $v_{\sigma(j)}$ et $w_{\theta(j)}$ pour avoir $g(p,q) \neq 0$ où p et q sont donnés par (b), (c), (d) et (e). L'expression de $g(p,q)$ à l'étape j peut être simplifiée sous la forme suivante

$$g(p,q) = g(v_{\sigma(j)}, w_{\theta(j)}) - \sum_{i=1}^{j-1} g(p_i, w_{\theta(j)}) g(v_{\sigma(j)}, q_i)/g(p_i, q_i). \tag{1.12}$$

Cette expression fait intervenir les quantités $g(p_i, w_{\theta(j)})$ et $g(v_{\sigma(j)}, q_i)$, ce qui rend le calcul de $g(p,q)$ très coûteux. Donc, lorsque $g(p,q) = 0$, le coût en opérations qui correspond au choix des vecteurs $v_{\sigma(j)}$ et $w_{\theta(j)}$ est en général élevé. Mais nous allons voir plus loin que ce choix n'est pas coûteux pour des espaces L et L' particuliers, par exemple les espaces de Krylov.

Afin de réduire l'expression de $g(p,q)$ et par conséquent rendre facile le choix des vecteurs $v_{\sigma(j)}$ et $w_{\theta(j)}$ (ou le choix de $\sigma(j)$ et $\theta(j)$), nous avons pensé à modifier le processus GBOR en un autre processus GBOD que nous allons décrire maintenant. Dans cette description, nous signalerons la différence entre GBOD et GBOR.

Tout d'abord remarquons que GBOD suppose que $L \times L'$ est régulier et que L et L' sont tous les deux de dimension finie (c.à.d $l < \infty$ ou $l' < \infty$). Soient $N = \{v_1, v_2, \ldots, v_l\}$ et $D = \{w_1, w_2, \ldots, w_{l'}\}$ deux bases respectives de L et L'.

$L \times L'$ est supposé régulier, donc il existe deux indices i_1 et j_1 pour lesquels nous avons $g(v_{i_1}, w_{j_1}) \neq 0$. Ainsi, nous attribuons les valeurs de i_1 et j_1 respectivement à $\sigma(1)$ et $\theta(1)$. Nous normalisons les deux vecteurs $v_{\sigma(1)}$ et $w_{\theta(1)}$ pour obtenir respectivement les vecteurs normalisés p_1 et q_1. La différence entre GBOD et GBOR est qu'ici, tous les autres vecteurs de la base N (resp. D) sont biorthogonalisés à droite (resp. à gauche), par rapport à g, à un vecteur $\tilde{p}_1$ (resp. $\tilde{q}_1$) que nous choisissons dans $L_1 \backslash L_0$ (resp. $L'_1 \backslash L'_0$). Ceci se fait suivant les deux relations suivantes

$$v_j = v_j - g(v_j, \tilde{q}_1)/g(p_1, \tilde{q}_1) p_1, \quad \text{pour} \quad j \in \{1, 2, \ldots, l\} \backslash \{\sigma(1)\}$$

et

$$w_j = w_j - \overline{g(\tilde{p}_1, w_j)/g(\tilde{p}_1, q_1)}q_1, \quad \text{pour } j \in \{1, 2, \ldots, l'\}\backslash\{\theta(1)\}.$$

$L \times L'$ est régulier, donc il existe deux entiers i_2 et j_2 pour lesquels $g(v_{i_2}, w_{j_2}) \neq 0$. Ainsi, nous attribuons les valeurs de i_2 et j_2 respectivement à $\sigma(2)$ et $\theta(2)$. Ensuite nous normalisons les deux vecteurs $v_{\sigma(2)}$ et $w_{\theta(2)}$ pour obtenir respectivement les vecteurs normalisés p_2 et q_2. Et ainsi de suite.

Nous pouvons maintenant énoncer le processus GBOD qui transforme deux bases arbitraires $N = \{v_1, v_2, \ldots, v_l\}$ de L et $D = \{w_1, w_2, \ldots, w_{l'}\}$ de L' en deux bases g-biorthogonales $M = \{p_1, p_2, \ldots, p_l\}$ de L et $Q = \{q_1, q_2, \ldots, q_{l'}\}$ de L' en supposant que le sous-espace $L \times L'$ est régulier.

Processus GBOD

1. **Initialisation** : Initialiser σ et θ à la permutation identité. Choisir deux entiers k_1 et k_1' pour lesquels $g(v_{\sigma(k_1)}, w_{\theta(k_1')}) \neq 0$. Attribuer les valeurs $\sigma(k_1)$ et $\theta(k_1')$ respectivement à $\sigma(1)$ et $\theta(1)$ et inversement. Normaliser $v_{\sigma(1)}$ et $w_{\theta(1)}$ par c_{11} et c_{11}' pour obtenir respectivement $p_1 = v_{\sigma(1)}/c_{11}$ et $q_1 = w_{\theta(1)}/c_{11}'$.

2. **Boucle principale** : Pour $j = 2, \ldots, \min\{l, l'\}$,

   (a) choisir $\tilde{p}_{j-1} \in L_{j-1}\backslash L_{j-2}$ et $\tilde{q}_{j-1} \in L'_{j-1}\backslash L'_{j-2}$,

   (b) calculer $c'_{j-1i} = \overline{g(\tilde{p}_{j-1}, w_{\theta(i)})/g(\tilde{p}_{j-1}, q_{j-1})}$ et $w_{\theta(i)} \leftarrow w_{\theta(i)} - c'_{j-1i}q_{j-1}$ pour $i = j, j+1, \ldots, l'$,

   (c) $c_{j-1i} = g(v_{\sigma(i)}, \tilde{q}_{j-1})/g(p_{j-1}, \tilde{q}_{j-1})$ et $v_{\sigma(i)} \leftarrow v_{\sigma(i)} - c_{j-1i}p_{j-1}$ pour $i = j, j+1, \ldots, l$,

   (d) choisir deux entiers $k_j \geq j$ et $k_j' \geq j$ pour lesquels $g(v_{\sigma(k_j)}, w_{\theta(k_j')}) \neq 0$,

   (e) attribuer les valeurs $\sigma(k_j)$ et $\theta(k_j')$ respectivement à $\sigma(j)$ et $\theta(j)$ et inversement,

   (f) normaliser $v_{\sigma(j)}$ et $w_{\theta(j)}$ par c_{jj} et c_{jj}' pour obtenir respectivement les vecteurs normalisés $p_j = v_{\sigma(j)}/c_{jj}$ et $q_j = w_{\theta(j)}/c_{jj}'$,

   fin de la boucle principale.

3. **Boucles complémentaires:**

Si $l < l'$ alors
pour $j = l+1, \ldots, l'$:

normaliser $w_{\theta(j)}$ par c'_{jj} pour obtenir $q_j = w_{\theta(j)}/c'_{jj}$,

fin (pour),
fin (si).

Si $l > l'$ alors
pour $j = l'+1, \ldots, l$:

normaliser $v_{\sigma(j)}$ par c_{jj} pour obtenir $p_j = v_{\sigma(j)}/c_{jj}$,

fin (pour),
fin (si).

Fin des boucles complémentaires.

Il est important de noter que cet algorithme nous permet d'obtenir les mêmes factorisations que celles données par le processus GBOR.

Remarque 1.4 Dans le cas où $L \times L'$ est régulier avec $l < l'$, le Théorème 1.6 montre qu'il existe un sous-espace L'' de L' de dimension $l'' \geq l$ tel que $L \times L''$ soit régulier. Alors, si nous prenons $D = \{w_1, w_2, \ldots, w_{l'}\}$ comme famille génératrice de L'' au lieu de la prendre comme une base de L', GBOR et GBOD restent toujours valables. Ainsi, les factorisations correspondantes nous permettent aussi de résoudre les systèmes linéaires dont les matrices sont rectangulaires. Un résultat analogue est obtenu si $l' < l$.

Dans la suite de ce chapitre nous supposons que $\tilde{p}_k = p_k$ et $\tilde{q}_k = q_k$.

1.3 Projection

Considérons deux sous-espaces vectoriels V_i et M_{m-i} de E avec $dim\ V_i = i$ et $dim\ M_{m-i} = m - i$ pour $i = 1, 2, \ldots, m$. Si la dimension m de E est infinie, alors M_{m-i} est de dimension infinie. Supposons que $V_i \oplus M_{m-i} = E$, ainsi chaque vecteur y de E peut se décomposer d'une manière unique sous forme d'une somme d'un élément y_1 de V_i et un élément y_2 de M_{m-i}. La transformation $\wp_i$ qui fait correspondre y_1 à y est un *projecteur*

sur V_i suivant M_{m-i} ($\wp_i^2 = \wp_i$), dont l'*image* est définie par

$$Im(\wp_i) = \{\wp_i(y)/ \quad y \in E\} = V_i$$

et le *noyau* par

$$Ker(\wp_i) = \{y \in E/ \quad \wp_i(y) = 0\} = M_{m-i}.$$

Supposons que $(E \times F, g)$ soit régulier et posons $W_i = M_{m-i}^{\perp_g}$ sous l'hypothèse $m = m' < \infty$ où m' désigne toujours la dimension de l'espace F. D'après le Corollaire 1.1, nous avons $M_{m-i} = W_i^{g\perp}$ et $dim\ W_i = i$. Donc le projecteur $\wp_i$ peut être caractérisé complètement par

$$\wp_i(y) \in V_i \quad \text{et} \quad (id_E - \wp_i)(y) \in W_i^{g\perp} \tag{1.13}$$

où id_E désigne l'identité sur E. Par analogie, en utilisant l'orthogonalité à droite, nous définissons le projecteur $\wp_i'$ par

$$\wp_i'(y) \in W_i \quad \text{et} \quad (id_F - \wp_i')(y) \in V_i^{\perp_g} \tag{1.14}$$

où id_F désigne l'identité sur F.
Un cas particulier souvent utilisé correspond au fait d'avoir d'une part $V_i = W_i$ et d'autre part $E = F$ avec $m < \infty$. Dans ce cas particulier, les deux projecteurs $\wp_i$ et $\wp_i'$ sont appelés projecteurs orthogonaux. Dans les autres cas, nous parlons de projecteurs obliques.
Si la forme g est hermitienne et $V_i = W_i$, alors d'après (1.13) et (1.14), $\wp_i' = \wp_i$.
Si pour tout $y, z \in E$, nous avons $g(\wp_i(y), z) = g(y, \wp_i(z))$, alors le projecteur $\wp_i$ est dit hermitien (ou autoadjoint, ou symétrique dans le cas réel) par rapport à g.

Théorème 1.7 *Si (E, g) est régulier avec g une forme hermitienne, alors un projecteur oblique $\wp_i$ est orthogonal si et seulement s'il est hermitien par rapport à g.*

Preuve :

Supposons tout d'abord que $\wp_i$ soit orthogonal et montrons qu'il est hermitien.
Pour tout $y, z \in E$, nous avons

$$\begin{aligned} g(\wp_i(y), z) - g(y, \wp_i(z)) &= g(\wp_i(y) - y, z) + g(y, z) - g(y, \wp_i(z)) \\ &= g(\wp_i(y) - y, z) + g(y, z - \wp_i(z)). \end{aligned}$$

$\wp_i$ est supposé orthogonal, donc $g(\wp_i(y) - y, z) = g(\wp_i(y) - y, z - \wp_i(z))$ et comme g est hermitienne $g(y, z - \wp_i(z)) = -g(\wp_i(y) - y, z - \wp_i(z))$. D'où $g(\wp_i(y) - y, z) + g(y, z - \wp_i(z)) = 0$, ce qui implique que $g(\wp_i(y), z) = g(y, \wp_i(z))$. Par conséquent $\wp_i$ est hermitien.

Réciproquement, supposons que $\wp_i$ soit hermitien et montrons qu'il est orthogonal. Pour tout $x \in V_i$, il existe $z \in E$ tel que $x = \wp_i(z)$. Donc pour tout $y \in E$,

$$g(y - \wp_i(y), x) = g(y - \wp_i(y), \wp_i(z)) = g(y, \wp_i(z)) - g(\wp_i(y), \wp_i(z)).$$

$\wp_i$ est un projecteur supposé hermitien, donc $g(\wp_i(y), \wp_i(z)) = g(y, \wp_i(z))$.
D'où $g(y - \wp_i(y), x) = 0$ pour tout $x \in V_i$ et tout $y \in E$. Ce qui équivaut à dire que pour tout $y \in E$, $y - \wp_i(y) \in V_i^{g\perp}$. Comme (E, g) est régulier, nous avons de plus $V_i \oplus V_i^{g\perp} = E$. Donc $\wp_i$ est orthogonal. ∎

Si nous supposons que $\wp_i$ est un projecteur orthogonal et que (E, g) est un espace régulier, alors l'adjoint $\wp_i^*$ de $\wp_i$ est aussi un projecteur orthogonal. Si de plus la forme g est hermitienne, alors d'après le Théorème 1.7, $\wp_i$ est autoadjoint.

1.3.1 Projection biorthogonale

Soit $L \times L'$ un sous-espace régulier de $E \times F$ avec $dim\ L = dim\ L' = l < \infty$. Soient $\{v_1, v_2, \ldots, v_l\}$ une base de L et $\{w_1, w_2, \ldots, w_l\}$ une base de L'. Nous avons montré précédemment qu'il existe deux permutations σ et θ telles que les espaces V_i et W_i engendrés respectivement par $N_i = \{v_{\sigma(1)}, v_{\sigma(2)}, \ldots, v_{\sigma(i)}\}$ et $D_i = \{w_{\theta(1)}, w_{\theta(2)}, \ldots, w_{\theta(i)}\}$ vérifient

$$V_i \oplus W_i^{g\perp} = E \quad \text{et} \quad W_i \oplus V_i^{\perp g} = F. \tag{1.15}$$

Ceci veut dire aussi que $V_i \times W_i$ est régulier pour $i = 1, 2, \ldots, l$.
Pour tout $i \in \{1, 2, \ldots, l\}$, nous allons donc considérer deux projecteurs, $\wp_i$ défini par $Im(\wp_i) = V_i$, $Ker(\wp_i) = W_i^{g\perp}$ et $\wp_i'$ défini par $Im(\wp_i') = W_i$, $Ker(\wp_i') = V_i^{\perp g}$.

Définition 1.5 *Nous appelons projection biorthogonale par rapport à g ou projection g-biorthogonale toute application de $E \times F$ vers $L \times L'$ qui fait correspondre la paire $(\wp_i(y), \wp_i'(y'))$ à tout couple de vecteurs (y, y') où*

- *$\wp_i(y)$ est la projection du vecteur y sur L suivant $L'^{g\perp}$,*
- *$\wp_i'(y')$ est la projection du vecteur y' sur L' suivant $L^{\perp g}$.*

Il est possible d'appliquer la méthode GBO (GBOR ou GBOD) pour obtenir deux bases biorthogonales $P = \{p_1, p_2, \ldots, p_l\}$ de L et $Q = \{q_1, q_2, \ldots, q_l\}$ de L'. Ce qui nous permet d'avoir l'algorithme suivant qui donne la projection g-biorthogonale d'un couple

de vecteurs (y, y') sur $L \times L'$ et que nous appelons *algorithme général de projection biorthogonale*. Nous désignons cet algorithme par GBOP (General BiOrthogonal Projection).

Algorithme GBOP

1. **Initialisation** : Choisir $y \in E$, $y' \in F$ et poser $\wp_0 = \wp'_0 = 0$.
2. **Boucle** : Pour $j = 1, \ldots, l$
   (Le calcul de $\wp_j(y)$ et $\wp'_j(y')$)
   (*a*) Utiliser la méthode GBO (GBOR ou GBOD) pour calculer p_j et q_j,
   (*b*) $c_j = g(y - \wp_{j-1}(y), q_j)/g(p_j, q_j)$,
   (*c*) $\wp_j(y) = \wp_{j-1}(y) + c_j p_j$,
   (*d*) $c'_j = \overline{g(p_j, y' - \wp'_{j-1}(y'))/g(p_j, q_j)}$,
   (*e*) $\wp'_j(y') = \wp'_{j-1}(y') + c'_j q_j$.

Remarque 1.5 1. Les coefficients c_j et c'_j à l'étape j peuvent être calculés par d'autres relations équivalentes à (*b*) et (*d*). Nous donnons comme exemple les deux expressions suivantes $c_j = g(y, q_j)/g(p_j, q_j)$ et $c'_j = \overline{g(p_j, y')/g(p_j, q_j)}$.

2. Les vecteurs y et y' peuvent être choisis comme solutions d'un problème linéaire, par conséquent ils seront des inconnues à approcher respectivement par les vecteurs $\wp_l(y)$ et $\wp'_l(y')$. Même si y et y' sont des inconnues, le calcul des quantités $g(y - \wp_{j-1}(y), q_j)$ et $g(p_j, y' - \wp'_{j-1}(y'))$ est possible suivant un choix de la forme g. Par exemple, si y et y' sont solutions respectives de $Gy = b$ et $G^* y' = b'$ avec $b \in L'$ et $b' \in L$, alors nous choisissons la forme g qui a pour matrice G^T.
   Le couple de vecteurs $(y - \wp_j(y), y' - \wp'_j(y'))$ représente l'erreur commise sur la projection g-biorthogonale de (y, y') sur $L \times L'$ à l'étape j.

Notons que le rôle des deux permutations σ et θ dans GBOR et GBOD est d'assurer que la condition (1.15) est satisfaite. Si la forme sesquilinéaire g est hermitienne définie positive avec $V_j = W_j$ et $y = y'$, alors le vecteur erreur $y - \wp_j(y)$ est minimisé en g-norme, c'est-à-dire

$$\forall y, z \quad \min_{z \in V_j} g(y - z, y - z) = g(y - \wp_j(y), y - \wp_j(y)) = g(y - \wp'_j(y), y - \wp'_j(y)). \quad (1.16)$$

Dans ce cas, nous avons aussi $\wp_j = \wp'_j$ pour tout j. Ceci est dû à la propriété particulière $V_i^{\perp_g} = V_i^{g\perp}$ d'une forme hermitienne.

Nous supposerons dans la suite de ce chapitre que $E = F$.

1.4 Méthodes directes

Dans cette section, nous allons montrer que, pour résoudre $Ax = b$, plusieurs méthodes directes peuvent être déduites de l'algorithme GBOP si nous choisissons convenablement les paramètres g, E, F, l, y et y' avec les deux bases $N_l = \{v_1, v_2, \ldots, v_l\}$ et $D_l = \{w_1, w_2, \ldots, w_l\}$ respectives de L et L'. Ici, nous allons juste déduire les deux factorisations les plus utilisées LU et QR de la matrice A du système (0.1).

1.4.1 Factorisation LU

Choisissons les paramètres
- $E = F = \mathbb{C}^n$, $l = n, y = y' = x$ (x est la solution du système $Ax = b$),
- $v_j = w_j = e_j = (0, \ldots, 1, \ldots, 0)^T$ (le j-ième vecteur de la base canonique de E),
- $g(u, v) =< Au, v >_n$ pour tout $u, v \in E$.

Premièrement, pour l'obtention des deux bases g-biorthogonales $P = \{p_1, p_2, \ldots, p_l\}$ et $Q = \{q_1, q_2, \ldots, q_l\}$, nous supposons que le processus GBOR peut être appliqué sans avoir à utiliser aucune permutation σ ou θ, ce qui est équivalent à prendre $v_{\sigma(j)} = v_j$ et $w_{\theta(j)} = w_j$ pour tout entier j. Ici, ceci veut dire ne pas échanger et les lignes et les colonnes de la matrice A.

Comme la solution x de $Ax = b$ est inconnue, nous ne pouvons pas effectuer (d) et (e) de l'algorithme GBOP. Nous avons vu que $p_i \in V_i = Vect(e_1, e_2, \ldots, e_i)$ et $q_i \in W_i = Vect(e_1, e_2, \ldots, e_i)$. Par conséquent, la matrice $M_p = (p_1 p_2 \ldots p_n)$ de colonnes p_i et la matrice $M_q = (q_1 q_2 \ldots q_n)$ de colonnes q_i sont deux matrices triangulaires supérieures. Imposons une normalisation sur les p_i et les q_i pour avoir $g(p_i, q_i) = 1$ pour tout i, par exemple (1.5). La condition de la g-biorthogonalité $< Ap_i, q_j >_n = \delta_{ij}$ nous donne

$$M_q^* \, A \, M_p = I_n \quad \text{or} \quad A = M_q^{*-1} \, M_p^{-1},$$

où I_n est la matrice identité de dimension n. Soit D la matrice diagonale $n \times n$ dont les termes diagonaux sont ceux de la matrice M_q^*. La factorisation $A = LU$ est unique lorsque L est triangulaire inférieure à diagonale unité et U est triangulaire supérieure. D'où les égalités suivantes

$$L = M_q^{*-1} \, D \quad \text{et} \quad U = D^{-1} \, M_p{}^{-1}.$$

Il est important de noter que nous obtenons aussi la factorisation de A^{-1} suivante

$$A^{-1} = M_p \, M_q^*. \tag{1.17}$$

Si $M_{qi} = (q_1 q_2 \dots q_i)$ et $M_{pi} = (p_1 p_2 \dots p_i)$, alors à la i-ème étape de l'algorithme GBOP utilisé avec GBOR, M_{qi} et M_{pi} vérifient $I_i = M_{qi}^* \, A \, M_{pi}$ où I_i est la matrice identité de $\mathbb{C}^{i \times i}$.

Nous pouvons obtenir la factorisation LU en multipliant tout simplement les deux membres de l'égalité (1.8) par la matrice A. De plus, si nous multiplions (1.9) par la matrice A^*, alors nous obtenons la factorisation LU de A^*. Nous pouvons aussi choisir les vecteurs v_j comme vecteurs colonnes de la matrice A au lieu de les choisir comme étant les vecteurs de la base canonique de E et de prendre $g =< .,. >_n$. Dans ce cas, la factorisation LU de A est obtenue directement à partir de (1.8).

L'algorithme GBOP donne la solution $\wp_n(x) = x$ de $Ax = b$ à la n-ième étape au plus. L'algorithme résultant de GBOP, sous les hypothèses ci-dessus, peut être considéré comme une variante de la méthode de Gauss.
Le remède apporté au problème de l'instabilité numérique de la méthode de Gauss est la technique du *pivotage partiel* ou *total*. Ici, pour le pivotage partiel, les lignes sont échangées par la permutation θ. Tandis que pour le pivotage total, les colonnes sont aussi échangées par la permutation σ, si nécessaire, pour rendre le pivot $g(p,q)$ le plus grand possible en valeur absolue. Le nombre d'opérations utilisées pour résoudre $Ax = b$ est le même que pour la méthode de Gauss sauf qu'ici, lorsque nous utilisons GBOR au lieu de GBOD dans GBOP, la recherche du pivot est plus chère en opérations.
En fait, avec les paramètres que nous avons choisis ci-dessus, GBOP utilisé avec GBOD nous permet de retrouver exactement la méthode de Gauss.

1.4.2 Factorisation QR

Afin de retrouver la factorisation QR, nous choisissons
- $E = F = \mathbb{C}^n, l = n, y = y' = x$ (x est la solution du système $Ax = b$),
- $v_j = w_j = e_j = (0, \dots, 1, \dots, 0)^T$ (le j-ième vecteur de la base canonique de E),
- $g(u,v) =< Au, Av >_n$ pour tout $u, v \in E$.

Clairement, g est hermitienne et définie positive. Donc, si les deux permutations σ et θ sont supposées égales à l'identité, GBOR et la normalisation (1.5) nous donnent deux bases g-biorthogonales identiques $P = \{p_1, p_2, \dots, p_n\}$ et $Q = \{q_1, q_2, \dots, q_n\}$. Ceci veut dire que $p_i = q_i$ pour $i = 1, \dots, n$.
Nous avons vu que $p_i \in V_i = Vect(e_1, e_2, \dots, e_i)$, donc la matrice $M_p = (p_1 p_2 \dots p_n)$ de colonnes p_i est triangulaire supérieure. $< Ap_i, Aq_j >_n = \delta_{ij}$ est la condition de la

g-biorthogonalité qui se traduit par

$$(A\ M_p)^*\ A\ M_p = I_n.$$

Nous en déduisons que $A\ M_p$ est une matrice unitaire et que

$$A = (A\ M_p)\ M_p^{-1}.$$

La factorisation QR est donc obtenue avec M_p^{-1}, qui est la matrice triangulaire supérieure R de la factorisation.

D'une manière plus simple, nous obtenons la factorisation QR en multipliant les deux membres de l'égalité (1.8) par la matrice A. Nous pouvons aussi choisir les vecteurs $v_j = w_j$ comme étant les colonnes de la matrice A au lieu de les choisir comme étant les vecteurs de la base canonique de E et de prendre $g =< ., . >_n$. Dans ce cas la factorisation QR de A est obtenue directement à partir de (1.8).

1.5 Méthodes itératives

Plusieurs méthodes itératives ont en commun le fait de projeter la solution sur l'espace de Krylov

$$K_i = K_i(A, r_0) = Vect(r_0, Ar_0, \ldots, A^{i-1}r_0), \quad \text{pour } i = 1, 2, \ldots,$$

où $r_0 = b - Ax_0$ avec x_0 est un vecteur choisi arbitrairement. Mais ces méthodes diffèrent par la manière de projeter. Dans cette section nous allons retrouver quelques unes de ces méthodes itératives à partir de l'algorithme GBOP utilisé avec le processus GBOR.

1.5.1 Méthode du gradient conjugué (CG)

La méthode du gradient conjugué est très utile pour avoir la solution d'un système linéaire, creux, hermitien et défini positif. Elle est efficace surtout lorsqu'elle est utilisée avec un préconditionnement. Diverses généralisations de la méthode du gradient conjugué classique ont été proposées par plusieurs auteurs [2, 51, 52, 74, 99, 127, 37, 50]. Plus récemment, l'auteur [66] a décrit les relations entre ces généralisations. L'algorithme CG classique [70] pour résoudre $Ax = b$ est donné dans [60] comme suit

1. **Initialisation** : $k = 0$, $x_0 = 0$ et $r_0 = b$.

2. **Boucle** : Tant que $(r_k \neq 0)$

$k = k + 1$

si $k = 1$

$p_1 = r_0$

sinon

$\alpha_k = r_{k-1}^* r_{k-1} / r_{k-2}^* r_{k-2}$

$p_k = r_{k-1} + \alpha_k p_{k-1}$

fin (si)

$\beta_k = r_{k-1}^* r_{k-1} / p_k^* A p_k$

$x_k = x_{k-1} + \beta_k p_k$

$r_k = r_{k-1} - \beta_k A p_k$

fin (Tant que).

Ici A est supposée hermitienne et définie positive. Cet algorithme donne l'itéré x_k dans l'espace de Krylov $K_k(A, r_0)$ avec la condition de minimisation suivante

$$\|x - x_k\|_A = \min_{z \in K_k} \|x - z\|_A,$$

où $\|z\|_A$ désigne la A-norme ($\|z\|_A =< Az, z >_n$). Cette équation et celle de (1.16) sont les mêmes si nous choisissons dans l'algorithme GBOP

- $E = F = \mathbb{C}^n$, $l = n$, $y = y' = x$ (x est la solution du système $Ax = b$),
- $v_1 = w_1 = r_0$ et $v_j = w_j = Ap_{j-1}$ pour $j = 2, 3, \ldots, n$ (v_j et w_j sont choisis récursivement),
- $g(u, v) =< Au, v >_n$ pour tout $u, v \in E$.

L'itéré $\wp_k(x)$ de l'algorithme GBOP est la projection de la solution x de $Ax = b$ sur $K_k(A, r_0)$, donc il est égal à x_k.

Comme la matrice A est hermitienne, (c) et (e) du processus GBOR deviennent deux relations équivalentes à trois termes et, à l'aide de petites modifications dans l'algorithme GBOP, nous retrouvons exactement l'algorithme CG classique.

Si nous appliquons CG aux équations normales $A^*Ax = A^*b$ et $AA^*z = b$, alors nous obtenons les deux méthodes $CGNR$ [51] et $CGNE$ [2]. La méthode $CGNR$ minimise la norme du vecteur résidu. La méthode $CGNE$, qui est souvent appelée la méthode de Craig [41], minimise la norme du vecteur erreur.

1.5.2 Méthode du résidu conjugué généralisé (GCR) ou ORTHOMIN

La méthode ORTHOMIN a été donnée par Vinsome [118]. Les résultats théoriques sur la convergence de cette méthode ont été obtenus par Axelsson [3] et les auteurs de [48, 49]. Cette méthode est aussi connue sous le nom GCR [49] dont l'algorithme peut être brièvement décrit comme suit

1. **Initialisation** : Choisir $x_0 \in E$ et poser $p_0 = r_0 = b - Ax_0$.
2. **Boucle** : Pour $i = 0, 1, \ldots$ jusqu'à la convergence :
   - (a_1) calculer $\alpha_i =< r_i, Ap_i >_n / < Ap_i, Ap_i >_n$,
   - (a_2) $x_{i+1} = x_i + \alpha_i p_i$,
   - (a_3) $r_{i+1} = r_i - \alpha_i Ap_i$,
   - (a_4) $p_{i+1} = r_{i+1} - \sum_{j=0}^{i} \beta_{ij} p_j$,
     où β_{ij} sont choisis pour que $< Ap_i, Ap_j >_n = 0$ pour $i \neq j$.

Si nous choisissons dans l'algorithme GBOP (utilisé avec GBOR)
- $E = F = \mathbb{C}^n$, $l = n$, $y = y' = x - x_0$, $v_1 = w_1 = r_0$,
- $v_j = w_j = Ap_{j-1}$, pour $j = 2, 3, \ldots, n$ (v_j et w_j sont choisis récursivement),
- $g(u, v) =< Au, Av >_n$ pour tout $u, v \in E$,

alors, à partir des résultats précédents, il est clair que nous retrouvons exactement GCR. A l'étape i, l'algorithme GCR présente une DPZ (division par zéro) si $\alpha_{i-1} = 0$ sans que r_i soit nul. Nous pouvons éviter cette DPZ en replaçant r_{i+1} dans (a_4) par Ap_i.
Dans le cas hermitien (c.à.d A est hermitienne), l'algorithme GCR est équivalent à la variante CR du CG. L'algorithme CR [70], sans aucune DPZ, peut être décrit et résumé comme suit

1. **Initialisation** : Choisir $x_0 \in E$ et poser $r_0 = b - Ax_0$, $\gamma_0 = \|Ar_0\|$, $p_{-1} = 0$, $p_0 = r_0/\gamma_0$.

2. **Boucle** : Pour $i = 0, 1, \ldots$ jusqu'à la convergence :

   (d_1) calculer $\alpha_i =< r_i, Ap_i >_n$,

   (d_2) $x_{i+1} = x_i + \alpha_i p_i$,

   (d_3) $r_{i+1} = r_i - \alpha_i Ap_i$,
   si $|\alpha_i| < tol$ pour une tolérance tol, alors

   (d_4) $z_i = Ap_i - \gamma_i p_{i-1}$,

   (d_5) $\beta_i =< Az_i, Ap_i >_n$,

   (d_6) $\gamma_{i+1} p_{i+1} = z_i - \beta_i p_i$,
   sinon

   (d_7) $\beta_i =< Ar_{i+1}, Ap_i >_n$,

   (d_8) $\gamma_{i+1} p_{i+1} = r_{i+1} - \beta_i p_i$,
   où γ_{i+1} est choisi tel que $\|Ap_{i+1}\| = 1$,
   fin (si).

Remarque 1.6 Lorsque $\alpha_i = 0$ à la i-ème itération, nous ne pouvons pas avoir $\alpha_{i+1} = 0$ sauf si la solution x de $Ax = b$ est obtenue. En fait, ceci est un avantage du cas hermitien.

Il est bien connu que l'algorithme CR est mathématiquement équivalent à l'algorithme du résidu minimum MINRES [88] et l'algorithme GCR est mathématiquement équivalent à GMRES. Les deux algorithmes GMRES et GCR sont respectivement deux généralisations de MINRES et CR. Dans l'algorithme précédent de CR, nous remarquons que (d_4), (d_5) et (d_6) définissent le processus hermitien de Lanczos [87, 88, 89] qui peut être retrouvé à partir du processus GBOR.
Donnons maintenant le processus hermitien de Lanczos comme il a été décrit dans plusieurs articles et travaux.

Processus hermitien de Lanczos

1. **Initialisation** : Choisir $v_1 \in E$ et poser $\beta_1 = \|v_1\|$, $p_1 = v_1/\beta_1$, $p_0 = 0$.
2. **Boucle** : Pour $i = 1, 2, \ldots, m$:

   calculer $z_i = Ap_i - \beta_i p_{i-1}$,

   $\alpha_i =< z_i, p_i >_n$,

   $\beta_{i+1} p_{i+1} = z_i - \alpha_i p_i$,
   où β_{i+1} est choisi tel que $\|p_{i+1}\| = 1$.

1.5.3 Processus non hermitien de Lanczos

Si nous choisissons dans le processus GBOR les paramètres
- $E = F = \mathbb{C}^n$, $l = n$,
- $v_j = Ap_{j-1}$, $w_j = A^* q_{j-1}$ pour $j = 2, 3, \ldots, n$ (v_j et w_j sont choisis récursivement),
- $g(u, v) =< u, v >_n$ pour tout $u, v \in E$,

et s'il n'y a pas de DPZ (c.à.d $< p_i, q_i >_n \neq 0$), alors il en résulte. le processus non hermitien de Lanczos [90].

Processus non hermitien de Lanczos

1. **Initialisation** : Choisir $v_1, w_1 \in E$ et poser $p_1 = v_1/\beta_1$, $q_1 = w_1/\rho_1$, $p_0 = q_0 = 0$, $h_1 = 1$ (ρ_1 et η_1 sont choisis pour avoir les vecteurs normalisés v_1 et w_1).
2. **Boucle** : Pour $i = 1, 2, \ldots, m$:

   calculer $z_i = Ap_i - \overline{\rho_i} h_i p_{i-1}$,

   $z_i' = A^* q_i - \overline{\beta_i h_i} q_{i-1}$,

   $\alpha_i =< z_i, q_i >_n / < p_i, q_i >_n$,

   $\beta_{i+1} p_{i+1} = z_i - \alpha_i p_i$,

   $\rho_{i+1} q_{i+1} = z_i' - \overline{\alpha_i} q_i$,

   $h_{i+1} =< p_{i+1}, q_{i+1} >_n / < p_i, q_i >_n$,
   où β_{i+1} et ρ_{i+1} sont choisis pour avoir les vecteurs normalisés p_{i+1} et q_{i+1}.

Si nous utilisons la simple normalisation (1.5), alors $\rho_i = \overline{\beta_i}$ et $h_i = 1$. Une autre variante du processus non hermitien de Lanczos [57] est obtenue en utilisant la normalisation suivante de p_i et q_i.

$$\begin{cases} p_i' = p_i/\|p_i\|, \\ q_i' = q_i/\|q_i\|. \end{cases} \tag{1.18}$$

Ceci signifie que $\beta_i = \|p_i\|$ et $\rho_i = \|q_i\|$. Nous savons que l'algorithme QMR [54] est basé sur cette variante du processus non hermitien de Lanczos. Si à la i-ème itération $< p_i, q_i >_n = 0$ sans que $p_i = 0$ ou $q_i = 0$, alors l'algorithme QMR présente une DPZ. Pour cette raison, plusieurs stratégies de "look-ahead" ont été proposées par les auteurs [90, 16, 56]. Dans le deuxième chapitre nous allons voir comment nous pouvons éviter ce "look-ahead" (voir aussi [4]).

Un résultat important de l'application de ce formalisme mathématique, est que le processus non hermitien de Lanczos peut se ramener au processus hermitien de Lanczos. En effet, si nous choisissons dans le processus GBOR
$E = F = \mathbb{C}^{2n}$, $l = n$, $w_j = v_j$ pour $j = 2, 3, \ldots, n$ (v_j et w_j sont choisis récursivement),

$$v_j = \begin{pmatrix} Ap_{j-1} \\ A^T q_{j-1} \end{pmatrix}, \quad g(u,v) = v_2^T u_1 + v_1^T u_2 \text{ pour tout } u = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}, v = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \in E,$$

alors la relation à trois termes suivante est déduite du processus GBOR

$$\begin{pmatrix} p_{j+1} \\ q_{j+1} \end{pmatrix} = \begin{pmatrix} A & 0 \\ 0 & A^T \end{pmatrix} \begin{pmatrix} p_j \\ q_j \end{pmatrix} - \alpha_j \begin{pmatrix} p_j \\ q_j \end{pmatrix} - \beta_j \begin{pmatrix} p_{j-1} \\ q_{j-1} \end{pmatrix}. \tag{1.19}$$

Les expressions de α_j et β_j sont déduites de la condition d'orthogonalité par rapport à g. La relation précédente, que nous retrouvons aussi dans [25], est à trois termes. Ceci est justifié par le fait que la matrice étendue ou augmentée

$$\begin{pmatrix} A & 0 \\ 0 & A^T \end{pmatrix}$$

est symétrique par rapport à la forme g choisie. Nous pouvons voir facilement que l'équation ci-dessus n'est autre que celle qui intervient dans le processus non hermitien de Lanczos. Il y a deux façons de normaliser : l'une est celle utilisée dans [57] qui consiste à normaliser les vecteurs p_j et q_j, l'autre consiste à normaliser le vecteur de composantes p_j et q_j qui n'est autre que la normalisation du processus hermitien de Lanczos.

1.6 Conclusion

Dans ce chapitre, nous avons étudié les espaces sesquilinéaires réguliers. Ces espaces réguliers nous ont permis de bien définir une projection biorthogonale. Par exemple, pour

une forme hermitienne, une projection orthogonale sur un espace régulier existe toujours. Une telle existence de projection sur des espaces sesquilinéaires réguliers nous a permis de définir la méthode GBOP. Cette méthode GBOP nous donne la projection biorthogonale sur un espace régulier V en utilisant des projections biorthogonales sur des sous-espaces réguliers de V. Pour avoir récursivement une telle projection biorthogonale, GBOP utilise la méthode GBO qui construit une paire de bases biorthogonales. La manière de construire ces deux bases biorthogonales nous a permis d'avoir les deux processus GBOR et GBOD. Ces deux processus GBOR et GBOD ont plusieurs applications, notamment celles données au deuxième chapitre.

Chapitre 2

Les polynômes orthogonaux formels et le problème de la DPZ

Dans ce chapitre, nous allons introduire une méthode qui évite les stratégies connues du "look-ahead" dans la programmation des polynômes orthogonaux formels. Cette méthode est appelée ALA. Elle fait intervenir de nouveaux polynômes qui ont des propriétés particulières. Signalons aussi qu'en choisissant E et F comme étant des espaces de polynômes, ALA est une application de la méthode GBO. Ceci va nous permettre de donner trois mises en oeuvre principales de la méthode ALA.

2.1 Introduction

Il est bien connu que les polynômes orthogonaux formels interviennent implicitement dans les méthodes de type Lanczos, dans la programmation des approximants de Padé, ainsi que dans la théorie des fractions continues, en particulier les P-fractions de Magnus [79, 80] qui sont les fractions continues associées aux diverses entrées des diagonales de la table non normale de Padé. Les blocs d'une table non normale de Padé et le problème de la DPZ des méthodes de type Lanczos sont dus à la non existence et à la singularité de quelques polynômes orthogonaux. Un travail considérable sur ce sujet a été fait par Struble [115], Gragg [61], Nuttall et Singh [85], Draux [44], Gragg et Lindquist [62], Stahl [113, 114], et Gutknecht [65]. Gutknecht a traité aussi dans [65] la dégénérescence de la biorthogonalisation de Lanczos. Le lien de la biorthogonalité de Lanczos avec la notion des polynômes orthogonaux a permis à Brezinski et Sadok de donner dans [18] divers algorithmes de type Lanczos. Certains polynômes peuvent exister et, cependant, sont impossibles à calculer par certaines relations de récurrence. C'est le *"ghost breakdown"*, voir [22].

Nous savons qu'il est possible d'obtenir le polynôme orthogonal régulier suivant à partir

de ceux qui le précèdent en résolvant un système linéaire régulier. Par exemple, voir la méthode de bordage [16] et l'élimination de Gauss [65, 20].
Toujours dans le cadre de l'extension des algorithmes connus et en tenant compte de la dégénérescence, Gutknecht et Hochbruck ont présenté dans [68] la généralisation des algorithmes de Levinson et Schur en utilisant la stratégie du "look-ahead" pour les matrices de Toeplitz non hermitiennes qui consiste à résoudre un système régulier. Nous pouvons dire alors que la stratégie du "look-ahead" n'est autre qu'une étape intermédiaire qui consiste à résoudre un système linéaire régulier.

Dans ce chapitre, nous allons montrer comment nous pouvons éviter la stratégie du "look-ahead" et par conséquent éviter la dégénérescence à l'aide de nouveaux polynômes intermédiaires qui ont des caractéristiques particulières. Ces polynômes sont biorthogonaux et satisfont une relation de récurrence simple à trois termes. Cette relation de récurrence relie des polynômes biorthogonaux dont les degrés ne sont pas successifs (sinon ils seraient orthogonaux). De plus ces polynômes peuvent être considérés comme une alternative des polynômes orthogonaux qui n'existent pas. A première vue, l'approche que nous proposons dans ce chapitre semble être comme une nouvelle formulation du "look-ahead". Ceci est partiellement vrai, puisque nous avons un test à faire pour savoir où se trouve le polynôme orthogonal régulier suivant. Cependant, la stratégie donnée ici est simple et nous permet de calculer récursivement tous les polynômes orthogonaux réguliers sans avoir à résoudre un système linéaire régulier et sans aucun problème de stockage.

Les coefficients qui apparaissent dans des différentes relations de récurrence sont donnés en général sous forme de fractions. Lorsqu'un dénominateur de ces fractions est proche de zéro, alors des erreurs d'arrondi peuvent affecter sérieusement l'algorithme correspondant. Les erreurs de cancellation peuvent aussi entraîner un mauvais calcul des polynômes et des vecteurs orthogonaux ou biorthogonaux. Nous disons dans ce cas que nous avons presque une DPZ, cette situation sera désignée par l'abréviation PDPZ (Presque Division Par Zéro). Ce dernier problème se traduit par une construction de blocs, voir [83].

Ce chapitre est organisé comme suit. Dans la section 2, nous introduisons quelques notations et définitions, et revoyons quelques propriétés de la programmation des polynômes biorthogonaux. Dans la section 3, nous étudions le cas de l'orthogonalité et nous décrivons une nouvelle méthode que nous appelons ALA (Avoiding Look-Ahead) qui nous permet d'éviter le "look-ahead" dans le calcul des polynômes orthogonaux formels. La section 4 traite le problème de DPZ dans la méthode de Lanczos. Dans la section 5, nous appli-

quons ALA au calcul des approximants de Padé et nous donnons quelques propriétés de ces derniers. A la fin de ce chapitre, dans la section 6, nous appliquons ALA pour obtenir les itérés d'indices pairs de l'epsilon algorithme.

2.2 Biorthogonalité

2.2.1 Résultats préliminaires et définitions

Soit $(L_i)_{i\in\mathbb{N}}$ une suite de fonctionnelles linéaires sur l'espace des polynômes $\mathbb{C}[X]$. Pour tout entier n, un polynôme P_n de degré inférieur ou égal à n est appelé biorthogonal par rapport à la suite $(L_i)_{i\in\mathbb{N}}$ si et seulement si

$$L_i(P_n) = 0, \quad i = 0, \ldots, n-1. \tag{2.1}$$

Cette biorthogonalité est similaire à celle donnée dans [15]. P_n n'est déterminé qu'à un coefficient de normalisation près. Si la suite $(L_i)_{i\in\mathbb{N}}$ satisfait

$$L_{i+d}(\psi) = L_i(x\psi), \quad i \in \mathbb{N}, \quad \forall \psi \in \mathbb{C}[X]$$

pour un entier fixe $d \geq 1$, alors P_n est un polynôme orthogonal vectoriel de dimension d [117]. Il suffit alors de connaître $L_0, L_1, \ldots, L_{d-1}$. Si $d = 1$, P_n est le polynôme orthogonal formel habituel par rapport à L_0. Dans la section 2.3, nous considérons le cas où d est égal à 1.

Le polynôme P_n est appelé régulier si le déterminant de Hankel

$$G_n = \begin{vmatrix} L_0(1) & \cdots & L_0(x^{n-1}) \\ L_1(1) & \cdots & L_1(x^{n-1}) \\ \cdots & \cdots & \cdots \\ L_{n-1}(1) & \cdots & L_{n-1}(x^{n-1}) \end{vmatrix}$$

est non nul. Cette condition entraîne que P_n est exactement de degré n. Notons aussi que P_n est appelé singulier si $G_n = 0$.

Lorsque G_n est nul, nous introduisons et développons de nouveaux polynômes que nous notons $P_n^\theta, n \in \mathbb{N}$ et qui sont biorthogonaux réguliers par rapport à la suite $(L_{\theta(i)})_{i\in\mathbb{N}}$, où θ est une permutation de $\mathbb{N}$ vers $\mathbb{N}$ que nous allons définir dans la prochaine section.

$$\theta : \begin{array}{ccc} \mathbb{N} & \longmapsto & \mathbb{N} \\ i & \longmapsto & \theta(i) \end{array}$$

Nous allons voir que ces polynômes $P_n^\theta, n \in \mathbb{N}$ vont servir au calcul des polynômes biorthogonaux réguliers par rapport à la suite initiale $(L_i)_{i\in\mathbb{N}}$.

2.2.2 Construction de P_n^θ

Nous allons maintenant voir comment la permutation θ est définie.
Posons

$$G_\infty = \begin{vmatrix} L_0(1) & L_0(x) & L_0(x^2) & \cdots \\ L_1(1) & L_1(x) & L_1(x^2) & \cdots \\ L_2(1) & L_2(x) & L_2(x^2) & \cdots \\ \vdots & \vdots & \vdots & \end{vmatrix}.$$

Nous voulons permuter les lignes de G_∞ afin d'avoir un nouveau déterminant G_∞^θ dont tous les déterminants principaux sont non nuls. Par conséquent, nous permutons

- la première ligne avec la ligne $(L_{j_0}(1), L_{j_0}(x), \ldots)$ de plus petit indice j_0 pour laquelle le premier terme $L_{j_0}(1)$ est non nul ; ainsi $\theta(0)$ est définie par $\theta(0) = j_0$. Si $L_0(1) \neq 0$, alors nous gardons la première ligne et $\theta(0) = 0$.

- la deuxième ligne avec la ligne $(L_{j_1}(1), L_{j_1}(x), \ldots)$ de plus petit indice j_1 qui rend le déterminant principal de dimension 2 non nul ; ainsi $\theta(1) = j_1$.

- et ainsi de suite pour les autres lignes.

Ce processus nous permet de donner la définition ci-dessous de l'application θ.
En pratique, le calcul des déterminants principaux de G_∞ se fait à l'aide des polynômes P_n^θ. Nous allons voir ceci plus tard, lorsque nous donnerons l'expression explicite des P_n^θ.

Définition 2.1 *θ est une application sur $\mathbb{N}$ telle que pour tout $n \in \mathbb{N}$, $\theta(n)$ est le plus petit entier j, pour lequel le déterminant*

$$\begin{vmatrix} L_{\theta(0)}(1) & \cdots & L_{\theta(0)}(x^n) \\ L_{\theta(1)}(1) & \cdots & L_{\theta(1)}(x^n) \\ \vdots & & \vdots \\ L_{\theta(n-1)}(1) & \cdots & L_{\theta(n-1)}(x^n) \\ L_j(1) & \cdots & L_j(x^n) \end{vmatrix}$$

est non nul.

Par la suite, nous supposons que pour tout $n \in \mathbb{N}$, $\theta(n)$ est fini (c.à.d. $\theta(n) < \infty$). Pour que $\theta(n)$ soit fini, il est nécessaire et suffisant qu'il existe $n+1$ fonctionnelles linéaires de la suite $(L_i)_{i \in \mathbb{N}}$ qui ont des restrictions à l'espace des polynômes de degré inférieur ou égal à n linéairement indépendantes.

Pour une suite $(L_i)_{i\in\mathbb{N}}$ donnée, notons G_n^θ le déterminant

$$G_n^\theta = \begin{vmatrix} L_{\theta(0)}(1) & \cdots & L_{\theta(0)}(x^{n-1}) \\ L_{\theta(1)}(1) & \cdots & L_{\theta(1)}(x^{n-1}) \\ \vdots & & \vdots \\ L_{\theta(n-1)}(1) & \cdots & L_{\theta(n-1)}(x^{n-1}) \end{vmatrix}.$$

Il est clair, à partir de la définition de θ, que G_n^θ est non nul pour tout $n \in \mathbb{N}^*$. L'application θ est définie récursivement puisque la connaissance de $\theta(n)$ nécessite celle des $\theta(j)$, $j \leq n-1$.

Remarque 2.1 Pour résoudre un système de Hankel, Rissanen a utilisé dans [94] une permutation similaire ayant, dans le cas de l'orthogonalité, les mêmes propriétés que θ. Simplement, il permute les colonnes au lieu des lignes.

Lemme 2.1 *S'il existe un entier $n \in \mathbb{N}$ tel que*

$$\theta(\{0,\ldots,n\}) = \{0,\ldots,n\},$$

alors P_{n+1} est régulier.

Preuve:

$\theta(\{0,\ldots,n\}) = \{0,\ldots,n\}$ implique que les lignes de G_{n+1}^θ sont celles de G_{n+1}. Et comme G_{n+1}^θ est non nul, alors G_{n+1} est aussi non nul. Il s'ensuit donc, par définition, que P_{n+1} est régulier. ∎

Nous allons définir maintenant deux suites d'indices $(p_l)_{l\in\mathbb{N}}$ et $(h_l')_{l\in\mathbb{N}}$ que nous retrouvons dans [44] avec les mêmes notations.

- pour $l = 0$
  si $G_1 \neq 0$, alors $p_0 = h_0' = 0$; sinon, nous posons $h_0' = -1$ et

$$p_0 = \max\{i \in \mathbb{N}, \forall n \in]h_0' + 1, i] \Rightarrow G_n = 0\}.$$

- pour $l = 1, 2, 3, \ldots$
  h_l' et p_l sont définis par les deux conditions

$$h_l' + 1 = \max\{i \in \mathbb{N}, \forall n \in]p_{l-1}, i] \Rightarrow G_n \neq 0\}$$

et

$$p_l = \max\{i \in \mathbb{N}, \forall n \in]h'_l + 1, i] \Rightarrow G_n = 0\}.$$

Ainsi, la connaissance de h'_l nécessite celle de p_{l-1} et connaissant h'_l, nous pouvons déterminer p_l.

Lemme 2.2 *i/ Les polynômes biorthogonaux formels P_n^θ, $n \in \mathbb{N}$, par rapport à la famille $(L_{\theta(i)})_{i \in \mathbb{N}}$, sont réguliers.*

ii/ Soit l un entier non nul. Pour tout entier $i \in \{p_{l-1}, p_{l-1}+1, \ldots, h'_l\}$, nous avons

$$\theta(\{0, \ldots, i\}) = \{0, \ldots, i\},$$

c'est-à-dire que P_{i+1}^θ est régulier.

Preuve :

i/ La définition de θ nous donne immédiatement le résultat.

ii/ $i \in \{p_{l-1}, \ldots, h'_l\}$ implique que $i+1 \in \{p_{l-1}+1, \ldots, h'_l+1\}$, donc dans ce cas G_{i+1} est non nul.
Par récurrence sur $j \in \{0, \ldots, i\}$, nous avons

- pour $j = 0$
  il existe un entier $k \in \{0, \ldots, i\}$ pour lequel $L_k(1) \neq 0$ car, sinon, les éléments de la première colonne de G_{i+1} seraient nuls et par conséquent G_{i+1} serait nul. Nous concluons que $\theta(0) \in \{0, \ldots, i\}$,
- nous supposons que pour $j \in \{0, \ldots, i-1\}$, l'hypothèse de récurrence est vraie (c.à.d $\theta(j) \in \{0, \ldots, i\}$),
- pour $j+1$
  nous savons que, par définition de θ, le déterminant

$$G_{j+1}^\theta = \begin{vmatrix} L_{\theta(0)}(1) & \cdots & L_{\theta(0)}(x^j) \\ L_{\theta(1)}(1) & \cdots & L_{\theta(1)}(x^j) \\ \vdots & & \vdots \\ L_{\theta(j)}(1) & \cdots & L_{\theta(j)}(x^j) \end{vmatrix}$$

est non nul. Il existe donc un entier $k \in \{0, \ldots, i\}$ pour lequel le déterminant

$$G_{k,j}^\theta = \begin{vmatrix} L_{\theta(0)}(1) & \cdots & L_{\theta(0)}(x^{j+1}) \\ L_{\theta(1)}(1) & \cdots & L_{\theta(1)}(x^{j+1}) \\ \vdots & & \vdots \\ L_{\theta(j)}(1) & \cdots & L_{\theta(j)}(x^{j+1}) \\ L_k(1) & \cdots & L_k(x^{j+1}) \end{vmatrix}$$

est non nul car, sinon,

$$G^{\theta}_{k,j} = 0, \text{ pour tout } k \in \{0,\dots,i\}\backslash\{\theta(0),\dots,\theta(j)\}$$

et par conséquent il existerait $j+1$ coefficients $\gamma_1,\dots,\gamma_{j+1}$, tels que

$$\begin{pmatrix} L_{\theta(0)}(x^{j+1}) \\ L_{\theta(1)}(x^{j+1}) \\ \vdots \\ L_{\theta(j)}(x^{j+1}) \\ L_k(x^{j+1}) \end{pmatrix} = \gamma_1 \begin{pmatrix} L_{\theta(0)}(1) \\ L_{\theta(1)}(1) \\ \vdots \\ L_{\theta(j)}(1) \\ L_k(1) \end{pmatrix} + \cdots + \gamma_{j+1} \begin{pmatrix} L_{\theta(0)}(x^j) \\ L_{\theta(1)}(x^j) \\ \vdots \\ L_{\theta(j)}(x^j) \\ L_k(x^j) \end{pmatrix}.$$

Par hypothèse de récurrence, $G^{\theta}_{j+1} \neq 0$. Donc, les coefficients $\gamma_1,\dots,\gamma_{j+1}$ sont toujours les mêmes pour tout $k \in \{0,\dots,i\}\backslash\{\theta(0),\dots,\theta(j)\}$. En conséquence

$$\begin{pmatrix} L_0(x^{j+1}) \\ L_1(x^{j+1}) \\ \vdots \\ L_i(x^{j+1}) \end{pmatrix} = \gamma_1 \begin{pmatrix} L_0(1) \\ L_1(1) \\ \vdots \\ L_i(1) \end{pmatrix} + \cdots + \gamma_{j+1} \begin{pmatrix} L_0(x^j) \\ L_1(x^j) \\ \vdots \\ L_i(x^j) \end{pmatrix},$$

ce qui entraîne que $G_{i+1} = 0$. Or il est impossible que ce déterminant G_{i+1} soit nul, d'où, $\theta(j+1) \in \{0,\dots,i\}$. Par conséquent, $\theta(\{0,\dots,i\}) = \{0,\dots,i\}$. Ceci implique, d'après le Lemme 2.1, que P^{θ}_{i+1} est régulier. ∎

Expression explicite de P^{θ}_i

P^{θ}_i est défini par $L_{\theta(k)}(P^{\theta}_i) = 0$, pour $k = 0,\dots,i-1$. Alors, pour chaque $i \in \mathbb{N}$, le polynôme biorthogonal régulier P^{θ}_i de degré i par rapport à la suite $(L_{\theta(i)})_{i\in\mathbb{N}}$ peut être écrit sous forme d'un rapport de deux déterminants

$$P^{\theta}_i(x) = \alpha \begin{vmatrix} L_{\theta(0)}(1) & \cdots & L_{\theta(0)}(x^i) \\ L_{\theta(1)}(1) & \cdots & L_{\theta(1)}(x^i) \\ \vdots & & \vdots \\ L_{\theta(i-1)}(1) & \cdots & L_{\theta(i-1)}(x^i) \\ 1 & \cdots & x^i \end{vmatrix} / G^{\theta}_i$$

où $\alpha \in \mathbb{C}^*$.

Remarque 2.2 Pour chaque $i \in \mathbb{N}$, $\theta(i)$ peut être déterminé comme étant le plus petit entier j tel que $L_j(P^{\theta}_i(x))$ est non nul. Il s'ensuit que $L_{\theta(i)}(P^{\theta}_i(x)) \neq 0$, pour tout $i \in \mathbb{N}$.

Voyons maintenant comment calculer les déterminants G_i^θ. Posons $h_i = L_{\theta(i)}(P_i^\theta(x))$ et $G_0^\theta = 1$. D'après l'expression explicite des P_i^θ, nous avons $h_i = G_{i+1}^\theta / G_i^\theta$. D'où l'égalité $G_{i+1}^\theta = h_i h_{i-1} \dots h_0$ pour tout $i \in \mathbb{N}$.
Par conséquent le calcul des déterminants G_i^θ se fait grâce à la relation de récurrence suivante

$$G_{i+1}^\theta = h_i G_i^\theta, \quad i = 1, 2, \dots,$$

avec $G_0^\theta = 1$. Les valeurs des quantités h_i s'obtiennent par la relation $h_i = L_{\theta(i)}(P_i^\theta(x))$ où les coefficients des polynômes P_i^θ se calculent à l'aide de la relation de récurrence donnée ci-dessous dans le Théorème 2.1.
Nous terminons cette section avec quelques propriétés de P_n^θ. Ces propriétés vont nous servir dans la prochaine section.
Pour une simple raison de simplification, nous supposons dans la suite que P_i^θ et P_i sont unitaires lorsqu'ils existent.

Théorème 2.1 *i/ Si* $i \in \{p_{l-1} + 1, \dots, h_l' + 1\}$, $l \in \mathbb{N}$ *alors* $P_i^\theta = P_i$.

ii/ Pour chaque $i \in \mathbb{N}$, *le polynôme* P_{i+1}^θ *est donné par*

$$P_{i+1}^\theta = x P_i^\theta + \sum_{j=0}^{i} \alpha_j P_j^\theta,$$

où

$$\alpha_j = -[L_{\theta(j)}(x P_i^\theta) + \sum_{k=0}^{j-1} \alpha_k L_{\theta(j)}(P_k^\theta)] / L_{\theta(j)}(P_j^\theta), \quad j = 0, \dots, i.$$

Preuve :

i/ Afin de prouver que $P_i^\theta = P_i$ pour $i \in \{p_{l-1} + 1, \dots, h_l' + 1\}$, $l \in \mathbb{N}$, il suffit de permuter les lignes des deux déterminants qui apparaissent dans l'expression de P_i^θ pour obtenir, grâce au Lemme 2.1, l'égalité $\theta(\{0, \dots, i-1\}) = \{0, \dots, i-1\}$.

ii/ Les polynômes P_i^θ sont unitaires et de degrés égaux à leurs indices. Donc ils forment une base de $\mathbb{C}[X]$, d'où

$$P_{i+1}^\theta = x P_i^\theta + \sum_{j=0}^{i} \alpha_j P_j^\theta, \tag{2.2}$$

où $\alpha_j \in \mathbb{C}$ pour $j = 0, \dots, i$. Comme $L_{\theta(k)}(P_j^\theta) = 0$, pour $k < j$, l'application successive des fonctionnelles linéaires $L_{\theta(0)}, \cdots, L_{\theta(i)}$ à (2.1) se traduit par le système

triangulaire suivant

$$\begin{cases} \alpha_0 L_{\theta(0)}(P_0^\theta) & & & & = & -L_{\theta(0)}(xP_i^\theta) \\ \alpha_0 L_{\theta(1)}(P_0^\theta) & + & \alpha_1 L_{\theta(1)}(P_1^\theta) & & = & -L_{\theta(1)}(xP_i^\theta) \\ \vdots & & \vdots & & & \vdots \\ \alpha_0 L_{\theta(i)}(P_0^\theta) & + & \alpha_1 L_{\theta(i)}(P_1^\theta) & \cdots + \alpha_i L_{\theta(i)}(P_i^\theta) & = & -L_{\theta(i)}(xP_i^\theta). \end{cases}$$

Les termes de la diagonale de ce système ci-dessus sont $L_{\theta(j)}(P_j^\theta)$, $j = 0, \ldots, i$. D'après la remarque précédente, nous savons que ces termes sont non nuls et par conséquent la détermination des coefficients de (2.1) se fait par la relation

$$\alpha_j = -[L_{\theta(j)}(xP_i^\theta) + \sum_{k=0}^{j-1} \alpha_k L_{\theta(j)}(P_k^\theta)]/L_{\theta(j)}(P_j^\theta), \quad j = 0, \ldots, i. \qquad \blacksquare$$

2.2.3 Lien avec la biorthogonalité par rapport à une forme sesquilinéaire

Dans cette sous-section, nous allons voir que la biorthogonalité définie ci-dessus et que nous allons utiliser dans ce chapitre n'est autre qu'un cas particulier de celle que nous avons traité au premier chapitre. Ce cas particulier est obtenu en faisant un choix particulier de la forme sesquilinéaire g.

Choix de la forme sesquilinéaire g

Tout d'abord commençons par choisir les deux espaces vectoriels E et F. Ici, E est l'espace des polynômes $\mathbb{C}[X]$ et F est l'espace des fonctionnelles semi-linéaires définies sur E c'est-à-dire, en d'autres termes, que F est l'espace dual conjugué $\overline{\widehat{E}}$ de E.
Nous définissons la forme sesquilinéaire g par

$$g(\psi, \overline{L}) = L(\psi), \quad \forall L \in \widehat{E}, \quad \forall \psi \in E.$$

La matrice de la restriction de la forme sesquilinéaire g aux sous-espaces V_n et W_n, engendrés respectivement par $\{1, x, \ldots, x^{n-1}\}$ et $\{\overline{L}_0, \overline{L}_1, \ldots, \overline{L}_{n-1}\}$, a pour déterminant G_n. L'application θ que nous utilisons ici est la même que celle du premier chapitre. La permutation σ est prise égale à l'identité. Le rôle de l'application θ est d'avoir un sous-espace régulier $V_n \times W_n^\theta$ de $E \times F$ avec W_n^θ étant le sous-espace engendré par $\{\overline{L}_{\theta(0)}, \overline{L}_{\theta(1)}, \ldots, \overline{L}_{\theta(n-1)}\}$. Alors, la matrice de la restriction de g à $V_n \times W_n^\theta$ est régulière et a pour déterminant G_n^θ qui est bien non nul. A l'aide du processus GBOR, nous pouvons alors construire deux bases g-biorthogonales $\{P_0^\theta, P_1^\theta, \ldots, P_{n-1}^\theta\}$ de V_n et $\{\overline{\phi}_0^\theta, \overline{\phi}_1^\theta, \ldots, \overline{\phi}_{n-1}^\theta\}$

de W_n^θ. Ces deux bases sont obtenues récursivement par les relations déduites du GBOR

$$\begin{cases} \alpha_i = \phi_i^\theta(xP_{j-1}^\theta)/\phi_i^\theta(P_i^\theta), & i = 0, \ldots, j-1 \\ P_j^\theta = xP_{j-1}^\theta - \sum_{i=0}^{j-1} \alpha_i P_i^\theta \end{cases}$$

et

$$\begin{cases} \beta_i = L_{\theta(j)}(P_i^\theta)/\phi_i^\theta(P_i^\theta), & i = 0, \ldots, j-1 \\ \phi_j^\theta = L_{\theta(j)} - \sum_{i=0}^{j-1} \beta_i \phi_i^\theta. \end{cases}$$

Dans la définition de la permutation θ, nous avons supposé que $\theta(n) < \infty$ pour tout entier n, ceci revient à supposer que $F^{g\perp} = \{0\}$, c'est-à-dire que $(E \times F, g)$ est régulier.

2.3 Orthogonalité

Supposons que les fonctionnelles L_j, $j \in \mathbb{N}$, utilisées dans la définition de la biorthogonalité vérifient la propriété suivante

$$L_j(\psi) = C(x^j\psi), \quad j \in \mathbb{N}, \quad \forall \psi \in \mathbb{C}[X],$$

où C est une fonctionnelle linéaire définie sur l'espace des polynômes $\mathbb{C}[X]$. Cette situation n'est autre que l'orthogonalité formelle habituelle [10].
Soit $(P_n)_{n\in\mathbb{N}}$ la famille des polynômes orthogonaux par rapport à C. Si tous les polynômes P_n, $n \in \mathbb{N}$ sont réguliers, alors ils vérifient une relation de récurrence à trois termes. Dans le cas contraire, quelques polynômes de la famille $(P_n)_{n\in\mathbb{N}}$ n'existent pas, et nous avons à résoudre un système linéaire régulier pour obtenir le polynôme régulier suivant.
Avec l'usage des polynômes $P_i^\theta, i \in \mathbb{N}$, définis précédemment, nous allons voir comment retrouver les polynômes orthogonaux réguliers de la famille $(P_n)_{n\in\mathbb{N}}$.

2.3.1 Relations de récurrence

Supposons qu'il existe un polynôme régulier P_k de degré k de la famille $(P_n)_{n\in\mathbb{N}}$, pour lequel la quantité $C(x^kP_k)$ est nulle, c'est-à-dire $\theta(k) > k$. Par définition, nous savons que $\theta(k)$ est le plus petit entier j tel que $C(x^jP_k)$ est non nul. Il est donc évident que le polynôme P_k vérifie

- $C(x^jP_k) = 0, \quad j = 0, \cdots, \theta(k) - 1$
- $C(x^{\theta(k)}P_k) \neq 0$
- $P_k = P_k^\theta$.

De plus, pour tout entier $n \in \mathbb{N}$, P_n^θ est défini et caractérisé par

$$\begin{cases} C(x^{\theta(j)} P_n^\theta) = 0, & j = 0, 1, \ldots, n-1 \\ C(x^{\theta(n)} P_n^\theta) \neq 0. \end{cases}$$

Considérons le résultat du Théorème 2.1 en supposant que $L_j(\psi) = C(x^j \psi)$ pour tout polynôme $\psi \in \mathbb{C}[X]$ et pour tout entier j. Nous pouvons déduire le théorème suivant en montrant qu'il y a des coefficients qui s'annulent dans la relation de récurrence du Théorème 2.1.

Théorème 2.2 *Si P_k est un polynôme régulier tel que $C(x^k P_k) = 0$, alors pour $i \in \{k, \ldots, \theta(k) - 1\}$, nous avons*

i/ la relation de récurrence à trois termes

$$P_{i+1}^\theta = x P_i^\theta + \alpha_i P_k$$

avec

$$\alpha_i = -C(x^{\theta(k)+1} P_i^\theta) / C(x^{\theta(k)} P_k),$$

ii/ la condition $C(x^j P_{i+1}^\theta) = 0$ pour $j = 0, \ldots, \theta(k)$ à l'exception de $j = \theta(k)+k-(i+1)$,

iii/ les deux propriétés $\theta(i+1) = \theta(k) + k - (i+1)$ et $C(x^{\theta(i+1)} P_{i+1}^\theta) = C(x^{\theta(k)} P_k) \neq 0$.

Preuve :

La preuve se fait par une récurrence sur i qui commence par $i = k$.

- Pour $i = k$.
  Nous posons $D = x P_k^\theta + \alpha P_k$ avec $\alpha = -C(x^{\theta(k)+1} P_k^\theta) / C(x^{\theta(k)} P_k)$.
  P_k est régulier, donc l'égalité $P_k = P_k^\theta$ est une conséquence du Théorème 2.1. Si nous multiplions D par x^j et que nous appliquons la fonctionnelle C, alors

$$C(x^j D) = C(x^{j+1} P_k) + \alpha C(x^j P_k).$$

  Ensuite, en utilisant les relations

$$C(x^j P_k) = 0 \quad \text{pour} \quad j \in \{0, \ldots, \theta(k) - 1\}$$

  et

$$C(x^{j+1} P_k) = 0 \quad \text{pour} \quad j \in \{0, \ldots, \theta(k) - 2\}$$

nous aurons $C(x^j D) = 0$ pour $j \in \{0, \ldots, \theta(k) - 2\}$.
$\alpha = -C(x^{\theta(k)+1} P_k^\theta)/C(x^{\theta(k)} P_k)$ implique que $C(x^{\theta(k)} D) = 0$, ce qui entraîne

$$C(x^j D) = 0 \quad \text{pour} \quad j \in \{0, \ldots, \theta(k)\} \quad \text{et} \quad j \neq \theta(k) - 1.$$

D est un polynôme de degré $k+1$ qui vérifie les propriétés de P_{k+1}^θ. P_{k+1}^θ étant déterminé d'une façon unique, donc $D = P_{k+1}^\theta$.D'où l'écriture

$$P_{k+1}^\theta = x P_k^\theta + \alpha P_k.$$

La multiplication de P_{k+1}^θ par $x^{\theta(k)-1}$ et l'application de la fonctionnelle C nous donne

$$C(x^{\theta(k)-1} P_{k+1}^\theta) = C(x^{\theta(k)} P_k) + \alpha C(x^{\theta(k)-1} P_k).$$

Et comme $C(x^{\theta(k)-1} P_k) = 0$, alors

$$C(x^{\theta(k)-1} P_{k+1}^\theta) = C(x^{\theta(k)} P_k) \neq 0.$$

Par conséquent, $\theta(k) - 1$ est le plus petit entier j, pour lequel $C(x^{\theta(j)} P_{k+1}^\theta)$ est non nul. D'où $\theta(k+1) = \theta(k) - 1 = \theta(k) + k - (k+1)$.

- Pour i.
  Nous supposons que le résultat du théorème est vrai.

- Pour $i+1$.
  Posons $D = x P_i^\theta + \alpha P_k$.
  La multiplication de D par x^j et l'application de la fonctionnelle C, nous donnent

$$C(x^j D) = C(x^{j+1} P_i^\theta) + \alpha C(x^j P_k)$$

et nous avons

  - $C(x^j P_k) = 0$ pour $j \in \{0, \ldots, \theta(k) - 1\}$,
  - $\alpha = -C(x^{\theta(k)+1} P_i^\theta)/C(x^{\theta(k)} P_k)$ implique que $C(x^{\theta(k)} D) = 0$,
  - $C(x^{j+1} P_i^\theta) = 0$ pour $j+1 \in \{0, \ldots, \theta(k)\}$ et $j+1 \neq \theta(k) + k - i$,
    ce qui donne
    $C(x^{j+1} P_i^\theta) = 0$ pour $j \in \{0, \ldots, \theta(k) - 1\}$ et $j \neq \theta(k) + k - (i+1)$.

Nous pouvons donc conclure que

$$C(x^j D) = 0 \text{ pour } j \in \{0, \ldots, \theta(k)\} \text{ et } j \neq \theta(k) + k - (i+1).$$

Le polynôme D est unitaire de degré $i+1$ et vérifie les propriétés de P_{i+1}^{θ}. Donc l'unicité de P_{i+1}^{θ} implique que $D = P_{i+1}^{\theta}$. D'où l'écriture

$$P_{i+1}^{\theta} = xP_i^{\theta} + \alpha P_k,$$

avec $\alpha = -C(x^{\theta(k)+1}P_i^{\theta})/C(x^{\theta(k)}P_k)$.
Si nous appliquons la fonctionnelle C à P_{i+1}^{θ} après l'avoir multiplié par $x^{\theta(k)+k-(i+1)}$, alors nous obtenons

$$C(x^{\theta(k)+k-(i+1)}P_{i+1}^{\theta}) = C(x^{\theta(k)+k-i}P_i^{\theta}) + \alpha C(x^{\theta(k)+k-(i+1)}P_k).$$

Par hypothèse de récurrence, $C(x^{\theta(k)+k-i}P_i^{\theta}) = C(x^{\theta(k)}P_k)$.
Comme $C(x^{\theta(k)+k-(i+1)}P_k) = 0$, alors il s'ensuit que $C(x^{\theta(k)+k-(i+1)}P_{i+1}^{\theta}) = C(x^{\theta(k)}P_k)$ est non nul.
D'où $\theta(i+1) = \theta(k) + k - (i+1)$. ∎

Corollaire 2.1 *$P_{\theta(k)+1}$ est régulier.*

Preuve :

P_k est supposé régulier d'avance, et d'après le Lemme 2.2, nous avons $\theta(\{0,\ldots,k-1\}) = \{0,\ldots,k-1\}$. Le Théorème 2.2 nous donne la propriété $\theta(i+1) = \theta(k) + k - (i+1)$, qui implique

$$\theta(\{0,\ldots,\theta(k)-1,\theta(k)\}) = \{0,\ldots,\theta(k)-1,\theta(k)\}.$$

Enfin, une simple application du Lemme 2.1 montre que $P_{\theta(k)+1}$ est régulier. ∎
Ce résultat est aussi donné par Draux [44] et Gutknecht [65].

En posant, dans le Théorème 2.1, $L_j(\psi) = C(x^j\psi)$ pour tout $\psi \in \mathbb{C}[X]$ et pour tout entier j, et en montrant qu'il y a des coefficients qui s'annulent dans la relation de récurrence du Théorème 2.1, nous pouvons déduire le théorème suivant.

Théorème 2.3 *Soit P_{k_0} le dernier polynôme régulier précédant P_k.*
La relation de récurrence suivante nous donne le premier polynôme régulier suivant P_k

$$P_{\theta(k)+1} = xP_{\theta(k)}^{\theta} + \sum_{i=k+1}^{\theta(k)} \alpha_i P_i^{\theta} + \alpha_k P_k + \alpha_{k-1}P_{k_0}$$

avec

$$\alpha_{k-1} = -C(x^{\theta(k)}P_k)/C(x^{\theta(k_0)}P_{k_0}) \neq 0,$$
$$\alpha_k = -(C(x^{\theta(k)+1}P_{\theta(k)}^{\theta}) + \alpha_{k-1}C(x^{\theta(k)}P_{k_0}))/C(x^{\theta(k)}P_k),$$
$$\alpha_i = C(x^{\theta(i)}P_{k_0})/C(x^{\theta(k_0)}P_{k_0}), \quad i = k+1,\ldots,\theta(k).$$

Preuve :

P_{k_0} est le polynôme régulier de plus haut degré précédant P_k, alors $\theta(k_0) = k-1$. Les polynômes P_i^θ forment une base de l'espace $\mathbb{C}[X]$, donc nous pouvons écrire

$$P_{\theta(k)+1} = xP^\theta_{\theta(k)} + \sum_{i=0}^{\theta(k)} \alpha_i P_i^\theta. \tag{2.3}$$

L'application de la fonctionnelle C à (2.2), après l'avoir multiplié par $x^{\theta(j)}$, nous donne

$$C(x^{\theta(j)}P_{\theta(k)+1}) = C(x^{\theta(j)+1}P^\theta_{\theta(k)}) + \sum_{i=0}^{\theta(k)} \alpha_i C(x^{\theta(j)}P_i^\theta).$$

Nous avons

- $C(x^{\theta(j)+1}P^\theta_{\theta(k)}) = 0$ pour $0 \leq \theta(j)+1 \leq \theta(k)$ et $\theta(j)+1 \neq k$

- $C(x^{\theta(j)}P^\theta_{\theta(i)}) = 0$ pour $j < i$, $i \in \mathbb{N}$.

Si $j \leq k-1$, alors $\theta(j) \leq k-1$, puisque nous avons vu précédemment que $\theta(\{0,\ldots,k-1\}) = \{0,\ldots,k-1\}$. Ainsi, en posant $s_{i,j} = C(x^{\theta(i)}P_j^\theta)$, nous obtenons le système linéaire suivant noté (S)

$$(S) \qquad \begin{pmatrix} s_{0,0} & & & & \\ \vdots & \ddots & & & \\ s_{k_0-1,0} & \cdots & s_{k_0-1,k_0-1} & & \\ s_{k_0+1,0} & \cdots & s_{k_0+1,k_0-1} & s_{k_0+1,k_0+1} & \\ \vdots & & \vdots & \vdots & \ddots \\ s_{k-1,0} & \cdots & s_{k-1,k_0-1} & s_{k-1,k_0+1} & \cdots s_{k-1,k-1} \end{pmatrix} \begin{pmatrix} \alpha_0 \\ \vdots \\ \alpha_{k_0-1} \\ \alpha_{k_0+1} \\ \vdots \\ \alpha_{k-1} \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

Le système (S) est triangulaire régulier avec un même nombre d'inconnues et d'équations, puisque la condition

$$C(x^{\theta(j)}P^\theta_{k_0}) = 0 \quad \text{pour } j = 0,\ldots,k-1 \quad \text{sauf lorsque } j = k_0$$

montre que α_{k_0} est éliminée et n'apparaît pas dans (S).
Par conséquent $\alpha_j = 0$ pour $j = 0,\ldots,k-1$ sauf lorsque $j = k_0$, et la relation (2.2) devient

$$P_{\theta(k)+1} = xP^\theta_{\theta(k)} + \sum_{i=k+1}^{\theta(k)} \alpha_i P_i^\theta + \alpha_k P_k + \alpha_{k_0} P_{k_0}. \tag{2.4}$$

Déterminons maintenant les coefficients α_i.

- Calcul de α_{k_0}.
  L'application de la fonctionnelle C à l'équation (2.3), après l'avoir multipliée par $x^{k-1} = x^{\theta(k_0)}$, nous donne

$$C(x^{k-1}P_{\theta(k)+1}) = 0 = C(x^k P^\theta_{\theta(k)}) + \sum_{i=k+1}^{\theta(k)} \alpha_i C(x^{k-1}P^\theta_i) + \alpha_k C(x^{k-1}P_k) + \alpha_{k_0} C(x^{k-1}P_{k_0}).$$

  Nous savons que

$$C(x^{k-1}P^\theta_i) = 0, \quad \text{pour } i = k, \ldots, \theta(k)$$

  et d'après le Théorème 2.2,

$$C(x^k P^\theta_{\theta(k)}) = C(x^{\theta(k)} P^\theta_k).$$

  D'où

$$\alpha_{k_0} = -C(x^{\theta(k)}P_k)/C(x^{\theta(k_0)}P_{k_0}) \neq 0.$$

- Calcul de α_k.
  L'expression de α_k est déduite immédiatement, en multipliant (2.3) par $x^{\theta(k)}$ et en appliquant ensuite la fonctionnelle C.

- Calcul de α_i, pour $i = k+1, \ldots, \theta(k)$.
  L'application de C à (2.3) après l'avoir multipliée par $x^{\theta(i)}$ implique

$$C(x^{\theta(i)+1}P^\theta_{\theta(k)}) + \sum_{j=k+1}^{\theta(k)} \alpha_j C(x^{\theta(i)}P^\theta_j) + \alpha_k C(x^{\theta(i)}P_k) + \alpha_{k_0} C(x^{\theta(i)}P_{k_0}) = 0.$$

  D'après le Théorème 2.2, nous avons

  - $\theta(i) = (\theta(k) + k - i) \in \{k, \ldots, \theta(k) - 1\}$ pour $i = k+1, \ldots, \theta(k)$,
    ce qui nous donne
    $C(x^{\theta(i)}P^\theta_j) = 0$ pour $j = k+1, \ldots, \theta(k)$ à l'exception de $j = i$.
  - $\theta(i) + 1 \in \{k+1, \ldots, \theta(k)\}$ montre que $C(x^{\theta(i)+1}P^\theta_{\theta(k)}) = 0$.
  - $\theta(i) < \theta(k)$ pour $i = k+1, \ldots, \theta(k)$, ce qui implique que
    $C(x^{\theta(i)}P_k) = 0$ pour $i = k+1, \ldots, \theta(k)$.

  Donc, la dernière relation nous donne

$$\alpha_i = -\alpha_{k_0} C(x^{\theta(i)}P_{k_0})/C(x^{\theta(i)}P^\theta_i) \quad \text{pour } i = k+1, \ldots, \theta(k)$$

ce qui est équivalent à

$$\alpha_i = C(x^{\theta(i)}P_{k_0})/C(x^{\theta(k_0)}P_{k_0}) \quad \text{pour } i = k+1,\ldots,\theta(k)$$

puisque

$$\alpha_{k_0} = -C(x^{\theta(k)}P_k)/C(x^{\theta(k_0)}P_{k_0})$$

et

$$C(x^{\theta(i)}P_i^\theta) = C(x^{\theta(k)}P_k) \quad \text{pour } i = k,\ldots,\theta(k).$$

■

Il est clair que les deux théorèmes 2.2 et 2.3 définissent un algorithme pour calculer tous les polynômes orthogonaux réguliers par rapport à une fonctionnelle C. Cet algorithme utilise les polynômes biorthogonaux intermédiaires pour le calcul des polynômes orthogonaux réguliers. Ainsi, nous appelons la méthode ALA (Avoiding the Look-Ahead) celle qui utilise des polynômes biorthogonaux intermédiaires pour retrouver tous les polynômes orthogonaux réguliers.
Pour tout k, posons $C(x^k) = c_k$ et écrivons P_k^θ sous la forme

$$P_k^\theta(x) = \sum_{j=0}^{k} p_j^{(k)} x^j, \quad p_k^{(k)} = 1.$$

Alors, en utilisant cette écriture dans les relations de récurrence des deux théorèmes 2.2 et 2.3, en identifiant les coefficients des différentes puissances en x, nous obtenons un algorithme qui est une mise en oeuvre de la méthode ALA pour calculer les coefficients de P_k^θ.

Algorithme 1

- Etape 1 : Initialisation
  $p_0^{(-1)} = p_{-1}^{(-1)} = 0$, $h_{-1} = 1$, $\theta(-1) = -1$,
  $p_0^{(0)} = 1$, $p_{-1}^{(0)} = p_1^{(0)} = 0$, $k = 0$.

- Etape 2 (Détermination de $\theta(k)$)
  1 $i = 0$
  $k_0 = \theta(k-1)$
  2 $h_{k+i} = \sum_{j=0}^{k} c_{k+i+j} p_j^{(k)}$
  $d_i = \sum_{j=0}^{k_0} c_{k+i+j} p_j^{(k_0)}$
  si $|h_{k+i}| < \varepsilon_1$ pour une certaine tolérance ε_1, alors
  $i = i + 1$
  aller à 2
  fin (si)
  $\theta(k) = k + i$.

- Etape 3 :
  $b = h_{\theta(k)}/h_{k-1}$
  $w_j = -p_j^{(k_0)}$, $j = 0, \ldots, k_0$
  $w_j = 0$, $j = k_0 + 1, \ldots, \theta(k)$
  pour $i = k, \ldots, \theta(k)$
  $d_{\theta(k)-i} = d_{\theta(k)-i}/h_{\theta(k)}$
  $w_j = w_j + d_{\theta(k)-i} p_j^{(i)}$, $j = 0, \ldots, i$
  $\beta_i = (\sum_{j=0}^{i} c_{\theta(k)+1+j} p_j^{(i)})/h_{\theta(k)}$
  $p_j^{(i+1)} = p_{j-1}^{(i)} - \beta_i p_j^{(k)}$, $j = 0, \ldots, k$
  $p_j^{(i+1)} = p_{j-1}^{(i)}$, $j = k+1, \ldots, i+1$
  $p_{-1}^{(i+1)} = 0$
  fin (pour)
  $k \leftarrow \theta(k) + 1$
  $p_j^{(k)} \leftarrow p_j^{(k)} + b w_j$, $j = 0, \ldots, k-1$
  aller à 1

fin.

Dans cet algorithme, $r \leftarrow s$ veut dire que nous attribuons à r la valeur de s. Nous allons utiliser cette notation dans les algorithmes que nous allons énoncer plus tard.
Lorsque $k = 0$, certains termes s'annulent dans cet algorithme, comme par exemple $d_0 = 0$. Le fait qu'ils soient nuls nous permet d'éviter l'utilisation d'une boucle avec "si" dans la troisième étape. Nous allons adapter cette technique à tous les algorithmes et processus que nous allons énoncer dans la suite.

Remarque 2.3 Pour avoir l'existence et l'unicité d'un polynôme orthogonal P_n, nous avons imposé la condition: P_n est de degré n. D'une manière similaire, nous pouvons imposer la condition de normalisation $P_n(0) = 1$, ce qui donne une autre permutation θ et par conséquent un autre polynôme intermédiaire P_n^θ.

Remarque 2.4 Il est facile de voir que les deux théorèmes 2.2 et 2.3 nous donnent la propriété suivante

$$P_{\theta(k)+1}(x) = W_{\theta(k)-k+1}(x)P_k(x) + \alpha_{k_0}P_{k_0}(x)$$

où $W_{\theta(k)-k+1}(x)$ est un polynôme de degré $\theta(k) - k + 1$. Pour la démonstration, il suffit de remarquer, dans le Théorème 2.2, que P_k divise P_i^θ pour $i = k, \ldots, \theta(k)$. Cette propriété est similaire à celle donnée par Draux [44], Gragg et Lindquist [62] et Gutknecht [65].

Remarque 2.5 Les coefficients $\alpha_i = d_{\theta(k)-i}$, $i = k+1, \ldots, \theta(k)$ du Théorème 2.3 dépendent uniquement de P_{k_0}. En conséquence, dans l'Algorithme 1, ils sont mémorisés pour calculer récursivement la somme $\sum_{i=k+1}^{\theta(k)} \alpha_i P_i^\theta$. Ceci montre que l'Algorithme 1 n'a aucun problème de stockage. En effet, si nous mémorisons les coefficients de chaque polynôme dans un vecteur, alors le codage de l'Algorithme 1 n'a besoin que de trois vecteurs sans compter le vecteur dont les composantes sont les coefficients d_i utilisés dans l'algorithme. Notons aussi que ce nombre de vecteurs est optimal, c'est-à-dire qu'il est impossible d'utiliser moins de trois vecteurs.

Il y a une autre manière de programmer le polynôme orthogonal régulier $P_{\theta(k)+1}$. Elle consiste à utiliser la relation de récurrence suivante

$$P_{\theta(k)+1} = xP_{\theta(k)}^\theta - \beta_k P_k - \beta_{k_0} P'_{k_0}, \tag{2.5}$$

où $\beta_k = C(x^{\theta(k)+1}P_{\theta(k)}^\theta)/C(x^{\theta(k)}P_k)$, $\beta_{k_0} = C(x^k P_{\theta(k)}^\theta)/C(x^{k-1}P'_{k_0})$, et P'_{k_0} s'obtient par la relation de récurrence suivante

$$\begin{cases} P'_{k_0} = P_{k_0} \\ \alpha_i = C(x^{\theta(i)}P'_{k_0})/C(x^{\theta(i)}P_i^\theta), \quad P'_{k_0} = P'_{k_0} - \alpha_i P_i^\theta, \quad i = k, \ldots, \theta(k). \end{cases}$$

2.3.2 Mise en oeuvre de la méthode ALA

Dans cette sous-section, nous allons montrer comment mettre en oeuvre la méthode ALA. Mais tout d'abord, d'une manière similaire à la biorthogonalité vue dans la sous-section 2.2.3, nous allons voir que l'orthogonalité définie ci-dessus n'est autre qu'un cas particulier de celle que nous avons traitée au premier chapitre et que ce cas particulier est obtenu en choisissant la forme sesquilinéaire g hermitienne.

Choix de la forme sesquilinéaire hermitienne g

Commençons par choisir les deux espaces vectoriels E et F. Ici, E est l'espace des polynômes $\mathbb{C}[X]$ et $F = E$.
Nous définissons la forme bilinéaire symétrique g_1 par

$$g_1(\psi, \varphi) = C(\psi\varphi), \quad \forall \psi, \varphi \in E,$$

La matrice de la restriction de la forme bilinéaire g_1 au sous-espace $V_n \times W_n$, où $V_n = W_n$ est engendré par $\{v_0 = w_0 = 1, v_1 = w_1 = x, \ldots, v_{n-1} = w_{n-1} = x^{n-1}\}$, a pour déterminant $H_n = G_n$. La permutation θ que nous utilisons ici est la même que celle du premier chapitre. Nous avons supposé que $\theta(i) < \infty$ pour tout $i \in \mathbb{N}$, notons bien que ceci revient à supposer que (E, g_1) est régulier. La permutation σ est prise égale à l'identité. Le rôle de la permutation θ est d'avoir un sous-espace régulier $V_n \times W_n^\theta$ de $E \times F$ avec W_n^θ comme sous-espace engendré par $\{w_{\theta(0)} = x^{\theta(0)}, w_{\theta(1)} = x^{\theta(1)}, \ldots, w_{\theta(n-1)} = x^{\theta(n-1)}\}$. Alors, la matrice de la restriction de g_1 à $V_n \times W_n^\theta$ est régulière et a pour déterminant $H_n^\theta = G_n^\theta$ qui est non nul. A l'aide du processus GBOR, nous pouvons alors construire deux bases g_1-orthogonales $\{P_0^\theta, P_1^\theta, \ldots, P_{n-1}^\theta\}$ de V_n et $\{Q_0^\theta, Q_1^\theta, \ldots, Q_{n-1}^\theta\}$ de W_n^θ. $\{P_0^\theta, P_1^\theta, \ldots, P_{n-1}^\theta\}$ et $\{\overline{Q^\theta}_0, \overline{Q^\theta}_1, \ldots, \overline{Q^\theta}_{n-1}\}$ sont aussi deux bases biorthogonales, mais cette fois ci, par rapport à la forme sesquilinéaire hermitienne g définie par

$$g(\psi, \varphi) = C(\psi\overline{\varphi}), \quad \forall \psi, \varphi \in E,$$

où $\overline{\varphi}$ représente le polynôme dont les coefficients sont les conjugués de ceux de φ.

La base $\{P_0^\theta, P_1^\theta, \ldots, P_{n-1}^\theta\}$ est la même que celle donnée par l'Algorithme 1, si la condition de normalisation utilisée dans GBOR consiste à imposer que les polynômes de cette base soient unitaires et de degrés égaux à leurs indices. L'Algorithme 1 se déduit aussi du processus GBOR : pour cela, il suffit de choisir récursivement les polynômes v_i en posant tout simplement, à l'étape j, $v_j = xP_{j-1}^\theta$.

A chaque choix de deux bases dénombrables $\{v_0, v_1, \ldots\}$ et $\{w_0, w_1, \ldots\}$ de $\mathbb{C}[X]$ correspondent deux autres bases dénombrables g_1-orthogonales. Mais le choix qui nous intéresse

ici est celui qui nous permet de retrouver tous les polynômes orthogonaux réguliers par rapport à la fonctionnelle C. C'est-à-dire le choix qui nous donne deux bases dénombrables g_1-biorthogonales $\{P_0^\theta, P_1^\theta, \ldots\}$ et $\{Q_{\theta(0)}^\theta, Q_{\theta(1)}^\theta, \ldots\}$ vérifiant $Q_i^\theta = P_i^\theta = \alpha_i P_i, \alpha_i \in \mathbb{C}$ pour tout polynôme orthogonal régulier P_i de degré i par rapport à la fonctionnelle C.

En plus du choix qui nous a permis d'établir l'Algorithme 1 et que nous considérons comme un premier choix, nous allons donner deux autres choix qui présentent une importance au point de vue utilisation et qui donnent deux manières différentes pour mettre en oeuvre la méthode ALA. Nous allons montrer dans le deuxième choix le résultat du théorème suivant qui généralise la relation de récurrence à trois termes classique reliant les polynômes orthogonaux réguliers.

Théorème 2.4 *Tout polynôme orthogonal régulier $P_{\theta(k)+1}^\theta$ vérifie une relation de récurrence de la forme*

$$P_{\theta(k)+1}^\theta = xP_{\theta(k)}^\theta - \alpha_{\theta(k)}P_k^\theta - \beta_{\theta(k)}P_{\theta(k-1)}^\theta,$$

où P_k^θ est le polynôme régulier de plus haut degré précédant $P_{\theta(k)+1}^\theta$. $P_{\theta(k)+1}^\theta$, $P_{\theta(k)}^\theta$, P_k^θ et $P_{\theta(k-1)}^\theta$ sont de degrés égaux à leurs indices respectifs.

Notons que les relations de récurrence que nous allons donner relient séparément les polynômes des deux bases $\{P_0^\theta, P_1^\theta, \ldots\}$ et $\{Q_{\theta(0)}^\theta, Q_{\theta(1)}^\theta, \ldots\}$. Nous les appliquerons à la méthode de Lanczos dans la section suivante. Ceci équivaut aussi, en changeant la variable x des polynômes Q_i^θ en y, à avoir deux bases biorthogonales par rapport à la forme bilinéaire g_2 définie sur $\mathbb{C}[X] \times \mathbb{C}[Y]$ par

$$g_2(x^i, y^j) = C(x^{i+j}), \quad \forall i, j \in \mathbb{N}.$$

♣ Le deuxième choix consiste à déterminer récursivement les polynômes des deux bases dénombrables $\{v_0, v_1, \ldots\}$ et $\{w_0, w_1, \ldots\}$ en posant

$$\begin{array}{l} v_j = xP_{j-1}^\theta, \quad j = 1, 2, \ldots, \\ w_j = x^{j-k}Q_k^\theta, \quad j = k+1, \ldots, \theta(k)+1, \quad k = 0, \theta(0)+1, \theta(\theta(0)+1)+1, \ldots, \end{array}$$

avec les indices des polynômes v_j et w_j représentant leurs degrés respectifs. En appliquant GBOR, nous déduisons les relations de récurrence suivantes

$$Q_j^\theta = x^{j-k}Q_k^\theta, \quad j = k+1, \ldots, \theta(k), \tag{2.6}$$

$$\left\{ \begin{array}{l} \alpha_{j-1} = C(xP_{j-1}^\theta Q_{\theta(k)}^\theta)/C(P_k^\theta Q_{\theta(k)}^\theta) \\ P_j^\theta = xP_{j-1}^\theta - \alpha_{j-1}P_k^\theta \end{array} \right. \quad j = k+1, \ldots, \theta(k), \tag{2.7}$$

$$\begin{cases} \alpha_{\theta(k)} = C(xP^\theta_{\theta(k)}Q^\theta_{\theta(k)})/C(P^\theta_k Q^\theta_{\theta(k)}) \\ \beta_{\theta(k)} = C(P^\theta_{\theta(k)}Q^\theta_k)/C(P^\theta_{\theta(k-1)}Q^\theta_{k-1}) \\ P^\theta_{\theta(k)+1} = xP^\theta_{\theta(k)} - \alpha_{\theta(k)}P^\theta_k - \beta_{\theta(k)}P^\theta_{\theta(k-1)} \\ Q^\theta_{\theta(k)+1} = xQ^\theta_{\theta(k)} - \sum_{i=k+1}^{\theta(k)} \alpha_{\theta(i)}Q^\theta_i - \alpha_{\theta(k)}Q^\theta_k - \beta_{\theta(k)}Q^\theta_{\theta(k-1)}. \end{cases} \tag{2.8}$$

En remplaçant les polynômes orthogonaux Q^θ_i par les P^θ_i puisque $Q^\theta_i = P^\theta_i = P_i$ pour tout i, ces trois équations nous donnent une mise en oeuvre de la méthode ALA dont le codage nécessite seulement trois vecteurs.

Les coefficients $\alpha_{\theta(i)}$ dans le calcul de $Q^\theta_{\theta(k)+1}$ peuvent être programmés d'une autre manière, en utilisant les polynômes de l'ensemble $\{P'^\theta_{k-1}, P'^\theta_k, \ldots, P'^\theta_{\theta(k)}, P'^\theta_{\theta(k)+1}\}$ qui est g_1-biorthogonal à $\{xQ^\theta_{\theta(k)}, Q^\theta_{\theta(k)}, Q^\theta_{\theta(k)-1}, \ldots, Q^\theta_k, Q^\theta_{\theta(k-1)}\}$ avec $P'^\theta_{k-1} = P^\theta_{k-1}$. La programmation des P'^θ_i se fait par la relation suivante qui les relie aux P^θ_i

$$\begin{cases} \lambda_i = C(P^\theta_i xQ^\theta_{\theta(k)})/C(P^\theta_k Q^\theta_{\theta(k)}) \\ P'^\theta_i = P^\theta_i - \lambda_i P^\theta_{k-1} \end{cases} \quad i = k, k+1, \ldots, \theta(k)+1. \tag{2.9}$$

A l'aide de ces polynômes, l'expression qui donne les $\alpha_{\theta(i)}$ est

$$\alpha_{\theta(i)} = -\beta_{\theta(k)}C(Q^\theta_{\theta(k-1)}P'^\theta_{\theta(i)})/C(Q^\theta_i P'^\theta_{\theta(i)}).$$

Si nous n'avons besoin que de calculer les polynômes P^θ_j, alors nous pouvons utiliser seulement la relation générale et simple suivante

$$\begin{cases} \alpha_j = C(x^{\theta(k)-k+1}P^\theta_j P^\theta_k)/C(x^{\theta(k)-k}P^\theta_k P^\theta_k) \\ \beta_j = C(P^\theta_j P^\theta_k)/C(P^\theta_{\theta(k-1)}P^\theta_{k-1}) \\ P^\theta_{j+1} = xP^\theta_j - \alpha_j P^\theta_k - \beta_j P^\theta_{\theta(k-1)} \end{cases} \quad j = k, k+1, \ldots, \theta(k), \tag{2.10}$$

pour $k = 0, \theta(0)+1, \theta(\theta(0)+1)+1, \ldots$. L'initialisation de cette relation de récurrence se fait en posant $P^\theta_0 = 1$ et $P^\theta_{-1} = 0$ avec $\theta(-1) = -1$.

♣ Le troisième choix consiste à donner les polynômes des deux bases dénombrables $\{v_0, v_1, \ldots\}$ et $\{w_0, w_1, \ldots\}$ d'une manière récursive en posant

$$\begin{aligned} v_j &= x^{j-k}P^\theta_k, \quad j = k+1, k+2, \ldots, n_k, \\ v_{j+1} &= xP^\theta_j, \quad j = n_k, n_k+1, \ldots, \theta(k), \\ w_j &= x^{j-k}Q^\theta_k, \quad j = k+1, k+2, \ldots, n_k, \\ w_{j+1} &= xQ^\theta_j, \quad j = n_k, n_k+1, \ldots, \theta(k), \end{aligned}$$

et ceci pour $k = 0, \theta(0)+1, \theta(\theta(0)+1)+1, \ldots$, avec $n_k = \lfloor(\theta(k)+k+1)/2\rfloor$ qui est la partie entière de $(\theta(k)+k+1)/2$. Le degré de chaque polynôme v_j est égal à son indice j, il en est de même pour chaque w_j. Dans ce troisième choix, la permutation

σ n'est plus prise égale à l'identité, mais égale à une permutation σ_1 que nous allons définir avec une autre permutation θ_1 qui remplace θ.

▷ Si $\theta(k)+k$ est pair (ce qui implique que $n_k = (\theta(k)+k)/2$), alors nous définissons σ_1 et θ_1 par

$$\begin{array}{l} \sigma_1(k+2j) = n_k + j, \quad j = 0, 1, \ldots, n_k - k, \\ \sigma_1(k+2j-1) = n_k - j, \quad j = 1, 2, \ldots, n_k - k, \\ \theta_1(k+2j) = n_k - j, \quad j = 0, 1, \ldots, n_k - k, \\ \theta_1(k+2j-1) = n_k + j, \quad j = 1, 2, \ldots, n_k - k \end{array} \tag{2.11}$$

et ceci pour $k = 0, \theta(0)+1, \theta(\theta(0)+1)+1, \ldots$. Les deux permutations σ_1 et θ_1 sont reliées à la permutation θ par deux relations

$$\theta o \theta_1 = \sigma_1 \quad \text{et} \quad \theta o \sigma_1 = \theta_1 \tag{2.12}$$

qui sont équivalentes puisque $\theta o \theta$ est la permutation identité. C'est la raison pour laquelle θ intervient dans la notation des deux bases g_1-biorthogonales $\{P^\theta_{\sigma_1(0)}, P^\theta_{\sigma_1(1)}, \ldots\}$ et $\{Q^\theta_{\theta_1(0)}, Q^\theta_{\theta_1(1)}, \ldots\}$ que nous obtenons en appliquant GBOR. Ces deux bases peuvent être écrites sous la forme

$$\begin{array}{cccccccc} \ldots & P^\theta_{n_k} & P^\theta_{n_k-1} & P^\theta_{n_k+1} & P^\theta_{n_k-2} & \ldots & P^\theta_k & P^\theta_{2n_k-k} \quad \ldots \\ \ldots & Q^\theta_{n_k} & Q^\theta_{n_k+1} & Q^\theta_{n_k-1} & Q^\theta_{n_k+2} & \ldots & Q^\theta_{2n_k-k} & Q^\theta_k \quad \ldots \end{array}$$

sans utiliser σ_1 et θ_1. Notons que pour deux entiers i et j, la quantité $C(P^\theta_i Q^\theta_{\theta(j)})$ ou $C(P^\theta_{\sigma_1(i)} Q^\theta_{\theta_1(j)})$ est nulle si $i \neq j$ et qu'elle est non nulle dans le cas contraire. Nous allons maintenant nous intéresser au calcul des éléments de ces deux bases qui sont donnés par les relations suivantes déduites du GBOR

$$\left\{ \begin{array}{l} P^\theta_j = x^{j-k} P^\theta_k \\ Q^\theta_j = x^{j-k} Q^\theta_k \end{array} \right. \quad j = k+1, k+2, \ldots, n_k, \tag{2.13}$$

$$\left\{ \begin{array}{l} \alpha_j = C(xP^\theta_{n_k+j} Q^\theta_{n_k+j}) / C(P^\theta_{n_k-j} Q^\theta_{n_k+j}) \\ \beta_j = C(xP^\theta_{n_k+j} Q^\theta_{n_k+j+1}) / C(P^\theta_{n_k-j-1} Q^\theta_{n_k+j+1}) \\ Q^\theta_{n_k+j+1} = xQ^\theta_{n_k+j} - \beta_{j-1} Q^\theta_{n_k-j+1} - \alpha_j Q^\theta_{n_k-j} \\ P^\theta_{n_k+j+1} = xP^\theta_{n_k+j} - \alpha_j P^\theta_{n_k-j} - \beta_j P^\theta_{n_k-j-1} \end{array} \right. \quad j = 0, 1, \ldots, n_k - k - 1, \tag{2.14}$$

avec $\beta_{-1} = 0$. Le polynôme orthogonal régulier $P^\theta_{\theta(k)+1}$ par rapport à C, ainsi que $Q^\theta_{\theta(k)+1}$ sont donnés par

$$\left\{ \begin{array}{l} \alpha_{n_k-k} = C(xP^\theta_{\theta(k)} Q^\theta_{\theta(k)}) / C(P^\theta_k Q^\theta_{\theta(k)}) \\ \beta_{n_k-k} = C(xP^\theta_{\theta(k)} Q^\theta_{k-1}) / C(P^\theta_{\theta(k-1)} Q^\theta_{k-1}) \\ \gamma = C(xQ^\theta_{\theta(k)} P^\theta_{\theta(k)-1}) / C(Q^\theta_{k+1} P^\theta_{\theta(k)-1}) \\ Q^\theta_{\theta(k)+1} = xQ^\theta_{\theta(k)} - \gamma Q^\theta_{k+1} - \alpha_{n_k-k} Q^\theta_k - \beta_{n_k-k} Q^\theta_{\theta(k-1)} \\ P^\theta_{\theta(k)+1} = xP^\theta_{\theta(k)} - \alpha_{n_k-k} P^\theta_k - \beta_{n_k-k} P^\theta_{\theta(k-1)}. \end{array} \right. \tag{2.15}$$

Les expressions des coefficients de ces relations de récurrence ont été obtenues en utilisant la propriété $C(P_i^\theta Q_{\theta(i)}^\theta) = C(P_k^\theta Q_{\theta(k)}^\theta)$ qui est toujours valable pour $i = k, \ldots, \theta(k)$.

▷ Si $\theta(k) + k$ est impair (ce qui implique que $n_k = (\theta(k) + k + 1)/2$), alors nous définissons σ_1 et θ_1 par

$$\begin{aligned}
&\sigma_1(k+2j) = n_k + j, \quad j = 0, 1, \ldots, n_k - k - 1,\\
&\sigma_1(k+2j-1) = n_k - j, \quad j = 1, 2, \ldots, n_k - k,\\
&\theta_1(k+2j) = n_k - j - 1, \quad j = 0, 1, \ldots, n_k - k - 1,\\
&\theta_1(k+2j-1) = n_k + j - 1, \quad j = 1, 2, \ldots, n_k - k,
\end{aligned}$$

et ceci pour $k = 0, \theta(0) + 1, \theta(\theta(0) + 1) + 1, \ldots$. Les deux permutations σ_1 et θ_1 sont toujours reliées à la permutation θ par (2.12). C'est la raison pour laquelle nous notons toujours $\{P_{\sigma_1(0)}^\theta, P_{\sigma_1(1)}^\theta, \ldots\}$ et $\{Q_{\theta_1(0)}^\theta, Q_{\theta_1(1)}^\theta, \ldots\}$ les deux bases g-biorthogonales que nous obtenons en appliquant GBOR. Ces deux bases peuvent être écrites sous la forme

$$\begin{array}{ccccccccc}
\ldots & P_{n_k}^\theta & P_{n_k-1}^\theta & P_{n_k+1}^\theta & P_{n_k-2}^\theta & \ldots & P_{2n_k-k-1}^\theta & P_k^\theta & \ldots\\
\ldots & Q_{n_k-1}^\theta & Q_{n_k}^\theta & Q_{n_k-2}^\theta & Q_{n_k+1}^\theta & \ldots & Q_k^\theta & Q_{2n_k-k-1}^\theta & \ldots
\end{array}$$

sans avoir besoin de σ_1 et θ_1. Nous allons maintenant nous intéresser au calcul des éléments de ces deux bases qui sont donnés par les relations suivantes déduites du GBOR

$$\left\{\begin{array}{l} P_j^\theta = x^{j-k} P_k^\theta \\ Q_{j-1}^\theta = x^{j-k-1} Q_k^\theta \end{array}\right. \quad j = k+1, k+2, \ldots, n_k \tag{2.16}$$

et

$$\left\{\begin{array}{l}
\alpha_j = C(xP_{n_k+j}^\theta Q_{n_k+j-1}^\theta)/C(P_{n_k-j}^\theta Q_{n_k+j-1}^\theta)\\
\beta_j = C(xP_{n_k+j}^\theta Q_{n_k+j}^\theta)/C(P_{n_k-j-1}^\theta Q_{n_k+j}^\theta)\\
Q_{n_k+j}^\theta = xQ_{n_k+j-1}^\theta - \beta_{j-1} Q_{n_k-j}^\theta - \alpha_j Q_{n_k-j-1}^\theta\\
P_{n_k+j+1}^\theta = xP_{n_k+j}^\theta - \alpha_j P_{n_k-j}^\theta - \beta_j P_{n_k-j-1}^\theta
\end{array}\right. \tag{2.17}$$

pour $j = 0, 1, \ldots, n_k - k - 1$, avec $\beta_{-1} = 0$. Le polynôme orthogonal régulier $P_{\theta(k)+1}^\theta$ par rapport à C, ainsi que $Q_{\theta(k)+1}^\theta$ sont donnés par

$$\left\{\begin{array}{l}
\alpha_{\theta(k)+1} = C(xP_{\theta(k)}^\theta Q_{\theta(k)}^\theta)/C(P_k^\theta Q_{\theta(k)}^\theta)\\
\beta_{\theta(k)+1} = C(xP_{\theta(k)}^\theta Q_{k-1}^\theta)/C(P_{\theta(k-1)}^\theta Q_{k-1}^\theta)\\
\gamma = C(xP_{\theta(k)}^\theta Q_{\theta(k)-1}^\theta)/C(P_{k+1}^\theta Q_{\theta(k)-1}^\theta)\\
Q_{\theta(k)+1}^\theta = xQ_{\theta(k)}^\theta - \alpha_{\theta(k)+1} Q_k^\theta - \beta_{\theta(k)+1} Q_{\theta(k-1)}^\theta\\
P_{\theta(k)+1}^\theta = xP_{\theta(k)}^\theta - \gamma P_{k+1}^\theta - \alpha_{\theta(k)+1} P_k^\theta - \beta_{\theta(k)+1} P_{\theta(k-1)}^\theta.
\end{array}\right. \tag{2.18}$$

Le calcul des coefficients de ces relations de récurrence se fait en utilisant la propriété $C(P_i^\theta Q_{\theta(i)}^\theta) = C(P_k^\theta Q_{\theta(k)}^\theta)$ qui est toujours valable pour $i = k, \ldots, \theta(k)$.

Changeons maintenant le choix que nous avons fait ci-dessus de la forme g. Notons g_1 la forme bilinéaire symétrique définie sur l'espace $\mathbb{C}[X]$ par

$$g_1(\psi,\varphi) = C(\psi\varphi), \quad \forall \psi,\varphi \in \mathbb{C}[X],$$

et considérons la forme bilinéaire symétrique g définie sur l'espace $E = \mathbb{C}[X] \times \mathbb{C}[X]$ par

$$g(\psi,\varphi) = g_1(\psi_1,\varphi_2) + g_1(\psi_2,\varphi_1), \quad \forall \psi = \begin{pmatrix} \psi_1 \\ \psi_2 \end{pmatrix}, \varphi = \begin{pmatrix} \varphi_1 \\ \varphi_2 \end{pmatrix} \in E. \tag{2.19}$$

Ce choix de la forme hermitienne g est similaire à celui qui nous a permis, à la fin du premier chapitre, de montrer que théoriquement le processus non hermitien de Lanczos n'est autre que le processus hermitien de Lanczos appliqué à un système hermitien étendu.

Choisissons d'une manière récursive deux familles dénombrables libres $\{v_0, v_1, \ldots\}$ et $\{w_0, w_1, \ldots\}$ de E qui vont nous donner deux familles g-biorthogonales $\{p_{\sigma_1(0)}, p_{\sigma_1(1)}, \ldots\}$ et $\{q_{\theta_1(0)}, q_{\theta_1(1)}, \ldots\}$ de E, où σ_1 et θ_1 sont deux permutations de $\mathbb{N}$.

$$\begin{aligned} v_j &= x^{j-k} p_k, \quad j = k+1, k+2, \ldots, n_k, \\ v_{j+1} &= x p_j, \quad j = n_k, n_k+1, \ldots, \theta(k), \\ w_j &= x^{j-k} q_k, \quad j = k+1, k+2, \ldots, n_k, \\ w_{j+1} &= x q_j, \quad j = n_k, n_k+1, \ldots, \theta(k), \end{aligned}$$

et ceci pour $k = 0, \theta(0)+1, \theta(\theta(0)+1)+1, \ldots$, avec toujours $n_k = \lfloor(\theta(k)+k+1)/2\rfloor$ étant la partie entière de $(\theta(k)+k+1)/2$.
Comme ci-dessus, pour définir récursivement les deux permutations σ_1 et θ_1, il y a deux cas qui se présentent

- ▷ Lorsque $\theta(k)+k$ est pair (ce qui implique que $n_k = (\theta(k)+k)/2$), σ_1 et θ_1 sont définis par (2.11). Donc, elles vérifient (2.12).
  Si nous supposons que les polynômes $\varphi_1, \varphi_2, \psi_1, \psi_2$ de $p_j = (\varphi_1, \varphi_2)^T$ et $q_j = (\psi_1, \psi_2)^T$ sont unitaires, alors le fait que g est symétrique nous permet d'avoir

$$p_j = q_j = x^{j-k} p_k, \quad j = k, k+1, \ldots, n_k, \tag{2.20}$$

et

$$\begin{cases} \alpha_j = g(x p_{n_k+j-1}, p_{n_k+j-1}) / g(p_{n_k-j+1}, p_{n_k+j-1}) \\ q_{n_k+j} = x p_{n_k+j-1} - \alpha_j p_{n_k-j+1} \\ \beta_j = g(q_{n_k+j}, q_{\theta(k_j)}) / g(p_{k_j}, q_{\theta(k_j)}) \\ p_{n_k+j} = q_{n_k+j} - \beta_j p_{k_j} \end{cases} \qquad j = 1, 2, \ldots, n_k - k + 1, \tag{2.21}$$

avec $k_j = n_k - j$ si $j \leq n_k - k$, $k_j = \theta(k-1)$ si $j = n_k - k + 1$ et ceci pour $k = 0, \theta(0)+1, \theta(\theta(0)+1)+1, \ldots$. Dans l'utilisation de cette relation nous n'avons pas besoin de tester si $j \leq n_k - k$ ou non. Nous pouvons aussi écrire la relation sans introduire k_j en posant $\theta(\theta(k)+1) = \theta(k-1)$, ce qui est possible puisque $\theta(\theta(k)+1)$ est calculée à l'étape suivante.

▷ Si $\theta(k) + k$ est impair (ce qui implique que $n_k = (\theta(k) + k + 1)/2$), alors nous définissons σ_1 et θ_1 par

$$\begin{aligned}
&\sigma_1(k+2j) = n_k - j - 1, \quad j = 0, 1, \ldots, n_k - k - 1,\\
&\sigma_1(k+2j-1) = n_k + j - 1, \quad j = 1, 2, \ldots, n_k - k,\\
&\theta_1(k+2j) = n_k + j, \quad j = 0, 1, \ldots, n_k - k - 1,\\
&\theta_1(k+2j-1) = n_k - j, \quad j = 1, 2, \ldots, n_k - k,
\end{aligned}$$

et ceci pour $k = 0, \theta(0)+1, \theta(\theta(0)+1)+1, \ldots$. Si nous supposons que les polynômes $\varphi_1, \varphi_2, \psi_1, \psi_2$ de $p_j = (\varphi_1, \varphi_2)^T$ et $q_j = (\psi_1, \psi_2)^T$ sont unitaires, alors comme pour le cas précédent nous aurons

$$q_j = p_j = x^{j-k} p_k, \quad j = k, k+1, \ldots, n_k - 1 \tag{2.22}$$

et

$$\left\{\begin{array}{l}
\alpha_{j-1} = g(xp_{n_k+j-1}, p_{n_k+j-1}) / g(p_{n_k-j}, p_{n_k+j-1})\\
q_{n_k+j} = xp_{n_k+j-1} - \alpha_{j-1} p_{n_k-j}\\
\beta_j = g(q_{n_k+j}, q_{\theta(k_j)}) / g(p_{k_j}, q_{\theta(k_j)})\\
p_{n_k+j} = q_{n_k+j} - \beta_j p_{k_j}
\end{array}\right. \qquad j = 0, 1, \ldots, n_k - k, \tag{2.23}$$

avec $k_j = n_k - j - 1$ si $j < n_k - k$, $k_j = \theta(k-1)$ si $j = n_k - k$ et ceci pour $k = 0, \theta(0)+1, \theta(\theta(0)+1)+1, \ldots$, avec $\alpha_{-1} = 0$.

Nous savons que pour tout entier j, p_j et q_j dépendent de la permutation θ. Il est donc naturel de poser

$$p_j = \begin{pmatrix} P_j^\theta \\ Q_j^\theta \end{pmatrix} \quad \text{et} \quad q_j = \begin{pmatrix} P_j'^\theta \\ Q_j'^\theta \end{pmatrix},$$

où les degrés des polynômes P_j^θ, Q_j^θ, $P_j'^\theta$ et $Q_j'^\theta$ sont égaux à leurs indices respectifs. Tous les polynômes orthogonaux réguliers par rapport à la fonctionnelle C sont retrouvés à l'aide de la famille $\{P_j^\theta\}_j$. Si le polynôme orthogonal régulier P_j, de degré j, que nous supposons unitaire existe alors il est égal à P_j^θ.

Signalons que les familles libres $\{p_{\sigma_1(0)}, p_{\sigma_1(1)}, \ldots\}$ et $\{\bar{q}_{\theta_1(0)}, \bar{q}_{\theta_1(1)}, \ldots\}$ que nous obtenons sont biorthogonales par rapport à la forme sesquilinéaire hermitienne qui, à tout élément (ψ, φ) de E, fait correspondre $g(\psi, \overline{\varphi})$.

Il est très important de noter que les polynômes P_j^θ, Q_j^θ, $P_j'^\theta$ et $Q_j'^\theta$ forment des bases g_1-biorthogonales, c'est-à-dire $\{P^\theta_{\sigma_1(0)}, P^\theta_{\sigma_1(1)}, \ldots\}$, $\{Q'^\theta_{\theta_1(0)}, Q'^\theta_{\theta_1(1)}, \ldots\}$ et $\{P'^\theta_{\theta_1(0)}, P'^\theta_{\theta_1(1)}, \ldots\}$, $\{Q^\theta_{\sigma_1(0)}, Q^\theta_{\sigma_1(1)}, \ldots\}$. Ceci simplifie le calcul des coefficients des relations (2.21) et (2.23). Cependant cette simplification est liée à la normalisation des vecteurs p_j et q_j et non pas aux polynômes P_j^θ, Q_j^θ, $P_j'^\theta$ et $Q_j'^\theta$ pour tout entier j. Donnons maintenant les équations équivalentes à (2.21) et (2.23) qui nous permettent de calculer et de normaliser séparément les polynômes P_j^θ, Q_j^θ, $P_j'^\theta$ et $Q_j'^\theta$ pour tout entier j. L'équation équivalente à (2.21) est

$$\left\{\begin{array}{l}
\alpha'_j = C(xP^\theta_{n_k+j-1}Q^\theta_{n_k+j-1})/C(P^\theta_{n_k-j+1}Q^\theta_{n_k+j-1}) \\
\beta'_j = C(xQ^\theta_{n_k+j-1}P^\theta_{n_k+j-1})/C(Q^\theta_{n_k-j+1}P^\theta_{n_k+j-1}) \\
P'^\theta_{n_k+j} = xP^\theta_{n_k+j-1} - \alpha'_j P^\theta_{n_k-j+1} \\
Q'^\theta_{n_k+j} = xQ^\theta_{n_k+j-1} - \beta'_j Q^\theta_{n_k-j+1} \\
\alpha_j = C(P'^\theta_{n_k+j}Q'^\theta_{\theta(k_j)})/C(P^\theta_{k_j}Q'^\theta_{\theta(k_j)}) \\
\beta_j = C(Q'^\theta_{n_k+j}P'^\theta_{\theta(k_j)})/C(Q^\theta_{k_j}P'^\theta_{\theta(k_j)}) \\
P^\theta_{n_k+j} = P'^\theta_{n_k+j} - \alpha_j P^\theta_{k_j} \\
Q^\theta_{n_k+j} = Q'^\theta_{n_k+j} - \beta_j Q^\theta_{k_j}
\end{array}\right. \qquad j = 1, 2, \ldots, n_k - k + 1, \qquad (2.24)$$

avec $k_j = n_k - j$ si $j \leq n_k - k$, $k_j = \theta(k-1)$ si $j = n_k - k + 1$. L'équation équivalente à (2.23) est

$$\left\{\begin{array}{l}
\alpha'_{j-1} = C(xP^\theta_{n_k+j-1}Q^\theta_{n_k+j-1})/C(P^\theta_{n_k-j}Q^\theta_{n_k+j-1}) \\
\beta'_{j-1} = C(xQ^\theta_{n_k+j-1}P^\theta_{n_k+j-1})/C(Q^\theta_{n_k-j}P^\theta_{n_k+j-1}) \\
P'^\theta_{n_k+j} = xP^\theta_{n_k+j-1} - \alpha'_{j-1}P^\theta_{n_k-j} \\
Q'^\theta_{n_k+j} = xQ^\theta_{n_k+j-1} - \beta'_{j-1}Q^\theta_{n_k-j} \\
\alpha_j = C(P'^\theta_{n_k+j}, Q'^\theta_{\theta(k_j)})/C(P^\theta_{k_j}, Q'^\theta_{\theta(k_j)}) \\
\beta_j = C(Q'^\theta_{n_k+j}P'^\theta_{\theta(k_j)})/C(Q^\theta_{k_j}P'^\theta_{\theta(k_j)}) \\
P^\theta_{n_k+j} = P'^\theta_{n_k+j} - \alpha_j P^\theta_{k_j} \\
Q^\theta_{n_k+j} = Q'^\theta_{n_k+j} - \beta_j Q^\theta_{k_j}
\end{array}\right. \qquad j = 0, 1, \ldots, n_k - k, \qquad (2.25)$$

avec $k_j = n_k - j - 1$ si $j < n_k - k$, $k_j = \theta(k-1)$ si $j = n_k - k$, $\beta'_{-1} = \alpha'_{-1} = 0$. Ces dernières équations nous permettent aussi de retrouver le troisième choix fait ci-dessus pour la forme bilinéaire g_1 et de construire ainsi d'une manière simple les deux bases g_1-biorthogonales correspondantes. Signalons que le codage relatif à la mise en oeuvre de ALA par ces deux équations nécessite seulement quatre vecteurs.

Il est possible d'écrire d'une manière simple les deux équations (2.21) et (2.23) en les regroupant en une seule. C'est-à-dire, en d'autres termes, regrouper le cas où $\theta(k)+k$ est pair et celui où il est impair. En effet, si nous posons $\theta(\theta(k)+1) = \theta(k-1)$ avant d'avoir calculé la vraie valeur de $\theta(\theta(k)+1)$, alors, par une simple identification, (2.21) et (2.23) sont regroupées dans la relation suivante

$$\left\{\begin{array}{l} \alpha_j = g(xp_j, p_j)/g(p_{\theta(j)}, p_j) \\ q_{j+1} = xp_j - \alpha_j p_{\theta(j)} \\ \beta_j = g(q_{j+1}, q_{\theta(\theta(j+1))})/g(p_{\theta(j+1)}, q_{\theta(\theta(j+1))}) \\ p_{j+1} = q_{j+1} - \beta_j p_{\theta(j+1)} \end{array}\right. \quad j = \theta(n_k), \theta(n_k)+1, \ldots, \theta(k), \qquad (2.26)$$

avec $\alpha_{n_k-1} = 0$. D'une manière analogue, nous regroupons (2.24) et (2.25) dans l'équation suivante

$$\left\{\begin{array}{l} \alpha'_j = C(xP^\theta_j Q^\theta_j)/C(P^\theta_{\theta(j)} Q^\theta_j) \\ \beta'_j = C(xQ^\theta_j P^\theta_j)/C(Q^\theta_{\theta(j)} P^\theta_j) \\ P'^\theta_{j+1} = xP^\theta_j - \alpha'_j P^\theta_{\theta(j)} \\ Q'^\theta_{j+1} = xQ^\theta_j - \beta'_j Q^\theta_{\theta(j)} \\ \alpha_j = C(P'^\theta_{j+1} Q'^\theta_{\theta(\theta(j+1))})/C(P^\theta_{\theta(j+1)} Q'^\theta_{\theta(\theta(j+1))}) \\ \beta_j = C(Q'^\theta_{j+1} P'^\theta_{\theta(\theta(j+1))})/C(Q^\theta_{\theta(j+1)} P'^\theta_{\theta(\theta(j+1))}) \\ P^\theta_{j+1} = P'^\theta_{j+1} - \alpha_j P^\theta_{\theta(j+1)} \\ Q^\theta_{j+1} = Q'^\theta_{j+1} - \beta_j Q^\theta_{\theta(j+1)} \end{array}\right. \quad j = \theta(n_k), \theta(n_k)+1, \ldots, \theta(k), \qquad (2.27)$$

avec $\alpha'_{n_k-1} = \beta'_{n_k-1} = 0$. Comme pour (2.26), dans (2.27) nous posons toujours $\theta(\theta(k)+1) = \theta(k-1)$.
Lorsque $n_k = k$, il est clair que nous retrouvons bien les relations de récurrence à trois termes du cas régulier. Les relations de l'équation (2.27) sont obtenues en choisissant la forme bilinéaire g au lieu de g_2. Dans le cas régulier, (2.27) est la double récurrence donnée dans [57] que Nachtigal et Freund ont utilisée pour mettre en oeuvre la méthode QMR.

Si nous n'avons besoin que de calculer les polynômes P^θ_j, alors nous utilisons l'équation suivante obtenue à partir de (2.27)

$$\left\{\begin{array}{l} \alpha'_j = C(xP^\theta_j P^\theta_j)/C(P^\theta_{\theta(j)} P^\theta_j) \\ P'^\theta_{j+1} = xP^\theta_j - \alpha'_j P^\theta_{\theta(j)} \\ \alpha_j = C(P'^\theta_{j+1} P'^\theta_{\theta(\theta(j+1))})/C(P^\theta_{\theta(j+1)} P'^\theta_{\theta(\theta(j+1))}) \\ P^\theta_{j+1} = P'^\theta_{j+1} - \alpha_j P^\theta_{\theta(j+1)} \end{array}\right. \quad j = \theta(n_k), \theta(n_k)+1, \ldots, \theta(k), \qquad (2.28)$$

avec $\alpha'_{n_k-1} = \beta'_{n_k-1} = 0$ et $\theta(\theta(k)+1) = \theta(k-1)$.
Pour le calcul des coefficients de cette relation de récurrence nous nous servons des égalités suivantes qui se déduisent de (2.28)

- $C(P^\theta_{\theta(j)} P^\theta_j) = C(x^{\theta(k)} P^\theta_k)$.
- $C(P^\theta_{\theta(j+1)} P'^\theta_{\theta(\theta(j+1))})$ est égal à $C(x^{\theta(k)} P^\theta_k)$ si $j \neq \theta(k)$ et à $C(x^{k-1} P^\theta_{\theta(k-1)})$ lorsque $j = \theta(k)$.

- $C(xP_j^\theta P_j^\theta) = \sum_{i=\theta(j)}^{j+1} p_{i-1}^{(j)} C(x^i P_j^\theta)$ avec $P_j^\theta = \sum_{i=0}^{j} p_i^{(j)} x^i$, $p_j^{(j)} = 1$.

- $C(P'^\theta_{j+1} P'^\theta_{\theta(\theta(j+1))}) = C(x^{\theta(k)} P_k^\theta)$ si $j \neq \theta(k)$. Dans le cas contraire nous avons $C(P'^\theta_{j+1} P'^\theta_{\theta(\theta(j+1))}) = C(xP_j^\theta P'^\theta_{j+1}) = \sum_{i=\theta(j)}^{j+2} p_{i-1}'^{(j+1)} C(x^i P_j^\theta)$ avec $P'^\theta_{j+1} = \sum_{i=0}^{j+1} p_i'^{(j+1)} x^i$, $p_{j+1}'^{(j+1)} = 1$.

Pour ces relations, nous avons besoin de connaître les quantités $C(x^i P_j^\theta)$, $i = \theta(j), \ldots, j+2$. Pour les calculer, nous nous servons des trois propriétés suivantes

$$\begin{aligned} &C(x^i P_j^\theta) = C(x^{i+1} P_{j-1}^\theta), \quad i = \theta(j), \ldots, j-2, \\ &C(x^{j-1} P_j^\theta) = C(x^j P_{j-1}^\theta) - \alpha'_{j-1} C(x^{\theta(k)} P_k^\theta), \\ &C(x^j P_j^\theta) = C(x^{j+1} P_{j-1}^\theta) - \alpha_{j-1} C(x^{\theta(k)} P_k^\theta) - \alpha'_{j-1} C(x^{\theta(k)+1} P_k^\theta). \end{aligned}$$

Les deux quantités restantes $C(x^{j+1} P_j^\theta)$ et $C(x^{j+2} P_j^\theta)$ se calculent d'une manière directe, c'est-à-dire en appliquant directement la fonctionnelle C.

2.4 Application à la méthode de Lanczos

Nous commençons tout d'abord par donner une description de la méthode de Lanczos. Les notations utilisées dans cette description sont celles de [18, 16, 23].

2.4.1 Description

Nous voulons trouver la solution du système linéaire suivant

$$Ax = b,$$

où $A \in \mathbb{C}^{n \times n}$ est supposée non singulière, $b \in \mathbb{C}^n$ et $x \in \mathbb{C}^n$.
Soient x_0 et y_0 deux vecteurs arbitraires de $\mathbb{C}^n$. Définissons les deux suites $(x_k)_k$ et $(r_k)_k$ de vecteurs par les deux conditions

$$x_k - x_0 \in K_k(A, r_0), \tag{2.29}$$

$$r_k = b - Ax_k \perp K_k(A^*, y_0), \tag{2.30}$$

où $K_k(A, r) = Vect(r, Ar, \cdots, A^{k-1} r)$. La méthode de Lanczos est complètement définie par les deux conditions (2.29) et (2.30). Elle consiste à projeter récursivement le résidu initial r_0 sur l'espace de Krylov $K_k(A, Ar_0)$ biorthogonalement à $K_k(A^*, y_0)$ par rapport au produit hermitien $< ., . >_n$ de $\mathbb{C}^n$. $< ., . >_n$ remplace la forme sesquilinéaire g introduite

dans le premier chapitre.
L'équation (2.4) nous permet d'écrire

$$x_k - x_0 = -\alpha_1 r_0 - \alpha_2 A r_0 - \cdots - \alpha_k A^{k-1} r_0. \tag{2.31}$$

En multipliant (2.31) par A et en soustrayant et en ajoutant b, nous obtenons

$$r_k = r_0 + \alpha_1 A r_0 + \cdots + \alpha_k A^k r_0.$$

L'équation (2.30) nous donne

$$< r_k, A^{*^i} y_0 >_n = 0 \quad \text{pour} \quad i = 0, \cdots, k-1. \tag{2.32}$$

Si nous posons $c_i =< A^i r_0, y_0 >_n$, alors (2.32) est équivalente au système suivant

$$(R) \begin{cases} \alpha_1 c_1 + \alpha_2 c_2 + \cdots + \alpha_k c_k = -c_0 \\ \alpha_1 c_2 + \alpha_2 c_3 + \cdots + \alpha_k c_{k+1} = -c_1 \\ \cdots \quad \cdots \quad \cdots \quad \cdots \quad \cdots \quad \cdots \quad \cdots \quad \cdots \quad \cdots \\ \alpha_1 c_k + \alpha_2 c_{k+1} + \cdots + \alpha_k c_{2k-1} = -c_{k-1}. \end{cases}$$

Si nous considérons le polynôme $P_k(\xi) = 1 + \alpha_1 \xi + \cdots + \alpha_k \xi^k$, alors $r_k = P_k(A) r_0$. Définissons la fonctionnelle C sur $\mathbb{C}[X]$ par

$$C(\xi^i) = c_i =< A^i r_0, y_0 >_n, \quad i = 0, 1, \ldots$$

et la fonctionnelle $C^{(1)}$ par

$$C^{(1)}(\xi^i) = C(\xi^{i+1}), \quad i = 0, 1, \ldots.$$

Le polynôme P_k vérifie

$$\begin{cases} C(\xi^i P_k(\xi)) = 0, \quad i = 0, \ldots, k-1, \\ P_k(0) = 1. \end{cases}$$

Donc, P_k est le polynôme orthogonal par rapport à la fonctionnelle linéaire C, normalisé par la condition $P_k(0) = 1$.
Soit $P_k^{(1)}$ le polynôme unitaire de degré k vérifiant

$$C^{(1)}(\xi^i P_k^{(1)}(\xi)) = 0 \quad \text{pour} \quad i = 0, \ldots, k-1.$$

Les familles $(P_k^{(1)})_k$ et $(P_k)_k$ sont appelées adjacentes [119]. Nous pouvons voir facilement que pour chaque $k \in \mathbb{N}^*$, P_k et $P_k^{(1)}$ existent et sont uniques si et seulement si le déterminant de Hankel

$$H_k^{(1)} = \begin{vmatrix} c_1 & c_2 & \cdots & c_k \\ c_2 & c_3 & \cdots & c_{k+1} \\ \vdots & \vdots & & \vdots \\ c_k & c_{k+1} & \cdots & c_{2k-1} \end{vmatrix}$$

est non nul. Pour définir d'une manière unique les deux suites $(P_k^\theta)_k$ et $(P_k^{(1)^\theta})_k$ avec seulement une permutation θ, $(P_k^\theta)_k$ et $(P_k^{(1)^\theta})_k$ vont être normalisés par les deux conditions $P_k^\theta(0) = 1$ et $P_k^{(1)^\theta}$ unitaire.
Si P_k est non régulier, alors nous posons

$$r_k = P_k^\theta(A)r_0.$$

Le polynôme P_k^θ vérifie

$$\begin{cases} C(\xi^{\theta(i)} P_k^\theta(\xi)) = 0 \quad \text{pour} \quad i = 0, \ldots, k-1, \\ P_k^\theta(0) = 1. \end{cases}$$

En conséquence, (2.32) devient

$$< r_k, A^{*^{\theta(i)}} y_0 >_n = 0 \quad \text{pour} \quad i = 0, \ldots, k-1. \tag{2.33}$$

(2.9) est équivalente au système linéaire suivant

$$(R') \begin{cases} \alpha_1 c_{\theta(0)+1} + \alpha_2 c_{\theta(0)+2} + \cdots + \alpha_k c_{\theta(0)+k} = -c_{\theta(0)} \\ \alpha_1 c_{\theta(1)+1} + \alpha_2 c_{\theta(1)+2} + \cdots + \alpha_k c_{\theta(1)+k} = -c_{\theta(1)} \\ \cdots \quad \cdots \quad \cdots \quad \cdots \quad \cdots \quad \cdots \quad \cdots \quad \cdots \quad \cdots \\ \alpha_1 c_{\theta(k-1)+1} + \alpha_2 c_{\theta(k-1)+2} + \cdots + \alpha_k c_{\theta(k-1)+k} = -c_{\theta(k-1)}. \end{cases}$$

En accord avec la définition de θ, le déterminant de ce système (R'), noté $H_k^{(1)^\theta}$, est non nul. Ceci montre que (R') a une solution unique.

Nous allons rappeler maintenant deux résultats fondamentaux en ce qui concerne la méthode de Lanczos.

Théorème 2.5 *Si les vecteurs $A^{*^{\theta(0)}} y_0, A^{*^{\theta(1)}} y_0, \ldots, A^{*^{\theta(n-1)}} y_0$ sont linéairement indépendants, alors il existe un entier $k \leq n$ tel que*

$$r_k = 0 \text{ et } x_k = x.$$

Théorème 2.6 *Si k est le plus petit entier tel que $r_0, Ar_0, \ldots, A^k r_0$ soient linéairement dépendants, alors*

$$r_k = 0 \text{ et } x_k = x.$$

Preuve :

k est le plus petit entier tel que $r_0, Ar_0, \ldots, A^k r_0$ soient linéairement dépendants, donc il existe $\beta_0, \beta_1, \ldots, \beta_k$ non tous nuls avec

$$\beta_0 r_0 + \beta_1 A r_0 + \cdots + \beta_k A^k r_0 = 0.$$

β_0 et β_k doivent être différents de zéro puisque dans le cas contraire, $r_0, Ar_0, \ldots, A^{k-1} r_0$ seraient linéairement dépendants.
Alors, il existe $\delta_1, \delta_2, \ldots, \delta_k$ pour lesquels

$$r_0 + \delta_1 A r_0 + \cdots + \delta_k A^k r_0 = 0$$

avec

$$\delta_i = \beta_i / \beta_0 \quad \text{pour} \quad i = 1, \ldots, k.$$

Ceci nous permet de conclure que les coefficients $\delta_1, \delta_2, \ldots, \delta_k$ forment une solution du système (R'). Comme la solution de (R') est unique, alors $r_k = 0$ et $x_k = x$. ∎

Remarque 2.6 Le Théorème 2.6 montre que le choix de x_0 est aussi important que celui de y_0, et que les coefficients $\delta_1, \delta_2, \ldots, \delta_k$ sont ceux du polynôme minimal, pour le vecteur r_0, de la restriction de A à $Vect(r_0, Ar_0, \ldots, A^{k-1} r_0)$.

Une étude sur les divers algorithmes de type Lanczos pour résoudre des systèmes d'équations linéaires a été faite dans [18]. Ici, nous présentons une application de ALA à la méthode Lanczos/Orthodir qui est décrite dans [74, 127].

2.4.2 La méthode Lanczos/Orthodir

Plusieurs algorithmes de type Lanczos/Orthodir peuvent être envisagés, notamment ceux donnés dans [18]. En particulier l'algorithme connu sous le nom de Biodir [65].
La première mise en oeuvre de ALA appliquée à la méthode de Lanczos n'a aucun problème de stockage, nous avons donc choisi de l'appliquer dans cette sous-section à la méthode Lanczos/Orthodir. Les autres mises en oeuvre de la méthode ALA vont être appliquées, dans la sous-section suivante, au processus non hermitien de Lanczos et, plus loin, aux approximants de Padé, à l'extension de l'algorithme *qd* et à l'epsilon algorithme.

Nous avons deux cas à distinguer

1. Si $\theta(k) = k$ (cas régulier).

   Le calcul de P_{k+1} se fait d'une manière similaire au MRZ dans [16]. C'est-à-dire que nous utilisons les deux relations suivantes

$$(F_1) \begin{cases} P_{k+1}(\xi) = P_k(\xi) - \lambda_k \xi P_k^{(1)}(\xi) \\ \lambda_k = C(\xi^k P_k(\xi)) / C(\xi^{k+1} P_k^{(1)}(\xi)) \end{cases}$$

et

$$(F_2)\begin{cases} P_{k+1}^{(1)}(\xi) = (\xi - a_k)P_k^{(1)}(\xi) - b_k P_{\theta(k-1)}^{(1)}(\xi) \\ b_k = C(\xi^{k+1}P_k^{(1)}(\xi))/C(\xi^k P_{\theta(k-1)}^{(1)}(\xi)) \\ a_k = [C(\xi^{k+2}P_k^{(1)}(\xi)) - b_k C(\xi^{k+1}P_{\theta(k-1)}^{(1)}(\xi))]/C(\xi^{k+1}P_k^{(1)}(\xi)). \end{cases}$$

2. Si $\theta(k) \neq k$(cas singulier).

   Pour $i = k, \ldots, \theta(k) - 1$, nous avons

$$P_{i+1}^{\theta}(\xi) = P_i^{\theta}(\xi) - \lambda_i \xi P_i^{(1)^{\theta}}(\xi)$$

   avec

$$\lambda_i = C(\xi^{\theta(i)}P_i^{\theta}(\xi))/C(\xi^{\theta(k)+1}P_k^{(1)^{\theta}}(\xi)).$$

   Par récurrence, nous montrons que

$$C(\xi^{\theta(i)}P_i^{\theta}(\xi)) = C(\xi^{\theta(i)}P_k^{\theta}(\xi)) \quad \text{pour} \quad i = k, \ldots, \theta(k).$$

   Ce qui nous donne la formule suivante

$$(F_3)\begin{cases} P_{i+1}^{\theta}(\xi) = P_i^{\theta}(\xi) - \lambda_i \xi P_i^{(1)^{\theta}}(\xi) \\ \lambda_i = C(\xi^{\theta(i)}P_k^{\theta}(\xi))/C(\xi^{\theta(k)+1}P_k^{(1)^{\theta}}(\xi)) \end{cases} \quad i = k, \ldots, \theta(k).$$

   A partir du Théorème 2.2 et du Théorème 2.3, nous obtenons respectivement les deux relations suivantes

$$(F_4)\begin{cases} P_{i+1}^{(1)^{\theta}}(\xi) = \xi P_i^{(1)^{\theta}}(\xi) - \beta_i P_k^{(1)}(\xi) \\ \beta_i = C(\xi^{\theta(k)+2}P_i^{(1)^{\theta}}(\xi))/C(\xi^{\theta(k)+1}P_k^{(1)^{\theta}}(\xi)) \end{cases} \quad i = k, \ldots, \theta(k) - 1$$

   et

$$(F_5)\begin{cases} P_{\theta(k)+1}^{(1)} = \xi P_{\theta(k)}^{(1)^{\theta}} + \sum_{i=k+1}^{\theta(k)} \alpha_i P_i^{(1)^{\theta}} + \alpha_k P_k^{(1)} + \alpha_{k-1} P_{\theta(k-1)}^{(1)} \\ \alpha_{k-1} = -C(\xi^{\theta(k)+1}P_k^{(1)})/C(\xi^k P_{\theta(k-1)}^{(1)}) \neq 0 \\ \alpha_k = -[C(\xi^{\theta(k)+2}P_{\theta(k)}^{(1)^{\theta}}) + \alpha_{k-1}C(\xi^{\theta(k)+1}P_{\theta(k-1)}^{(1)})]/C(\xi^{\theta(k)+1}P_k^{(1)}) \\ \alpha_i = C(\xi^{\theta(i)+1}P_{\theta(k-1)}^{(1)})/C(\xi^k P_{\theta(k-1)}^{(1)}), \quad i = k+1, \ldots, \theta(k). \end{cases}$$

Remarque 2.7 Ces deux cas peuvent être regroupés en un seul, car lorsque $\theta(k) = k$, (F_3) et (F_5) sont équivalentes respectivement à (F_1) et (F_2). Donc, pour appliquer ALA à la méthode Lanczos/Orthodir selon sa première mise en oeuvre, nous avons seulement besoin des formules (F_3), (F_4) et (F_5).

Maintenant, nous avons tout ce qu'il faut pour donner l'algorithme qui nous permet d'éviter le "Look-Ahead" dans Lanczos/Orthodir en utilisant la première mise en oeuvre. Cet algorithme comprend trois étapes

- la première étape est l'initialisation,
- la deuxième étape consiste à savoir où se trouve le polynôme orthogonal régulier suivant, c'est-à-dire déterminer $\theta(k)$ à l'itération k,
- la troisième étape consiste à calculer les itérés x_{k+1}, z_{k+1} et le vecteur résidu r_{k+1} à l'itération k.

Posons $z_k = P_k^{(1)^\theta}(A)r_0, k = 0, 1, \ldots$. Nous posons ceci pour que l'indice k de z_k soit égal au degré du polynôme $P_k^{(1)^\theta}$.

Algorithme 2

- Etape 1 : Initialisation
  choisir x_0 et y_0 arbitrairement dans $\mathbb{C}^n$,
  poser $r_0 = b - Ax_0$, $z_0 = r_0$, $z_{-1} = (0, 0, \ldots 0)^T$,
  $h_{-1} = 1$, $\theta(-1) = -1$ et $k = 0$.

- Etape 2 (Le calcul de $\theta(k)$)
  1 $i = 0$
  2 $e_i =< y_{k+i}, r_k >_n$
  $y_{k+i+1} = A^* y_{k+i}$
  $h_{k+i} =< y_{k+i+1}, z_k >_n$
  $d_i =< y_{k+i+1}, z_{\theta(k-1)} >_n$
  si $|h_{k+i}| < \varepsilon_1$ pour une tolérance ε_1, alors
  $i = i + 1$
  aller à 2
  fin (si)
  $\theta(k) = k + i$.

• Etape 3

$b_k = h_{\theta(k)}/h_{k-1}$
$w = -z_{\theta(k-1)}$
pour $i = k, \ldots, \theta(k)$
$\alpha_i = d_{\theta(k)-i}/h_{\theta(k)}$
$w = w + \alpha_i z_i$
$\lambda_i = e_{\theta(k)-i}/h_{\theta(k)}$
$x_{i+1} = x_i + \lambda_i z_i$
$r_{i+1} = r_i - \lambda_i A z_i$
$\beta_i = < y_{\theta(k)+1}, Az_i >_n / h_{\theta(k)}$
$z_{i+1} = Az_i - \beta_i z_k$
fin (pour)
$z_{\theta(k)+1} \leftarrow z_{\theta(k)+1} + b_k w$
$k \leftarrow \theta(k) + 1$
aller à 1
fin.

Il est important de noter qu'à chaque itération de cet algorithme nous effectuons 4 produits scalaires et deux produits d'une matrice par un vecteur dont l'un concerne A et l'autre A^*. Pour chaque itération k du cas régulier (c-à-d lorsque $\theta(\{0,1,\ldots,k-1\}) = \{0,1,\ldots,k-1\}$), nous effectuons 5 opérations de type "SAXPY" ; ce nombre est réduit à 4 dans le cas singulier (c-à-d lorsque $\theta(\{0,1,\ldots,k-1\}) \neq \{0,1,\ldots,k-1\}$).
Cet algorithme n'a besoin globalement que de 9 vecteurs : 7 dans $\mathbb{C}^n$ et les deux autres dans $\mathbb{C}^m$ avec $m = \max_k(\theta(k) - k + 1) < n$. Ce nombre de vecteurs est optimal (c'est-à-dire qu'il est le plus petit nombre de vecteurs que nous puissions utiliser).
Le vecteur w introduit dans l'Algorithme 2 est mémorisé à la place de $z_{\theta(k-1)}$.

Remarque 2.8 1. Durant la détermination de $\theta(k)$ à l'itération k, nous mémorisons les composantes des deux vecteurs $e = (e_j)_j$ et $d = (d_j)_j$ de $\mathbb{C}^m$ avec $m = \theta(k) - k + 1 < n$.

2. En pratique, au lieu de tester si la quantité $< y_k, Az_k >_n$ est égale à zéro, nous testons si $| < y_k, Az_k >_n |$ est plus petit qu'une tolérance $\varepsilon_1 > 0$.
Nous pouvons aussi avoir un problème de PDPZ dû au fait que $| < y_k, Az_k >_n | \leq \varepsilon_1$.
Dans ce cas, les relations correspondantes ne doivent pas être nécessairement les mêmes que pour la DPZ.
Il peut y avoir aussi une DPZ à laquelle nous ne pouvons remédier qu'en changeant

le choix de y_0 ou x_0.

D'une manière similaire, nous pouvons appliquer les deux autres mises en oeuvre de ALA à la méthode Lanczos/Orthodir. Nous obtenons alors deux applications différentes de ALA à l'algorithme connu sous le nom de Biodir [65]. Mais ces deux applications nécessitent malheureusement le stockage de $m+6$ vecteurs avec $m = \max_k(\theta(k) - k + 1) < n$. Ce nombre de vecteurs est plus petit que celui que nous utilisons pour une stratégie de "Look-Ahead" où il est égal à $2m+5$. Dans notre cas, ce nombre peut être réduit à 7 vecteurs, indépendamment de la valeur de m, si nous effectuons à chaque itération du cas singulier trois produits d'une matrice par un vecteur au lieu de deux.

Nous pouvons appliquer la technique de Horner à Lanczos/Orthodir pour obtenir un nouvel algorithme appelé HMRZ, voir [28]. Avec de petites modifications, ce nouvel algorithme HMRZ est exactement similaire à l'Algorithme 2 qui utilise la première mise en oeuvre de ALA. Mais la technique de Horner se limite à la programmation et ne donne aucune signification aux itérés du point de vue de l'orthogonalité ou de la biorthogonalité.

Signalons aussi que les trois mises en oeuvre de ALA s'appliquent également à Lanczos/Orthores et à Lanczos/Orthomin qui est le plus stable des trois algorithmes.

2.4.3 Résultats numériques

La programmation de tout ce travail a été faite sur un "Sun SparcStation10" dont la précision machine est égale à $2.22\ 10^{-16}$. Les tests ont été réalisés à l'aide du Compilateur FORTRAN 77. Nous désignons par $\|r_k\|_n$, dans cette sous-section, la norme du résidu obtenu à l'itération k de l'Algorithme 2.

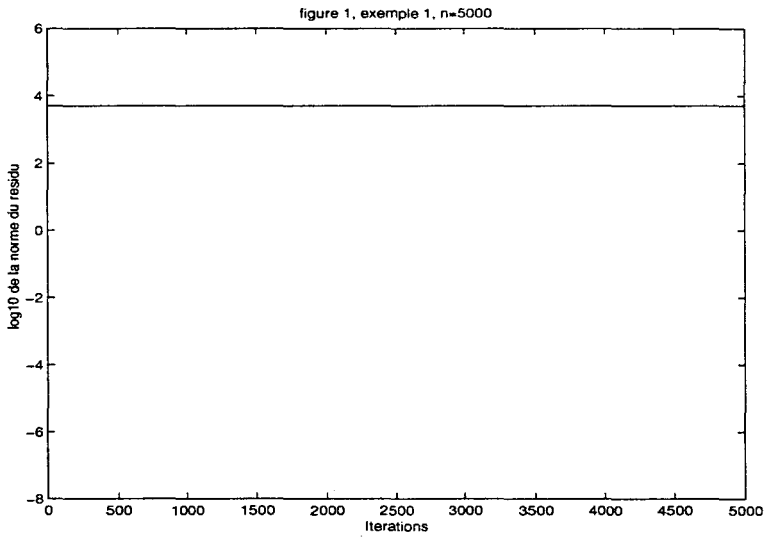
Exemple 1

Nous considérons l'exemple étudié dans [12].

$$\begin{pmatrix} 0 & 0 & 0 & \cdots & 0 & -1 \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \\ 3 \\ \vdots \\ n \end{pmatrix} = \begin{pmatrix} -n \\ 1 \\ 2 \\ \vdots \\ n-1 \end{pmatrix}.$$

Pour la dimension de la matrice et des vecteurs initiaux, nous choisissons $n = 5000$, $y_0 = (1, 0, 0, \ldots, 0, 0, 1)^T$ et $x_0 = (0, 0, \ldots, 0)^T$. Pour une valeur de *tol* égale à 10^{-8}, nous

obtenons la figure suivante



Les valeurs prises par la permutation θ sont

$$\theta(0) = 0, \quad \theta(k) = 4999 - k \text{ pour } k = 1, 2, \ldots, 4998 \text{ et } \theta(4999) = 4999.$$

Comme le tracé l'indique, il y a une stagnation à partir de $k = 1$ jusqu'à l'itération $k = 4998$. A la fin de cette stagnation, nous obtenons $\|r_{4999}\|_n = 1.77\ 10^5$ et $\|r_{5000}\|_n = 7.93\ 10^{-7}$.

Exemple 2

Nous considérons une matrice non symétrique obtenue à partir de la discrétisation de l'équation aux dérivées partielles à trois dimensions

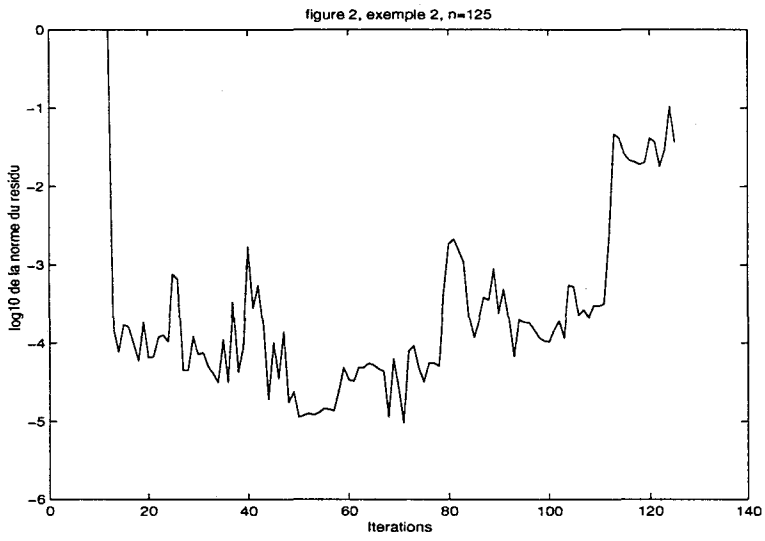
$$Lu = f \quad \text{sur } [0,1] \times [0,1] \times [0,1],$$

où

$$Lu = -\triangle u + x\frac{\partial u}{\partial x} + y\frac{\partial u}{\partial y} + z\frac{\partial u}{\partial z} - u,$$

avec les conditions de Dirichlet aux bords $u = 0$. L'opérateur a été discrétisé en utilisant un schéma à sept points sur une grille uniforme $5 \times 5 \times 5$ avec un pas h égal à $1/6$. Ceci nous donne une matrice creuse non symétrique de dimension $n = 125$, avec 725 éléments non nuls.
En utilisant l'Algorithme 2 avec $tol = 10^{-8}$, $y_0 = (0, 0, \ldots, 0, 0, 1)^T$, $x_0 = (0, 0, \ldots, 0)^T$ et

$b = (1, 0, 0, \ldots, 0)^T$, nous obtenons la courbe suivante



Les valeurs prises par la permutation θ sont

$$\theta(k) = 12 - k \text{ pour } k = 0, 1, \ldots, 12 \text{ et } \theta(k) = k \text{ pour } k = 13, 14, \ldots, 124.$$

Comme le début du tracé le montre, nous avons une stagnation de $k = 0$ jusqu'à $k = 12$. Après cette stagnation, nous remarquons que la courbe présente des pics ; ce qui est normal puisque ceux-ci caractérisent les méthodes de type Lanczos.
En arithmétique exacte, la norme du résidu doit être égale à zéro lorsque l'itération k coïncide avec la dimension n du système. Ici, ce n'est malheureusement pas le cas parce que nous travaillons avec une arithmétique finie. A cause de ceci, des erreurs d'arrondi empêchent le bon fonctionnement de l'Algorithme 2. Ceci explique, dans cet exemple, l'obtention de la valeur $\|r_{125}\|_n = 3.64\ 10^{-2}$.

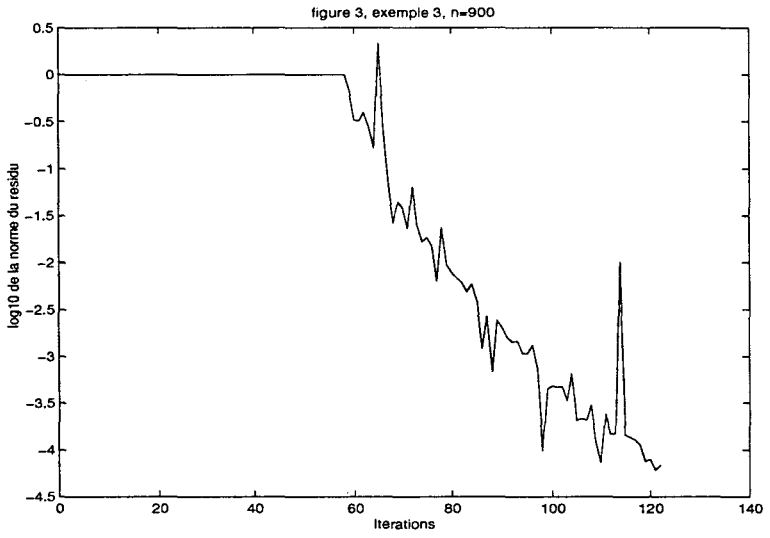
Exemple 3

Nous allons montrer que l'exécution de l'Algorithme 2 n'est pas à l'abri d'un problème d"'overflow" ou d"'underflow".
Nous considérons la matrice obtenue par discrétisation de l'équation elliptique aux dérivées partielles $Lu = f$ sur $[0, 1] \times [0, 1]$, où

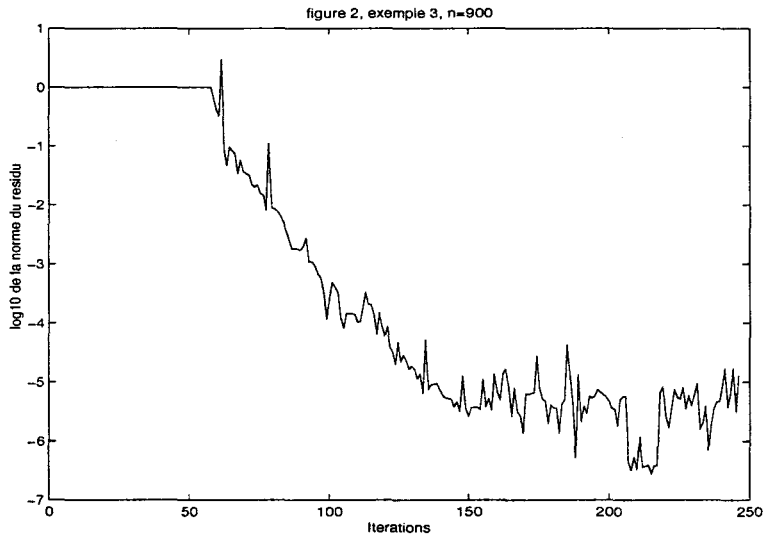
$$Lu = -\triangle u + s\frac{\partial u}{\partial x},$$

avec les conditions de Dirichlet $u = 0$ aux bords, en utilisant un schéma à cinq points sur une grille uniforme 30×30 avec un pas $h = 1/31$. Ceci nous donne une matrice creuse

non symétrique de dimension $n = 900$ avec 4380 éléments non nuls. Pour le paramètre s, nous choisissons $s = 10^4$. L'application de l'Algorithme 2 à cette matrice avec $tol = 10^{-8}$, $y_0 = (0, 0, \ldots, 0, 0, 1)^T$, $x_0 = (0, 0, \ldots, 0)^T$ et $b = (1, 0, 0, \ldots, 0)^T$ nous donne la figure suivante



Comme pour le deuxième exemple, il y a une stagnation au début de la figure, des pics apparaissent ensuite. Mais, à l'itération $k = 123$, un "overflow" affecte l'Algorithme 2. Cet "overflow" est dû à une multiplication successive de la matrice A^* par y_0, c'est-à-dire aux vecteurs $A^{*^k} y_0$. Il n'est pas dû à une division par un nombre voisin de zéro car, dans l'Algorithme 2, nous divisons par les quantités $h_{\theta(k)}$ qui sont controlées en valeur absolue par la tolérance tol prise égale à 10^{-8}. Essayons maintenant, en utilisant la norme euclidienne, de normaliser les vecteurs z_k. C'est-à-dire, qu'à chaque itération, z_k est normalisé en étant divisé par sa norme euclidienne.



Dans cet exemple, le problème d'overflow n'apparaît que plus tard. En effet, il n'affecte l'Algorithme 2 qu'à l'itération $k = 247$. Nous pouvons donc conclure que le choix d'une normalisation particulière peut contribuer à éviter d'atteindre le seuil d'underflow et d'overflow.

Il est toujours important de vérifier que nos estimations des normes des résidus soient admissibles. Pour ces exemples, nous avons remarqué qu'elles coïncident parfaitement avec les normes des vrais résidus.

2.4.4 Processus non hermitien de Lanczos

Nous avons déjà vu à la fin du premier chapitre que le processus non hermitien de Lanczos [90, 56, 54] n'est théoriquement que le processus hermitien de Lanczos en utilisant la forme bilinéaire g définie par

$$g(u,w) = w_2^T u_1 + w_1^T u_2 \quad \text{pour tout} \quad u = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}, w = \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \in \mathbb{C}^{2n}.$$

La forme g est l'équivalent de la forme donnée par (2.19) en remplaçant E par $\mathbb{C}^n$. Donc, avec la technique de normalisation utilisée dans le processus hermitien de Lanczos et les deux équations (2.21) et (2.23) regroupées en (2.26), nous obtenons une application de la méthode ALA au processus non hermitien de Lanczos. Nous donnons ici cette application sous la forme d'un processus. Posons

$$B = \begin{pmatrix} A & 0 \\ 0 & A^T \end{pmatrix}.$$

Processus 1

Choisir $v \in \mathbb{C}^{2n}$ et poser $\lambda_1 p_1 = v$, $p_0 = 0$, $k = 1$
(λ_1 est choisi pour que $\|p_1\|_{2n} = 1$).

Programmer
1 $i = 0$
$n_k = k$
2 si i est impair
$n_k = n_k + 1$
$\lambda_{n_k} p_{n_k} = B p_{n_k - 1}$
(λ_{n_k} est choisi pour que $\|p_{n_k}\|_{2n} = 1$)
$d_k = g(p_{n_k}, p_{n_k - 1})$
sinon
$d_k = g(p_{n_k}, p_{n_k})$
fin (si)
si $|d_k| < \varepsilon_1$ (pour une tolérance ε_1)
$i = i + 1$
aller à 2
fin (si)
$\theta(k + j) = k + i - j, \ j = 0, 1, \ldots, i$
poser $\theta(\theta(k) + 1) = \theta(k - 1)$.
Pour $j = \theta(n_k), \theta(n_k) + 1, \ldots, \theta(k)$:
$\alpha_j = g(Bp_j, p_j)/g(p_{\theta(j)}, p_j)$
$q_{j+1} = Bp_j - \alpha_j p_{\theta(j)}$
$\beta_j = g(q_{j+1}, q_{\theta(\theta(j+1))})/g(p_{\theta(j+1)}, q_{\theta(\theta(j+1))})$
$\lambda_{j+1} p_{j+1} = q_{j+1} - \beta_j p_{\theta(j+1)}$
(λ_{j+1} est choisi pour que $\|p_{j+1}\|_{2n} = 1$)
fin (pour)
$k = \theta(k) + 1$
aller à 1
fin.

Un processus pour la résolution des systèmes linéaires nous permet de triangulariser, tridiagonaliser la matrice du système en question ou la transformer en une matrice facile à manipuler, comme par exemple celle de Hessenberg. Ici, à chaque itération k du Processus 1, nous avons la factorisation suivante

$$B \begin{pmatrix} p_k & p_{k+1} & \cdots & p_{\theta(k)} \end{pmatrix} = \begin{pmatrix} p_{\theta(k-1)} & p_k & p_{k+1} & \cdots & p_{\theta(k)} & p_{\theta(k)+1} \end{pmatrix} \begin{pmatrix} \frac{d_k^T}{} \\ H_k \end{pmatrix} \qquad (2.34)$$

où $d_k^T = \beta_{\theta(k)}(0,0,\ldots,0,1) \in \mathbb{C}^{\theta(k)-k+1}$, la matrice $\widetilde{H_k}$ de $(\theta(k)-k+2)$ lignes et $(\theta(k)-k+1)$ colonnes est égale à

$$\begin{pmatrix}
0 & & & & & & \beta_{\theta(k)-1} & \alpha_{\theta(k)} \\
\lambda_{k+1} & 0 & & & & \iddots & \alpha_{\theta(k)-1} & \\
 & \lambda_{k+2} & \ddots & & \beta_{n_k+1} & \iddots & & \\
 & & \ddots & 0 \quad \beta_{n_k} & \alpha_{n_k+1} & & & \\
 & & & \lambda_{n_k} \quad \alpha_{n_k} & & & & \\
 & & & \lambda_{n_k+1} & 0 & & & \\
 & & & & \lambda_{n_k+2} & \ddots & & \\
 & & & & & \ddots & 0 & \\
 & & & & & & \lambda_{\theta(k)} & 0 \\
 & & & & & & & \lambda_{\theta(k)+1}
\end{pmatrix}$$

si $\theta(n_k) = n_k$, sinon elle est égale à

$$\begin{pmatrix}
0 & & & & & & \beta_{\theta(k)-1} & \alpha_{\theta(k)} \\
\lambda_{k+1} & 0 & & & & \iddots & \alpha_{\theta(k)-1} & \\
 & \lambda_{k+2} & \ddots & & \beta_{n_k+1} & \iddots & & \\
 & & \ddots & 0 \quad \beta_{n_k} & \alpha_{n_k+1} & & & \\
 & & & \lambda_{n_k} \quad \alpha_{n_k} & & & & \\
 & & & \lambda_{n_k+1} & 0 & & & \\
 & & & & \lambda_{n_k+2} & \ddots & & \\
 & & & & & \ddots & 0 & \\
 & & & & & & \lambda_{\theta(k)} & 0 \\
 & & & & & & & \lambda_{\theta(k)+1}
\end{pmatrix}.$$

Ce qui nous permet d'avoir la factorisation suivante

$$B\begin{pmatrix} p_1 & p_2 & \ldots & p_{\theta(k)} \end{pmatrix} = \begin{pmatrix} p_1 & p_2 & \ldots & p_{\theta(k)} & p_{\theta(k)+1} \end{pmatrix} \begin{pmatrix}
\widetilde{H_1} & D_2 & & & \\
 & \widetilde{H_2} & D_3 & & \\
 & & \widetilde{H_3} & \ddots & \\
 & & & \ddots & D_k \\
 & & & & \widetilde{H_k}
\end{pmatrix} \tag{2.35}$$

où D_i est la matrice $(d_i, 0)^T$ dont la première ligne est d_i^T et les autres lignes sont nulles. D'une manière analogue, si nous utilisons l'équation (2.27) et la technique de normalisation utilisée dans le processus non hermitien de Lanczos, nous obtenons le processus suivant

Processus 2

Choisir $v_1, v_2 \in \mathbb{C}^n$ et poser $\lambda_1 p_1 = v_1$, $\mu_1 q_1 = v_2$, $p_0 = 0$, $k = 1$
(λ_1 et μ_1 sont choisis pour que $\|p_1\|_n = \|q_1\|_n = 1$).

Programmer
1 $i = 0$, $n_k = k$
2 si i est impair
$n_k = n_k + 1$
$\lambda_{n_k} p_{n_k} = A p_{n_k - 1}$
$\mu_{n_k} q_{n_k} = A^* q_{n_k - 1}$
(λ_{n_k} et μ_{n_k} sont choisis pour que $\|p_{n_k}\|_n = \|q_{n_k}\|_n = 1$)
$d_k = g_1(p_{n_k}, q_{n_k - 1})$
sinon $d_k = g_1(p_{n_k}, q_{n_k})$
fin (si)
si $|d_k| < \varepsilon_1$ (pour une tolérance ε_1)
$i = i + 1$
aller à 2
fin (si)
$\theta(k + j) = k + i - j$, $j = 0, 1, \ldots, i$
poser $\theta(\theta(k) + 1) = \theta(k - 1)$.
Pour $j = \theta(n_k), \theta(n_k) + 1, \ldots, \theta(k)$:
$\alpha_j = g_1(Ap_j, q_j)/g_1(p_{\theta(j)}, q_j)$
$p'_{j+1} = Ap_j - \alpha_j p_{\theta(j)}$
$\beta_j = g_1(p'_{j+1}, q_{\theta(\theta(j+1))})/g_1(p_{\theta(j+1)}, q_{\theta(\theta(j+1))})$
$\lambda_{j+1} p_{j+1} = p'_{j+1} - \beta_j p_{\theta(j+1)}$
$\alpha'_j = g_1(A^* q_j, p_j)/g_1(q_{\theta(j)}, p_j)$
$q'_{j+1} = A^* q_j - \alpha'_j q_{\theta(j)}$
$\beta'_j = g_1(q'_{j+1}, p_{\theta(\theta(j+1))})/g_1(q_{\theta(j+1)}, p_{\theta(\theta(j+1))})$
$\mu_{j+1} q_{j+1} = q'_{j+1} - \beta'_j q_{\theta(j+1)}$
(λ_{j+1} et μ_{j+1} sont choisis pour que $\|p_{j+1}\|_n = \|q_{j+1}\|_n = 1$)
fin (pour)
$k = \theta(k) + 1$
aller à 1
fin.

g_1 désigne la forme bilinéaire symétrique définie par $g_1(u, w) = w^T u$ pour tout $u, w \in \mathbb{C}^n$. Les factorisations que nous obtenons à partir de ce processus sont de la même forme que celles données par le Processus 1. Le Processus 1 et le Processus 2 utilisent tous les

deux la troisième mise en oeuvre de ALA et ils diffèrent dans la normalisation. Si, en revanche, nous utilisons la deuxième mise en oeuvre de ALA, nous aurons le processus correspondant suivant.

Processus 3

Choisir $v_1, v_2 \in \mathbb{C}^n$ et poser $\lambda_1 p_1 = v_1$, $\mu_1 q_1 = v_2$, $p_0 = 0$, $k = 1$
(λ_1 et μ_1 sont choisis pour que $\|p_1\|_n = \|q_1\|_n = 1$).

Programmer
1 $i = 0$
2 $d_k = g_1(p_k, q_{k+i})$
si $|d_k| < \varepsilon_1$ (pour une tolérance ε_1)
$i = i + 1$
$\mu_{k+i} q_{k+i} = A^* q_{k+i-1}$ (μ_{k+i} est choisi pour que $\|q_{k+i}\|_n = 1$)
aller à 2
fin (si)
$\theta(k + j) = k + i - j, \ j = 0, 1, \ldots, i$
$q_{\theta(k)+1} = A^* q_{\theta(k)}$
Pour $j = k, k + 1, \ldots, \theta(k)$:
$\alpha_j = g_1(Ap_j, q_{\theta(k)})/g_1(p_k, q_{\theta(k)})$
$p_{j+1} = Ap_j - \alpha_j p_k$
$\beta_j = g_1(q_{\theta(k)+1}, p_j)/g_1(q_{\theta(j)}, p_j)$
$q_{\theta(k)+1} = q_{\theta(k)+1} - \beta_j q_{\theta(j)}$
si $j = \theta(k)$
$\alpha'_j = g_1(Ap_j, q_{k-1})/g_1(p_{\theta(k-1)}, q_{k-1})$
$p_{j+1} = p_{j+1} - \alpha'_j p_{\theta(k-1)}$
$\beta'_j = g_1(q_{\theta(k)+1}, p_{k-1})/g_1(q_{\theta(k-1)}, p_{k-1})$
$\mu_{\theta(k)+1} q_{\theta(k)+1} = q_{\theta(k)+1} - \beta'_j q_{\theta(k-1)}$
($\mu_{\theta(k)+1}$ est choisi pour que $\|q_{\theta(k)+1}\|_n = 1$)
fin (si)
$\lambda_{j+1} p_{j+1} = p_{j+1}$ (λ_{j+1} est choisi pour que $\|p_{j+1}\|_n = 1$)
fin (pour)
$k = \theta(k) + 1$
aller à 1
fin.

Ce processus nous donne la factorisation suivante

$$A \begin{pmatrix} p_k & p_{k+1} & \cdots & p_{\theta(k)} \end{pmatrix} = \begin{pmatrix} p_{\theta(k-1)} & p_k & p_{k+1} & \cdots & p_{\theta(k)} & p_{\theta(k)+1} \end{pmatrix} \begin{pmatrix} \widetilde{d'^t_k} \\ H'_k \end{pmatrix} \quad (2.36)$$

où $d_k'^t = \alpha'_{\theta(k)}(0,0,\ldots,0,1) \in \mathbb{C}^{\theta(k)-k+1}$, la matrice $\widetilde{H_k'}$ de $(\theta(k)-k+2)$ lignes et $(\theta(k)-k+1)$ colonnes est égale à

$$\begin{pmatrix} \alpha_k & \alpha_{k+1} & \cdots & \alpha_{\theta(k)} \\ \lambda_{k+1} & & & \\ & \lambda_{k+2} & & \\ & & \ddots & \\ & & & \lambda_{\theta(k)+1} \end{pmatrix}.$$

Regardons maintenant le critère d'arrêt de ces trois processus. Tout d'abord considérons les trois espaces de Krylov $W_1 = K_n(B,v) = Vect(v, Bv, B^2v\ldots)$, $W_2 = K_n(A,v_1) = Vect(v_1, Av_1, A^2v_1\ldots)$ et $W_3 = K_n(A^*,v_2) = Vect(v_2, A^*v_2, A^{*2}v_2,\ldots)$. Il y a deux cas à envisager

- le premier cas correspond à ne pas avoir une DPZ à une itération $k \leq \min\{l,l'\}$ avec $l = dim\ W_2$ et $l' = dim\ W_3$, ce qui revient à dire que les espaces $(W_1 \times W_1, g)$ et $(W_2 \times W_3, g_1)$ sont réguliers.

- le second cas correspond à avoir une DPZ incurable à laquelle nous ne pouvons pas apporter de remède (*incurable breakdown*). Ce qui revient à dire que nous avons une DPZ à une itération $k \leq \min\{l,l'\}$, ce qui signifie en d'autres termes que $(W_1 \times W_1, g)$ et $(W_2 \times W_3, g_1)$ ne sont pas réguliers. Il est donc clair que si $(W_1 \times W_1, g)$ (ou $(W_2 \times W_3, g_1)$) n'est pas régulier, alors les trois processus ne peuvent fonctionner et par conséquent il faut changer le choix fait sur les deux vecteurs v_1 et v_2 de $\mathbb{C}^n$ (ou v de $\mathbb{C}^{2n}$ pour Processus 1).

Précisons que la DPZ, appelée "la vraie DPZ" dans [22], est la seule évitée ici.

Remarque 2.9 Le codage de ces trois processus nécessite $\max_k\{\theta(k)-k\}+6$ vecteurs de $\mathbb{C}^n$. Dans le cas régulier, c'est-à-dire en n'ayant aucune DPZ dans le processus classique de Lanczos, nous n'avons besoin que de 6 vecteurs. Ceci coïncide bien avec $\max_k\{\theta(k)-k\}+6$, car dans le cas régulier, $\theta(k) = k$ pour tout entier k.

Notons que la factorisation du Processus 3 a aussi été trouvée par Ziegler dans [128, 129] où il parle d'un "look-ahead" spécial.

Remarque 2.10 Nous avons montré comment il faut appliquer ALA aux méthodes de type Lanczos. Nous pouvons faire la même chose pour les méthodes de type CGM (Conjugate Gradient Multiplied) qui sont connues aussi sous le nom de *"Lanczos-type product methods"*(LTPM), et qui ont été introduites simultanément par Brezinski [14] et Gutknecht [64]. Cette classe CGM contient la méthode CGS (Conjugate Gradient squared) due à Sonneveld [112] et la méthode Bi-CGSTAB due à Van Der Vorst [116].

2.5 Application aux approximants de Padé

Nous savons que les approximants de Padé ont de nombreuses applications dans divers domaines. Il est donc intéressant que nous donnions dans cette section l'application de la méthode ALA au calcul des approximants de Padé. Nous allons utiliser la première, la deuxième et la troisième mise en oeuvre de ALA pour trouver le numérateur et le dénominateur d'un approximant de Padé.

2.5.1 Définitions et notations

Soit f une série formelle de la variable t

$$f(t) = \sum_{i=0}^{\infty} c_i t^i, \quad c_i \in \mathbb{C}.$$

Nous cherchons une fraction rationnelle

$$R(t) = \frac{Q(t)}{P(t)} = \frac{a_0 + a_1 t + \ldots + a_p t^p}{b_0 + b_1 t + \ldots + b_q t^q}$$

telle que son développement en série suivant les puissances croissantes de la variable t coïncide avec celui de la fonction f aussi loin que possible, c'est-à-dire que nous devons avoir

$$f(t) - R(t) = \mathcal{O}(t^{p+q+1}) \quad (t \to 0).$$

Une telle fraction rationnelle s'appelle un approximant de Padé de f et nous le notons $[p/q]_f(t)$. Nous arrangeons ces approximants dans une table à double entrée comme l'a déjà proposé Padé. Cette table est connue sous le nom de table de Padé.

$$\begin{array}{cccc} [0/0] & [0/1] & [0/2] & \cdots \\ [1/0] & [1/1] & [1/2] & \cdots \\ [2/0] & [2/1] & [2/2] & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{array}$$

Les approximants de la première colonne de cette table sont exactement les sommes partielles de la série f. Il se peut que certains approximants de Padé soient identiques. S'il n'y a aucun bloc, alors la table de Padé est dite normale. Dans le cas contraire elle est dite non normale.

Pour chaque entier relatif $n \in \mathbb{Z}$, considérons la fonctionnelle linéaire $C^{(n)}$ définie sur l'espace des polynômes $\mathbb{C}[X]$ par

$$C^{(n)}(x^i) = c_{n+i}.$$

Par convention, nous posons $c_i = 0$ pour $i < 0$. A chaque fonctionnelle $C^{(n)}$, nous associons la série formelle

$$f_n(t) = c_n + c_{n+1}t^{n+1} + c_{n+2}t^{n+2} + \dots .$$

Comme les coefficients de $Q(t)$ et $P(t)$ sont définis à un facteur multiplicatif près, nous pouvons poser $b_0 = 1$. Ainsi, les coefficients du dénominateur $P(t)$ de $[p/q]_f(t)$ sont donnés par la solution du système linéaire de Hankel

$$\begin{pmatrix} c_{p-q+1} & c_{p-q+2} & c_{p-q+3} & \cdots & c_p \\ c_{p-q+2} & c_{p-q+3} & c_{p-q+4} & \cdots & c_{p+1} \\ c_{p-q+3} & c_{p-q+4} & c_{p-q+5} & \cdots & c_{p+2} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ c_p & c_{p+1} & c_{p+2} & \cdots & c_{p+q-1} \end{pmatrix} \begin{pmatrix} b_q \\ b_{q-1} \\ b_{q-2} \\ \vdots \\ b_1 \end{pmatrix} = \begin{pmatrix} -c_{p+1} \\ -c_{p+2} \\ -c_{p+3} \\ \vdots \\ -c_{p+q} \end{pmatrix}. \tag{2.37}$$

Les coefficients du numérateur $Q(t)$ de $[p/q]_f(t)$ s'obtiennent soit directement à partir de ceux du dénominateur $P(t)$ grâce aux équations suivantes

$$\begin{aligned} a_0 &= c_0 b_0 \\ a_1 &= c_1 b_0 + c_0 b_1 \\ &\vdots \\ a_p &= c_p b_0 + c_{p-1} b_1 + \dots + c_{p-q} b_q, \end{aligned} \tag{2.38}$$

soit en utilisant les relations de récurrence que $Q(t)$ et $P(t)$ vérifient. L'un des algorithmes que nous utilisons pour calculer $Q(t)$ et $P(t)$ est l'algorithme *qd* dont la forme progressive est plus stable numériquement d'après Henrici [69].

Une simple permutation des colonnes de la matrice de Hankel du système (2.37) nous permet d'obtenir une matrice de Toeplitz. Nous savons bien que les algorithmes classiques de Levinson (ou Levinson-Durbin) [78, 46] et Schur (ou Schur-Bareiss) [111, 7] sont basés principalement sur les sous-matrices d'une matrice de Toeplitz pour calculer les approximants de Padé en suivant deux lignes adjacentes de la table de Padé. Plus récemment, Chandrasekaran et Sayed ont montré dans [35] comment rendre stable l'algorithme généralisé de Schur. A la matrice de Hankel nous avons associé les polynômes orthogonaux. Par analogie, les polynômes associés à la matrice de Toeplitz sont les polynômes semi-orthogonaux que Draux a définis dans [44].

2.5.2 Relations de récurrence

Dans le cadre de l'extension au cas non normal des algorithmes qui permettent de calculer les approximants de Padé, Eliece et Shearer ont présenté dans [81] un algorithme

qui calcule les approximants de Padé le long d'une antidiagonale en se basant sur l'algorithme d'Euclide. Avec une idée similaire, Cordellier a aussi donné dans [38] un autre algorithme qui calcule les approximants de Padé en escalier le long de deux antidiagonales adjacentes et qui étend au cas non normal l'algorithme de Baker. Claessens et Wuytack ont proposé dans [36] un algorithme pour le calcul des approximants de Padé grâce aux propriétés des fractions continues et à l'algorithme *qd*. Malheureusement, cet algorithme concerne seulement les blocs entourés d'au moins deux rangées (colonnes et lignes) qui correspondent à des polynômes orthogonaux réguliers. Bultheel a donné dans [32, 33] des algorithmes pour se déplacer le long d'une diagonale, en escalier suivant deux diagonales et en dents de scie suivant deux lignes. Tous ces algorithmes sont retrouvés dans le cadre de la théorie des polynômes orthogonaux formels, voir [10], dans le cas régulier. Ils ont été étendus par Draux [44] au cas non régulier. Draux a étudié les blocs dans [44] et il a donné avec Van Ingelandt dans [45] les algorithmes qui nous permettent de nous déplacer suivant un chemin arbitraire choisi par l'utilisateur.
Récemment Gutknecht et Hochbruck ont donné dans [50] des algorithmes récursifs stables qui étendent les deux algorithmes de Levinson et Schur au cas d'une table non normale de Padé en utilisant une stratégie de look-ahead.

Ici, nous allons étudier la méthode ALA pour établir des relations de récurrence qui nous permettent de nous déplacer dans la table non normale de Padé. Tout d'abord, nous allons commencer par utiliser la première mise en oeuvre.

Selon cette mise en oeuvre, la famille des polynômes unitaires $\{P_q^{\theta_n}\}_q$ est définie par les conditions

$$C^{(n)}(x^{\theta_n(i)}P_q^{\theta_n}(x)) = 0, \quad i = 0, 1, \ldots, q-1$$

où θ_n est la permutation associée à la fonctionnelle $C^{(n)}$ qui est définie ci-dessus. Pour chaque polynôme $P_q^{\theta_n}$, nous considérons le polynôme associé

$$Q_q^{\theta_n}(t) = C^{(n)}\left(\frac{P_q^{\theta_n}(x) - P_q^{\theta_n}(t)}{x-t}\right)$$

où $C^{(n)}$ agit sur x et t est un paramètre. $Q_q^{\theta_n}$ est un polynôme en t de degré au plus $q-1$. Dans le cas normal, les polynômes $P_q^{\theta_n}$ sont orthogonaux réguliers par rapport à $C^{(n)}$. Plusieurs relations de récurrence relient ces polynômes orthogonaux et permettent le calcul récursif des approximants $\{[p/q]_f\}$ en suivant n'importe quel chemin dans la table normale de Padé. Pour plus de détails, voir par exemple [10, 58].
D'une manière simple, nous avons précédemment étendu quelques unes de ces relations

au cas non normal en utilisant les polynômes $P_q^{\theta_n}$ avec la permutation θ_n.

Disposons les polynômes $P_q^{\theta_n}$ dans un tableau à une seule entrée que nous appelons table P.

$$\begin{array}{cccc}
\vdots & \vdots & \vdots & \iddots \\
P_{-1}^{\theta_0} & P_0^{\theta_{-1}} & P_1^{\theta_{-2}} & \cdots \\
P_{-1}^{\theta_1} & P_0^{\theta_0} & P_1^{\theta_{-1}} & \cdots \\
P_{-1}^{\theta_2} & P_0^{\theta_1} & P_1^{\theta_0} & \cdots \\
\vdots & \vdots & \vdots & \ddots
\end{array}$$

Supposons que cette table P ait un bloc à la k-ième colonne. Ceci peut être illustré par le schéma suivant

$$\begin{array}{c|cc}
P_k^{\theta_n} & P_{k+1}^{\theta_{n-1}} & \cdots \\
\hline
P_k^{\theta_{n+1}} & P_{k+1}^{\theta_n} & \cdots \\
P_k^{\theta_{n+2}} & P_{k+1}^{\theta_{n+1}} & \cdots \\
\vdots & \vdots & \ddots
\end{array}$$

où $P_k^{\theta_n}$ est supposé régulier. A l'aide des résultats précédents donnés dans la sous-section 2.4.2, nous déduisons les deux relations de récurrence suivantes

$$\left\{\begin{array}{l} P_k^{\theta_{m+1}} = P_k^{\theta_m} - e_k^{\theta_m} P_{\theta_{m+1}(k-1)}^{\theta_{m+1}} \\ e_k^{\theta_m} = C^{(m)}(x^k P_k^{\theta_m}) / C^{(m+1)}(x^{k-1} P_{\theta_{m+1}(k-1)}^{\theta_{m+1}}) \end{array}\right. \quad m = n, \ldots, \theta_n(k) - n \qquad (2.39)$$

et

$$\left\{\begin{array}{l} P_{i+1}^{\theta_{m-1}} = x P_i^{\theta_m} - q_{i+1}^{\theta_{m-1}} P_{\theta_{m-1}(i)}^{\theta_{m-1}} \\ q_{i+1}^{\theta_{m-1}} = C^{(m)}(x^i P_i^{\theta_m}) / C^{(m-1)}(x^i P_{\theta_{m-1}(i)}^{\theta_{m-1}}) \end{array}\right. \quad i = k, \ldots, \theta_n(k), \ m = n + k - i. \qquad (2.40)$$

Nous allons voir ci-dessous que ces deux dernières relations nous permettent d'énoncer dans le théorème suivant quelques propriétés des blocs de la table P.

Théorème 2.7 *Pour tout $n \in \mathbb{Z}$, si la table P a un bloc à sa k-ième colonne comme nous l'avons décrit précédemment, alors nous aurons*

$$P_k^{\theta_{n+i}} = P_k^{\theta_n}, \quad i = 0, \ldots, \theta_n(k) - k,$$
$$P_{k+i}^{\theta_{n-i}} = x^i P_k^{\theta_n}, \quad i = 0, \ldots, \theta_n(k) - k$$

avec

$$\theta_{n-i}(k+i) = \theta_n(k) \text{ et } \theta_{n+i}(k) = \theta_n(k) - i \quad \text{pour} \quad i = 0, \ldots, \theta_n(k) - k.$$

Preuve :

Ce théorème a été donné et prouvé par Draux dans [44]. Ici, nous avons seulement fait la connection avec la permutation θ_n, ce qui simplifie l'écriture de ces relations de récurrence. Mais signalons, qu'ici, la démonstration de ce théorème est très simple, elle n'est qu'une conséquence des équations de (2.39) et (2.40).
En effet, faisons une récurrence sur i

- pour $i = 0$, le résultat du théorème est évident,
- pour $i < \theta(k) - k$, supposons que

$$\theta_{n-i}(k+i) = \theta_n(k), \quad \theta_{n+i}(k) = \theta_n(k) - i, \quad P_k^{\theta_{n+i}} = P_k^{\theta_n} \text{ et } P_{k+i}^{\theta_{n-i}} = x^i P_k^{\theta_n},$$

- pour $i+1$, si nous remplaçons dans (2.39) m par $n+i$, alors $C^{(m)}(x^k P_k^{\theta_m}) = 0$ car, d'après l'hypothèse de récurrence, nous avons $\theta_{n+i}(k) = \theta_n(k) - i > k$. Ceci entraîne que $e_k^{\theta_m} = 0$, donc $P_k^{\theta_{n+i+1}} = P_k^{\theta_{n+i}} = P_k^{\theta_n}$.
  D'une manière analogue, (2.40) et l'hypothèse de récurrence impliquent que $q_{k+i+1}^{\theta_{n-i-1}}$ est nul, ce qui nous permet de déduire que $P_{k+i+1}^{\theta_{n-i-1}} = x P_{k+i}^{\theta_{n-i}} = x^{i+1} P_k^{\theta_n}$.
  En conséquence,

$$C^{(n-i-1)}(x^j P_{k+i+1}^{\theta_{n-i-1}}) = C^{(n)}(x^j P_k^{\theta_n}) = 0$$

  si $j < \theta_n(k)$ et

$$C^{(n-i-1)}(x^{\theta_n(k)} P_{k+i+1}^{\theta_{n-i-1}}) = C^{(n)}(x^{\theta_n(k)} P_k^{\theta_n}) \neq 0,$$

  ce qui implique que $\theta_{n-i-1}(k+i+1) = \theta_n(k)$.
  De même, nous avons

$$C^{(n+i+1)}(x^j P_k^{\theta_{n+i+1}}) = C^{(n)}(x^{j+i+1} P_k^{\theta_n}) = 0$$

  si $j < \theta_n(k) - i - 1$ et

$$C^{(n+i+1)}(x^{\theta_n(k)-i-1} P_k^{\theta_{n+i+1}}) = C^{(n)}(x^{\theta_n(k)} P_k^{\theta_n}) \neq 0,$$

  ce qui prouve que $\theta_{n+i+1}(k) = \theta_n(k) - i - 1$.

■

Comme Padé l'a prouvé dans sa thèse [86] d'une toute autre façon, la propriété de θ_n dans Théorème 2.7 montre que chaque bloc fini, dans la table de Padé, est carré. Dans le schéma précédent, à l'intérieur des blocs de la table P, nous avons arrangé les nouveaux polynômes qui ont été définis et introduits avant. Tous ces polynômes sont reliés par les

relations du théorème suivant.

Théorème 2.8 *Pour tout $n \in \mathbb{Z}$, si la table P a un bloc à la k-ième colonne comme nous l'avons décrit précédemment, alors nous aurons pour $i = k, \ldots, \theta_n(k)$ et $m = 0, \ldots, \theta_n(k) - i$*

$$P_i^{\theta_{n+m}} = P_i^{\theta_n}, \quad P_{i+m}^{\theta_{n-m}} = x^m P_i^{\theta_n}$$

et

$$\theta_{n-m}(i+m) = \theta_n(i), \quad \theta_{n+m}(i) = \theta_n(i) - m.$$

Preuve :

La preuve de ce théorème se fait en utilisant les propriétés de la permutation θ_n donnée dans le Théorème 2.7 et le Théorème 2.2. En effet, d'après le Théorème 2.2, $\theta_{n-m}(i+m) = \theta_{n-m}(i+m-1)+1$ et d'après le Théorème 2.7, $\theta_{n-m}(i+m-1)+1 = \theta_{n-m+1}(i+m-1)$. Ainsi, en appliquant le Théorème 2.2 puis le Théorème 2.7 m fois, nous obtenons l'égalité $\theta_{n-m}(i+m) = \theta_n(i)$.
Le Théorème 2.7 nous donne $\theta_{n+m}(i) = \theta_{n+m-1}(i) - 1$. En appliquant le Théorème 2.7 m fois, nous aurons donc $\theta_{n+m}(i) = \theta_n(i) - m$.
Ces relations entre les permutations θ_j nous permettent de voir que $P_i^{\theta_n}$ et $x^m P_i^{\theta_n}$ vérifient respectivement les mêmes propriétés que celles de $P_i^{\theta_{n+m}}$ et $P_{i+m}^{\theta_{n-m}}$, c'est-à-dire $P_i^{\theta_{n+m}} = P_i^{\theta_n}$ et $P_{i+m}^{\theta_{n-m}} = x^m P_i^{\theta_n}$. ∎

Il est clair que nous pouvons considérer le Théorème 2.8 comme une généralisation du Théorème 2.7.

Les relations de récurrence (2.39) et (2.40) peuvent être généralisées de sorte qu'elles soient aussi valables à l'intérieur d'un bloc. Pour cela, il suffit d'identifier $P_i^{\theta_m} - e_i^{\theta_m} P_{r_{i-1}}^{\theta_{m+1}}$ à $P_i^{\theta_{m+1}}$ et $xP_i^{\theta_{m+1}} - q_{i+1}^{\theta_m} P_{r_i}^{\theta_m}$ à $P_{i+1}^{\theta_m}$, c'est-à-dire écrire

$$P_i^{\theta_{m+1}} = P_i^{\theta_m} - e_i^{\theta_m} P_{r_{i-1}}^{\theta_{m+1}} \tag{2.41}$$

et

$$P_{i+1}^{\theta_m} = xP_i^{\theta_{m+1}} - q_{i+1}^{\theta_m} P_{r_i}^{\theta_m} \tag{2.42}$$

où $e_i^{\theta_m}$ et $q_{i+1}^{\theta_{m-1}}$ se calculent à l'aide de la condition de biorthogonalité ou d'orthogonalité imposée à $P_i^{\theta_{m+1}}$ et $P_{i+1}^{\theta_m}$. Pour tout entier i et tout $m \in \mathbb{Z}$, $P_{r_i}^{\theta_m}$ désigne le polynôme régulier de plus haut degré précédant $P_{i+1}^{\theta_m}$ qui se trouve sur la même diagonale que $P_{i+1}^{\theta_m}$. Malheureusement, la relation de récurrence (2.42) est incapable de calculer les polynômes

orthogonaux réguliers qui se trouvent à l'est d'un bloc et (2.41) est incapable de calculer les polynômes orthogonaux réguliers qui se trouvent au sud d'un bloc. Dans la sous-section suivante, nous allons montrer que ce problème peut être évité en choisissant la deuxième mise en oeuvre de ALA.

2.5.3 Extension de l'algorithme *qd*

L'algorithme *qd* a été étendu au cas non normal pour la première fois par Claessens et Wuytack [36]. Une étude bien détaillée sur l'extension de cet algorithme a été faite par Draux dans [44]. Ici, nous allons montrer que la méthode ALA nous permet d'étendre l'algorithme *qd* au cas non normal d'une manière simple et tout à fait similaire au cas normal. Contrairement à l'extension connue [44], celle que nous présentons ici est caractérisée par le fait que tous les éléments du tableau *qd* sont définis. Ceci bien sûr est dû aux polynômes biorthogonaux que nous avons introduits dans les blocs de la table P.

Nous allons considérer la première et la deuxième mise en oeuvre de ALA.

Première mise en oeuvre

D'après l'équation (2.41), nous pouvons remplacer $P_i^{\theta_{m+1}}$ par $P_i^{\theta_m} - e_i^{\theta_m} P_{r_{i-1}}^{\theta_{m+1}}$ dans (2.42) et obtenir

$$P_{i+1}^{\theta_m} = xP_i^{\theta_m} - e_i^{\theta_m} xP_{r_{i-1}}^{\theta_{m+1}} - q_{i+1}^{\theta_m} P_{r_i}^{\theta_m}.$$

Si $P_{r_{i-1}}^{\theta_{m+1}}$ et $P_{r_i}^{\theta_m}$ sont à l'ouest d'un bloc, alors d'après le Théorème 2.8 nous aurons $P_{r_{i-1}}^{\theta_{m+1}} = P_{r_i}^{\theta_m}$ et $e_i^{\theta_m} = 0$. Dans le cas contraire, nous pouvons appliquer (2.42) pour avoir

$$P_{r_i}^{\theta_m} = xP_{r_{i-1}}^{\theta_{m+1}} - q_{r_i}^{\theta_m} P_{\theta_m(r_i-1)}^{\theta_m}.$$

Dans ces deux cas, nous retrouvons la relation de récurrence à trois termes suivante

$$P_{i+1}^{\theta_m} = xP_i^{\theta_m} - (q_{i+1}^{\theta_m} + e_i^{\theta_m}) P_{r_i}^{\theta_m} - e_i^{\theta_m} q_{r_i}^{\theta_m} P_{\theta_m(r_i-1)}^{\theta_m}, \tag{2.43}$$

reliant trois polynômes qui se suivent sur une même diagonale de la table P.
D'après (2.41), $P_{i+1}^{\theta_{m+1}} = P_{i+1}^{\theta_m} - e_{i+1}^{\theta_m} P_{r_i}^{\theta_{m+1}}$. Grâce à (2.42), nous pouvons remplacer $P_{i+1}^{\theta_m}$ par $xP_i^{\theta_{m+1}} - q_{i+1}^{\theta_m} P_{r_i}^{\theta_m}$. Donc, nous aurons

$$P_{i+1}^{\theta_{m+1}} = xP_i^{\theta_{m+1}} - q_{i+1}^{\theta_m} P_{r_i}^{\theta_m} - e_{i+1}^{\theta_m} P_{r_i}^{\theta_{m+1}}.$$

Si $P_{r_i}^{\theta_{m+1}}$ et $P_{r_i}^{\theta_m}$ sont au nord d'un bloc, alors d'après le Théorème 2.8 nous aurons $P_{r_i}^{\theta_m} = xP_{r_i}^{\theta_{m+1}}$ et $q_{i+1}^{\theta_m} = 0$. Dans le cas contraire, nous pouvons appliquer (2.41) pour avoir

$$P_{r_i}^{\theta_{m+1}} = P_{r_i}^{\theta_m} - e_{r_i}^{\theta_m} P_{\theta_{m+1}(r_i-1)}^{\theta_{m+1}}.$$

Dans ces deux cas, nous obtenons la relation de récurrence à trois termes suivante

$$P_{i+1}^{\theta_{m+1}} = xP_i^{\theta_{m+1}} - (q_{i+1}^{\theta_m} + e_{i+1}^{\theta_m})P_{r_i}^{\theta_{m+1}} - q_{i+1}^{\theta_m} e_{r_i}^{\theta_m} P_{\theta_{m+1}(r_i-1)}^{\theta_{m+1}}. \tag{2.44}$$

Ici, c'est le lien entre les coefficients des relations de récurrence qui nous intéresse. Alors, comme dans le cas normal, l'identification de (2.43) avec (2.44) nous permet de déduire les deux relations suivantes

$$e_i^{\theta_{m+1}} + q_{i+1}^{\theta_{m+1}} = q_{i+1}^{\theta_m} + e_{i+1}^{\theta_m}$$
$$e_i^{\theta_{m+1}} q_{r_i}^{\theta_{m+1}} = e_{r_i}^{\theta_m} q_{i+1}^{\theta_m}$$

qui définissent une extension de l'algorithme *qd*. Le seul inconvénient de cette extension est qu'elle utilise la première mise en oeuvre de ALA, pour laquelle, les polynômes orthogonaux réguliers se trouvant au sud et à l'est d'un bloc dépendent de plusieurs éléments de la table P. En effet, ces polynômes vérifient la relation de récurrence plus longue donnée au Théorème 2.3. Afin d'échapper à ce désavantage, nous allons utiliser la deuxième mise en oeuvre.

Deuxième mise en oeuvre

La deuxième mise en oeuvre de ALA nous donne les mêmes propriétés que celles citées ci-dessus pour la première mise en oeuvre. Nous avons vu que (2.42) est incapable de calculer les polynômes orthogonaux réguliers qui se trouvent à l'est d'un bloc et (2.41) est incapable de calculer les polynômes orthogonaux réguliers qui se trouvent au sud d'un bloc. Le fait que les deux relations (2.41) et (2.42) puissent être étendues, afin d'être valables pour tous les éléments de la table P, est l'une des caractéristiques de la deuxième mise en oeuvre. En effet, pour le calcul d'un polynôme $P_{i+1}^{\theta_m}$ se trouvant à l'est d'un bloc, nous remplaçons dans l'équation (2.42) $P_{r_i}^{\theta_m}$ par $P_{\theta(r_i-1)}^{\theta_m}$ que nous notons $P_{r_i'}^{\theta_m}$ pour simplifier l'écriture.

$$P_{i+1}^{\theta_m} = xP_i^{\theta_{m+1}} - q_{i+1}^{\theta_m} P_{r_i'}^{\theta_m}. \tag{2.45}$$

Pour tout entier i et tout $m \in \mathbb{Z}$, $P_{r_i'}^{\theta_m}$ désigne alors le polynôme régulier de plus haut degré précédant $P_{r_i}^{\theta_m}$ et qui se trouve sur la même diagonale que $P_{i+1}^{\theta_m}$. D'une manière analogue, nous aurons pour (2.41) l'équation

$$P_{i+1}^{\theta_{m+1}} = P_{i+1}^{\theta_m} - e_{i+1}^{\theta_m} P_{r_i'}^{\theta_{m+1}} \tag{2.46}$$

qui nous donnera les polynômes orthogonaux réguliers se trouvant au sud d'un bloc. Maintenant, essayons d'établir les relations de récurrence reliant les coefficients $e_i^{\theta_m}$ et

$q_i^{\theta_m}$. Pour cela nous distinguons quatre cas

- $P_{i+1}^{\theta_{m+1}}$ est à l'intérieur d'un bloc.
  Dans ce cas, comme pour la première mise en oeuvre, les deux relations

$$e_i^{\theta_{m+1}} + q_{i+1}^{\theta_{m+1}} = q_{i+1}^{\theta_m} + e_{i+1}^{\theta_m},$$
$$e_i^{\theta_{m+1}} q_{r_i}^{\theta_{m+1}} = e_{r_i}^{\theta_m} q_{i+1}^{\theta_m},$$

  de l'algorithme *qd* dans le cas normal sont toujours valables. Seulement, à l'intérieur d'un bloc, ces deux relations sont réduites à la première car, d'après le Théorème 2.2 et le Théorème 2.4, $e_i^{\theta_{m+1}} q_{r_i}^{\theta_{m+1}} = e_{r_i}^{\theta_m} q_{i+1}^{\theta_m} = 0$. Même si la deuxième relation ne nous fournit aucune information sur les coefficients $e_i^{\theta_m}$ et $q_i^{\theta_m}$, la première est complètement suffisante pour cela. En effet

  * Si $P_{i+1}^{\theta_{m+1}}$ est dans la partie inférieure du bloc (c.à.d celle située au-dessous de la diagonale du bloc), alors, d'après le Théorème 2.8 et l'équation (2.41), nous avons $e_i^{\theta_{m+1}} = e_{i+1}^{\theta_m} = 0$, ce qui implique que $q_{i+1}^{\theta_{m+1}} = q_{i+1}^{\theta_m}$ puisque $e_i^{\theta_{m+1}} + q_{i+1}^{\theta_{m+1}} = q_{i+1}^{\theta_m} + e_{i+1}^{\theta_m}$.
  * Si $P_{i+1}^{\theta_{m+1}}$ est dans la partie supérieure du bloc (c.à.d celle située au-dessus de la diagonale du bloc), alors le Théorème 2.8 et (2.42) nous donnent $q_{i+1}^{\theta_{m+1}} = q_{i+1}^{\theta_m} = 0$, ce qui entraîne que $e_i^{\theta_{m+1}} = e_{i+1}^{\theta_m}$.
  * Si $P_{i+1}^{\theta_{m+1}}$ est sur la diagonale du bloc, alors toujours d'après le Théorème 2.8 et (2.42) nous avons $e_i^{\theta_{m+1}} = q_{i+1}^{\theta_m} = 0$, ce qui implique que $q_{i+1}^{\theta_{m+1}} = e_{i+1}^{\theta_m}$.

  Comme pour la première mise en oeuvre, $P_{i+1}^{\theta_{m+1}}$ vérifie la relation de récurrence à trois termes suivante

$$P_{i+1}^{\theta_{m+1}} = xP_i^{\theta_{m+1}} - (q_{i+1}^{\theta_{m+1}} + e_i^{\theta_{m+1}})P_{r_i}^{\theta_{m+1}} - e_i^{\theta_{m+1}} q_{r_i}^{\theta_{m+1}} P_{\theta_m(r_i-1)}^{\theta_{m+1}}.$$

- $P_{i+1}^{\theta_{m+1}}$ est à l'est d'un bloc et non aux coins.
  L'extension des deux relations (2.41) et (2.42) nous permet d'avoir la relation à quatre termes donnant $P_{i+1}^{\theta_{m+1}}$ suivante

$$P_{i+1}^{\theta_{m+1}} = xP_i^{\theta_{m+1}} - e_i^{\theta_{m+1}} P_{r_i}^{\theta_{m+1}} - q_{i+1}^{\theta_{m+1}} P_{r_i'}^{\theta_{m+1}}, \tag{2.47}$$

  avec $q_{i+1}^{\theta_{m+1}}$ non nul. En effet, la relation (2.45) nous permet d'écrire

$$P_{i+1}^{\theta_{m+1}} = xP_i^{\theta_{m+2}} - q_{i+1}^{\theta_{m+1}} P_{r_i'}^{\theta_{m+1}}.$$

  D'après (2.41), nous pouvons remplacer $P_i^{\theta_{m+2}}$ par $P_i^{\theta_{m+1}} - e_i^{\theta_{m+1}} P_{r_{i-1}}^{\theta_{m+2}}$. Comme $P_{r_i}^{\theta_{m+1}}$ et $P_{r_i}^{\theta_{m+1}}$ sont au nord d'un bloc, c'est-à-dire $P_{r_i}^{\theta_{m+1}} = xP_{r_i}^{\theta_{m+2}}$, nous déduisons

l'équation ci-dessus (2.47).
Une relation équivalente à (2.47), dont les coefficients dépendent de l'indice m au lieu de $m+1$, peut être déduite de l'équation suivante

$$P_{i+1}^{\theta_{m+1}} = xP_i^{\theta_{m+1}} - e_{i+1}^{\theta_m} P_{r_i}^{\theta_{m+1}} - q_{i+1}^{\theta_m} P_{r_i'}^{\theta_m}. \tag{2.48}$$

Cette équation est aussi obtenue à l'aide de l'extension de (2.41) et (2.42). En effet, d'après (2.41), nous avons $P_{i+1}^{\theta_{m+1}} = P_{i+1}^{\theta_m} - e_{i+1}^{\theta_m} P_{r_i}^{\theta_{m+1}}$. En se servant de (2.45), nous pouvons remplacer $P_{i+1}^{\theta_m}$ par $xP_i^{\theta_{m+1}} - q_{i+1}^{\theta_m} P_{r_i'}^{\theta_m}$ pour obtenir (2.48).

Pour pouvoir identifier les coefficients de ces deux relations (2.47) et (2.48), nous allons exprimer $P_{r_i'}^{\theta_m}$ en fonction de $P_{r_i}^{\theta_{m+1}}$ et $P_{r_i'}^{\theta_{m+1}}$. Pour cela, il est facile de voir que nous avons deux cas à distinguer

a/ $P_{r_i'}^{\theta_m} = P_{r_i'}^{\theta_{m+1}}$.
Ce qui équivaut à dire que $P_{r_i'}^{\theta_m}$ et $P_{r_i'}^{\theta_{m+1}}$ sont à l'ouest d'un bloc avec $deg(P_{r_i'}^{\theta_m}) < deg(P_{r_i}^{\theta_{m+1}})$. En remplaçant $P_{r_i'}^{\theta_m}$ par $P_{r_i'}^{\theta_{m+1}}$ dans (2.48) et en identifiant les coefficients des équations (2.47) et (2.48), nous obtenons

$$q_{i+1}^{\theta_{m+1}} = q_{i+1}^{\theta_m} \neq 0, \quad e_{i+1}^{\theta_m} = e_i^{\theta_{m+1}}.$$

b/ $P_{r_i}^{\theta_{m+1}} = P_{r_i'}^{\theta_m} - e_{r_i}^{\theta_m} P_{r_i'}^{\theta_{m+1}}$.
C'est le cas qui correspond au fait d'avoir $deg(P_{r_i'}^{\theta_m}) = deg(P_{r_i}^{\theta_{m+1}})$. En remplaçant $P_{r_i'}^{\theta_m}$ par $P_{r_i}^{\theta_{m+1}} + e_{r_i}^{\theta_m} P_{r_i'}^{\theta_{m+1}}$ dans (2.48), une simple identification des coefficients des équations (2.47) et (2.48) nous donne

$$q_{i+1}^{\theta_m} + e_{i+1}^{\theta_m} = e_i^{\theta_{m+1}}, \quad q_{i+1}^{\theta_m} e_{r_i}^{\theta_m} = q_{i+1}^{\theta_{m+1}} \neq 0.$$

Signalons qu'il est impossible que $P_{r_i'}^{\theta_m}$ et $P_{r_i'}^{\theta_{m+1}}$ soient au nord d'un bloc lorsque $P_{r_i}^{\theta_m}$ et $P_{r_i}^{\theta_{m+1}}$ sont au nord d'un autre bloc.

- $P_{i+1}^{\theta_{m+1}}$ est au sud d'un bloc et non aux coins.
L'extension des deux relations (2.41) et (2.42) nous permet toujours d'avoir la relation à quatre termes suivante

$$P_{i+1}^{\theta_{m+1}} = xP_i^{\theta_{m+1}} - q_{i+1}^{\theta_m} P_{r_i}^{\theta_{m+1}} - e_{i+1}^{\theta_m} P_{r_i'}^{\theta_{m+1}}, \tag{2.49}$$

avec $e_{i+1}^{\theta_m}$ non nul. Une relation équivalente à cette dernière, dont les coefficients dépendent de l'indice $m+1$ au lieu de m, est la suivante

$$P_{i+1}^{\theta_{m+1}} = xP_i^{\theta_{m+1}} - q_{i+1}^{\theta_{m+1}} P_{r_i}^{\theta_{m+1}} - e_i^{\theta_{m+1}} xP_{r_i'}^{\theta_{m+2}}. \tag{2.50}$$

Ajoutons seulement que, lorsque $P_i^{\theta_{m+2}}$ est au coin sud-ouest du bloc, cette équation n'est valable qu'en remplaçant $P_{r_i'}^{\theta_{m+2}}$ par $P_{r_i}^{\theta_{m+2}}$. Mais comme il n'y a pas de confusion et pour une raison de simplification, nous désignons $P_{r_i}^{\theta_{m+2}}$ par $P_{r_i'}^{\theta_{m+2}}$ lorsque $P_i^{\theta_{m+2}}$ est au coin sud-ouest du bloc.
Afin de pouvoir identifier les coefficients des deux équations équivalentes (2.49) et (2.50), nous allons exprimer $xP_{r_i'}^{\theta_{m+2}}$ en fonction de $P_{r_i}^{\theta_{m+1}}$ et $P_{r_i'}^{\theta_{m+1}}$. Pour cela nous avons deux cas à distinguer

$a'/$ $xP_{r_i'}^{\theta_{m+2}} = P_{r_i'}^{\theta_{m+1}}$.
Ce qui équivaut à dire que $P_{r_i'}^{\theta_{m+2}}$ et $P_{r_i'}^{\theta_{m+1}}$ sont au nord d'un bloc (si $P_i^{\theta_{m+2}}$ est au coin sud-ouest d'un bloc, alors $xP_{r_i}^{\theta_{m+2}} = P_{r_i'}^{\theta_{m+1}}$). En remplaçant $xP_{r_i'}^{\theta_{m+2}}$ par $P_{r_i'}^{\theta_{m+1}}$ dans (2.50) et en identifiant les coefficients des équations (2.49) et (2.50), nous obtenons

$$q_{i+1}^{\theta_{m+1}} = q_{i+1}^{\theta_m}, \quad e_{i+1}^{\theta_m} = e_i^{\theta_{m+1}} \neq 0.$$

$b'/$ $xP_{r_i'}^{\theta_{m+2}} = P_{r_i}^{\theta_{m+1}} + q_{r_i}^{\theta_{m+1}} P_{r_i'}^{\theta_{m+1}}$.
C'est le cas qui correspond au fait d'avoir $deg(xP_{r_i'}^{\theta_{m+2}}) = deg(P_{r_i}^{\theta_{m+1}})$ (si $P_i^{\theta_{m+2}}$ est au coin sud-ouest du bloc, alors $xP_{r_i}^{\theta_{m+2}} = P_{r_i}^{\theta_{m+1}} + q_{r_i}^{\theta_{m+1}} P_{r_i'}^{\theta_{m+1}}$).
En remplaçant $xP_{r_i'}^{\theta_{m+2}}$ par $P_{r_i}^{\theta_{m+1}} + q_{r_i}^{\theta_{m+1}} P_{r_i'}^{\theta_{m+1}}$ dans (2.50), une simple identification des coefficients des équations (2.49) et (2.50) nous donne

$$q_{i+1}^{\theta_m} = e_i^{\theta_{m+1}} + q_{i+1}^{\theta_{m+1}}, \quad e_{i+1}^{\theta_m} = e_i^{\theta_{m+1}} q_{r_i}^{\theta_{m+1}} \neq 0.$$

Signalons qu'il est impossible que $P_{r_i'}^{\theta_{m+2}}$ et $P_{r_i'}^{\theta_{m+1}}$ soient à l'ouest d'un bloc lorsque $P_{r_i}^{\theta_{m+2}}$ et $P_{r_i}^{\theta_{m+1}}$ sont à l'ouest d'un autre bloc.

- $P_{i+1}^{\theta_{m+1}}$ est au coin sud-est d'un bloc.
Comme pour les cas précédents, nous utilisons l'extension des équations (2.41) et (2.42) pour trouver la relation à quatre termes suivante

$$P_{i+1}^{\theta_{m+1}} = xP_i^{\theta_{m+1}} - e_{i+1}^{\theta_m} P_{r_i}^{\theta_{m+1}} - q_{i+1}^{\theta_m} P_{r_i'}^{\theta_m}. \tag{2.51}$$

Une relation équivalente à cette dernière, dont les coefficients dépendent de l'indice $m+1$ au lieu de m, est la suivante

$$P_{i+1}^{\theta_{m+1}} = xP_i^{\theta_{m+1}} - q_{i+1}^{\theta_{m+1}} P_{r_i}^{\theta_{m+1}} - e_i^{\theta_{m+1}} xP_{r_i'}^{\theta_{m+2}}. \tag{2.52}$$

Nous remarquons que (2.51) et (2.52) sont respectivement identiques aux équations (2.48) et (2.50), ce qui s'explique par le fait que $P_{i+1}^{\theta_{m+1}}$ est à la fois au sud et à l'est du bloc. En combinant les cas d'un polynôme à l'est d'un bloc et ceux d'un polynôme au sud que nous avons cités ci-dessus, nous exprimons $xP_{r_i'}^{\theta_{m+2}}$ et $P_{r_i'}^{\theta_m}$ en fonction de $P_{r_i}^{\theta_{m+1}}$ et $P_{r_i'}^{\theta_{m+1}}$. Nous aurons alors au total quatre cas pour pouvoir identifier les coefficients des équations (2.51) et (2.52).

a/ et *a'/*. Ce cas est impossible car les blocs sont tous carrés.

a/ et *b'/*. Par identification des équations (2.51) et (2.52), nous obtenons

$$e_{i+1}^{\theta_m} = e_i^{\theta_{m+1}} + q_{i+1}^{\theta_{m+1}}, \quad q_{i+1}^{\theta_m} = e_i^{\theta_{m+1}} q_{r_i}^{\theta_{m+1}} \neq 0.$$

b/ et *a'/*. Pour ce cas, nous trouvons les relations suivantes

$$q_{i+1}^{\theta_m} + e_{i+1}^{\theta_m} = q_{i+1}^{\theta_{m+1}}, \quad q_{i+1}^{\theta_m} e_{r_i}^{\theta_m} = e_i^{\theta_{m+1}} \neq 0.$$

b/ et *b'/*. Toujours par identification des équations (2.51) et (2.52), nous retrouvons dans ce cas la forme classique suivante

$$\begin{aligned} e_i^{\theta_{m+1}} + q_{i+1}^{\theta_{m+1}} &= q_{i+1}^{\theta_m} + e_{i+1}^{\theta_m} \\ e_i^{\theta_{m+1}} q_{r_i}^{\theta_{m+1}} &= e_{r_i}^{\theta_m} q_{i+1}^{\theta_m} \end{aligned}$$

de l'algorithme *qd*. Il est important de noter que les résultats de ce cas sont aussi valables lorsqu'il n'y a pas de bloc, c'est-à-dire que les polynômes aux coins sont adjacents et orthogonaux.

2.5.4 Propriétés des blocs dans la table de Padé

Posons

$$[p/q]_f^{\theta}(t) = \sum_{i=0}^{i=m-1} c_i t^i + t^m \tilde{Q}_q^{\theta_m}(t)/\tilde{P}_q^{\theta_m}(t)$$

où

$$m = p - q + 1, \quad \tilde{Q}_q^{\theta_m}(t) = t^{q-1} Q_q^{\theta_m}(t^{-1}), \quad \tilde{P}_q^{\theta_m}(t) = t^q P_q^{\theta_m}(t^{-1}).$$

Avec cette notation nous avons $[q-1/q]_{f_m}^{\theta}(t) = \tilde{Q}_q^{\theta_m}(t)/\tilde{P}_q^{\theta_m}(t)$.
Si $P_q^{\theta_m}$ est un polynôme orthogonal régulier avec $m = p - q + 1$, alors il est évident que nous aurons $[p/q]_f^{\theta}(t) = [p/q]_f$.
Supposons maintenant que la table de Padé a un bloc à la k-ième colonne qui peut être décrit comme suit

$$
\begin{array}{llll}
[n+k-1/k]_f & [n+k-1/k+1]_f & \dots & [n+k-1/\theta_n(k)]_f \\
[n+k/k]_f & [n+k/k+1]_f^\theta & \dots & [n+k/\theta_n(k)]_f^\theta \\
[n+k+1/k]_f & [n+k+1/k+1]_f^\theta & \dots & [n+k+1/\theta_n(k)]_f^\theta \\
\vdots & \vdots & & \vdots \\
[n+\theta_n(k)-2/k]_f & [n+\theta_n(k)-2/k+1]_f^\theta & \dots & [n+\theta_n(k)-2/\theta_n(k)]_f^\theta \\
[n+\theta_n(k)-1/k]_f & [n+\theta_n(k)-1/k+1]_f^\theta & \dots & [n+\theta_n(k)-1/\theta_n(k)]_f^\theta \\
[n+\theta_n(k)/k]_f & [n+\theta_n(k)/k+1]_f & \dots & [n+\theta_n(k)/\theta_n(k)]_f
\end{array}
\qquad
\begin{array}{l}
[n+k-1/\theta_n(k)+1]_f \\
[n+k/\theta_n(k)+1]_f \\
[n+k+1/\theta_n(k)+1]_f \\
\vdots \\
[n+\theta_n(k)-2/\theta_n(k)+1]_f \\
[n+\theta_n(k)-1/\theta_n(k)+1]_f \\
[n+\theta_n(k)/\theta_n(k)+1]_f
\end{array}
$$

A l'aide du théorème précédent et de la Remarque 3, nous déduisons les égalités suivantes qui caractérisent un bloc.

$$[n+k-1+i/k+j]_f^\theta = [n+k-1/k]_f, \quad i,j = 0,\dots,\theta_n(k)-k.$$

L'approximant $[n+k-1/k]_f$ est donc le meilleur que nous puissions obtenir dans ce cas puisque d'une part le calcul d'un approximant de Padé $[p/q]_f$ nécessite la connaissance des moments $c_0, c_1, \dots, c_{p+q}$ et d'autre part $n+2k-1 \leq n+2k-1+i+j$. Cet approximant $[n+k-1/k]_f$ est appelé forme de Padé par Gilewicz [58].

2.5.5 Polynômes orthogonaux réciproques

Nous savons que les polynômes orthogonaux réciproques servent à calculer récursivement les numérateurs des approximants de Padé. Dans cette sous-section, nous allons introduire dans la famille des polynômes orthogonaux réciproques les polynômes biorthogonaux que nous avons déjà étudié avant. Ceci va nous permettre d'avoir des propriétés intéressantes pour le calcul récursif des numérateurs des approximants de Padé.
Considérons la série réciproque g de $t^{-\theta(0)}f$ définie par

$$t^{-\theta(0)}f(t)g(t) = 1.$$

Posons $g(t) = \sum_{i=0}^{\infty} d_i t^i$ et définissons une fonctionnelle D sur $\mathbb{C}[X]$ par

$$D(x^i) = d_i, \quad i = 0,1,\dots.$$

D est appelée fonctionnelle réciproque de C. Par convention, nous posons $d_i = c_i = 0$ si $i < 0$. Soit η la permutation associée à la fonctionnelle D que nous appelons permutation

réciproque de θ. Nous remarquons que la définition de la fonctionnelle réciproque D nous donne $\eta(0) = 0$. Nous allons voir plus tard la relation qui relie les deux permutations η et θ.
Les nombres complexes d_i peuvent être calculés par les relations suivantes

$$c_{\theta(0)}d_0 = 1 \tag{2.53}$$

et

$$c_{\theta(0)}d_i + c_{\theta(0)+1}d_{i-1} + \ldots + c_{\theta(0)+i}d_0 = 0, \quad i = 1, 2, \ldots. \tag{2.54}$$

Signalons que (2.54) nous donne les d_i d'une manière unique et qu'elle est équivalente à la relation suivante

$$c_0 d_i + c_1 d_{i-1} + c_2 d_{i-2} + \ldots + c_i d_0 = 0, \quad i = \theta(0)+1, \theta(0)+2, \ldots. \tag{2.55}$$

Nous allons maintenant étudier le lien entre les deux familles $\{P_k\}_k$ et $\{R_k\}_k$ de polynômes unitaires et orthogonaux respectivement par rapport à C et D. Vu la singularité et la non existence de certains polynômes orthogonaux, nous allons étendre l'étude aux deux familles $\{P_k^\theta\}_k$ et $\{R_k^\eta\}_k$ données par la méthode ALA. Pour cette étude, nous allons considérer les trois mises en oeuvre de ALA.

Comme pour C, nous définissons la fonctionnelle $D^{(n)}$ par

$$D^{(n)}(x^i) = d_{n+i}, \quad i = 0, 1, \ldots,$$

à laquelle nous associons la permutation η_n et la fonctionnelle g_n où $n \in \mathbb{Z}$. Les polynômes orthogonaux, par rapport à la fonctionnelle $D^{(n)}$, seront alors notés $R_k^{\eta_n}$. La permutation θ_0 n'est autre que θ et η_0 n'est autre que η. Nous verrons dans cette sous-section qu'il est possible, par une simple relation, de connaître η_n à partir de θ_n.
Tout d'abord, donnons une propriété des polynômes associés par rapport à une fonctionnelle.

Le polynôme $Q_q^{\theta_n}(t)$ associé à $P_q^{\theta_n}$, par rapport à la fonctionnelle $C^{(n)}$, est donné par

$$Q_q^{\theta_n}(t) = C^{(n)}[(P_q^{\theta_n}(x) - P_q^{\theta_n}(t))/(x-t)],$$

où $C^{(n)}$ agit sur x.

Lemme 2.3 *Si $Q_k^{\theta_n}$ est associé au polynôme $P_k^{\theta_n}$ de degré k, alors*

$$Q_k^{\theta_n}(t) = \sum_{i=0}^{m} t^i C^{(n-i-1)}(P_k^{\theta_n}(x)),$$

où $m = n + k - 1 - \theta_n(0)$. $Q_k^{\theta_n}(t)$ est de degré m si $m \geq 0$, sinon, $Q_k(t) = 0$.

Proof :

$Q_k^{\theta_n}(t)$ est égal à $C^{(n)}[(P_k^{\theta_n}(x) - P_k^{\theta_n}(t))/(x-t)]$. En utilisant l'égalité

$$1/(x-t) = x^{-1}\sum_{i=0}^{\infty}(x^{-1}t)^i,$$

nous montrons que

$$Q_k^{\theta_n}(t) = C^{(n)}\left([P_k^{\theta_n}(x) - P_k^{\theta_n}(t)]x^{-1}\sum_{i=0}^{n+k-1}(x^{-1}t)^i\right),$$

où $C^{(n)}$ agit sur x. Enfin, en tenant compte du fait que $c_i = 0$ lorsque $i < \theta(0)$, nous obtenons le résultat du lemme. ∎

Dans le théorème suivant, nous allons voir les relations existantes entre les polynômes orthogonaux réguliers par rapport à $C^{(n)}$ et ceux par rapport à $D^{(n)}$.

Théorème 2.9 *Le polynôme $P_k^{\theta_{\theta(0)+n+1}}$ est régulier si et seulement si $R_{n+k}^{\eta_{-n+1}}$ est régulier. De même, $P_{n+k}^{\theta_{\theta(0)-n+1}}$ est régulier si et seulement si $R_k^{\eta_{n+1}}$ est régulier.*
Si nous supposons que $P_k^{\theta_{\theta(0)+n+1}}$ et $P_{n+k}^{\theta_{\theta(0)-n+1}}$ sont orthogonaux réguliers, alors nous aurons les égalités suivantes

$$S_{n+k}^{\eta_{-n+1}} = d_0 P_k^{\theta_{\theta(0)+n+1}}, \quad Q_{n+k}^{\theta_{\theta(0)-n+1}} = c_{\theta(0)} R_k^{\eta_{n+1}}, \quad n = 1, 2, \ldots,$$

$$\begin{cases} c_{\theta(0)} R_{n+k}^{\eta_{-n+1}} = P_k^{\theta_{\theta(0)+n+1}} \sum_{i=0}^{n} c_{\theta(0)+i} x^{n-i} + Q_k^{\theta_{\theta(0)+n+1}} \\ d_0 P_{n+k}^{\theta_{\theta(0)-n+1}} = R_k^{\eta_{n+1}} \sum_{i=0}^{n} d_i x^{n-i} + S_k^{\eta_{n+1}} \end{cases} \qquad n = 0, 1, \ldots.$$

Preuve :

Pour démontrer ce théorème, il suffit de remarquer que $f_{\theta(0)}$ est la série réciproque de g (c'est-à-dire que $C^{(\theta(0))}$ est la fonctionnelle réciproque de D) et ainsi se ramener au cas étudié dans [10, 44]. Lorsque $\theta(0) = 0$, la démonstration donnée dans [10] des égalités de ce théorème est longue, elle consiste à transformer les déterminants des expressions explicites des polynômes orthogonaux. Une démonstration simple, utilisant uniquement le Lemme 2.3, est donnée dans la preuve du Théorème 2.12 qui est une généralisation du

Théorème 2.9. ∎

D'après ce théorème, il est clair que pour un entier n fixé, $R_{n+k}^{\eta_{-n+1}}$, $S_{n+k}^{\eta_{-n+1}}$, $P_k^{\theta_{\theta(0)+n+1}}$ et $Q_k^{\theta_{\theta(0)+n+1}}$ vérifient les mêmes relations de récurrence avec des initialisations différentes. Il en est de même pour $P_{n+k}^{\theta_{\theta(0)-n+1}}$, $Q_{n+k}^{\theta_{\theta(0)-n+1}}$, $R_k^{\eta_{n+1}}$ et $S_k^{\eta_{n+1}}$. Si nous posons, pour tout $n, k \in \mathbb{N}$,

$$N_k^{\eta_{n+2}} = c_{\theta(0)} R_k^{\eta_{-n}} \quad \text{et} \quad N_k^{\eta_{-n+1}} = c_{\theta(0)} R_k^{\eta_{n+1}},$$

alors un approximant de Padé $[p/q]_f$ peut s'écrire sous la forme

$$[p/q]_f = \widetilde{N}_p^{\eta_{p-q+1}} / \widetilde{P}_q^{\theta_{\theta(0)+p-q+1}}$$

lorsque nous supposons que $P_q^{\theta_{\theta(0)+p-q+1}}$ est régulier, voir [10].

Nous déduisons de ce qui précède qu'aussi bien en l'absence de blocs qu'en leur présence dans la table de Padé, le numérateur d'un approximant de Padé peut être calculé récursivement à l'aide des relations de récurrence qui nous permettent d'obtenir son dénominateur. Notons que ces relations de récurrence dépendent du choix d'une mise en oeuvre de ALA.

Corollaire 2.2 *Nous pouvons connaître les permutations η_n en fonction des θ_n grâce aux relations suivantes*

$$\eta_{-n+1}(i) = \theta_{\theta(0)+n+1}(i-n) + n, \quad \theta_{\theta(0)-n+1}(i) = \eta_{n+1}(i-n) + n, \quad \textit{pour} \quad i \geq n, \ n \geq 1,$$

$$\theta_{\theta(0)-n+1}(i) = \eta_{-n+1}(i) = n - 1 - i, \quad \textit{pour} \quad i = 0, 1, \ldots, n-1.$$

Preuve :

D'après la définition des deux permutations θ_n et η_n et en utilisant le Théorème 2.9, nous déduisons le résultat du Corollaire 2.2 puisque la connaissance des degrés des polynômes orthogonaux réguliers implique celle des deux permutations η_n et θ_n, voir le Théorème 2.2.
Signalons que dans le Théorème 2.2, pour que nous puissions parler de η_n, il suffit de remplacer la fonctionnelle C par $D^{(n)}$, θ par η_n et les polynômes orthogonaux par rapport à C par ceux qui sont orthogonaux par rapport à $D^{(n)}$. ∎

Nous avons montré comment utiliser les trois mises en oeuvre de ALA pour calculer récursivement les numérateurs et les dénominateurs des approximants de Padé. Le calcul

récursif des numérateurs fait intervenir les quantités

$$Q_{n+k}^{\theta_{\theta(0)-n+1}} \quad \text{et} \quad P_k^{\theta_{\theta(0)+n+1}} \sum_{i=0}^{n} c_{\theta(0)+i} x^{n-i} + Q_k^{\theta_{\theta(0)+n+1}}.$$

Ces quantités ne vérifient pas les égalités du Théorème 2.9 lorsque $P_k^{\theta_{\theta(0)+n+1}}$ et $P_{n+k}^{\theta_{\theta(0)-n+1}}$ ne sont pas orthogonaux réguliers. C'est pour cette raison que nous allons maintenant donner une signification à ces quantités au point de vue de l'orthogonalité, en utilisant la première mise en oeuvre de ALA. Nous n'allons pas considérer la deuxième et la troisième mise en oeuvre de ALA car la manipulation des expressions explicites des polynômes biorthogonaux intermédiaires relatifs à ces deux mises en oeuvre est complexe.

Pour tout $n \in \mathbb{Z}$ et $k \in \mathbb{N}$, nous considérons des polynômes unitaires $R_k^{'\eta_n}$ que nous définissons ici pour la première mise en oeuvre de ALA. Le rôle de ces polynômes est de remplacer les $R_k^{\eta_n}$ dans les égalités du Théorème 2.9 afin que ce dernier reste valable même dans le cas où $P_k^{\theta_{\theta(0)+n+1}}$ et $P_{n+k}^{\theta_{\theta(0)-n+1}}$ ne sont pas des polynômes orthogonaux réguliers. Nous définissons les $R_k^{'\eta_n}$ par

$$D^{(n)}(R_k^{'\eta_n} t^{\eta_n(j)}) + \alpha_{n,k} d_{\eta_n(j)-\eta_n(k)} = 0, \quad j = 0, \ldots, k-1, \tag{2.56}$$

avec $\alpha_{n,k}$ est une constante qui nous permet seulement d'avoir $R_k^{'\eta_n}$ unitaire.

Il est clair que la famille ${R_k^{'\eta_n}}_{n,k}$ est consruite de telle sorte qu'elle contienne tous les polynômes orthogonaux réguliers par rapport à la fonctionnelle $D^{(n)}$. D'après la définition de ces polynômes, nous pouvons voir facilement que leur condition d'existence et d'unicité est la même que celle des $R_k^{\eta_n}$. Par conséquent, tout polynôme $R_k^{'\eta_n}$ existe et est unique de degré égal à son indice k.
Commençons tout d'abord par nous intéresser aux expressions explicites des polynômes des deux familles $\{P_k^{\theta}\}_k$ et $\{R_k^{'\eta}\}_k$.

En utilisant la première mise en oeuvre de ALA, nous obtenons le résultat du théorème suivant qui est semblable à celui que nous trouvons en cas normal.

Théorème 2.10 *Le polynôme $c_{\theta(0)}R_k^{'\eta}(x)$ peut s'écrire sous la forme explicite suivante*

$$\begin{vmatrix} c_{\theta(0)+\eta(1)+1} & c_{\theta(0)+\eta(1)+2} & \cdots & c_{\theta(0)+\eta(1)+k} \\ c_{\theta(0)+\eta(2)+1} & c_{\theta(0)+\eta(2)+2} & \cdots & c_{\theta(0)+\eta(2)+k} \\ \vdots & \vdots & \cdots & \vdots \\ c_{\theta(0)+\eta(k-1)+1} & c_{\theta(0)+\eta(k-1)+2} & \cdots & c_{\theta(0)+\eta(k-1)+k} \\ c_{\theta(0)}x + c_{\theta(0)+1} & c_{\theta(0)}x^2 + c_{\theta(0)+1}x + c_{\theta(0)+2} & \cdots & c_{\theta(0)}x^k + c_{\theta(0)+1}x^{k-1} + \ldots + c_{\theta(0)+k} \end{vmatrix}$$

$$\begin{vmatrix} c_{\theta(0)+\eta(1)+1} & c_{\theta(0)+\eta(1)+2} & \cdots & c_{\theta(0)+\eta(1)+k-1} \\ c_{\theta(0)+\eta(2)+1} & c_{\theta(0)+\eta(2)+2} & \cdots & c_{\theta(0)+\eta(2)+k-1} \\ \vdots & \vdots & \cdots & \vdots \\ c_{\theta(0)+\eta(k-1)+1} & c_{\theta(0)+\eta(k-1)+2} & \cdots & c_{\theta(0)+\eta(k-1)+k-1} \end{vmatrix}.$$

Preuve :

Nous pouvons transformer le numérateur et le dénominateur de la forme explicite de $R_k^{'\eta}(x)$ pour obtenir la forme donnée dans le théorème. Mais pour une démonstration plus simple et compréhensible, nous donnons une preuve qui ressemble à celle qui se trouve dans [10] en cas normal.
Nous voulons montrer que

$$D(x^{\eta(p)}R_k^{'\eta}(x)) + \alpha_{0,k}d_{\eta(p)-\eta(k)} = 0, \quad \text{pour} \quad p = 0, 1, \ldots, k-1.$$

En utilisant la forme de $R_k^{'\eta}$ dans le théorème, nous pouvons avoir $D(x^{\eta(p)}R_k^{'\eta}(x))$ sous forme d'un rapport de deux déterminants. Les éléments de la dernière ligne de celui qui est au numérateur sont donnés par

$$\begin{aligned} & D(x^{\eta(p)}(c_{\theta(0)}x^m + c_{\theta(0)+1}x^{m-1} + \ldots + c_{\theta(0)+m})) \\ =& c_{\theta(0)}d_{\eta(p)+m} + c_{\theta(0)+1}d_{\eta(p)+m-1} + \ldots + c_{\theta(0)+m}d_{\eta(p)} \\ =& -\sum_{i=0}^{\eta(p)-1} d_i c_{\theta(0)+m+\eta(p)-i} \end{aligned}$$

et ceci pour $m = 1, 2, \ldots, k$. A l'aide de cette relation, en écrivant

$$\alpha_{0,k} = d_0 C^{(\theta(0)+2)}(x^{\theta_{\theta(0)+2}(k-1)}P_{k-1}^{\theta_{\theta(0)+2}}(x))$$

sous forme d'un rapport de deux déterminants, nous pouvons voir que $D(x^{\eta(p)}R_k^{'\eta}(x))$ n'est autre que $-\alpha_{0,k}d_{\eta(p)-\eta(k)}$. Par conséquent, $D(x^{\eta(p)}R_k^{'\eta}(x)) + \alpha_{0,k}d_{\eta(p)-\eta(k)} = 0$ pour $p = 0, 1, \ldots, k-1$. ∎

Soit $\{S_k^{'\eta}\}_k$ la famille des polynômes associés par rapport à la fonctionnelle D aux polynômes de la famille $\{R_k^{'\eta}\}_k$.

Théorème 2.11 *Le polynôme $c_{\theta(0)}S_k^{'\eta}$, associé par rapport à la fonctionnelle D au polynôme $c_{\theta(0)}R_k^{'\eta}$, est donné par le rapport*

$$\frac{\begin{vmatrix} c_{\theta(0)+\eta(1)+1} & c_{\theta(0)+\eta(1)+2} & \cdots & c_{\theta(0)+\eta(1)+k} \\ c_{\theta(0)+\eta(2)+1} & c_{\theta(0)+\eta(2)+2} & \cdots & c_{\theta(0)+\eta(2)+k} \\ \vdots & \vdots & \ldots & \vdots \\ c_{\theta(0)+\eta(k-1)+1} & c_{\theta(0)+\eta(k-1)+2} & \cdots & c_{\theta(0)+\eta(k-1)+k} \\ 1 & x & \ldots & x^{k-1} \end{vmatrix}}{\begin{vmatrix} c_{\theta(0)+\eta(1)+1} & c_{\theta(0)+\eta(1)+2} & \cdots & c_{\theta(0)+\eta(1)+k-1} \\ c_{\theta(0)+\eta(2)+1} & c_{\theta(0)+\eta(2)+2} & \cdots & c_{\theta(0)+\eta(2)+k-1} \\ \vdots & \vdots & \ldots & \vdots \\ c_{\theta(0)+\eta(k-1)+1} & c_{\theta(0)+\eta(k-1)+2} & \cdots & c_{\theta(0)+\eta(k-1)+k-1} \end{vmatrix}}.$$

Preuve:

Par définition d'un polynôme associé, nous avons

$$c_{\theta(0)}S_k^{'\eta}(t) = D\left(\frac{c_{\theta(0)}R_k^{'\eta}(x) - c_{\theta(0)}R_k^{'\eta}(t)}{x-t}\right).$$

Si nous écrivons $(c_{\theta(0)}R_k^{'\eta}(x) - c_{\theta(0)}R_k^{'\eta}(t))/(x-t)$ comme rapport de deux déterminants, alors les éléments de la dernière ligne de celui qui est au numérateur sont donnés par

$$\begin{aligned}\psi_m(t,x) &= c_{\theta(0)}\sum_{j=0}^{m-1} x^{m-1-j}t^j + c_{\theta(0)+1}\sum_{j=0}^{m-2} x^{m-2-j}t^j + \ldots + c_{\theta(0)+m-1} \\ &= \sum_{i=0}^{m-1} c_{\theta(0)+i}\sum_{j=0}^{m-1-i} x^{m-1-i-j}t^j\end{aligned}$$

pour $m = 1, 2, \ldots, k$. Appliquons à $\psi_m(t,x)$ la fonctionnelle réciproque D agissant sur la variable x

$$D(\psi_m(t,x)) = \sum_{i=0}^{m-1} c_{\theta(0)+i}\sum_{j=0}^{m-1-i} d_{m-1-i-j}t^j,$$

D'une manière simple, nous pouvons aussi écrire $D(\psi_m(t,x))$ comme un polynôme en t en nous servant du fait que $d_i = 0$ lorsque $i < 0$

$$D(\psi_m(t,x)) = \sum_{j=0}^{m-1}(\sum_{i=0}^{m-1} c_{\theta(0)+i}d_{m-1-i-j})t^j.$$

Or

$$\sum_{i=0}^{m-1} c_{\theta(0)+i}d_{m-1-i-j} = \sum_{i=0}^{m-1-j} c_{\theta(0)+i}d_{m-1-j-i} = \delta_{m-1j}$$

donc $D(\psi_m(t,x)) = t^{m-1}$, d'où le résultat du théorème. ∎

Corollaire 2.3 *Nous avons les égalités suivantes*

$$c_{\theta(0)} S_k^{'\eta} = P_{k-1}^{\theta_{\theta(0)+2}}$$

et

$$c_{\theta(0)} R_{k-1}^{\eta_2} = Q_k^{'\theta_{\theta(0)}}$$

où nous désignons par $Q_k^{'\theta_n}$ le polynôme associé à $P_k^{'\theta_n}$ par rapport à la fonctionnelle $C^{(n)}$.

Preuve :

Tout d'abord, remarquons que la deuxième égalité se déduit de la première en considérant la série réciproque $f_{\theta(0)}$ de g, c'est-à-dire que nous devons avoir

$$d_0 Q_k^{'\theta_{\theta(0)}} = R_{k-1}^{\eta_2}$$

puisque $\eta(0) = 0$.
La première égalité est une conséquence du Théorème 2.11. En effet, il suffit de voir que pour tout entier $p > 0$, nous avons

$$c_{\theta(0)+\eta(p)+1} \in \{c_{\theta(0)+2}, c_{\theta(0)+3}, c_{\theta(0)+4}, \ldots\}.$$

∎

Le résultat du Corollaire 2.3 nous permet d'affirmer que les polynômes $S_k^{'\eta}$ et $R_k^{'\eta}$ vérifient les mêmes relations de récurrence que $P_{k-1}^{\theta_{\theta(0)+2}}$ et $Q_{k-1}^{\theta_{\theta(0)+2}}$ avec bien sûr des initialisations différentes.

D'une manière plus générale, essayons maintenant de donner, pour la première mise en oeuvre de ALA, les relations existantes entre les polynômes des familles

$$\{P_k^{\theta_n}\}_{n,k}, \quad \{R_k^{'\eta_n}\}_{n,k}, \quad \{Q_k^{\theta_n}\}_{n,k} \text{ et } \{S_k^{'\eta_n}\}_{n,k}$$

en utilisant celles déjà connues entre les polynômes orthogonaux réguliers.

Théorème 2.12 *Pour la première mise en oeuvre de ALA, nous avons*

$$Q_{n+k}^{\theta_{\theta(0)-n+1}} = c_{\theta(0)} R_k^{'\eta_{n+1}}, \quad S_{n+k}^{'\eta_{-n+1}} = d_0 P_k^{\theta_{\theta(0)+n+1}}, \quad n = 1, 2, \ldots,$$

$$\begin{cases} c_{\theta(0)} R_{n+k}^{'\eta_{-n+1}} = P_k^{\theta_{\theta(0)+n+1}} \sum_{i=0}^{n} c_{\theta(0)+i} x^{n-i} + Q_k^{\theta_{\theta(0)+n+1}} \\ d_0 P_{n+k}^{\theta_{\theta(0)-n+1}} = R_k^{'\eta_{n+1}} \sum_{i=0}^{n} d_i x^{n-i} + S_k^{'\eta_{n+1}} \end{cases} \quad n = 0, 1, \ldots.$$

Preuve :

Montrons que $Q_{n+k}^{\theta_{\theta(0)-n+1}} = c_{\theta(0)} R_k^{'\eta_{n+1}}$. Afin de ne pas compliquer la démonstration, commençons par supposer que $\theta(0) = 0$. Dans ce cas, grâce au Lemme 2.3, nous avons pour $j = 0, 1, \ldots, k-1$

$$\begin{aligned} D^{(n+1)}[Q_{n+k}^{\theta_{-n+1}}(t) t^{\eta_{n+1}(j)}] &= D^{(n+1)}[\sum_{i=0}^{k} t^{i+\eta_{n+1}(j)} C^{(-n-i)}(P_{n+k}^{\theta_{-n+1}}(x))] \\ &= \sum_{l=0}^{n+k} a_l \sum_{i=0}^{k} d_{i+\eta_{n+1}(j)+n+1} c_{-n-i+l} \end{aligned}$$

où $D^{(n+1)}$ agit sur t, $C^{(-n-i)}$ agit sur x et $P_{n+k}^{\theta_{-n+1}}(x) = \sum_{l=0}^{n+k} a_l x^l$. $\theta(0)$ est supposé nul, donc d'après le Corollaire 2.2 $\theta_{-n+1}(j+n) = \eta_{n+1}(j) + n$. D'où

$$\begin{aligned} D^{(n+1)}[Q_{n+k}^{\theta_{-n+1}}(t) t^{\eta_{n+1}(j)}] &= \sum_{l=0}^{n+k} a_l \sum_{i=0}^{l-n} d_{i+\theta_{-n+1}(j+n)+1} c_{-n-i+l} \\ &= -\sum_{l=0}^{n+k} a_l \sum_{i=0}^{\theta_{-n+1}(j+n)} d_{\theta_{-n+1}(j+n)-i} c_{-n+1+l-i} \\ &= -\sum_{i=0}^{\theta_{-n+1}(j+n)} d_{\theta_{-n+1}(j+n)-i} C^{(-n+1)}(x^i P_{n+k}^{\theta_{-n+1}}(x)) \\ &= -d_{\theta_{-n+1}(j+n)-\theta_{-n+1}(k+n)} C^{(-n+1)}(x^{\theta_{-n+1}(k+n)} P_{n+k}^{\theta_{-n+1}}(x)) \\ &= -d_{\eta_{n+1}(j)-\eta_{n+1}(k)} C^{(-n+1)}(x^{\theta_{-n+1}(k+n)} P_{n+k}^{\theta_{-n+1}}(x)). \end{aligned}$$

En conséquence, nous obtenons $Q_{n+k}^{\theta_{-n+1}} = c_0 R_k^{'\eta_{n+1}}$.
Dans le cas où $\theta(0) \neq 0$, $f_{\theta(0)}$ est la série réciproque de g. Nous pouvons donc suivre le même raisonnement que celui de ci-dessus et conclure ensuite que

$$Q_{n+k}^{\theta_{\theta(0)-n+1}} = c_{\theta(0)} R_k^{'\eta_{n+1}}.$$

D'une manière tout à fait similaire, nous montrons que

$$S_{n+k}^{'\eta_{-n+1}} = d_0 P_k^{\theta_{\theta(0)+n+1}}.$$

Les deux dernières égalités qui restent à démontrer se déduisent des deux précédentes. En effet, d'après les deux égalités précédentes, pour n fixé, les polynômes $R_{n+k}^{'\eta_{-n+1}}$, $P_k^{\theta_{\theta(0)+n+1}}$ et $Q_k^{\theta_{\theta(0)+n+1}}$ vérifient les mêmes relations de récurrence. Il en est de même pour les polynômes

$P_{n+k}^{\theta_{\theta(0)-n+1}}$, $R_k^{'\eta_{n+1}}$ et $S_k^{'\eta_{n+1}}$. Donc les deux dernières égalités sont vraies si elles le sont initialement, ce qui est le cas puisque

$$S_0^{'\eta_{n+1}} = Q_0^{\theta_{\theta(0)+n+1}} = 0, \quad P_0^{\theta_{\theta(0)+n+1}} = R_0^{'\eta_{n+1}} = 1$$

et

$$R_n^{'\eta_{-n+1}} = \sum_{i=0}^{n} d_i x^{n-i}, \quad P_n^{\theta_{\theta(0)-n+1}} = \sum_{i=0}^{n} c_{\theta(0)+i} x^{n-i}.$$

■

Si nous posons, pour tout $n, k \in \mathbb{N}$,

$$N_k^{'\eta_{n+2}} = c_{\theta(0)} R_k^{'\eta_{-n}} \quad \text{et} \quad N_k^{'\eta_{-n+1}} = c_{\theta(0)} R_k^{'\eta_{n+1}},$$

alors nous pouvons conclure à partir de ce théorème que tout approximant de Padé $[p/q]_f^{\theta}$ peut s'écrire sous la forme

$$[p/q]_f^{\theta} = \widetilde{N}_p^{'\eta_{p-q+1}} / \widetilde{P}_q^{\theta_{\theta(0)+p-q+1}}.$$

Remarque 2.11 Pour les autres mises en oeuvre de ALA, nous pouvons définir les polynômes $R_k^{'\eta_n}$ par

$$Q_{n+k+1}^{\theta_{\theta(0)-n}} = c_{\theta(0)} R_k^{'\eta_{n+2}}, \quad c_{\theta(0)} R_{n+k}^{'\eta_{-n+1}} = P_k^{\theta_{\theta(0)+n+1}} \sum_{i=0}^{n} c_{\theta(0)+i} x^{n-i} + Q_k^{\theta_{\theta(0)+n+1}}, \quad n = 0, 1, \ldots.$$

Grâce à cette définition, ces polynômes se calculent récursivement à l'aide des mêmes relations de récurrence que celles qui nous fournissent les coefficients des polynômes $P_k^{\theta_n}$, ce qui est important puisque la famille $\{R_k^{'\eta_n}\}_{n,k}$ contient tous les polynômes orthogonaux réguliers par rapport à la fonctionnelle $D^{(n)}$.

Nous pouvons calculer les coefficients du numérateur d'un approximant de Padé de deux manières différentes

1. La première utilise les relations de récurrence reliant les polynômes $P_q^{\theta_{\theta(0)+p-q+1}}$ qui se trouvent sur une même diagonale de la table P. Ceci est tout à fait possible car, à partir du Théorème 2.12, nous savons que ces relations de récurrence s'appliquent aussi aux polynômes $N_p^{'\eta_{p-q+1}}$ avec des initialisations différentes.
   Pour les trois mises en oeuvre de la méthode ALA, l'obtention des coefficients du numérateur de l'approximant de Padé $[p/q]_f^{\theta}$ exige au plus $2p-1$ multiplications et $2p-1$ additions et suppose la connaissance de deux approximants de Padé non identiques précédant $[p/q]_f^{\theta}$ et se trouvant sur la même diagonale que ce dernier.

2. La seconde utilise les équations de (2.38) qui nous donnent directement les coefficients a_i du numérateur de l'approximant de Padé

$$[p/q]_f^\theta(t) = \frac{a_0 + a_1 t + \ldots + a_p t^p}{b_0 + b_1 t + \ldots + b_q t^q}$$

à partir de ceux du dénominateur. Dans ce cas, le coût algorithmique est de $(q+1)(q+2)/2 + (p-q)(q+1)$ multiplications et $q(q+1)/2 + (p-q)q$ additions si $p \geq q$ et il est de $(p+1)(p+2)/2 + (q-p)(p+1)$ multiplications et $p(p+1)/2 + (q-p)p$ additions si $p \leq q$.

Au point de vue coût algorithmique, il est préférable d'utiliser la seconde méthode si nous voulons calculer un seul approximant. En revanche, lorsqu'il s'agit de calculer plusieurs approximants de Padé, il est évident que la première est nettement moins coûteuse.

En pratique, nous ne savons pas au départ l'approximant qui nous convient, il est donc plus commode de calculer récursivement par les mêmes relations de récurrence les numérateurs et les dénominateurs des approximants de Padé. D'où la préférence de l'utilisation de la première méthode.

Nous allons voir maintenant le comportement de l'erreur commise sur le calcul des coefficients a_i et b_i.
Pour la première méthode, nous remarquons que les deux erreurs commises respectivement sur le calcul des coefficients a_i et b_i ont le même comportement parce qu'ils sont la conséquence de l'utilisation des mêmes relations de récurrence.
Pour la deuxième méthode, l'erreur commise sur le calcul des coefficients a_i est donnée par le système triangulaire (2.38). Elle dépend de l'erreur commise sur le calcul des coefficients b_i. C'est-à-dire que nous ajoutons à l'erreur commise sur le calcul des b_i celle due au produit de la matrice triangulaire de (2.38) par un vecteur.
Ceci donne aussi la préférence à l'utilisation de la première méthode.

2.5.6 Résultats numériques

Nous allons utiliser les trois mises en oeuvre de la méthode ALA dans le calcul des coefficients du dénominateur

$$P(t) = b_0 + b_1 t + \ldots + b_q t^q$$

et du numérateur

$$Q(t) = a_0 + a_1 t + \ldots + a_p t^p$$

d'un approximant de Padé $[p/q]_f^\theta$ arbitraire. Les coefficients du numérateur et ceux du dénominateur de $[p/q]_f^\theta$ sont calculés à l'aide des mêmes relations de récurrence et non pas directement en utilisant les équations de (2.38). Les trois algorithmes résultants, correspondants aux trois mises en oeuvre de la méthode ALA, nous permettent de nous déplacer suivant n'importe quelle diagonale de la table de Padé. Sachant p et q et les moments c_i, $i = 0, 1, \ldots, p+q$, ces trois algorithmes nous donnent les coefficients du numérateur et du dénominateur de l'approximant de Padé $[p/q]_f^\theta$. Ils suivent la diagonale de la table de Padé qui contient $[p/q]_f^\theta$.

Exemple 1

Pour ce premier exemple, nous considérons la série suivante

$$f(t) = 1 + t^5 + t^9/2 + t^{13}/4 + t^{17}/8 + t^{21}/16 + \ldots$$

qui converge vers la fonction $(1-t^4/2+t^5)/(1-t^4/2)$ pour $t \in]-\sqrt[4]{2}, \sqrt[4]{2}[$. Pour le seuil de détection des blocs, nous posons $tol = 10^{-12}$. Suivant les trois mises en oeuvre de ALA, la table de Padé possède plusieurs blocs. Nous obtenons

$$\begin{aligned}
&[p/q]_f^\theta = [0/0]_f^\theta = 1 \quad \text{pour} \quad p, q \in \{0,1,2,3,4\},\\
&[p/q]_f^\theta = [4i+1/0]_f^\theta = 1 + t^5 \sum_{j=0}^{i-1} t^{4j}/2^j \quad \text{pour} \quad 4i+1 \geq p \leq 4i+4,\ q \in \{0,1,2,3\},\ i \geq 1,\\
&[p/q]_f^\theta = [0/5]_f^\theta = 1/(1-t^5) \quad \text{pour} \quad p \in \{0,1,2,3\},\ q \in \{5,6,7,8\},\\
&[p/q]_f^\theta = [5/4]_f^\theta = (1-t^4/2+t^5)/(1-t^4/2) \quad \text{pour} \quad p \geq 5,\ q \geq 4.
\end{aligned}$$

Les approximants $[0/0]_f^\theta$, $[5/0]_f^\theta$, $[0/5]_f^\theta$, $[5/4]_f^\theta$ sont aux coins d'un bloc d'order 4.
Pour une série qui représente une fonction rationnelle, nous savons que les approximants de Padé $[p/q]_f^\theta$ redonnent la fonction lorsque p et q sont plus grands (ou égaux) respectivement aux degrés du numérateur et du dénominateur de cette fonction rationnelle. Ici, $(1-t^4/2+t^5)/(1-t^4/2)$ est une fonction rationnelle, ce qui explique le fait que $[p/q]_f^\theta = (1-t^4/2+t^5)/(1-t^4/2)$ pour $p \geq 5$ et $q \geq 4$.
Nous allons donner maintenant, suivant les trois mises en oeuvre de ALA, les numérateurs et les dénominateurs que nous obtenons pour deux approximants qui se trouvent à l'intérieur d'un même bloc.

$[p/q]_f^\theta = P/Q$	$[2/6]_f^\theta$	$[3/7]_f^\theta$
Première mise en oeuvre	$P = 1+2t$ $Q = 1+2t-t^5-2t^6$	$P = 1+2t+4t^2$ $Q = 1+2t+4t^2-t^5-2t^6-4t^7$
Deuxième mise en oeuvre	$P = 1+2t$ $Q = 1+2t-t^5-2t^6$	$P = 1+2t+4t^2$ $Q = 1+2t+4t^2-t^5-2t^6-4t^7$
Troisième mise en oeuvre	$P = 1$ $Q = 1-t^5$	$P = 1+2t+4t^2$ $Q = 1+2t+4t^2-t^5-2t^6-4t^7$

A partir de ces résultats, nous pouvons dire qu'à l'intérieur d'un bloc, P et Q peuvent être différents d'une mise en oeuvre à l'autre, même si la fraction $P/Q = [2/6]_f^\theta = [3/7]_f^\theta$ garde une même valeur. Nous expliquons ceci par le fait que les trois mises en oeuvre de ALA utilisent, à l'intérieur des blocs, des polynômes biorthogonaux différents. Ceci est illustré dans l'exemple suivant.

Exemple 2

Nous allons considérer la série étudiée dans [26].

$$f(t) = 1 + at + at^2 + \ldots + at^{m-1} + t^m + at^{m+1} + at^{m+2} + \ldots + at^{2m-1} + t^{2m} + at^{2m+1} + \ldots$$

Cette série converge vers la fonction rationnelle $(1 + at + at^2 + \ldots + at^{m-1})/(1 - t^m)$ pour $t \in]-1, 1[$. En posant $a = 0.001$, $tol = 10^{-16}$ et $m = 7$, l'application des trois mises en oeuvre de ALA nous donne les valeurs suivantes des coefficients du numérateur et du dénominateur de $[4/4]_f^\theta$ qui se trouve à l'intérieur d'un bloc d'ordre 4.

a_i, b_i	Premième mise en oeuvre	Deuxième mise en oeuvre	Troisième mise en oeuvre
a_0	0.1000000000000000D+01	0.1000000000000000D+01	0.1000000000000000D+01
a_1	0.9000000000001229D-03	0.9009000000000003D+00	0.9009000000000003D+00
a_2	0.8100000000001994D-03	0.8116200000000002D+00	0.8116200000000005D+00
a_3	0.7290000000002017D-03	0.7311870000000003D+00	-0.2184813000000001D+01
a_4	-0.6554439000000004D+00	-0.2621775600000001D+01	0.0000000000000000D+00
b_0	0.1000000000000000D+01	0.1000000000000000D+01	0.1000000000000000D+01
b_1	0.1110223024625157D-15	0.9000000000000002D+00	0.9000000000000002D+00
b_2	0.1110223024625157D-15	0.8100000000000001D+00	0.8100000000000003D+00
b_3	0.1110223024625157D-15	0.7290000000000001D+00	-0.2187000000000001D+01
b_4	-0.6561000000000003D+00	-0.2624400000000001D+01	0.0000000000000000D+00

Nous déduisons de ces résultats que les trois mises en oeuvre nous donnent des numérateurs et des dénominateurs qui sont différents pour un même approximant $[4/4]_f^\theta$. Ceci différencie les trois mises en oeuvre.

Exemple 3

Soit la série

$$f(t) = \sum_{i=0}^{\infty} c_i t^i = t^3 - t^6/2 + t^9/3 - \dots$$

qui converge vers $\mathrm{Log}(1+t^3)$ si $t \in]-1,1]$.
Nous appliquons les trois mises en oeuvre de ALA à la fonction f pour le calcul des éléments de la diagonale principale (celle qui contient $[0/0]_f^\theta$) de la table de Padé. L'exécution des algorithmes qui correspondent aux trois mises en oeuvre avec $tol = 10^{-12}$, dans le but d'obtenir l'approximant $[10/10]_f^\theta$, rencontre quatre blocs et nous montre que l'approximant $[10/10]_f^\theta$ est à l'intérieur du quatrième bloc qui est d'ordre 2. $[9/9]_f^\theta$ est au coin nord-est de ce bloc.
Pour l'approximant $[10/10]_f^\theta$, nous trouvons

a_i, b_i	Premième mise en oeuvre	Deuxième mise en oeuvre	Troisième mise en oeuvre
a_3	0.1000000000000000D+01	0.1000000000000000D+01	0.1000000000000000D+01
a_6	0.9999999999999951D+00	0.9999999999999951D+00	0.1000000000000002D+01
a_9	0.1833333333333311D+00	0.1833333333333311D+00	0.1833333333333351D+00
b_0	0.1000000000000000D+01	0.1000000000000000D+01	0.1000000000000000D+01
b_3	0.1499999999999995D+01	0.1499999999999995D+01	0.1500000000000002D+01
b_6	0.5999999999999954D+00	0.5999999999999954D+00	0.6000000000000030D+00
b_9	0.4999999999999938D-01	0.4999999999999934D-01	0.5000000000000070D-01

Les autres coefficients sont nuls. La première et la deuxième mises en oeuvre nous donnent les mêmes valeurs des coefficients a_i, b_i à l'exception de celle de b_9 pour laquelle il y a une légère différence. Nous remarquons aussi une légère différence entre la troisième mise en oeuvre et les deux autres, concernant les valeurs de a_6, a_9, b_3, b_6 et b_9.
Pour $[14/14]_f^\theta$, nous obtenons les valeurs suivantes

a_i, b_i	Premième mise en oeuvre	Deuxième mise en oeuvre	Troisième mise en oeuvre
a_3	0.1000000000000000D+01	0.1000000000000000D+01	0.1000000000000000D+01
a_6	0.1499999999999689D+01	0.1499999999999689D+01	0.1499999999999844D+01
a_9	0.6190476190472924D+00	0.6190476190472927D+00	0.6190476190474557D+00
a_{12}	0.5952380952374431D-01	0.5952380952374434D-01	0.5952380952377828D-01
b_0	0.1000000000000000D+01	0.1000000000000000D+01	0.1000000000000000D+01
b_3	0.1999999999999689D+01	0.1999999999999689D+01	0.1999999999999845D+01
b_6	0.1285714285713804D+01	0.1285714285713804D+01	0.1285714285714047D+01
b_9	0.2857142857140833D+00	0.2857142857140832D+00	0.2857142857141869D+00
b_{12}	0.1428571428569599D-01	0.1428571428569599D-01	0.1428571428570568D-01

Pour cet approximant, nous remarquons aussi une légère différence entre les valeurs des a_i, b_i données par les trois mises en oeuvre.

Pour $t = 1$, nous avons Log(2) = 0.6931471805599453.... En utilisant les approximants de Padé $[k + 1/k]_f^\theta$ obtenus par les trois mises en oeuvre de ALA, nous allons chercher une approximation de la valeur exacte de Log(2).

k	$S_{2k+1} = \sum_{i=0}^{2k+1} c_i$	1-ère, 2-ème et 3-ème mise en oeuvre
3	0.5000	0.66
6	0.5833	0.6923
9	0.6166	0.693121
12	0.6345	0.6931464
15	0.6456	0.693147158
18	0.6532	0.693147179
21	0.6587	0.693147180540
24	0.6628	0.69314718055935
27	0.6628	0.693147180559927
30	0.6687	0.6931471805599447
32	0.7163	0.6931471805599454

Ici, les trois mises en oeuvre de ALA nous donnent exactement les mêmes résultats. Ces résultats nous permettent de remarquer que la suite $([k + 1/k]_f^\theta)_k$ des approximants de Padé converge plus rapidement que $(S_{2k+1})_k$. Ce phénomène caractérise les approximants de Padé des séries de Stieltjes et explique leur utilisation dans la pratique.

Si nous considérons la série

$$h(t) = \sum_{i=0}^{\infty} c_i' t^i = t - t^2/2 + t^3/3 - \dots$$

qui converge vers Log$(1 + t)$ si $t \in]-1, 1]$, alors nous aurons $f(t) = h(t^3)$. Cette égalité montre que $[p/q]_h^\theta(t^3)$ est l'approximant de Padé $[3p/3q]_f^\theta(t)$ de f pour tout $p, q \in \mathbb{N}$. L'application des trois mises en oeuvre à h montre qu'en posant $t = 1$,

$$[k + 1/k]_h^\theta(1) = [3k + 3/3k + 2]_f^\theta(1) \quad \text{pour} \quad k = 1, 2, \dots.$$

Pour cet exemple, nous trouvons

$$[2/1]_h^\theta(1) = [6/5]_f^\theta(1) = 0.7, \quad [6/5]_h^\theta(1) = [18/17]_f^\theta(1) = 0.6931471849621315.$$

2.6 Application à l'epsilon algorithme

L'epsilon algorithme, dû à Wynn [126], est défini par la relation suivante

$$\varepsilon_{k+1}^{(n)} = \varepsilon_{k-1}^{(n+1)} + (\varepsilon_k^{(n+1)} - \varepsilon_k^{(n)})^{-1}. \tag{2.57}$$

Si $\varepsilon_0^{(n)} = \sum_{i=0}^{n} c_i t^i$ et $\varepsilon_{-1}^{(n)} = 0$, alors nous aurons l'égalité suivante

$$\varepsilon_{2k}^{(n)} = [n + k/k]_f(t)$$

où $f(t) = \sum_{i=0}^{\infty} c_i t^i$.

Dans notre cas, les quantités $\varepsilon_{2k+1}^{(n)}$ ne présentent pas d'intérêt. L'epsilon algorithme, sous sa forme (2.57), souffre d'une instabilité numérique. En effet, des erreurs de cancellation importantes dues à la différence $\varepsilon_k^{(n+1)} - \varepsilon_k^{(n)}$ peuvent affecter l'algorithme. Une telle instabilité numérique peut être évitée, soit en utilisant la forme progressive suivante

$$\varepsilon_{k+1}^{(n+1)} = \varepsilon_{k+1}^{(n)} + (\varepsilon_{k+2}^{(n)} - \varepsilon_k^{(n+1)})^{-1} \tag{2.58}$$

de l'epsilon algorithme, soit en utilisant les règles particulières données par Wynn. Pour plus de détails sur la stabilité numérique des processus d'accélération de la convergence, voir [40, 21].

Dans cette section, nous allons nous intéresser au calcul des itérés d'indices pairs $\varepsilon_{2j}^{(n)}$. Ces itérés peuvent être calculés en utilisant l'extension de la méthode de bordage que Piñar et Ramirez ont donné dans [92] et qui est une extension de la méthode qui se trouve dans [9]. Nous pouvons aussi utiliser les extensions de la méthode de bordage obtenues grâce aux mises en oeuvre de la méthode ALA. Ici, nous allons appliquer l'extension de la méthode de bordage basée sur la première et la deuxième mise en oeuvre de ALA, afin de donner deux algorithmes simples qui serviront à calculer les itérés $\varepsilon_{2j}^{(n)}$ de l'epsilon algorithme.

Pour que $\varepsilon_{2j}^{(n)}$ soit défini pour tout $n \in \mathbb{Z}$ et tout $j \in \mathbb{N}$, nous posons

$$\varepsilon_{2j}^{(n)} = [n + j/j]_f^{\theta}(t).$$

Ce qui nous donne

$$\varepsilon_{2j}^{(n-1)} = \sum_{i=0}^{i=n-1} c_i t^i + t^n [j - 1/j]_{f_n}^{\theta}(t) \tag{2.59}$$

avec

$$[j - 1/j]_{f_n}^{\theta}(t) = \tilde{Q}_j^{\theta_n}(t)/\tilde{P}_j^{\theta_n}(t), \quad \tilde{Q}_j^{\theta_n}(t) = t^{j-1} Q_j^{\theta_n}(t^{-1}), \quad \tilde{P}_j^{\theta_n}(t) = t^j P_j^{\theta_n}(t^{-1}),$$

où la série formelle f_n est définie par

$$f_n(t) = c_n + c_{n+1} t^{n+1} + c_{n+2} t^{n+2} + \ldots.$$

La famille des polynômes $\{P_j^{\theta_n}\}_j$ est donnée par la première ou la deuxième mise en oeuvre de la méthode ALA en imposant maintenant, à chaque polynôme $P_j^{\theta_n}$, la condition de

normalisation $P_j^{\theta_n}(1) = 1$ au lieu d'être unitaire de degré j. Cette famille contient tous les polynômes orthogonaux $P_j = P_j^{\theta_n}$ qui existent par rapport à la fonctionnelle $C^{(n)}$ et vérifient la condition de normalisation $P_j(1) = 1$.

Considérons maintenant la famille $\{P'_j\}_j$ de polynômes orthogonaux par rapport à la fonctionnelle $C'^{(n)} = C^{(n+1)} - C^{(n)}$, en imposant à chaque polynôme P'_j d'être unitaire et de degré j. D'une manière tout à fait similaire à ce que nous avons vu au début de la section consacrée à l'application de ALA à la méthode de Lanczos, nous pouvons montrer que P'_j existe si et seulement si P_j existe aussi. Donc, par la première ou la deuxième mise en oeuvre de ALA, nous aurons une même permutation θ_n et par conséquent la famille $\{P_j'^{\theta_n}\}_j$.

Pour tout entier j, considérons $Q_j^{\theta_n}$ et $Q_j'^{\theta_n}$ comme étant les deux polynômes associés respectivement aux polynômes $P_j^{\theta_n}$ et $P_j'^{\theta_n}$ par rapport à la fonctionnelle $C^{(n)}$, c'est-à-dire

$$Q_j^{\theta_n}(t) = C^{(n)}\left(\frac{P_j^{\theta_n}(t) - P_j^{\theta_n}(x)}{t-x}\right), \quad Q_j'^{\theta_n}(t) = C^{(n)}\left(\frac{P_j'^{\theta_n}(t) - P_j'^{\theta_n}(x)}{t-x}\right),$$

avec $C^{(n)}$ agissant sur x.

Deuxième mise en oeuvre

Nous allons calculer les polynômes de la famille $\{P_j'^{\theta_n}\}_j$ à l'aide de la relation suivante qui est équivalente à (2.10)

$$\begin{cases} \alpha_j = C'^{(n)}(x^{\theta_n(k)-k+1} P_j'^{\theta_n} P_k'^{\theta_n})/C'^{(n)}(x^{\theta_n(k)} P_k'^{\theta_n}) \\ \beta_j = C'^{(n)}(x^k P_j'^{\theta_n})/C'^{(n)}(x^{k-1} P'^{\theta_n}_{\theta_n(k-1)}) \\ P'^{\theta_n}_{j+1} = x P_j'^{\theta_n} - \alpha_j P_k'^{\theta_n} - \beta_j P'^{\theta_n}_{\theta_n(k-1)} \end{cases} \qquad j = k, k+1, \ldots, \theta_n(k), \tag{2.60}$$

pour $k = 0, \theta_n(0)+1, \theta_n(\theta_n(0)+1)+1, \ldots$. L'initialisation de cette relation de récurrence se fait en posant $P_0'^{\theta_n} = 1$ et $P_{-1}'^{\theta_n} = 0$ avec $\theta_n(-1) = -1$.

Le calcul des polynômes de la famille $\{P_j^{\theta_n}\}_j$ se fait grâce à la relation suivante

$$\begin{cases} \lambda_j = C^{(n)}(P_k'^{\theta_n})/C'^{(n)}(x^{\theta_n(k)} P_k'^{\theta_n}) \\ P^{\theta_n}_{j+1} = P_j^{\theta_n} - \lambda_j (x-1) P_j'^{\theta_n} \end{cases} \qquad j = k, k+1, \ldots, \theta_n(k), \tag{2.61}$$

pour $k = 0, \theta_n(0)+1, \theta_n(\theta_n(0)+1)+1, \ldots$. Nous n'allons pas démontrer cette relation car elle est de même type que la formule $(F3)$ de la sous-section 2.4.2.

En nous servant des deux relations de récurrence (2.60) et (2.61), nous montrons que les polynômes associés $Q_j'^{\theta_n}$ et $Q_j^{\theta_n}$ vérifient les deux relations de récurrence suivantes

$$Q'^{\theta_n}_{j+1}(t) = t Q_j'^{\theta_n}(t) - \alpha_j Q_k'^{\theta_n}(t) - \beta_j Q'^{\theta_n}_{\theta_n(k-1)}(t) + C^{(n)}(P_j'^{\theta_n}(x)) \tag{2.62}$$

et

$$Q^{\theta_n}_{j+1}(t) = Q_j^{\theta_n}(t) - \lambda_j[(t-1) Q_j'^{\theta_n}(t) + C^{(n)}(P_j'^{\theta_n}(x))] \tag{2.63}$$

pour $j = k, k+1, \ldots, \theta_n(k)$ et $k = 0, \theta_n(0)+1, \theta_n(\theta_n(0)+1)+1, \ldots$. L'initialisation se fait avec $Q_{-1}^{'\theta_n} = 0$, $Q_0^{\theta_n} = Q_0^{'\theta_n} = 0$ et $\theta_n(-1) = -1$. Les coefficients α_j, β_j et λ_j sont les mêmes que ceux des deux relations précédentes (2.60) et (2.61).
La quantité $C^{(n)}(P_j^{'\theta_n}(x))$ peut être calculée récursivement en utilisant la relation de récurrence (2.60). En effet, par une simple application de la fonctionnelle $C^{(n)}$ à (2.60), nous obtenons

$$C^{(n)}(P_{j+1}^{'\theta_n}) = C^{(n)}(xP_j^{'\theta_n}) - \alpha_j C^{(n)}(P_k^{'\theta_n}) - \beta_j C^{(n)}(P_{\theta_n(k-1)}^{'\theta_n})$$

or, $C^{(n)}(xP_j^{'\theta_n}) = C^{(n)}((x-1)P_j^{'\theta_n}) + C^{(n)}(P_j^{'\theta_n})$ est égal à $C^{(n)}(P_j^{'\theta_n})$ si $j \neq \theta_n(0)$ et est égal à $C^{(n)}(P_j^{'\theta_n}) + c_{n+\theta_n(0)+1} - c_{n+\theta_n(0)}$ si $j = \theta_n(0)$. Nous pouvons alors écrire la relation de récurrence suivante

$$a_{j+1} = a_j - \alpha_j a_k - \beta_j a_{\theta_n(k-1)} + \delta_{j\theta_n(0)}(c_{n+\theta_n(0)+1} - c_{n+\theta_n(0)}), \tag{2.64}$$

où δ_{ij} est le symbole de Kronecker et $a_i = C^{(n)}(P_i^{'\theta_n})$ pour tout entier i.

Prenons maintenant le cas simple $t = 1$. En conséquence, la relation (2.59) nous donne

$$\varepsilon_{2j}^{(n-1)} = \sum_{i=0}^{n-1} c_i + Q_j^{\theta_n}(1)$$

et en utilisant (2.63), nous aurons la relation de récurrence suivante

$$\varepsilon_{2j+2}^{(n-1)} = \varepsilon_{2j}^{(n-1)} - \lambda_j a_j \tag{2.65}$$

pour $j = k, k+1, \ldots, \theta_n(k)$ et $k = 0, \theta_n(0)+1, \theta_n(\theta_n(0)+1)+1, \ldots$. Pour l'initialisation de cette relation de récurrence nous posons

$$\varepsilon_0^{(n-1)} = \sum_{i=0}^{n-1} c_i.$$

D'après la relation (2.61), l'expression donnant le coefficient λ_j ne dépend pas des polynômes $P_j^{\theta_n}$. Donc, pour calculer les itérés $\varepsilon_{2j}^{(n-1)}$, nous avons besoin des relations de récurrence (2.65), (2.64) et (2.60) auxquelles nous ajoutons la relation de (2.61) qui nous donne le coefficient λ_j. Nous avons maintenant tout ce qu'il faut pour énoncer l'algorithme ci-après qui calcule les éléments d'indices pairs $\varepsilon_{2j}^{(n-1)}$ du tableau de l'epsilon algorithme suivant une diagonale quelconque.

Pour tout entier i, nous posons dans l'algorithme ci-dessous

$$P_i^{'\theta_n}(x) = \sum_{j=0}^{i} q_j^{(i)} x^j, \quad q_i^{(i)} = 1.$$

Algorithme 3

- Etape 1 : Initialisation

  Choisir un entier n et poser $q_0^{(-1)} = q_{-1}^{(-1)} = 0$, $h_{-1} = 1$, $k_0 = -1$,

  $a_{-1} = 0$, $a_0 = c_n$, $q_0^{(0)} = 1$, $q_{-1}^{(0)} = 0$, $k = 0$, $\varepsilon_0^{(n-1)} = \sum_{i=0}^{i=n-1} c_i$.

- Etape 2 (Détermination de $\theta_n(k)$)

  1 $i = 0$

  2 $h_{k+i} = \sum_{j=0}^{k} (c_{n+k+i+j+1} - c_{n+k+i+j}) q_j^{(k)}$

  si $|h_{k+i}| < tol$ pour une certaine tolérance tol, alors

  $i = i + 1$

  aller à 2

  fin (si)

  $\theta_n(k) = k + i$

  $d_{m-1} = \sum_{j=0}^{k} (c_{n+\theta_n(k)+m+j+1} - c_{n+\theta_n(k)+m+j}) q_j^{(k)}$, $m = 1, \ldots, i+1$

- Etape 3

  $b_k = a_k / h_{\theta_n(k)}$

  pour $i = k, \ldots, \theta_n(k)$

  $\varepsilon_{2i+2}^{(n-1)} = \varepsilon_{2i}^{(n-1)} - b_k a_i$

  $\alpha_i = q_{k-1}^{(i)} + (\sum_{j=k}^{i} d_{j-k} q_j^{(i)}) / h_{\theta_n(k)}$

  $\beta_i = \sum_{j=0}^{i} (c_{n+j+k+1} - c_{n+j+k}) q_j^{(i)} / h_{k-1}$

  $q_j^{(i+1)} = q_{j-1}^{(i)}$, $j = 1, \ldots, i+1$

  $q_j^{(i+1)} \leftarrow q_j^{(i+1)} - \alpha_i q_j^{(k)}$, $j = 0, \ldots, k$

  $q_j^{(i+1)} \leftarrow q_j^{(i+1)} - \beta_i q_j^{(k_0)}$, $j = 0, \ldots, k_0$

  $a_{i+1} = a_i - \alpha_i a_k - \beta_i a_{k_0}$

  fin (pour)

  si $k = 0$ alors $a_{\theta_n(0)+1} \leftarrow a_{\theta_n(0)+1} + (c_{n+\theta_n(0)+1} - c_{n+\theta_n(0)})$

  fin (si)

  $k_0 \leftarrow k$

  $k \leftarrow \theta_n(k) + 1$

  aller à 1

fin.

Première mise en oeuvre

Selon cette mise en oeuvre, le calcul des coefficients des polynômes $P_j^{'\theta_n}$ se fait à l'aide de l'Algorithme 1, c'est-à-dire en utilisant les deux équations suivantes

$$\begin{cases} \beta_j = C'^{(n)}(x^{\theta_n(k)+1}P_j^{'\theta_n})/C'^{(n)}(x^{\theta_n(k)}P_k^{'\theta_n}) \\ P_{j+1}^{'\theta_n} = xP_j^{'\theta_n} - \beta_j P_k^{'\theta_n} \end{cases} \quad j = k, \ldots, \theta_n(k)-1 \tag{2.66}$$

et

$$\begin{cases} \alpha_{k-1} = -C'^{(n)}(x^{\theta_n(k)}P_k^{'\theta_n})/C'^{(n)}(x^{k-1}P_{\theta_n(k-1)}^{'\theta_n}) \neq 0 \\ \alpha_k = -[C'^{(n)}(x^{\theta_n(k)+1}P_{\theta_n(k)}^{'\theta_n}) + \alpha_{k-1}C'^{(n)}(x^{\theta_n(k)}P_{\theta_n(k-1)}^{'\theta_n})]/C'^{(n)}(x^{\theta_n(k)}P_k^{'\theta_n}) \\ \alpha_i = C'^{(n)}(x^{\theta_n(i)}P_{\theta_n(k-1)}^{'\theta_n})/C'^{(n)}(x^{k-1}P_{\theta_n(k-1)}^{'\theta_n}), \quad i = k+1, \ldots, \theta_n(k) \\ P_{\theta_n(k)+1}^{'\theta_n} = xP_{\theta_n(k)}^{'\theta_n} + \sum_{i=k+1}^{\theta_n(k)} \alpha_i P_i^{'\theta_n} + \alpha_k P_k^{'\theta_n} + \alpha_{k-1}P_{\theta_n(k-1)}^{'\theta_n} \end{cases} \tag{2.67}$$

déduites des Théorèmes 2.2 et 2.3. Les polynômes $P_j^{\theta_n}$ se calculent grâce à la relation suivante

$$\begin{cases} \lambda_j = C^{(n)}(x^{\theta_n(j)}P_k^{\theta_n})/C'^{(n)}(x^{\theta_n(k)}P_k^{'\theta_n}) \\ P_{j+1}^{\theta_n} = P_j^{\theta_n} - \lambda_j(x-1)P_j^{'\theta_n} \end{cases} \quad j = k, k+1, \ldots, \theta_n(k), \tag{2.68}$$

dont la démonstration est de même type que celle de $(F3)$ de la sous-section 2.4.2.
A l'aide de (2.68), il est clair que la relation de récurrence (2.63) est aussi valable pour cette mise en oeuvre, elle nous permet de calculer les polynômes associés $Q_j^{\theta_n}$. De même (2.65) reste aussi valable.
Comme pour (2.64), nous appliquons la fonctionnelle $C^{(n)}$ aux équations (2.66) et (2.67) pour obtenir les deux relations suivantes

$$a_{j+1} = a_j - \beta_j a_k, \quad j = k, \ldots, \theta_n(k)-1 \tag{2.69}$$

et

$$a_{\theta_n(k)+1} = a_{\theta_n(k)} + \sum_{i=k+1}^{\theta_n(k)} \alpha_i a_i + \alpha_k a_k + \alpha_{k-1}a_{\theta_n(k-1)} + \delta_{k0}(c_{n+\theta_n(0)+1} - c_{n+\theta_n(0)}). \tag{2.70}$$

dont les coefficients β_i et α_i sont ceux de (2.66) et (2.67).

Enfin, nous concluons que pour utiliser (2.65), nous avons besoin des relations (2.66), (2.67), (2.68), (2.69) et (2.70). Ces relations définissent l'algorithme que nous allons énoncer ci-après et qui est une application de l'extension de la méthode de bordage selon la

première mise en oeuvre de ALA au calcul des itérés d'indices pairs de l'epsilon algorithme.
Pour tout entier i, nous posons, dans l'algorithme ci-après,

$$P_i'^{\theta_n}(x) = \sum_{j=0}^{i} q_j^{(i)} x^j \quad \text{et} \quad P_i^{\theta_n}(x) = \sum_{j=0}^{i} p_j^{(i)} x^j$$

avec $q_i^{(i)} = 1$ et $p_i^{(i)} = 1$.

Algorithme 4

- Etape 1 : Initialisation
  Choisir un entier n et poser $q_0^{(-1)} = q_{-1}^{(-1)} = 0$, $h_{-1} = 1$, $k_0 = -1$, $a_{-1} = 0$, $a_0 = c_n$, $q_0^{(0)} = p_0^{(0)} = 1$, $q_{-1}^{(0)} = 0$, $k = 0$, $\varepsilon_0^{(n-1)} = \sum_{i=0}^{i=n-1} c_i$.
- Etape 2 (Détermination de $\theta_n(k)$)
  1 $i = 0$
  2 $h_{k+i} = \sum_{j=0}^{k} (c_{n+k+i+j+1} - c_{n+k+i+j}) q_j^{(k)}$
  $d_i = \sum_{j=0}^{k_0} (c_{n+k+i+j+1} - c_{n+k+i+j}) q_j^{(k_0)}$, $e_i = \sum_{j=0}^{k} c_{n+k+i+j} p_j^{(k)}$
  si $|h_{k+i}| < \varepsilon_1$ pour une certaine tolérance ε_1, alors
  $i = i + 1$
  aller à 2
  fin (si)
  $\theta_n(k) = k + i$.

- Etape 3

$b = h_{\theta_n(k)}/h_{k-1}$, $r = 0$
$w_j = -q_j^{(k_0)}$, $j = 0, \ldots, k_0$
$w_j = 0$, $j = k_0 + 1, \ldots, \theta_n(k)$
pour $i = k, \ldots, \theta_n(k)$
$e_{\theta_n(k)-i} \leftarrow e_{\theta_n(k)-i}/h_{\theta_n(k)}$, $\varepsilon_{2i+2}^{(n-1)} = \varepsilon_{2i}^{(n-1)} - e_{\theta_n(k)-i}a_i$
$p_j^{(i+1)} = p_j^{(i)} - e_{\theta_n(k)-i}(q_{j-1}^{(i)} - q_j^{(i)})$, $j = 0, \ldots, i$
$p_{i+1}^{(i+1)} = -e_{\theta_n(k)-i}$
$w_j \leftarrow w_j + (d_{\theta_n(k)-i}/h_{\theta_n(k)})q_j^{(i)}$, $j = 0, \ldots, i$
$r \leftarrow r + (d_{\theta_n(k)-i}/h_{\theta_n(k)})a_i$, $\beta_i = (\sum_{j=0}^{i} c_{\theta_n(k)+1+j}q_j^{(i)})/h_{\theta_n(k)}$
$q_j^{(i+1)} = q_{j-1}^{(i)} - \beta_i q_j^{(k)}$, $j = 0, \ldots, k$
$q_j^{(i+1)} = q_{j-1}^{(i)}$, $j = k+1, \ldots, i+1$
$q_{-1}^{(i+1)} = 0$, $a_{i+1} = a_i - \beta_i a_k$
fin (pour)
si $k = 0$ alors $a_{\theta_n(0)+1} \leftarrow a_{\theta_n(0)+1} + (c_{n+\theta_n(0)+1} - c_{n+\theta_n(0)})$
fin (si)
$k_0 = k$, $k = \theta_n(k) + 1$, $a_k \leftarrow a_k + br$
$q_j^{(k)} \leftarrow q_j^{(k)} + bw_j$, $j = 0, \ldots, k-1$
aller à 1

fin.

Nous signalons que l'Algorithme 3 et l'Algorithme 4 sont applicables aux suites. Si par exemple nous voulons les appliquer à une suite $\{U_i\}_{i \in \mathbb{N}}$, alors il suffit de poser $c_0 = U_0$ et $c_i = U_i - U_{i-1}$ pour $i > 0$ et $t = 1$.

2.6.1 Résultats numériques

Exemple 1

Nous commençons par donner l'application de l'Algorithme 3 et l'Algorithme 4 à la série

$$f(t) = 1 + at + at^2 + \ldots + at^{m-1} + t^m + at^{m+1} + at^{m+2} + \ldots + at^{2m-1} + t^{2m} + at^{2m+1} + \ldots$$

qui a été étudiée dans [26]. Cette série converge vers la fonction rationnelle

$$(1 + at + at^2 + \ldots + at^{m-1})/(1 - t^m)$$

pour $t \in]-1,1[$. Nous posons $t = 0.9, a = 0.001$, $tol = 10^{-16}$ et $m = 7$. Dans le but de trouver une approximation de la valeur exacte $1.924882238575926\ldots$ de $f(0.9)$, nous appliquons l'Algorithme 3 et l'Algorithme 4 en posant $c_i = at^i$. Nous obtenons les résultats suivants.

$\varepsilon_{2k}^{(0)}$	Algorithme 4	Algorithme 3
$\varepsilon_0^{(0)}$	0.1000000000000000D+01	0.1000000000000000D+01
$\varepsilon_2^{(0)}$	0.1009000000000000D+01	0.1009000000000000D+01
$\varepsilon_4^{(0)}$	0.1009000000000000D+01	0.1009000000000000D+01
$\varepsilon_6^{(0)}$	0.1009000000000000D+01	0.1009000000000000D+01
$\varepsilon_8^{(0)}$	0.1009000000000000D+01	0.1009000000000000D+01
$\varepsilon_{10}^{(0)}$	0.1009000000000000D+01	0.1009000000000000D+01
$\varepsilon_{12}^{(0)}$	0.9999745519928722D+00	0.9999745519928722D+00
$\varepsilon_{14}^{(0)}$	0.1924882238577011D+01	0.1924882238573816D+01

Dans cet exemple, une DPZ est évitée à l'itération $k = 2$ jusqu'à $k = 5$.
Comme la série f représente une fonction rationnelle, à partir des propriétés des approximants de Padé, nous déduisons qu'en arithmétique exacte la suite $\{\varepsilon_{2k}^{(0)}\}_k$ redonne à la septième itération la valeur exacte de $f(0.9)$. Ceci explique le fait que $\varepsilon_{14}^{(0)}$ nous donne une bonne approximation de $f(0.9)$ en utilisant l'Algorithme 4 ou l'Algorithme 3.

Exemple 2

Nous choisissons la série

$$f(t) = \sum_{i=0}^{\infty} 4t^{5i}/(2i+1)$$

qui converge vers la fonction S définie par

$$S(t) = \begin{cases} (4/\sqrt{t^5})\text{Arcth}(\sqrt{t^5}) \quad \text{pour} \quad t \in]0,1[, \\ (4/\sqrt{-t^5})\text{Arctg}(\sqrt{-t^5}) \quad \text{pour} \quad t \in]-1,0[, \\ S(0) = 4, \ S(-1) = \pi = 3.14159265358979323846264 33\ldots. \end{cases}$$

Pour $t = -1$, l'application de l'Algorithme 3 et de l'Algorithme 4 à cette série avec $tol = 10^{-14}$ donne les résultats suivants.

k	S_{2k-1}	$\varepsilon_{2k}^{(-1)}$ obtenu par Algo 4	$\varepsilon_{2k}^{(-1)}$ obtenu par Algo 3
1	4.000	4.000000000000000	4.000000000000000
6	2.666	3.166666666666667	3.166666666666667
11	3.466	3.142342342342343	3.142342342342341
16	2.895	3.141614906832299	3.141614906832296
21	3.339	3.141593311879929	3.141593311879925
26	2.976	3.141592673030337	3.141592673030332
31	3.283	3.141592654163369	3.141592654163364
36	3.017	3.141592653606708	3.141592653606703
41	3.252	3.141592653590294	3.141592653590289
46	3.041	3.141592653589810	3.141592653589791

Pour ces résultats, il est évident que l'Algorithme 3 et l'Algorithme 4 accélèrent la convergence de la suite des nombres $S_1 = 4$, $S_{11} = 4 - 4/3$, $S_{21} = 4 - 4/3 + 4/5, \ldots$, donnée par Leibniz (1646-1716), vers la valeur exacte de π. Nous remarquons aussi dans cet exemple que l'Algorithme 3 et l'Algorithme 4 ont presque le même comportement.

2.7 Conclusion

Dans le cas normal, le calcul des polynômes orthogonaux peut se faire à l'aide d'une récurrence simple à trois termes. Dans le cas non normal, cette relation à trois termes n'est plus valable lorsqu'il s'agit de calculer un polynôme orthogonal régulier se trouvant à l'est ou au sud d'un bloc, à cause de la non existence et la singularité de certains polynômes orthogonaux. Nous pouvons chercher le polynôme orthogonal suivant en résolvant un système triangulaire régulier. Mais, grâce à la méthode ALA étudiée dans ce chapitre, nous pouvons éviter de résoudre un système linéaire en remplaçant les polynômes singuliers et ceux qui n'existent pas par des polynômes biorthogonaux qui vérifient des relations de récurrence à trois termes. Grâce à la deuxième mise en oeuvre de ALA, nous pouvons même avoir une relation de récurrence à quatre termes au plus, aussi bien pour les polynômes orthogonaux réguliers que pour les polynômes biorthogonaux introduits. La première, la deuxième et la troisième mise en oeuvre de la méthode ALA nous ont permis d'étendre d'une manière simple la méthode de bordage que nous avons utilisée pour calculer les itérés d'indices pairs de l'epsilon algorithme, suivant une diagonale. Nous avons aussi montré que la première, la deuxième et la troisième mise en oeuvre de ALA nous permettent de programmer récursivement les approximants de Padé. Une extension simple de l'algorithme *qd* est donnée en utilisant la deuxième mise en oeuvre de ALA. Nous avons aussi donné dans ce chapitre l'application des trois mises en oeuvre de ALA à la méthode de Lanczos. Ce que nous devons aussi retenir de ce chapitre est que ces trois mises en oeuvre de ALA sont simples du point de vue programmation.

Chapitre 3

Systèmes étendus hermitiens et préconditionnement ILU

3.1 Introduction

Plusieurs méthodes directes et itératives pour la résolution des systèmes d'équations linéaires sont connues. Un système linéaire hermitien nous donne plusieurs simplifications et la possibilité d'adapter la programmation à l'ordinateur pour une bonne performance. Une de ces simplifications est le fait que nous utilisons des relations à trois termes qui sont faciles à programmer.
Donc le thème de ce chapitre est de considérer les méthodes dérivées d'un système hermitien augmenté - aussi appelé système hermitien étendu - et faire une comparaison avec la méthode GMRES (Generalized Minimal Residual method). Nous allons aussi montrer comment appliquer le préconditionnement ILU au système hermitien étendu, ce qui améliore la convergence de la méthode étudiée dans ce chapitre.

Nous voulons résoudre un système linéaire non hermitien

$$Ax = b, \tag{3.1}$$

où la matrice $A \in \mathbb{C}^{n\times n}$ est non singulière et où $b \in \mathbb{C}^n$. Souvent, lorsque la matrice A est large et creuse, nous préférons les méthodes itératives aux méthodes directes pour diverses raisons

- le stockage en mémoire de l'ordinateur dépend en grande partie de la structure de la matrice A,
- souvent, une approximation de la solution x de (3.1) est obtenue en un petit nombre d'itérations, c'est-à-dire à petit coût algorithmique,

- en général, les méthodes itératives sont faciles à mettre en oeuvre et à adapter aux différents types de problèmes. Par exemple, nous pouvons les utiliser pour obtenir récursivement une approximation des valeurs propres.

En général, pour les systèmes provenant des équations aux dérivées partielles en trois dimensions, les méthodes directes seules sont trop coûteuses, aussi bien en stockage qu'en coût algorithmique. Cependant, le problème principal de l'utilisateur est de devoir choisir des méthodes itératives adaptées au problème en question.
La méthode GMRES introduite par Saad et Schultz [100] est l'une des méthodes itératives les plus efficaces, sa mise en oeuvre utilise le processus d'Arnoldi et la base de Newton pour construire les bases orthonormales de certains sous-espaces de Krylov [5]. Mais la rapidité de la convergence de la méthode GMRES n'est pas toujours garantie comme c'est le cas également pour les autres méthodes itératives. Théoriquement, la méthode GMRES peut stagner et donc ne pas donner une bonne approximation de la solution de (3.1) en peu d'itérations. Ceci est l'une des raisons pour lesquelles nous considérons dans ce chapitre un système étendu hermitien.

3.2 Systèmes hermitiens

Dans cette section, nous voulons transformer (3.1) en un système hermitien, ce qui est équivalent à transformer la matrice A en une matrice hermitienne. Pour cela, nous allons considérer un système augmenté ou étendu.

3.2.1 Systèmes étendus

Un simple système hermitien étendu associé à (3.1) est

$$A_{e\lambda}x_e = b_e, \quad A_{e\lambda} = \begin{pmatrix} \lambda I & A^* \\ A & 0 \end{pmatrix}, \quad b_e = \begin{pmatrix} b' \\ b \end{pmatrix}, \quad x_e = \begin{pmatrix} x \\ x' \end{pmatrix}, \tag{3.2}$$

où I est la matrice identité de $\mathbb{C}^{n\times n}$, λ est un réel positif arbitraire, et b' un vecteur arbitraire de $\mathbb{C}^n$. Ce système étendu, avec $\lambda = 0$, a été proposé pour la première fois par Hestenes et Stiefel dans [70]. Saunders a donné dans [108] quelques propriétés d'un système étendu similaire. Joly et Meurant [75] ont montré que les méthodes de type gradient conjugué, appliquées à (3.2), convergent lentement. Nous allons montrer comment utiliser un préconditionnement convenable pour obtenir une convergence acceptable.
Considérons le vecteur initial x_{0_e} et le vecteur résidu initial $r_{0_e} = b_e - A_{e\lambda}x_{0_e}$ de $\mathbb{C}^{2n}$ comme suit

$$r_{0_e} = \begin{pmatrix} r'_0 \\ r_0 \end{pmatrix}, \quad x_{0_e} = \begin{pmatrix} x_0 \\ x'_0 \end{pmatrix}.$$

Mathématiquement, le processus hermitien de Lanczos, appliqué à (3.2) avec $r_0' = 0$ et $\lambda = 0$, est exactement la bidiagonalisation donnée par Golub et Kahan [59]. Cette bidiagonalisation construit, à la i-ième étape, deux bases orthogonales de deux sous-espaces de Krylov $K_{i+1}(AA^*, r_0)$ et $K_{i+1}(A^*A, A^*r_0)$. Donc, la convergence des méthodes qui projettent sur ces espaces est souvent inacceptablement lente. Par exemple, CGNR (qui peut être bénéfiquement mis en oeuvre par l'algorithme LSQR [89] en utilisant la bidiagonalisation de Golub-Kahan) et CGNE. Les deux méthodes USYMLQ et USYMQR [107] projettent le résidu sur la somme de deux espaces de Krylov. Les récurrences à trois termes utilisées dans ces deux méthodes construisent deux bases orthogonales $\{p_1, p_2, \ldots, p_n\}$ et $\{q_1, q_2, \ldots, q_n\}$ telles que

$$\begin{cases} \beta_{j+1} p_{j+1} = Aq_j - \alpha_j p_j - \gamma_j p_{j-1} \\ \gamma_{j+1} q_{j+1} = A^* p_j - \alpha_j q_j - \beta_j q_{j-1} \end{cases} \quad j = 1, 2, \ldots, n-1, \tag{3.3}$$

avec $q_0 = p_0 = 0$, $p_1 = r_0/\|r_0\|_n, q_1 = r_0'/\|r_0'\|_n$, $\beta_1 = \gamma_1 = 0$, $\alpha_j =< Aq_j, p_j >_n$, où $\beta_{j+1} > 0$ et $\gamma_{j+1} > 0$ sont choisis tels que p_{j+1} et q_{j+1} sont des vecteurs unitaires. De ces deux récurrences à trois termes, nous obtenons

$$p_{2i-1} \in K_i(AA^*, r_0) + K_{i-1}(AA^*, Ar_0'), \quad p_{2i} \in K_i(AA^*, r_0) + K_i(AA^*, Ar_0'), \tag{3.4}$$

$$q_{2i-1} \in K_{i-1}(A^*A, A^*r_0) + K_i(A^*A, r_0'), \quad q_{2i} \in K_i(A^*A, A^*r_0) + K_i(A^*A, r_0'), \tag{3.5}$$

$$< p_k, p_i >_n =< q_k, q_i >_n = 0 \quad \text{for } k < i, \tag{3.6}$$

$$< Aq_k, p_i >_n =< A^* p_k, q_i >_n = 0 \quad \text{for } k < i-1. \tag{3.7}$$

Cependant, il est intéressant de noter que ces deux récurrences à trois termes peuvent avoir un problème de DPZ. Cette description nous permet de remarquer que CGNR, CGNE, LSQR, USYMLQ et USYMQR ont une convergence lente lorsqu'elles sont combinées avec un préconditionnement comme dans [107]. Ceci a été observé dans les résultats numériques donnés dans [107]. C'est pourquoi nous proposons d'utiliser un préconditionnement à la matrice étendue $A_{e\lambda}$ pour les raisons suivantes

1. les valeurs propres de $A_{e\lambda}$ sont données par l'équation

$$\mu(A_{e\lambda}) = \lambda/2 \pm \sqrt{\sigma_i^2 + \lambda^2/4}, \tag{3.8}$$

où $\sigma_1 \leq \sigma_2 \leq \ldots \leq \sigma_n$ sont les valeurs singulières de A. Pour la démonstration de ce résultat, voir [108]. Dans le cas $\lambda = 0$, les valeurs propres de A_{e0} en valeur absolue sont les valeurs singulières de A,

2. le conditionnement de $A_{e\lambda}$ est donné par l'expression

$$cond(A_{e\lambda}) = \frac{\lambda/2 + \sqrt{\sigma_n^2 + \lambda^2/4}}{-\lambda/2 + \sqrt{\sigma_1^2 + \lambda^2/4}},$$

qui peut être réduite à

$$cond(A_{e\lambda}) = \frac{(\lambda/2 + \sqrt{\sigma_n^2 + \lambda^2/4})(\lambda/2 + \sqrt{\sigma_1^2 + \lambda^2/4})}{\sigma_1^2}.$$

A partir de cette dernière expression, nous remarquons que le conditionnement de la matrice $A_{e\lambda}$ augmente avec la valeur de λ et que nous avons $cond(A_{e0}) = cond(A)$ pour $\lambda = 0$. Par conséquent, nous recommandons à l'utilisateur d'attribuer à λ une valeur qui n'est pas trop grande, spécialement lorsque la matrice A est mal conditionnée. Nous allons voir que l'intérêt d'avoir $\lambda \neq 0$ est de pouvoir appliquer le préconditionnement ILU sans pivotage.

3.2.2 La méthode du résidu minimum (MINRES)

La méthode MINRES a été définie par Paige et Saunders dans [88]. Elle est mathématiquement équivalente à la méthode CR et elle est basée sur le processus hermitien de Lanczos [87, 88, 89].
L'intégralité de l'algorithme correspondant à cette méthode MINRES est

MINRES

1. **Initialisation** : Choisir $x_0 \in \mathbb{C}^n$ et poser $r_0 = b - Ax_0$, $\beta_1 = \|r_0\|_n$, $p_1 = r_0/\beta_1$, $p_0 = q_0 = q_{-1} = 0$, $f_0 = \rho_0 = \beta_1$, $\rho_{-1} = 1$, $\rho_0' = s_0 = s_{-1} = c_{-1} = 0$, $c_0 = -1$.

2. **Boucle** : Pour $i = 1, 2, \ldots$ jusqu'à convergence

   (a_1) (le processus hermitien de Lanczos)

   calculer $z_i = Ap_i - \beta_i p_{i-1}$,

   $\alpha_i = < Ap_i, p_i >_n$,

   $\beta_{i+1} p_{i+1} = z_i - \alpha_i p_i$,
   où β_{i+1} est choisi tel que $\|p_{i+1}\|_n = 1$,

   (a_2) (construire et appliquer la transformation orthogonale suivante)

   $d_i = (\alpha_i - \rho_{i-1}' c_{i-2}) s_{i-1}$,

   $h_i = \beta_i s_{i-2}$,

   $\rho_i' = -\beta_i c_{i-2} s_{i-1} - \alpha_i c_{i-1}$,

   $\rho_i = (\rho_i'^2 + \beta_{i+1}^2)^{1/2}$,

   $c_i = \rho_i'/\rho_i$,

   $s_i = \beta_{i+1}/\rho_i$,

   (a_3) (calcul des itérés x_i et q_i)

   $z_i' = p_i - h_i/\rho_{i-2} q_{i-2}$,

   $q_i = z_i' - d_i/\rho_{i-1} q_{i-1}$,

   $t_i = f_{i-1} c_i$,

   $f_i = f_{i-1} s_i$,

   $x_i = x_{i-1} + t_i/\rho_i q_i$.

Le critère d'arrêt est donné à la i-ième itération par l'estimation $|f_i|$ de la norme du vecteur résidu $r_i = b - Ax_i$. Dans l'algorithme MINRES, si nous supposons que $t_i = 0$, alors t_{i+1} ne peut être nul sauf si la solution de (3.1) est obtenue. Donc, théoriquement, MINRES n'a pas de problème de stagnation.
Pour les raisons précédentes, nous proposons d'appliquer la méthode MINRES à (3.2). La méthode qui en résulte sera appelée la méthode du résidu étendu minimum et désignée

en abréviation par MINERES(λ). De façon similaire, si nous appliquons la méthode CR à (3.2), nous obtenons une nouvelle méthode appelée ici la méthode du résidu étendu conjugué et désignée par CER(λ).
A première vue, nous avons l'impression que MINERES(0) est équivalente à USYMQR. Ceci n'est pas vrai puisque USYMQR construit deux bases orthogonales de $\mathbb{C}^n$ alors que MINERES(0) construit une base orthogonale de $\mathbb{C}^{2n}$. A la i-ième étape du processus hermitien de Lanczos dans l'algorithme MINERES(λ), la matrice $A_{e\lambda}$ est réduite à la matrice tridiagonale T_i=tridiag($\beta_i, \alpha_i, \beta_{i+1}$). A partir des valeurs propres de T_i nous pouvons obtenir une approximation des valeurs singulières de A lorsque i décroît. Cependant, il est important de noter qu'en arithmétique exacte, MINERES(λ) est équivalente à GMRES lorsque cette dernière est appliquée à (3.2).
Donnons maintenant quelques propriétés de la méthode MINERES(λ).

Théorème 3.1 *i/ Si $r_0' = 0$, alors les relations de récurrence (3.3) utilisées dans USYMQR peuvent être maintenues jusqu'à $j = 2n$, ainsi nous aurons la première bidiagonalisation de Golub et Kahan.*

ii/ Si $r_0' = 0$, alors LSQR $\Longleftrightarrow$ MINERES(0).

iii/ Si la matrice A est hermitienne et $r_0' = r_0$, alors USYMQR $\Longleftrightarrow$ MINERES(0) et, dans ce cas, MINERES(0) et USYMQR sont réduites à MINRES.

La notation $M_1 \Longleftrightarrow M_2$ signifie que M_1 est mathématiquement équivalente à M_2.

Preuve :

i/ Dans [107], il est clair que si nous choisissons $r_0' = 0$ dans l'algorithme USYMQR, alors USYMQR présente à la première itération une DPZ. Cependant, la récurrence (3.3) peut être maintenue si nous posons $\gamma_j = 0$ quand $q_j = 0$ et $\beta_j = 0$ lorsque $p_j = 0$. Dans ce cas, la formule (3.3) nous donne $p_{2j} = q_{2j+1} = 0$, ce qui prouve que $\alpha_j = 0$ dans (3.3) pour tout j. Par conséquent, si (3.3) est maintenue jusqu'à $j = 2n$, alors nous obtenons exactement la première bidiagonalisation de Golub et Kahan appelée Bidiag 1 dans [59].

ii/ Si, dans le processus hermitien de Lanczos appliqué à (3.2), nous choisissons $r_0' = 0$ et $\lambda = 0$, alors nous obtenons $\alpha_i = 0$ pour tout i. A partir de ce résultat, nous remarquons que Bidiag 1 dans [59] est aussi mathématiquement équivalente au processus hermitien de Lanczos appliqué à (3.2) avec $r_0' = 0$. En conséquence, il s'ensuit que LSQR $\Longleftrightarrow$ MINERES(0) quand $r_0' = 0$.

iii/ Si nous remplaçons r_0' par r_0 et A^* par A dans les équations (3.4), (3.5), alors la condition d'orthogonalité (3.6) implique que $p_j = q_j$ et $\gamma_j = \beta_j$ pour tout j. En conséquence, la formule (3.3) est la récurrence utilisée dans le processus hermitien de Lanczos, et donc nous avons USYMQR $\Longleftrightarrow$ MINERES(0).
Si la matrice A est hermitienne, $r_0' = r_0$ et

$$r_{k_e} = \begin{pmatrix} r_k' \\ r_k \end{pmatrix}, \quad x_{k_e} = \begin{pmatrix} x_k \\ x_k' \end{pmatrix}$$

sont les itérés de MINERES(0) à l'étape k, alors il est facile de voir par récurrence que $r_k' = r_k$ et $x_k' = x_k$ sont les itérés de MINRES. De même, nous avons que CER(0) est réduit à CR. ∎

De plus, à partir du théorème précédent, il est clair que MINERES(λ) peut être considérée comme une généralisation de MINRES et LSQR.

Remarque 3.1 Le système (3.2) avec $r_0' = 0$ et $\lambda = 0$ a déjà été étudié par Cullum et Greenbaum dans [42, 43]. Mais le résultat du théorème précédent n'a jamais été mentionné nulle part ; seulement, un résultat similaire sur l'équivalence LSQR $\Leftrightarrow$ MINERES(0) a aussi été prouvé par Sadok [104].

Parallélisation de MINERES(λ)

Nous savons que le processus d'Arnoldi utilise une relation de récurrence longue dont le nombre de termes augmente d'un terme à chaque itération. En revanche le processus de Lanczos utilise une relation à trois termes. C'est ceci qui donne l'avantage à la parallélisation du processus de Lanczos par rapport à celui d'Arnoldi. Donc, en ce qui concerne la parallélisation des processus utilisés par exemple dans GMRES et MINRES(λ), nous avons un avantage en faveur de MINRES(λ).

La parallélisation du produit de la matrice étendue $A_{e\lambda}$ par un vecteur, que nous utilisons dans MINERES(λ), se fait en stockant la matrice A et en effectuant le produit Ay puis A^*z.

3.2.3 Préconditionnement *ILU*

Posons $A = LU + E$, où L est une matrice triangulaire inférieure et U est une matrice triangulaire supérieure. Si L et U ont respectivement les mêmes structures que la partie inférieure et supérieure de la matrice A et si elles sont obtenues conformément

à la factorisation standard LU de A, alors cette factorisation est appelée la *factorisation incomplète* LU et désignée par $ILU(0)$ [101, 102]. Cette factorisation incomplète $ILU(0)$ a été généralisée en introduisant le concept du niveau de remplissage. Initialement, tout élément non nul de A a un niveau de remplissage égal à zéro. Le niveau de remplissage d'un élément est récursivement défini comme un plus la somme des niveaux des éléments de L et U à partir desquels il est obtenu dans le processus d'élimination de Gauss. Chaque remplissage, par un élément dont le niveau dépasse un certain entier k, est annulé. Ceci définit la factorisation incomplète $ILU(k)$ qui présente trois difficultés expliquées dans [102]. Par exemple elle ne peut pas distinguer les petites valeurs des grandes et donc ne peut pas contrôler l'erreur E. Afin de remédier à ces difficultés, Saad a proposé une factorisation incomplète LU avec un seuil de remplissage τ (The dual threshold incomplete LU factorization), que nous désignons par $ILUT(k,\tau)$ [102] où

- k est un entier tel que chaque ligne de L et chaque ligne de U aura au maximum k éléments ajoutés aux éléments non nuls qui correspondent à la structure de la matrice A,

- τ est une tolérance utilisée pour éliminer les éléments de L et U. Un élément z de L ou de U est éliminé si $|z| < \|ligne\|_n \tau$ où *ligne* est la ligne de A qui subit l'élimination de Gauss.

Comme $ILU(0)$, $ILU(k)$ et $ILUT(k,\tau)$ n'existent pas pour la matrice A_{e0}, nous utilisons, dans nos résultats numériques, la version $ILUT(k,\tau,\tau')$ obtenue à partir de la factorisation $ILUT(k,\tau)$ en permutant les colonnes de la matrice. Nous permutons la i-ième et la j-ième colonne de $A_{e\lambda}$ si $|A_{e\lambda ij}|\tau' > |A_{e\lambda ii}|$. Une autre version de la factorisation incomplète LU est donnée dans [91] où les auteurs introduisent un nouveau paramètre pour éliminer les éléments de L et U. Il s'ensuit l'impossibilité d'appliquer le préconditionnement ILU à MINERES(0) sans permuter les colonnes de la matrice A_{e0}. Ceci est un désavantage de MINERES(0). Afin de remédier à cette difficulté, nous avons introduit le paramètre λ et considéré MINERES(λ). La matrice $A_{e\lambda}$ associée à MINERES(λ) possède la factorisation standard LU sans aucune permutation

$$\begin{pmatrix} \lambda I & A^* \\ A & 0 \end{pmatrix} = \begin{pmatrix} I & 0 \\ \lambda^{-1}A & L_1 \end{pmatrix} \begin{pmatrix} \lambda I & A^* \\ 0 & -U_1 \end{pmatrix},$$

où L_1 est la matrice triangulaire inférieure et U_1 est la matrice triangulaire supérieure de la factorisation LU de $\lambda^{-1}AA^*$. Ainsi, il est clair que la factorisation LU de $A_{e\lambda}$ est très simple.

Remarque 3.2 Si $\lambda = 1$, alors nous obtenons la factorisation LU de AA^*. Par conséquent, la factorisation incomplète LU de AA^* peut être obtenue à partir de celle de A_{el}. Ceci est un nouveau résultat intéressant qui nous permet d'appliquer le préconditionnement ILU à CGNR et CGNE.

3.3 Résultats numériques

Ici, nous considérons le cas de matrices réelles et nous présentons les résultats pour quatre exemples. Pour nos tests, nous choisissons $x_0 = 0$, $x_0' = 0$ et $b' = b$. Le second membre de (3.1) est choisi pour que la solution exacte x soit égale à $(1, 2, 3, \ldots, n)^T$. Si la matrice A est symétrique, alors l'algorithme MINERES(0) avec le choix $b' = b$ devient exactement l'algorithme MINRES. Pour faire une comparaison entre MINERES(λ) et GMRES, nous utilisons la mise en oeuvre de la méthode GMRES comme elle a été décrite dans [100]. Nos tests ont été réalisés en utilisant le compilateur FORTRAN 77 sur un"Sun SparcStation10" avec la précision machine égale à $2.22\ 10^{-16}$. Dans toutes les figures de nos exemples, la courbe de la méthode GMRES est représentée par une ligne continue, celle du MINERES(0) par des tirets et celle de MINERES(λ) lorsque $\lambda \neq 0$ par des tirets pointillés.

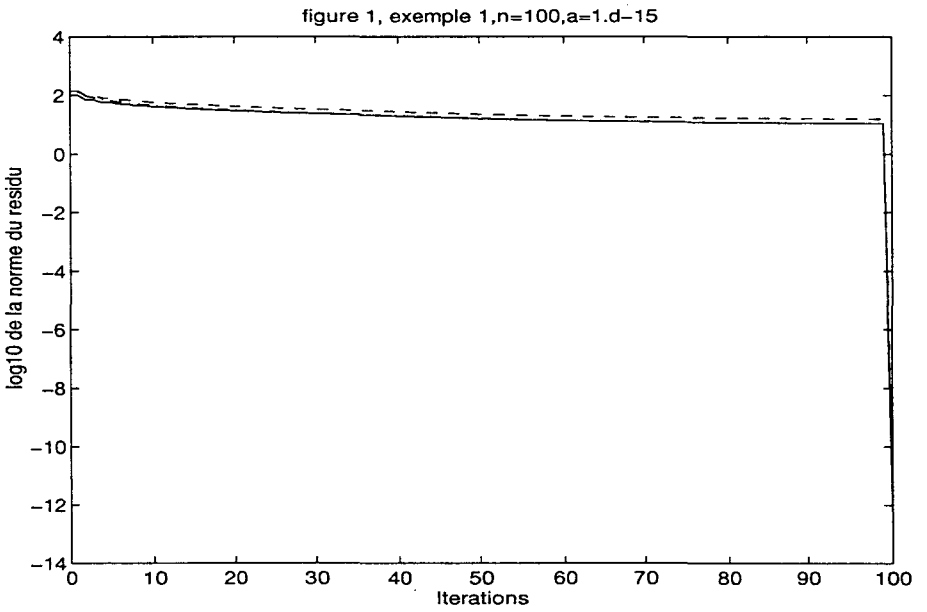
Exemple 1

Considérons la matrice $n \times n$ suivante

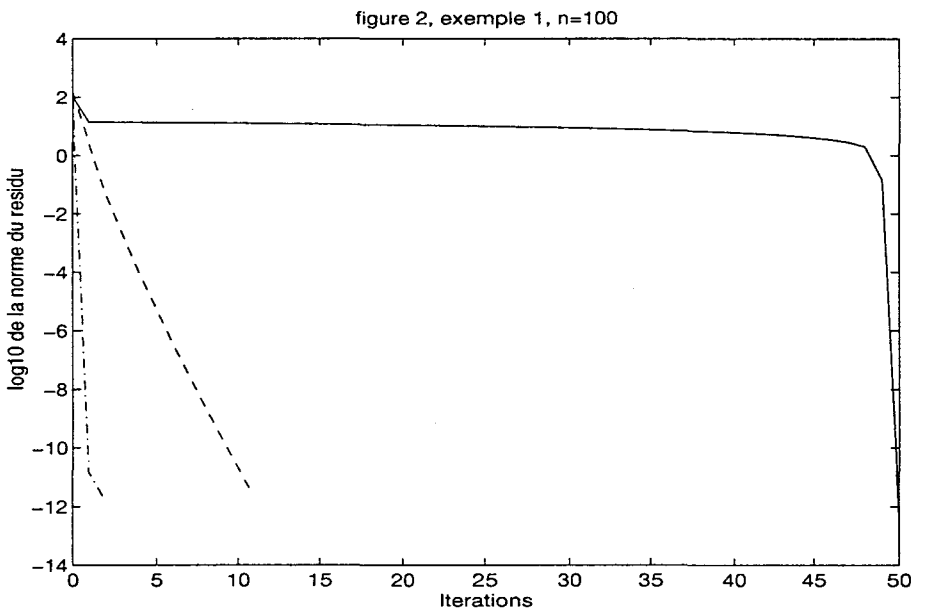
$$A = \begin{pmatrix} a & 1 & & & \\ -1 & a & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & a & 1 \\ & & & -1 & a \end{pmatrix}.$$

Cet exemple a été donné par Brown [31] pour illustrer la stagnation de l'algorithme GMRES. Si aucun préconditionnement n'est utilisé, alors pour tout n et a, aussi bien MINERES(λ) que GMRES ont une convergence similaire quand λ n'est pas trop grand. Mais, comme les tracés de GMRES, MINERES(0) et MINERES(1) l'indiquent, la perfor-

mance de ces méthodes est très réduite lorsque $a < 10^{-2}$.



Nous traçons dans la figure 2 la convergence des trois méthodes GMRES, MINERES(0) et MINERES(1) avec préconditionnement. Nous utilisons le préconditionnement $ILUT(k, \tau, \tau')$ à droite avec $k = 5$, $\tau = 10^{-1}$ et $\tau' = 10^{-1}$.



Pour tout n, nous obtenons des résultats similaires à ceux donnés dans la figure 2 lorsque $a \leq 10^{-4}$. Après quelques itérations seulement, nous remarquons que MINERES(1) et MINERES(0) ont une bonne convergence, tandis que GMRES stagne et ne montre aucun

signe de convergence. En fait, cet exemple est étudié ici pour mettre en valeur l'importance de la méthode MINERES(λ) quand elle est utilisée avec un préconditionnement convenable. La même courbe de convergence pour MINERES(1) est obtenue lorsque nous appliquons le préconditionnement $ILUT(k,\tau)$ à droite avec $k = 10$ et $\tau = 10^{-3}$. Ceci explique la meilleure performance de MINERES(1) puisque, lorsque nous permutons les colonnes de la matrice $A_{e\lambda}$, la matrice résultante n'est pas vraiment symétrique et donc, durant l'application du processus hermitien de Lanczos, l'orthogonalité peut être réellement perdue.

Exemple 2

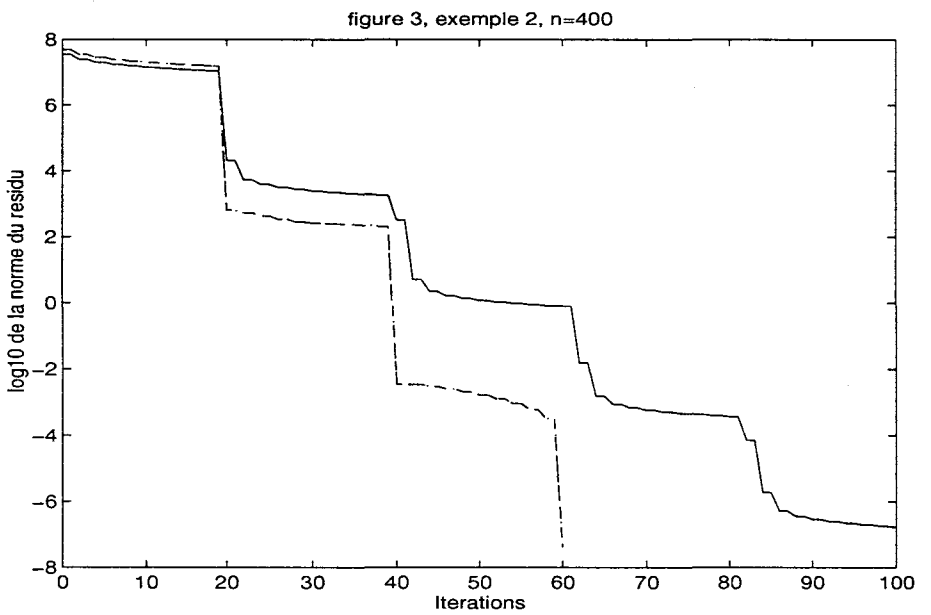
Nous considérons la matrice obtenue par discrétisation de l'équation elliptique aux dérivées partielles suivante

$$Lu = f \quad \text{sur} \quad [0,1] \times [0,1],$$

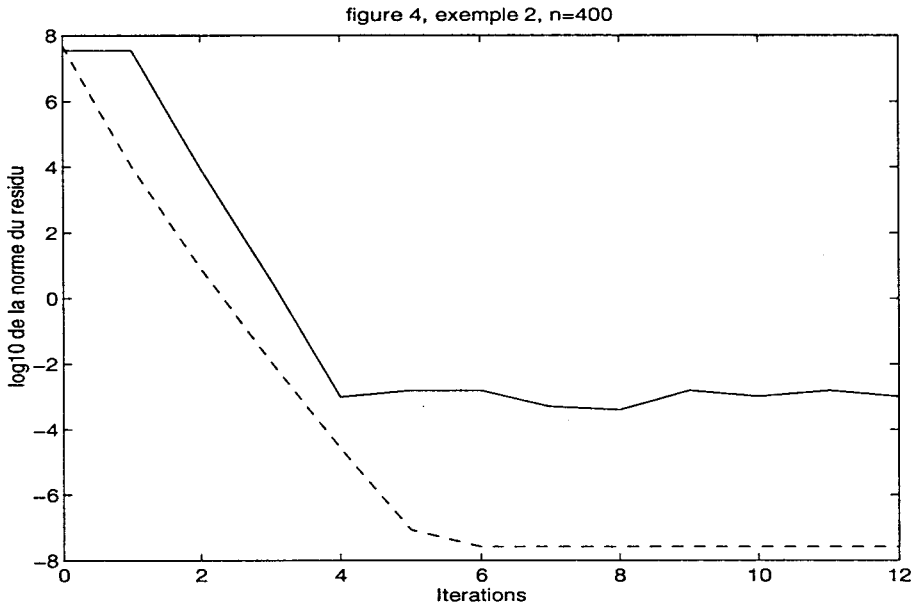
où

$$Lu = -\triangle u + \sigma \frac{\partial u}{\partial x},$$

avec les conditions de Dirichlet $u = 0$ aux bords, en utilisant un schéma à cinq points sur une grille uniforme 20×20 avec un pas $h = 1/21$. Ceci nous donne une matrice creuse non symétrique de dimension $n = 400$ avec 1920 éléments non nuls. Pour le paramètre σ, nous choisissons $\sigma = 10^6$. Si nous résolvons le système linéaire résultant de dimension 400 sans préconditionnement, alors nous obtenons les courbes de convergence suivantes pour MINERES(1), MINERES(0) et GMRES.



Comme les courbes le montrent, aussi bien MINERES(1) que MINERES(0) donnent des résultats similaires. Au début des tracés, la convergence des courbes de MINERES(1) et GMRES coïncident. Après 20 itérations, nous observons une différence entre elles en faveur de MINERES(1). Utilisons maintenant le préconditionnement à droite $ILUT(k, \tau)$ avec $k = 5$ comme niveau de remplissage et $\tau = 10^{-1}$ comme seuil de remplissage.



La figure 4 nous montre qu'après un nombre raisonnable d'itérations, MINERES(1) converge vers une bonne approximation de la solution x de (3.1) et que la courbe de GMRES présente une stagnation numérique.

Exemple 3

Nous appliquons MINERES(λ) et GMRES à une matrice non symétrique obtenue à partir de la discrétisation de l'équation aux dérivées partielles à trois dimensions

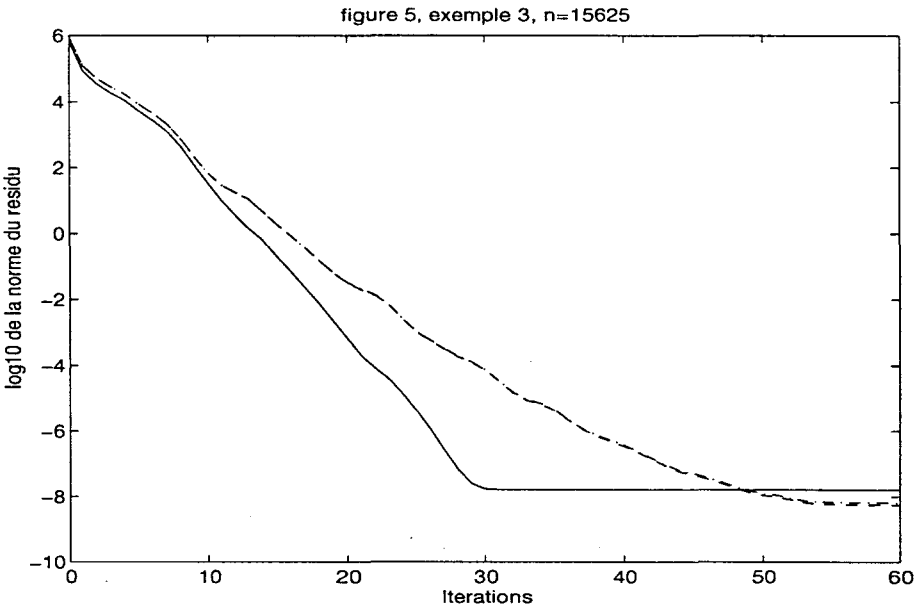
$$Lu = f \quad \text{sur} \quad [0,1] \times [0,1] \times [0,1],$$

où

$$Lu = -\triangle u + x\frac{\partial u}{\partial x} + y\frac{\partial u}{\partial y} + z\frac{\partial u}{\partial z} - u,$$

avec les conditions de Dirichlet aux bords $u = 0$. L'opérateur a été discrétisé en utilisant un schéma à sept points sur une grille uniforme $25 \times 25 \times 25$ avec un pas h égal à $1/26$. Ceci nous donne une matrice creuse non symétrique de dimension $n = 15625$, avec 105625 éléments non nuls. En utilisant le préconditionnement à droite $ILUT(k, \tau, \tau')$ avec

$k = 10$, $\tau = 10^{-3}$ et $\tau' = 10^{-1}$ nous obtenons les courbes de convergence de MINERES(0), MINERES(10^{-5}) et GMRES tracées dans la figure 5 suivante.



Clairement, les courbes de convergence de MINERES(0) et MINERES(10^{-5}) sont pratiquement les mêmes. Dans cet exemple, la méthode GMRES ne présente pas de stagnation avant la 30-ième itération, et sa convergence est meilleure que celle de MINERES(0). Ici, nous choisissons λ petit puisque nous savons que la matrice de cet exemple est mal conditionnée. Par exemple, si nous prenons $\lambda = 1$, alors nous observerons une stagnation numérique de MINERES(1).

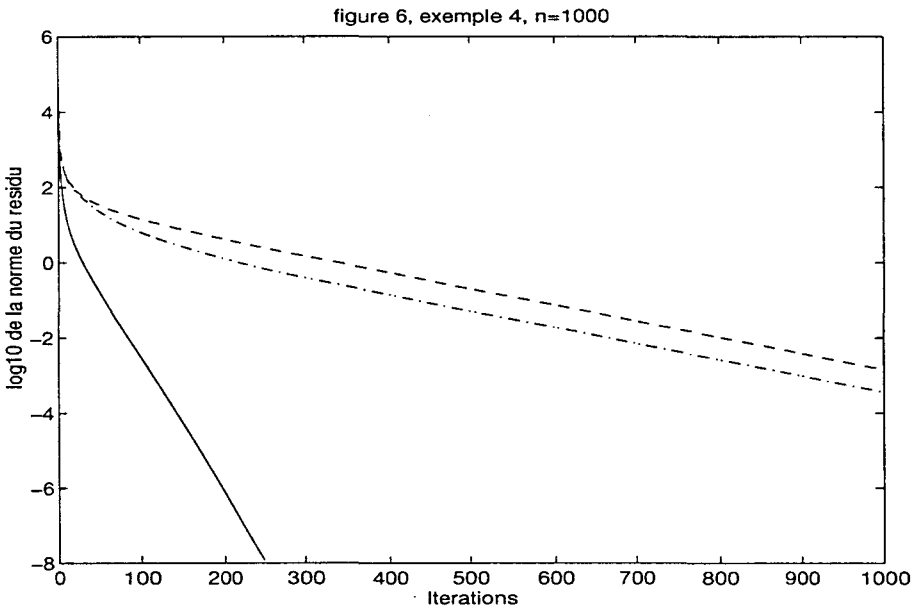
Exemple 4

Considérons la matrice carrée tridiagonale de dimension $n = 1000$ donnée dans [84].

$$A = \begin{pmatrix} 5.1 & 3 & & & \\ 2 & 5.1 & 3 & & \\ & \ddots & \ddots & \ddots & \\ & & 2 & 5.1 & 3 \\ & & & 2 & 5.1 \end{pmatrix}$$

Cet exemple est choisi ici pour montrer que, si aucun préconditionnement n'est utilisé, la convergence de MINERES(λ) peut être lente. Nous remarquons ceci dans la figure suivante où nous traçons les courbes correspondantes à MINERES(0), MINERES(1) et

GMRES.



Dans cet exemple, GMRES ne stagne pas et donne de bons résultats. En revanche MINERES(0) et MINERES(1) ont une convergence lente. Mais, si nous utilisons le préconditionnement à droite $ILUT(k,\tau,\tau')$ avec $k = 10$, $\tau = 10^{-1}$, $\tau' = 10^{-1}$ ou $ILUT(k,\tau)$ avec $k = 10$, $\tau = 10^{-1}$, aussi bien MINERES(λ)(avec λ pas trop grand) que GMRES nous donnent, à la première itération, une très bonne approximation de la solution exacte de (3.1).

3.4 Conclusion

Dans ce chapitre, nous avons proposé la méthode MINERES(λ) pour un système linéaire non hermitien. Théoriquement, MINERES(λ) évite le problème de stagnation que nous pouvons rencontrer dans les méthodes itératives de projection sur les sous-espaces de Krylov $K_i(A, r_0)$ pour $i = 1, 2, \ldots, n$. Les itérés de MINERES(λ) utilisent seulement des récurrences courtes et donc MINERES(λ) n'a pas besoin de stocker autant de vecteurs que GMRES. A partir de nos essais numériques, il est important de noter que nous remarquons souvent une faible performance de MINERES(λ), si aucun préconditionnement n'est utilisé. D'autre part, MINERES(λ) combinée avec un préconditionnement convenable nous donne une bonne performance avec une convergence rapide. Il est aussi important de noter que MINERES(λ) peut être considérée comme une généralisation de MINRES et LSQR.

Bibliographie

[1] W. ARNOLDI, *The principle of minimized iterations in the solution of the matrix eigenvalue problem*, Quart. Appl. Math., 9 (1951) 17-29.

[2] S. F. ASHBY, T. A. MANTEUFFEL, P. E. SAYLOR, *A taxonomy for conjugate gradient methods*, SIAM J. Numer. Anal., 27 (1990) 1542-1568.

[3] O. AXELSSON, *Conjugate gradient type methods for unsymmetric and inconsistent systems of linear equations*, Linear Alg. Appl., 29 (1980) 1-16.

[4] E. H. AYACHOUR, *Avoiding the look-ahead in the Lanczos method*, Pub. ANO-363, Université des Sciences et Technologies de Lille, (1996).

[5] Z. BAI, D. HU, L. REICHEL, *A Newton basis GMRES implementation*, IMA J. Numer. Anal., 14 (1994) 563-581.

[6] G. A. BAKER, *The Padé approximant method and some related generalizations*, In "The Padé Approximant in Theoretical Physics", G. A. Baker Jr. and J. L. Gammel eds., Academic Press, New York, (1970) 1-39.

[7] E. H. BAREISS, *Numerical solution of linear equations with Toeplitz and vector Toeplitz matrices*, Numer. Math., 13 (1969) 404-424.

[8] J. BOGNÁR, *Indefinite inner product spaces*, Springer, Berlin, (1974).

[9] C. BREZINSKI, *Computation of Padé approximants and continued fractions*, J. Comp. Appl. Math., 2 (1976) 113-123.

[10] C. BREZINSKI, *Padé-Type Approximation and General Orthogonal Polynomials*, Birkäuser Verlag, Basel, (1980).

[11] C. BREZINSKI, *Other manifestations of the Schur complement*, Linear Alg. Appl., 111 (1988) 231-247.

[12] C. BREZINSKI, M. REDIVO ZAGLIA, H. SADOK, *Avoiding breakdown and near-breakdown in Lanczos type algorithms*, Numer. Algorithms, 1 (1991) 261-284.

[13] C. BREZINSKI, *History of Continued Fractions and Padé Approximants*, Springer Verlag, Berlin, (1991).

[14] C. BREZINSKI, *CGM: a whole class of Lanczos-type solvers for linear systems*, Pub. ANO-253, Université des Sciences et Technologies de Lille, (1991).

[15] C. BREZINSKI, *Biorthogonality and its Applications to Numerical Analysis*, Marcel Dekker, New York, (1992).

[16] C. BREZINSKI, M. REDIVO ZAGLIA, H. SADOK, *A breakdown-free Lanczos type algorithm for solving linear systems*, Numer. Math., 63 (1992) 29-38.

[17] C. BREZINSKI, J. VAN ISEGHEM, *Padé Approximations*, In Handbook of Numerical Analysis, vol. III, P. G. Ciarlet and J. L. Lions eds. North-Holland, Amsterdam, (1992) 47-222.

[18] C. BREZINSKI, H. SADOK, *Lanczos-type algorithms for solving systems of linear equations*, Appl. Numer. Math., 11 (1993) 443-473.

[19] C. BREZINSKI, *Biorthogonality and conjugate gradient-type algorithms,* in *Contributions in Numerical Mathematics,* R.P. Agarval ed., World Scientific, Singapore, (1993) 55-70.

[20] C. BREZINSKI, *Treatment of near-breakdown in the CGS algorithm*, Numer. Algorithms, 7 (1994) 33-73.

[21] C. BREZINSKI, M. REDIVO ZAGLIA, *Extrapolation Methods - Theory and Practice*, North-Holland, Amsterdam, (1994).

[22] C. BREZINSKI, M. REDIVO ZAGLIA, *Breakdowns in the computation of orthogonal polynomials*, in Nonlinear Numerical Methods and Rational Approximation II, A. Cuyt ed., Kluwer, Dordrecht, (1994) 49-59.

[23] C. BREZINSKI, *The methods of Vorobyev and Lanczos*, Linear. Alg. Appl., 234 (1996) 21-41.

[24] C. BREZINSKI, M. REDIVO ZAGLIA, *Look-ahead in Bi-CGSTAB and other methods for linear systems*, BIT, 35 (1995) 169-201.

[25] C. BREZINSKI, *Projection methods for linear systems*, J. Comp. Appl. Math., 77 (1997) 35-51.

[26] C. BREZINSKI, M. REDIVO ZAGLIA, *A look-ahead strategy for the implementation of old and new extrapolation methods*, Numer. Algorithms, 11 (1996) 35-55.

[27] C. BREZINSKI, M. REDIVO ZAGLIA, H. SADOK, *Breakdowns in the implementation of the Lanczos method for solving linear systems*, Comp. & Maths. with Appls., 33 (1997) 31-44.

[28] C. BREZINSKI, M. REDIVO ZAGLIA, H. SADOK, *New look-ahead Lanczos-type algorithms for linear systems*, soumis.

[29] C. BREZINSKI, C. MUSSCHOOT, *Biorthogonal polynomials and the bordering method for linear systems*, Rend. Sem. Mat. Fis. Milano, 64 (1994) 85-98.

[30] H. BREZIS, *Analyse Fonctionnelle : Théorie et Applications*, Masson, Paris, (1983).

[31] P. N. BROWN, *A theoretical comparison of the Arnoldi and the GMRES algorithms*, SIAM J. Sci. Stat. Comput., 12 (1991) 58-78.

[32] A. BULTHEEL, *Recursive algorithms for non normal tables*, SIAM J. Appl. Math., 39 (1980) 106-118.

[33] A. BULTHEEL, *Division algorithms for continued fractions and the Padé table*, J. Comp. Appl. Math., 6 (1980) 259-266.

[34] L. CHAMBADAL, J. L. OVAERT, *Algèbre Linéaire et Algèbre Tensorielle*, Dunod, Paris, (1968).

[35] S. CHANDRASEKARAN, A. H. SAYED, *Stabilizing the generalized Schur algorithm*, SIAM J. Matrix Anal. Appl. 17 (1996) 950-983.

[36] G. CLAESSENS, L. WUYTACK, *On the computation of non normal Padé approximants*, J. Comp. Appl. Math., 5 (1979) 283-289.

[37] P. CONCUS, G. H. GOLUB, *A generalized conjugate gradient method for nonsymmetric systems of linear equations*, Lecture Notes in Economics and Mathematical systems 134, R. Glowinski and J. L. Lions, ed., Springer-verlag, New York, (1976) 56-65.

[38] F. CORDELLIER, *Deux algorithmes de calcul récursif des éléments d'une table de Padé non normale*, Exposé au colloque sur les approximants de Padé - Lille, 28-30 mars, (1978).

[39] F. CORDELLIER, *Démonstration algébrique de l'extension de l'identité de Wynn aux tables de Padé non normales*, In L. Wuytack, ed., Padé approximation and its Applications, LNM, Springer-Verlag, Berlin, 765 (1979) 36-60.

[40] F. CORDELLIER, *Interpolation Rationnelle et Autres Questions: Aspects Algorithmiques et Numériques*, Thèse d'Etat, Université des Sciences et Technologies de Lille, (1989).

[41] E. J. CRAIG, *The N-step iteration procedures*, J. Math. Physics, 34 (1955) 64-73.

[42] J. K. CULLUM, *Peaks, plateaus, numerical instabilities in Galerkin/minimal residual pair of methods for solving Ax=b*, Appl. Numer. Math., 19 (1995) 255-278.

[43] J. K. CULLUM, A. GREENBAUM, *Relations between Galerkin and norm-minimizing iterative methods for solving linear systems*, SIAM J. Matrix Anal. Appl., 17 (1996) 223-247.

[44] A. DRAUX, *Polynômes Orthogonaux Formels. Applications*, LNM vol. 974, Springer-Verlag, Berlin, (1983).

[45] A. DRAUX, P. VAN INGELANDT, *Polynômes Orthogonaux et Approximants de Padé. Logiciels*, Technip, Paris, (1987).

[46] J. DURBIN, *The fitting of time-series models*, Rev. Inst. Internat. Statist., 28 (1959) 229-249.

[47] F. E. EHLERS, W. H. WEATHERILL, E. L. YIP, *Development and application of algorithms for calculating the transonic flow about harmonically oscillating wings*, Tech. Report, The Boeing commercial Aireplane Company, Seattle, WA, October 1983.

[48] S. C. EISENSTAT, H. C. ELMAN, M. H. SCHULTZ, A. SHERMAN, *Solving approximations to the convection diffusion equation*, Proc. Fifth SPE Symposium on Reservoir Simulation, Denver, Colorado, (1979) 127-132.

[49] S. C. EISENSTAT, H. C. ELMAN, M. H. SCHULTZ, *Variational iterative methods for nonsymmetric systems of linear equations*, SIAM J. Numer. Anal., 20 (1983) 345-357.

[50] S. C. EISENSTAT, *A note on the generalized conjugate gradient method*, SIAM J. Numer. Anal., 20 (1983) 358-361.

[51] H. C. ELMAN, *Iterative Methods for Large, Sparse, Nonsymmetric Systems of Linear Equations*, Ph.D. thesis, Computer Science Department, Yale Univesity, New Haven, CT, (1982).

[52] V. FABER, T. A. MANTEUFFEL, *orthogonal error methods*, SIAM J. Numer. Anal., 24 (1987) 170-187.

[53] R. FLETCHER, *Conjugate Gradient methods for indefinite systems*, in Numerical Analysis, G. A. Watson ed. LNM 506, Springer Verlag, Berlin, (1976) 73-89.

[54] R. W. FREUND, N. M. NACHTIGAL, *QMR: A quasi-minimal residual method for non-hermitian linear systems*, Numer. Math., 60 (1991) 315-339.

[55] R. W. FREUND, *Conjugate gradient type-methods for linear systems for complex symmetric coefficient matrices*, SIAM J. Sci. Stat. Comp., 13 (1992) 425-448.

[56] R. W. FREUND, M. H. GUTKNECHT, N. M. NACHTIGAL, *An implementation of the look-ahead Lanczos algorithm for non-hermitian matrices*, SIAM J. Sci. Stat. Comput., 14 (1993) 137-158.

[57] R. W. FREUND, N. M. NACHTIGAL, *An implementation of the QMR method based on coupled two-term recurrences*, SIAM J. Sci. Comput., 15 (1994) 313-337.

[58] J. GILEWICZ, *Approximants de Padé*, Lecture Notes in Mathematics 667, Springer-Verlag, Heidelberg, (1978).

[59] G. H. GOLUB, W. KAHAN, *Calculating the singular values and pseudoinverse of a matrix*, SIAM J. Numer. Anal., 2 (1965) 205-224.

[60] G. H. GOLUB, C. F. VAN LOAN, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, 2nd ed., (1989).

[61] W. B. GRAGG, *Matrix interpretations and applications of the continued fraction algorithm*, Rocky Mountain J. Math., 4 (1974) 213-225.

[62] W. B. GRAGG, A. LINDQUIST, *On the partial realization problem*, Linear. Alg. Appl., 50 (1983) 277-319.

[63] P. R. GRAVES-MORRIS, *A "Look-around Lanczos" algorithm for solving a system of linear equations*, Numer. Algorithms, 15 (1997) 247-274.

[64] M. H. GUTKNECHT, *Variants of Bi-CGSTAB for matrices with complex spectrum*, SIAM J. Sci. Comput., 193 (1993) 1020-1033.

[65] M. H. GUTKNECHT, *A completed theory of the unsymmetric Lanczos process and related algorithms*, part I, SIAM J. Matrix Anal. Appl. 13 (1992) 594-639.

[66] M. H. GUTKNECHT, *Changing the norm in conjugate gradient type algorithms*, SIAM J. Numer. Anal., 30 (1993) 40-56.

[67] M. H. GUTKNECHT, M. HOCHBRUCK, *Look-ahead Levinson and Schur type recurrences in the Padé table*, Elec. Trans. Numer. Anal., 2 (1994) 104-129.

[68] M. H. GUTKNECHT, M. HOCHBRUCK, *Look-ahead Levinson and Schur algorithms for non-hermitian Toeplitz systems*, Numer. Math., 70 (1995) 181-227.

[69] P. HENRICI, *Applied and Computational Complex Analysis*, I. Wiley, New-York, (1974).

[70] M. R. HESTENES, E. STIEFEL, *Methods of conjugate gradients for solving linear systems*, J. Res. Nat. Bur. Standards., 49 (1952) 409-435.

[71] M. HEYOUNI, H. SADOK, *On variable smoothing procedure for Krylov subspaces methods*, Linear Alg. Appl., 268 (1998) 131-149.

[72] M. HEYOUNI, *Méthode de Hessenberg Généralisée et Applications*, Thèse, Université des Sciences et Technologies de Lille, (1996).

[73] A. S. HOUSEHOLDER, F. L. BAUER, *On certain methods for expanding the caracteristic polynomial*, Numer. Math., 1 (1959) 29-37.

[74] K. C. JEA, D. M. YOUNG, *On the simplification of generalized conjugate-gradient methods for nonsymmetrizable linear systems*, Linear Alg. Appl., 52 (1983) 399-417.

[75] P. JOLY, G. MEURANT, *Complex conjugate gradient methods*, Numer. Algorithms, 4 (1993) 379-406.

[76] C. LANCZOS, *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*, J. Res. Nat. Bur. stand., 45 (1950) 255-282.

[77] C. LANCZOS, *Solution of systems of linear equations by minimized iteration*, J. Res. Nat. Bur. stand., 49 (1952) 33-53.

[78] N. LEVINSON, *The Wiener rms (root-mean-square) error criterion in filter design and prediction*, J. Math. Phys., 25 (1947) 261-278.

[79] A. MAGNUS, *Certain continued fractions associated with the Padé table*, Math. Z., 78 (1962) 361-374.

[80] A. MAGNUS, *Expansion of power series into P-fractions*, Math. Z., 80 (1962) 209-216.

[81] R. J. MC ELIECE, J. B. SHEARER, *A property of Euclid's algorithm and an application to Padé approximation*, SIAM J. Appl. Math., 34 (1978) 611-616.

[82] R. MONTESSUS DE BALLORE, *Sur les fractions continues algébriques*, Bull. Soc. Math., France, 30 (1902) 28-36.

[83] N. M. NACHTIGAL, *A Look-Ahead Variant of the Lanczos Algorithm and its Application to the Quasi-Minimal Residual Method for Non-Hermitian Linear Systems*, Ph.D. thesis, Massachusetts Institute of Technology, (1991).

[84] N. M. NACHTIGAL, L. REICHEL, L. N. TREFETHEN, *A hybrid GMRES algorithm for nonsymmetric linear systems*, SIAM J. Matrix Anal. Appl., 13 (1992) 796-825.

[85] J. NUTTALL, S. R. SINGH, *Orthogonal polynomials and padé approximants associated with a system of arcs*, J. Approx. Theory, 21 (1977) 1-42.

[86] H. PADÉ, *Oeuvres*, Librairie Scientifique et Technique A. Blanchard, Paris, 1984.

[87] C. C. PAIGE, *Computational variants of the Lanczos method for the eigenproblem*, J. Inst. Math. Appl., 10 (1972) 373-381.

[88] C. C. PAIGE, M. A. SAUNDERS, *Solution of sparse indefinite systems of linear equations*, SIAM J. Numer. Anal., 12 (1975) 617-629.

[89] C. C. PAIGE, M. A. SAUNDERS, *LSQR: An algorithm for sparse linear equations and sparse least squares*, ACM Trans. Math. Softw., 8 (1982) 43-71.

[90] B. N. PARLETT, D. R. TAYLOR, Z. A. LIU, *A look-ahead Lanczos algorithm for unsymmetric matrices*, Math. Comp., 44 (1985) 105-124.

[91] S. PETITON, C. WEILL-DUFLOS, *Massively parallel preconditioners for the sparse conjugate gradient method*, In: Parallel Processing: COMPAR92-VAPP V, éd. by L. Bougé, Y. R. M. Cosnard, D. Trystram. Springer-Verlag, (1992) 373-378.

[92] M. A. PIÑAR, V. RAMIREZ, *Recursive inversion of Hankel matrices*, Monogr. Acad. Ciencias Zaragoza, 1 (1988) 119-128.

[93] M. PINDOR, *A simplified algorithm for calculating the Padé table derived from Baker and Longman schemes*, J. Comp. Appl. Math., 2 (1976) 255-257.

[94] J. RISSANEN, *Solution of linear equations with Hankel and Toeplitz matrices*, Numer. Math., 22 (1974) 361-366.

[95] H. RUTISHAUSER, *Der Quotienten-Differenzen Algorithmus*, Birkäuser Verlag, Basel, (1957).

[96] Y. SAAD, *Variations on Arnoldi's method for computing eigenelements of large unsymmetric matrices*, Linear Alg. Appl., 34 (1980) 269-295.

[97] Y. SAAD, *Krylov subspace methods for solving unsymmetric linear systems*, Math. Comp., 37 (1981) 105-126.

[98] Y. SAAD, *Practical use of some Krylov subspace methods for solving indefinite and nonsymmetric linear systems*, SIAM. J. Sci. Stat. Comp., 5 (1984) 203-228.

[99] Y. SAAD, M. H. SCHULTZ, *Conjugate gradient-like algorithms for solving nonsymmetric linear systems*, Math. Comp., 44 (1985) 417-424.

[100] Y. SAAD, M. H. SCHULTZ, *GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Stat. Comput., 7 (1986) 856-869.

[101] Y. SAAD, *Krylov subspace methods on supercomputers*, SIAM J. Sci. Stat. Comput., 10 (1989) 1200-1232.

[102] Y. SAAD, *ILUT: A dual threshold incomplete ILU factorization*, Linear Alg. Appl., 1 (1994) 387-402.

[103] Y. SAAD, K. WU, *DQGMRES: A Direct Quasi-Minimal Residual algorithm based on Incomplete Orthogonalization*, to appear.

[104] H. SADOK, Private communication.

[105] H. SADOK, *CMRH: A new method for solving nonsymmetric linear systems based on the Hessenberg reduction algorithm*, soumis.

[106] E. B. SAFF, *Aspects of Contemporary Complex Analysis*, D. A. Brannan and J. G. Clunie eds., Academic Press, New York, (1980).

[107] M. A. SAUNDERS, H. D. SIMON, E. L. YIP, *Two conjugate-gradient-type methods for unsymmetric linear equations*, SIAM J. Numer. Anal., 25 (1988) 927-940.

[108] M. A. SAUNDERS, *Solution of sparse rectangular systems using LSQR and CRAIG*, BIT, 35 (1995) 588-604.

[109] W. SCHARLAU, *Quadratic and hermitian forms*, vol. 270, Springer-Verlag, Berlin, (1985).

[110] W. SCHÖNAUER, *Scientific Computing on Vectors Computers*, North-Holland, Amsterdam, (1987).

[111] I. SCHUR, *Ueber Potenzreihen, die im Innern des Einheitskreises beschränkt sind*, I, J. Reine Angew. Math., 147 (1917) 205-232.

[112] P. SONNEVELD, *CGS: a fast Lanczos-type solver for nonsymmetric linear systems*, SIAM J. Sci. Stat. Comp., 10 (1989) 36-52.

[113] H. STAHL, *Divergence of diagonal Padé approximants and the asymptotic behavior of orthogonal polynomials associated with nonpositive measures*, Constr. Approx., 1 (1985) 249-270.

[114] H. STAHL, *Orthogonal polynomials with complex-valued weight function*, Constr. Approx., 2 (1986) I: 225-240 II: 241-251.

[115] G. W. STRUBLE, *Orthogonal polynomials: Variable-signed weight functions*, Numer. Math., 5 (1963) 88-94.

[116] H. A. VAN DER VORST, *Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems*, SIAM J. Sci. Stat. Comp., 13 (1992) 631-644.

[117] J. VAN ISEGHEM, *Vector Padé Approximants*, in *Numerical Mathematics and Applications*, R.Vichnevestky and J.Vignes ed., North Holland, Amsterdam, 1985, pp. 73-77.

[118] P. K. W. VINSOME, *ORTHOMIN: an iterative method for solving sparse sets of simultaneous linear equations*, Proc. Fourth SPE Symposium on Reservoir Simulation, Los Angeles, (1976) 149-160.

[119] H. VAN ROSSUM, *Contiguous orthogonal systems*, Koninkl. Nederl. Akad. Weten., ser. A, 63 (1960) 323-332.

[120] P. J. S. WATSON, *Algorithms for differentiation and integration*, In "Padé Approximants and their Applications", P. R. Graves-Morris ed., Academic Press, New York, (1973) 93-98.

[121] R. WEISS, *Convergence Behavior of Generalized Conjugate Gradient methods*, Ph.D. thesis, University of Karlsruhe, (1990).

[122] R. WEISS, W. SCHÖNAUER, *Accelerating generalized conjugate gradient methods by smoothing*, In Iterative Methods in Linear Algebra, R. Beauwens and P. de Groen eds., North-Holland, (1992) 283-292.

[123] R. WEISS, *Relations between Smoothing and QMR*, Internal report 53/94, University of Karlsruhe, Computing center (1994).

[124] J. H. WILKINSON, *The Algebraic Eigenvalue Problem*, Clarendon Press, Oxford, (1965).

[125] P. WYNN, *The rational approximation of functions which are formally defined by a power series expansion*, Math. Comp., 14 (1960) 147-186.

[126] P. WYNN, *Upon systems of recursions which obtain among the quotients of Padé table*, Numer. Math., 8 (1966) 264-269.

[127] D. M. YOUNG, K. C. JEA, *Generalized conjugate-gradient acceleration for non-symmetrizable iterative methods*, Linear Alg. Appl., 34 (1980) 159-194.

[128] M. ZIEGLER, *Generalized biorthogonal bases and tridiagonalisation of matrices*, Universität Tübingen, Biomathematik, Germany, Report Nr. 22, (1995).

[129] M. ZIEGLER, *Generalized biorthogonal bases and tridiagonalisation of matrices*, Numer. Math., 77 (1997) 407-421.