Preconditioning of linear systems for massively parallel computers (CM-2)

Olivier Perlot

THÈSE DE DOCTORAT DE L'UNIVERSITÉ PIERRE ET MARIE CURIE -PARIS VI-

SPÉCIALITÉ

MATHÉMATIQUES

PRÉSENTÉE PAR

Olivier PERLOT

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ PIERRE ET MARIE CURIE

Sujet de la thèse :

PRÉCONDITIONNEMENTS DE SYSTÈMES LINÉAIRES POUR CALCULATEURS MASSIVEMENT PARALLÈLES (CM-2)

Soutenue le 10 Juillet 1995 devant le jury composé de :

Monsieur	**O.PIRONNEAU**	
Monsieur	**P.LECA**	Rapporteur
Monsieur	**S.PETITION**	Rapporteur
Monsieur	**P.JOLY**	
Monsieur	**G.MEURANT**	

Remerciements

J'exprime toute ma gratitude à O. Pironneau d'avoir dirigé cette thèse.

Je tiens à remercier Messieurs P. Leca et S. Petiton de leur présence dans le jury et d'avoir accepté avec beaucoup de gentillesse la tâche fastidieuse de rapporter sur cette thèse. Je les remercie également pour tout l'intérêt qu'ils ont su apporter à mes travaux.

Je suis honoré de la présence dans le jury de G. Meurant, initiateur de nombreuses méthodes utilisées dans cette thèse. Je le remercie d'avoir orienté mon travail, tout en me laissant une grande liberté.

Ma plus profonde reconnaissance va à P. Joly, qui a su stimuler au moment opportun mes convictions et suivre avec soins mes idées, et au contact duquel j'ai beaucoup appris. Je le remercie également pour les multiples corrections qu'il a apporté à toutes mes tentatives de rédaction.

Je remercie également le SEH, en particulier F. Robin du CEA, de m'avoir procurer le temps nécessaire à la mise au point des nombreux algorithmes présents dans cette thèse. J'ai ainsi pu me rendre compte de la disponibilité de tous les membres du SEH ainsi que celle des ingénieurs de TMC.

Toute cette lumière n'aurait pu apparaitre sans l'étincelle produite par J. C. Lor, qui m'a initié aux joies des méthodes numériques. J'ai notamment pu apprécié sa disponibilité et son enthousiasme pour suivre mes travaux.

Je remercie finalement tous les membres du laboratoire d'analyse numérique pour leur présence quotidienne tout au long de cette thèse, en particulier A. Perronnet, M. Legendre, D. Boullic, M. Legras, L. Ruprecht, C. David, P. Maroni, C. Doursat, X. Juvigny, O. Lebigre, N. Bouhamou, D. Errate...

Que Carole, mon père, mes beaux-parents, Carl... recoivent toute ma gratitude et ma reconnaissance. Sans eux je n'aurais pas été capable de faire face aux difficultés rencontrées au cours de cette thèse.

Enfin j'exprime toute ma sympathie à mes amis qui m'ont énormément soutenu au cours de mes études, Manu, Christophe, Dominique...

Je remercie la Ville de Reims pour l'effort qu'elle a fourni pour aider l'achèvement de cette thèse.

SOMMAIRE

INTRODUCTION

Durant les vingt dernières années, l'introduction des supercalculateurs vectoriels puis des supercalculateurs parallèles a donné une nouvelle dimension au calcul scientifique. Des problèmes, qui étaient impossibles à simuler numériquement, peuvent maintenant être résolus efficacement. Mais, bien que ces machines aient de très grandes capacités de calculs (plusieurs centaines de millions d'opérations à la seconde), elles sont souvent sous-utilisées car les algorithmes de résolution ne sont pas exprimés de manière vectorielle ou ne contiennent pas assez de parallélisme. Ainsi, de nombreux travaux restent à réaliser et actuellement beaucoup d'analystes numériciens recherchent des algorithmes parallèles ou effectuent des modifications sur des méthodes déjà existantes pour une utilisation optimale des ordinateurs parallèles.

Nous nous sommes intéressés dans cette thèse à la résolution de systèmes linéaires provenant de la discrétisation par différences finies d'équations aux dérivées partielles elliptiques sur ordinateurs à architecture massivement parallèle à parallélisme sur les données (data parallel). Le modèle d'une telle machine sera pour nous la Connection Machine (CM-2). Dans ce cas, la méthode du gradient conjugué est, comme on le verra par la suite dans les expériences numériques, relativement bien adaptée à la CM-2 (ou réciproquement). Introduite en 1952 par Hestenes et Stiefel [HeSt52], cette méthode permet la résolution de systèmes linéaires $Ax = b$ dont la matrice A est symétrique définie positive. Cependant, le G.C n'est pas efficace si les valeurs propres de la matrice ont des ordres de grandeur très différents. Par conséquent, nous avons utilisé la généralisation de cette méthode popularisée par Concus, Golub et O'Leary [CoGO76] : le gradient conjugué préconditionné.

Le problème crucial, qui se pose pour l'efficacité de cette méthode, est celui du choix du préconditionnement. Pour le problème étudié, nous possédons aujourd'hui des méthodes extrêmement efficaces : la décomposition de Cholesky incomplète par points ou par blocs, voir par exemple [CoGM85]. De plus, ces techniques ont été adaptées à l'utilisation de machines vectorielles [Meur89a], [Meur89b]. Malheureusement, elles sont peu adaptées à une architecture massivement parallèle comme la CM-2, voir notamment [Weil94]. En effet, elles comportent toutes à des degrés divers des récurrences par points ou par

blocs, qui créent des dépendances de données qui font que le degré de parallélisme disponible est assez faible. Sur la CM-2, il faut donc se tourner vers d'autres types de préconditionnements. Toutefois il faut être conscient du fait que, puisque nous résolvons des problèmes elliptiques où la solution en un point dépend des autres points du domaine, il y a en quelque sorte antinomie entre parallélisme et efficacité. La solution "idéale" du point du vue du parallélisme est le préconditionnement Diagonal, qui n'est certes pas la plus rentable au niveau du gain en nombre d'itérations de la méthode. Des études ont déjà été réalisées sur la recherche de préconditionnements adaptés à la CM-2, voir par exemple [ChKT89], [Tong89], [PeWe92] et [Weil94]. Hélas, les résultats obtenus n'ont pas considérablement diminuer les temps de résolution du G.C.P préconditionné par le préconditionnement Diagonal sur la CM-2. Dans cette thèse nous allons montré que l'on peut trouver un préconditionnement tel que le G.C.P soit beaucoup plus efficace (au point de vue temps de résolution mais également vitesse de calcul) qu'avec le préconditionnement Diagonal sur la CM-2. D'autre part, le G.C préconditionné par la diagonale a déjà été comparé à d'autres méthodes de résolution de systèmes linéaires sur la CM-2, notamment par Mac Bryan [McBr88] et [McBr89] avec les méthodes multigrilles. Il apparaît que pour obtenir de bons résultats avec ce type de méthodes il faut utiliser des grilles assez fines. Ainsi, les performances atteintes, sur la CM-2, par le G.C sont supérieures à celles des méthodes multigrilles grâce à son plus grand degré de parallélisme.

Le modèle de progammation (data parallel) de la CM-2 étant très restrictif, nous avons dégagé trois types de préconditionnements. La première partie de cette thèse expose donc des préconditionnements adaptées à la CM-2, tout d'abord le préconditionnement référence au niveau du parallélisme : le préconditionnement Diagonal, ensuite des préconditionnements basés sur des renumérotations Rouge-Noir de préconditionnements déjà connus, [ChKT89]. Enfin une famille importante de préconditionnements adaptés à la CM-2 est celle des préconditionnements polynômiaux, déjà étudiés par de nombreux auteurs particulièrement [Adam82], [Adam85], [Ashb87], [Ashb91], [AsMO92], [CiMP93], [ChKT89], [DuGR79], [Freu89], [FrFi92], [HoVB90], [JoMP83], [Meur89b], [PeWe92], [Saad85], [Saad87], [Tong89] et [Weil94]. Le

principe fondamental des préconditionnements polynômiaux est la recherche d'une approximation de l'inverse d'une matrice A par un polynôme de degré k en cette matrice $P_k(A)$. Cette famille de préconditionnements prendra une part très importante tout au long de cette thèse, notamment les schémas d'évaluation, qui permettent d'éviter toute forme de remplissage de la matrice $P_k(A)$ en n'effectuant que des produits matrice vecteur. En effet, au lieu d'assembler la matrice A^2, qui comporte treize diagonales si la matrice initiale est pentadiagonale, nous pouvons décomposer le produit $z = A^2x$ en deux étapes : $y = Ax$ puis $z = Ay$. Ainsi si le produit matrice vecteur est parallélisable, ou bien adapté à l'architecture de la machine, le G.C préconditionné par un polynôme est facilement parallélisable. D'autre part, afin de comparer leurs taux de convergence avec des préconditionnements pas nécessairement adaptés à l'architecture massivement parallèle de la CM-2 mais possédant des caractéristiques communes, nous avons étudié des approximations de l'inverse exact, des préconditionnements à base de factorisations incomplètes par blocs [CoGM85] et des préconditionnements basés sur des compléments de Schur. Et pour vérifier le bon taux de convergence des préconditionnements massivement parallèles par rapport aux autres références, nous les avons tous étudiés sur une machine séquentielle. Puis, après avoir éliminé les préconditionnements peu efficaces (en tout cas moins efficaces que le préconditionnement diagonal), nous avons utilisé ces méthodes sur la CM-2. Devant les résultats encourageants obtenus par les préconditionnements polynômiaux, notamment pour les degrés assez élevés, nous avons décidé d'introduire des schémas d'évaluation beaucoup plus stables [AsMO92] et [CiMP93] que le simple schéma de Hörner. En effet, si le degré du polynôme est élevé ou si la précision de calcul est insuffisante, voire même si la matrice est mal conditionnée, ce schéma peut conduire à des problèmes, voir [AsMO92] et [Saad85] mais également [Gaut86], [Gaut90] et [StBu80] pour l'instabilité du schéma de Hörner simple. Or, sur la CM-2, le produit matrice vecteur s'est révélé très efficace, d'où une augmentation du taux de parallélisme et aussi de la vitesse de calcul avec le degré du polynôme utilisé. Il s'est donc avéré intéressant d'utiliser des polynômes de degré plus important que sur ordinateur séquentiel ou vectoriel [AsMO92], [ChKT89], [JoMP83], [Meur89b], [PeWe92], [Saad85] et [Weil94].

Un des principaux défauts des préconditionnements polynômiaux est qu'ils nécessitent une estimation a priori des valeurs propres (ou au moins la plus petite et la plus grande de la matrice du problème). Reprenant l'idée de Saad [Saad85], nous avons pris comme estimation de la plus petite valeur propre la valeur zéro. Puis en normalisant symétriquement la matrice par sa diagonale nous avons obtenu une matrice à diagonale unitaire et dominante. Ainsi avec le Théorème de Gershgörin, nous avons estimé la plus grande valeur propre de ce nouveau système par la valeur 2. Finalement, nous avons énoncé grâce à ces deux estimations un préconditionnement polynômial efficace indépendant des valeurs propres de la matrice.

Cette thèse se poursuit par l'étude d'un "double" préconditionnement. Il consiste à appliquer d'abord un premier préconditionnement (soit un polynôme de très faible degré soit une factorisation de Cholesky incomplète par points ou par blocs) destiné à obtenir une distribution des valeurs propres mieux adaptée aux polynômes utilisés dans la résolution effective. Puis dans un second temps, il suffit de résoudre ce nouveau système linéaire à l'aide de la méthode du gradient conjugué préconditionné par un préconditionnement polynômial. Le but de ce "double" préconditionnement est, bien entendu, d'atteindre un nombre d'itérations encore plus faible.

Ensuite nous complétons les résultats sur les schémas stables par une étude sur l'influence du produit matrice vecteur sur l'efficacité de ces schémas. En effet les schémas stables sont très efficaces grâce à la régularité du problème (matrices pentadiagonales) qui permet l'écriture d'un produit matrice vecteur très rapide. Ainsi, si on perturbe un peu la numérotation du maillage ou si on utilise un produit matrice vecteur moins efficace on constate une certaine dégradation des performances.

Enfin, les derniers travaux effectués proposent d'améliorer l'efficacité du G.C.P dans les problèmes d'évolution de la façon suivante. D'abord nous stockons quelques directions de descente lors de la résolution d'un système $Ax = f(t)$ avec le G.C.P. Ensuite, pour la résolution de $Ax = f(t + \Delta t)$ avant de lancer le G.C.P nous utilisons ces directions stockées afin d'approximer à moindre coût la solution et donc de gagner

quelques itérations dans le G.C.P.

Cette thèse comporte dix chapitres. Les deux premiers rappellent quelques notions sur les équations aux dérivées partielles elliptiques résolues et sur le gradient conjugué préconditionné, puis décrivent les supports informatiques. Le chapitre III expose les divers préconditionnements utilisés. Une étude numérique sur machine séquentielle puis parallèle constitue les deux chapitres suivants. Le chapitre VI introduit des algorithmes d'évaluation stables pour les préconditionnements polynômiaux. Dans le septième chapitre nous obtenons un préconditionnement polynômial indépendant des valeurs propres de la matrice utilisée. Le chapitre VIII propose un "double" préconditionnement. L'étude du chapitre IX sur l'influence du produit matrice vecteur dans l'efficacité des schémas stables complète le chapitre VI. Finalement, le chapitre X étudie l'intérêt du G.C.P dans un problème d'évolution.

En conclusion, nous proposons un bilan de nos travaux puis nous décrivons quelques perspectives.

CHAPITRE I

PAPPELS SUR LES ÉQUATIONS AUX DÉRIVÉES PARTIELLES ELLIPTIQUES

I.1 Présentation

La première partie de ce chapitre propose quelques rappels sur la théorie des équations aux dérivées partielles elliptiques linéaires. Dans un second temps, nous nous intéresserons aux problèmes résolus dans cette thèse et nous regarderons plus particulièrement leur discrétisation par la méthode des différences finies avec le classique schéma à cinq points.

Dans ce chapitre, nous reprendrons les notations définies par Raviart et Thomas dans [RaTh83].

I.2 Approche variationnelle

Afin de résoudre le problème elliptique, nous allons présenter le cadre abstrait qui se révèle bien adapté à ce type de problèmes. On se munit

(i) d'un espace de Hilbert V de norme $\| \ . \ \|$ sur R.

(ii) d'une forme bilinéaire $a(\ . \ , \ . \ )$ continue sur $V \times V$.

(iii) d'une forme linéaire L continue sur V.

Alors le problème $(\mathcal{P})$ variationnel considéré est :
trouver $u \in V$ tel que :

$$\forall v \in V, a(u, v) = L(v).$$

Donnons maintenant une condition suffisante sur la forme bilinéaire a afin d'obtenir une solution unique à ce problème $\mathcal{P}$.

Définition I.1 *La forme bilinéaire $a(\ .\ ,\ .\ )$ est dite V-elliptique s'il existe une constante réelle α strictement positive telle que :*

$$\forall v \in V, a(v,v) \geq \alpha \| v \|^2.$$

Nous pouvons alors utiliser le Lemme de Lax-Milgram

Lemme I.1 *Si $a(\ .\ ,\ .\ )$ est V-elliptique alors le problème $(\mathcal{P})$ admet une solution unique $u \in V$. De plus, l'application linéaire qui à L associe u est continue de V' dans V.*

Preuve I.1 *voir par exemple[RaTh83].*

Définition I.2 *La forme bilinéaire $a(\ .\ ,\ .\ )$ est dite symétrique si*

$$\forall v \in V, a(u,v) = a(v,u)$$

I.3 Problèmes aux limites elliptiques d'ordre 2

Soit Ω un ouvert borné de R^2 (les résultats restent vrais dans R^3) et $\partial\Omega$ sa frontière Lipschitz continue (ce qui inclus le carré unité).

$$\text{Posons } V = H_0^1(\Omega) \text{ et } \| v \| = \| \overrightarrow{grad}\ v \|_{0,\Omega}$$

En notant $\| \ . \ \|_{0,\Omega}$ la norme usuelle de $L^2(\Omega)$ dans R, $H^1(\Omega)$ étant bien entendu l'espace des fonctions de $L^2(\Omega)$ dont le gradient est dans $L^2(\Omega)^2$, puis $H_0^1(\Omega)$ la fermeture de $\mathcal{D}(\Omega)$, l'espace des fonctions de $\mathcal{C}^\infty(\Omega)$ à support compact, dans $H^1(\Omega)$ et finalement $H^{-1}(\Omega)$ est le dual de $H_0^1(\Omega)$ dans $\mathcal{D}'(\Omega)$, l'espace des distributions définies sur Ω.

On se donne des fonctions $a_{i\,j}, 1 \leq i,j \leq 2$ de l'espace $\mathrm{L}^\infty(\Omega)$ et on pose :

$$\mathcal{A} = \begin{pmatrix} a_{1\,1} & a_{1\,2} \\ a_{2\,1} & a_{2\,2} \end{pmatrix}$$

De plus, on suppose :

$$\exists \alpha > 0, \forall x \in \Omega, \forall v \in \mathrm{R}^2, (\mathcal{A}(x)v, v)_2 \geq \alpha \| v \|_2^2 \qquad \text{(I.1)}$$

Nota $(\, . \, , \, . \,)_2$ est le produit scalaire usuel sur R^2 et $\| \, . \, \|_2$ sa norme induite.

soit $f \in H^{-1}(\Omega)$, nous étudions le problème de Dirichlet $(\mathcal{D})$ suivant

$$(\mathcal{D}) \quad \begin{cases} -div(\mathcal{A}\, \overrightarrow{grad}\ u) = f \text{ dans } \Omega \\ u = 0 \text{ sur } \partial\Omega \end{cases}$$

Nous raccordons cette formulation faible à l'approche variationnelle du paragraphe précédent en posant :

$$a_D(u,v) = \int_\Omega \left(\sum_{i,j=1}^{2} a_{ij} \frac{\partial u}{\partial x_i} \frac{\partial v}{\partial x_j} \right) d\omega$$

$$\text{et } L_D(v) = < g, v >$$

avec $< \, . \, , \, . \, >$ le crochet de dualité entre $H^{-1}(\Omega)$ et $H_0^1(\Omega)$

Alors nous avons :

Théorème I.1 *Le problème $(\mathcal{D})$ est équivalent au problème suivant :*

$$\textit{Trouver } u \in V, \forall v \in V, a_D(u,v) = L_D(v)$$

Preuve I.2 *voir [GiRa86].*

Nous pouvons ensuite appliquer le Lemme de Lax-Milgram, car $a_D(\, . \, , \, . \,)$ est clairement bilinéaire continue sur $V \times V$ et V-elliptique d'après (I.1) et (I.2), et de plus $L_D(v)$ est linéaire sur V. Donc le problème $(\mathcal{D})$ admet une unique solution dans V.

I.4 Approximation par différences finies

I.4.1 Problèmes résolus

Nous allons tout d'abord préciser les types de problèmes que nous avons effectivement résolus. Le domaine étudié est le carré unité du plan $[0,1] \times [0,1]$ (on pourrait facilement généraliser ensuite à des rectangles quelconques). L'opérateur $\mathcal{A}$ est diagonal, et il s'écrit :

$$\mathcal{A} = \begin{pmatrix} a_{1\,1} & 0 \\ 0 & a_{2\,2} \end{pmatrix}. \tag{I.2}$$

Donc la forme bilinéaire $a(\ .\ ,\ .\ )$ est symétrique, par commodité d'utilisation nous noterons dans ce chapitre $a_{1\,1} = c$ et $a_{2\,2} = d$.

Lors de la mise en oeuvre informatique, nous avons résolu effectivement deux problèmes. Le premier problème est le problème de Poisson classique. Le second problème semble a priori plus difficile à résoudre numériquement car il présente des sauts de coefficients importants.

Problème 1

$$\begin{cases} c \equiv 1 \text{ dans } [0,1] \times [0,1], \\ d \equiv 1 \text{ dans } [0,1] \times [0,1], \end{cases}$$

Problème 2

$$\begin{cases} c(x,y) = \begin{cases} 1000, \text{ si } x \in]\frac{1}{4}, \frac{3}{4}[\\ 1, \text{ sinon} \end{cases} \\ \\ d \equiv 1 \end{cases}$$

Le second problème servira de modèle lors de l'étude sur ordinateur séquentiel au chapitre III, car il engendre un écart en nombre d'itérations bien plus conséquent que le problème 1.

Remarque I.1 *Les différences observées dans la résolution sur la CM-2 de ces deux problèmes, nous montreront la nécessité d'utiliser des préconditionnements polynômiaux de degré élevé pour les problèmes mal conditionnés. Elles nous amèneront ainsi à introduire des schémas d'évaluation plus stables pour ces polynômes.*

Remarque I.2 *Un problème est appelé problème mal conditionné, si son nombre de conditionnement est élevé, Dans ce cas lors de la résolution effective la méthode itérative nécessite un nombre important (par rapport à la taille du problème) d'itérations avant convergence. Nous verrons, par exemple aux chapitres IV et V, que le problème 2 est nécessite un nombre important d'itérations avant convergence.*

Remarque I.3 *La résolution du second problème à l'aide des préconditionnements polynômiaux est beaucoup plus réaliste que celle du premier problème. En effet pour la définition des préconditionnements polynômiaux nous avons besoin d'une estimation de la plus petite et de la plus grande valeur propre de la matrice initiale. Or en général ces valeurs ne sont pas connues a priori, sauf dans des cas classiques comme le problème 1. Néanmoins de nombreux et efficaces méthodes existent pour calculer ces valeurs, mais leur utilisation nous empecherait sans doute d'optimiser le temps de calcul. Donc pour ce second problème nous avons estimé ces valeurs a priori.*

I.4.2 Schéma de résolution

La méthode des différences finies fournit une approximation de la solution en un nombre fini de points de l'ouvert Ω. Pour cela, on commence par établir un maillage régulier de pas $h = \frac{1}{n+1}$ du domaine, voir figure I.1, avec la numérotation dite *naturelle* (de gauche à droite puis de bas en haut).

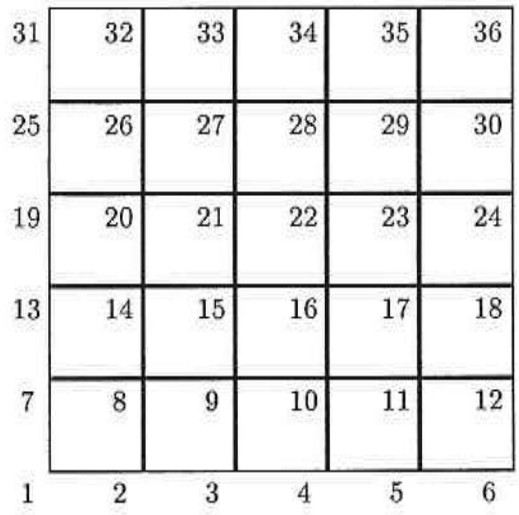

Figure I.1: Maillage utilisé, avec la numérotation dite *naturelle*

Notons $u_i^j = u(ih, jh)$, $0 \leq i, j \leq n$

La discrétisation du problème $(\mathcal{D})$ par différences finies avec le schéma usuel à cinq points, voir figure I.2, découle en fait de la formule d'approximation de dérivées suivante :

$$\frac{\partial u^j_{i+\frac{1}{2}}}{\partial x_1} \approx \frac{u^j_{i+1} - u^j_i}{h}. \tag{I.3}$$

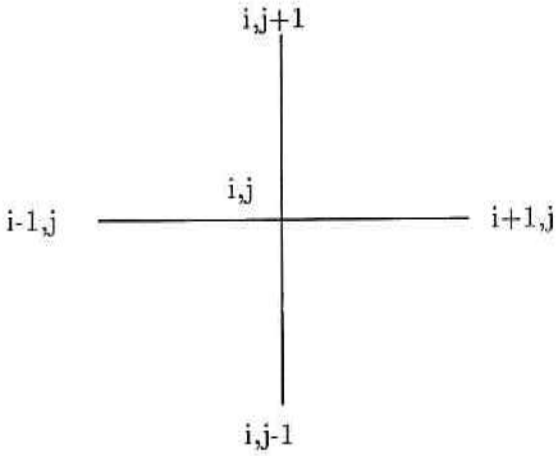

Figure I.2: Schéma à cinq points

Avec ce schéma centré on a :

$$-\frac{\partial}{\partial x_1}\left(c\frac{\partial u}{\partial x_1}\right)^j_i \approx -\frac{1}{h}\left(c^j_{i+\frac{1}{2}}\left(\frac{\partial u}{\partial x_1}\right)^j_{i+\frac{1}{2}} - c^j_{i-\frac{1}{2}}\left(\frac{\partial u}{\partial x_1}\right)^j_{i-\frac{1}{2}}\right) \tag{I.4}$$

$$\approx -\frac{1}{h}\left(c^j_{i+\frac{1}{2}}\frac{u^j_{i+1} - u^j_i}{h} - c^j_{i-\frac{1}{2}}\frac{u^j_i - u^j_{i-1}}{h}\right) \tag{I.5}$$

$$\approx -\frac{1}{h}\left(c^j_{i+\frac{1}{2}}u^j_{i+1} + c^j_{i-\frac{1}{2}}u^j_{i-1} - (c^j_{i+\frac{1}{2}} + c^j_{i-\frac{1}{2}})u^j_i\right) \tag{I.6}$$

avec

$$c^j_{i+\frac{1}{2}} = c((i+\frac{1}{2})h, jh) \tag{I.7}$$

Nous procédons de même sur la deuxième dimension, ainsi la discrétisation de $-div(\mathcal{A}\ \overrightarrow{grad}\ u)$ utilisée est la suivante :

$$\frac{1}{h^2}\Big((d_i^{j+\frac{1}{2}} + c^j_{i+\frac{1}{2}} + c^j_{i-\frac{1}{2}} + d_i^{j-\frac{1}{2}})u_i^j$$
$$-d_i^{j+\frac{1}{2}}u_i^{j+1} - c^j_{i+\frac{1}{2}}u^j_{i+1} - c^j_{i-\frac{1}{2}}u^j_{i-1} - d_i^{j-\frac{1}{2}}u_i^{j-1}\Big) \tag{I.8}$$

A l'aide du schéma de discrétisation (I.8) et à cause de la numérotation régulière du maillage utilisée, nous obtenons un système linéaire de la forme :

$$Ax = b \tag{I.9}$$

dans lequel A est une matrice symétrique définie positive tridiagonale par blocs et d'ordre $N = n^2$.

$$A = \begin{pmatrix} D_1 & -A_2^T & & & \\ -A_2 & D_2 & -A_3^T & & \\ & \ddots & \ddots & \ddots & \\ & & -A_{n-1} & D_{n-1} & -A_n^T \\ & & & -A_n & D_n \end{pmatrix} \tag{I.10}$$

Chaque matrice D_i est une matrice symétrique tridiagonale par points d'ordre n et les matrices A_i sont diagonales par points d'ordre n .

Dans ce cas A peut être considérée comme une matrice symétrique pentadiagonale. Lorsque nous considérerons la matrice A de cette façon, nous noterons les éléments non nuls de la ligne i respectivement de gauche à droite :

$$A = penta_N(cb_i, cg_i, cc_i, cd_i, ch_i)$$

avec les notations suivantes :

Définition I.3 *On note* $penta_N(a_i, b_i, c_i, d_i, e_i)$ *la matrice pentadiagonale d'ordre* N*, de même structure creuse que* A*, dont les éléments non nuls de la ligne* i *sont* a_i, b_i, c_i, d_i, e_i.

Définition I.4 *De la même manière, on note* $trid_n(a_i, b_i, c_i)$ *la matrice tridiagonale d'ordre* n *suivante :*

$$trid_n(a_i, b_i, c_i) = \begin{pmatrix} b_1 & c_1 & & & \\ a_2 & b_2 & c_2 & & \\ & \ddots & \ddots & \ddots & \\ & & a_{n-1} & b_{n-1} & c_{n-1} \\ & & & a_n & b_n \end{pmatrix}$$

Remarque I.4 *La matrice* A *est symétrique, nous avons donc* $cb_{i+n} = ch_i$ *et* $cg_{i+1} = cd_i$. *Nous simplifierons la notation pentadiagonale de* A *en notant*

$$A = penta_N(c_i, b_i, a_i, b_{i+1}, c_{i+n})$$

Remarque I.5 *Dans les expériences numériques du chapitre IV, la matrice* A *sera stockée par diagonale, chaque diagonale étant représentée par un vecteur. Avec l'introduction des zéros nécessaires , le produit matrice vecteur* $ap \leftarrow A \times p$ *s'écrit de la manière suivante :*

pour $i = 1, \ldots, N$

$$\begin{aligned} ap(i) \;=\; & cb(i) * p(i-n) + cg(i) * p(i-1) + cc(i) * p(i) \\ & + cd(i) * p(i+1) + ch(i) * p(i+n) \end{aligned} \qquad \text{(I.11)}$$

En fait, nous verrons dans la partie du chapitre suivant consacrée à la CM-2, que pour améliorer encore les performances de ce produit matrice vecteur, les diagonales seront stockées dans des tableaux bidimensionnels, dans les chapitres V à VIII, voir également le chapitre IX.

Pour conclure ce chapitre, considérons la matrice du problème 1. En fait on a

$$A = \frac{1}{h^2} \begin{pmatrix} D_n & -I_n & & & \\ -I_n & D_n & -I_n & & \\ & \ddots & \ddots & \ddots & \\ & & -I_n & D_n & -I_n \\ & & & -I_n & D_n \end{pmatrix} \tag{I.12}$$

avec I_n la matrice identité de dimension n et

$$\forall i = 1, \ldots, n, D_n = trid_n(-1, 4, -1)$$

Les propriétés de cette matrice, appelée communément matrice du Laplacien en dimension deux, sont bien connues. En particulier, ses valeurs propres sont données, voir [GoMe83] ou [Joly90], par :

$$\lambda_{i,j} = 4 \left(sin \left(\frac{i\pi}{2(n+1)} \right)^2 + sin \left(\frac{j\pi}{2(n+1)} \right)^2 \right), \ 1 \leq i, j \leq n \tag{I.13}$$

en particulier d'après (I.13), la plus petite valeur propre λ_{min} de cette matrice est

$$\lambda_{min} = 8 \ sin \left(\frac{\pi}{2(n+1)} \right)^2$$

et sa plus grande λ_{max}

$$\lambda_{max} = 8 \ sin \left(\frac{n\pi}{2(n+1)} \right)^2.$$

Ainsi, on obtient très facilement $\kappa(A)$ le conditionnement de cette matrice symétrique définie positive

$$\kappa(A) = \frac{sin \left(\frac{n\pi}{2(n+1)} \right)^2}{sin \left(\frac{\pi}{2(n+1)} \right)^2} = O(1/h^2) \tag{I.14}$$

Remarque I.6 *La connaissance de ces valeurs propres nous aidera lors de la mise en oeuvre des préconditionnements polynômiaux.*

CHAPITRE II

MÉTHODE DE RÉSOLUTION CHOISIE ET ORDINATEURS UTILISÉS

II.1 Introduction

Après nous être intéressé aux problèmes aux limites elliptiques dans le chapitre précédent, nous allons maintenant regarder la résolution numérique du type de problème suivant :

soit $A \in \mathrm{R}^N \times \mathrm{R}^N$ une matrice, $b \in \mathrm{R}^N$ un vecteur quelconque. On veut résoudre

$$\text{(P1)} \quad \begin{cases} \text{Trouver } x \in \mathrm{R}^n, \text{ solution de} \\ Ax = b \end{cases}$$

Dans ce but, nous rappelerons quelques principes de base de la méthode du gradient conjugué puis de la méthode du gradient conjugué préconditionné. Enfin nous conclurons ce chapitre par une présentation rapide de la CM-2 en insistant sur les contraintes que son utilisation entraine.

II.2 Le gradient conjugué

La matrice A possède une propriété essentielle : elle est symétrique définie positive. Dans ce cas, nous pouvons utiliser la méthode du gradient conjugué (G.C), due à Hestenes et Stiefel [HeSt52]. La méthode du G.C est une méthode de descente. A chaque itération, on détermine un vecteur p_k (direction de descente) et un scalaire α_k, qui permettent de calculer x_{k+1} (la solution approchée à l'étape $k+1$) par :

$$x_{k+1} = x_k + \alpha_k p_k$$

α_k et p_k étant calculés avec pour objectif de minimiser une fonctionnelle.

Ainsi on introduit la fonctionnelle $J : \mathrm{R}^N \to \mathrm{R}$, définie par

$$\forall u \in \mathrm{R}^N, J(u) = \frac{1}{2}(Au, u) - (b, u)$$

où (. , .) est le produit scalaire usuel de R^N

Alors nous pouvons énoncer le théorème suivant

Théorème II.1 *(i) J est strictement convexe sur R^N*

(ii) J admet un minimum unique, noté $x \in R^N$

(iii) $J'(x) = 0$

(iv) x est l'unique solution du problème (P1)

Preuve II.1 *Voir par exemple [Ciar82] ou [Joly90].*

Le problème (P1) est donc équivalent au problème de minimisation

$$\text{(P2)} \quad \begin{cases} \text{Trouver } x \in \mathrm{R}^n, \text{ solution de} \\ \forall z \in \mathrm{R}^n, z \neq x \;\; J(x) < J(z) \end{cases}$$

Notons $\{d^1, \ldots, d^N\}$ une base de R^N

$$\text{alors } \forall (x, x_0) \in \mathrm{R}^N \times \mathrm{R}^N, x - x_0 = \sum_{k=1}^{N} \alpha_k d^k$$

ainsi

$$\begin{aligned} J(x) &= J(x_0 + \sum_{k=1}^{N} \alpha_k d^k) \\ &= J(x_0) + \sum_{k=1}^{N} \alpha_k (Ax_0 - b, d^k) + \frac{1}{2} \sum_{k=1}^{N} \sum_{l=1}^{N} \alpha_k \alpha_l (Ad^k, d^l) \end{aligned}$$

Si on suppose de plus que la base est une base orthogonale associée au produit scalaire $(A\ .\ ,\ .\ )$, c'est à dire $(Ad^k, d^l) = 0$ pour $l \neq k$ alors

$$J(x) = J(x_0) + \sum_{k=1}^{N} \alpha_k (Ax_0 - b, d^k) + \frac{1}{2} \sum_{k=1}^{N} {\alpha_k}^2 (Ad^k, d^k)$$

ainsi pour x_0 fixé, $J(x)$ est une forme quadratique en α_k, et

$$\frac{\partial J}{\partial \alpha_k} = (Ax_0 - b, d^k) + \alpha_k (Ad^k, d^k)$$

donc

$$\frac{\partial J}{\partial \alpha_k} = 0 \Leftrightarrow \alpha_k = \frac{(b - Ax_0, d^k)}{(Ad^k, d^k)}.$$

Le minimum de J est donc atteint pour le vecteur x tel que

$$x = x_0 + \sum_{k=1}^{N} \frac{(b - Ax_0, d^k)}{(Ad^k, d^k)} d^k$$

Nous pouvons construire un algorithme de minimisation de la fonctionnelle J, en construisant de manière itérative une base $\{d^1, \ldots, d^N\}$ de R^N orthogonale associée au produit scalaire $(A\ .\ ,\ .\ )$ puis en minimisant J à chaque itération k sur le sous-espace $x_0 + < d^1, \ldots, d^k >$.

Nota $< d^1, \ldots, d^k >$ est le sous-espace vectoriel engendré par les vecteurs $d^1, \ldots, d^k$. Et on note $E^k =< d^1, \ldots, d^k >$.

On pourrait maintenant écrire l'algorithme du G.C, mais celui-ci possède de nombreuses propriétés simplificatrices. On obtient une écriture plus classique de l'algorithme du G.C.

On peut remarquer notamment parmi ces propriétés que comme $dim\ E^k = k$ et x_k réalise le minimum de J sur $x_0 + E^k$, l'algorithme converge en au plus N itérations.

Remarque II.1 *On peut constater lors de la mise en œuvre informatique, que cette propriété ne reste pas toujours vérifiée. Ce problème est dû au codage des nombres sur machine et en particulier à la propagation des erreurs d'arrondi. Cette méthode qui pouvait être considérée*

comme une méthode directe, est par conséquent utilisée comme une méthode itérative, voir [Reid71]. Il s'agit alors de trouver un critère d'arrêt efficace pour que la solution calculée soit suffisamment proche de la solution exacte.

Dans cette thèse nous utiliserons un critère d'arrêt portant sur la norme du résidu $r_k = b - Ax_k$.

Nous pouvons maintenant énoncer l'algorithme du G.C

Algorithme du G.C

soit $\epsilon > 0$ donné

Initialisation

soit $x_0 \in \mathrm{R}^N$ quelconque,

$$\begin{aligned} r_0 &= b - Ax_0 \\ d^0 &= r_0 \end{aligned}$$

Répéter $k = 0, 1, \ldots$

- Minimisation de J sur $x_0 + < d^0, \ldots, d^k >$

$$\begin{aligned} \alpha_k &= (r_k, r_k)/(Ad^k, d^k) \\ x_{k+1} &= x_k + \alpha_k d^k \\ r_{k+1} &= r_k - \alpha_k A d^k \end{aligned}$$

- Calcul de la nouvelle direction de descente

$$\begin{aligned} \beta_{k+1} &= (r_{k+1}, r_{k+1})/(r_k, r_k) \\ d^{k+1} &= r_{k+1} + \beta_{k+1} d^k \end{aligned}$$

jusqu'à $\| r_{k+1} \| \leq \epsilon \| r_0 \|$

Evaluons rapidement le coût de calcul de cette méthode. Notons C le nombre moyen d'éléments non nuls par ligne de la matrice A. A chaque itération nous devons effectuer :

- un produit matrice vecteur : $2CN$ opérations
- deux produits scalaires : $4N$ opérations
- trois combinaisons linéaires : $6N$ opérations

Alors pour une convergence en it itérations nous effectuons approximativement $2it(C+5)N$ opérations. En fait pour une matrice pleine et une convergence en N itérations nous obtenons $2N^3$ opérations, ce qui est relativement important particulièrement par rapport à la méthode de Cholesky qui n'en nécessite que $N^3/6$.

Remarque II.2 *Ce cas de figure extrêmement défavorable ne se présente évidemment pas. Tout d'abord la matrice est très creuse* ($C = 5$). *Et de plus, nous verrons dans les expérimentations numériques qu'avec ou même sans préconditionnement, it est nettement inférieur à* N.

II.3 Le gradient conjugué préconditionné

La méthode du G.C est une méthode de résolution relativement efficace. Néanmoins, si les problèmes sont mal conditionnés, elle perd en partie son efficacité. En effet, on peut montrer la majoration suivante, voir [GoMe83] ou [LaTh86] :

$$\epsilon(x_k) \leq 4\left(\frac{\sqrt{\kappa(A)}-1}{\sqrt{\kappa(A)}+1}\right)^{2k} \epsilon(x_0) \qquad \text{(II.1)}$$

avec

$$\epsilon(x_k) = (A(x_k - x), x_k - x)$$

et $\kappa(A)$ étant le nombre de conditionnement de la matrice A. En fait, pour une matrice symétrique définie positive A on a :

$$\kappa(A) = \frac{\lambda_{max}}{\lambda_{min}}$$

λ_{min} étant la plus petite valeur propre de A et λ_{max} sa plus grande.

Par conséquent, la vitesse de convergence de l'algorithme du G.C est directement liée au nombre de conditionnement de la matrice A, de la manière suivante : plus $\kappa(A)$ est proche de un, plus la méthode converge rapidement.

Remarque II.3 *En fait, le nombre d'itérations effectué par le G.C.P ne dépend pas uniquement du conditionnement de la matrice. En effet, la distribution des valeurs propres influence également la convergence, voir [Jenn77].*

Ainsi un moyen de réduire le nombre d'itérations est d'essayer de réduire le nombre de conditionnement du système à résoudre. Le principe du préconditionnement d'une matrice A consiste à remplacer le problème (P1) par

$$\text{(P3)} \quad \begin{cases} \text{Trouver } x \in \mathrm{R}^N, \text{ solution de} \\ M^{-1}Ax = M^{-1}b \end{cases}$$

M^{-1} devant être choisie, dans la mesure du possible, telle que $\kappa(M^{-1}A)$ soit beaucoup plus petit que $\kappa(A)$. Ensuite si on suppose que M est symétrique définie positive alors $M^{-1}A$ n'est pas systématiquement symétrique définie positive. Néanmoins, la matrice $M^{-1/2}AM^{-t/2}$ semblable à $M^{-1}A$ est symétrique définie positive, avec $M^{-t/2}$ la matrice transposée de $M^{-1/2}$.

Nota La matrice $M^{-1/2}$ est la matrice telle que $M^{-1/2}M^{-t/2} = M^{-1}$, et $M^{-1/2}$ existe car M est symétrique définie positive.

Donc au lieu de résoudre (P3) on résout

$$\text{(P4)} \quad \begin{cases} \text{Trouver } y \in \mathrm{R}^N, \text{ solution de} \\ M^{-1/2}AM^{-t/2}y = M^{-1/2}b \end{cases} .$$

Alors x solution de (P1) est obtenue en résolvant $M^{t/2}x = y$.

Etant donné que la matrice $M^{-1/2}AM^{-t/2}$ est symétrique définie positive on peut appliquer la méthode du G.C au problème (P4), ce qui revient à écrire l'algorithme du G.C.P dû à Concus Golub et O'Leary [CoGO76]

Algorithme du G.C.P

soit $\epsilon > 0$ donné

Initialisation

soit $x_0 \in \mathrm{R}^N$ quelconque,

$$\begin{aligned} r_0 &= b - Ax_0 \\ Mz_0 &= r_0 \\ p^0 &= z_0 \end{aligned}$$

Répéter $k = 0, 1, \ldots$

- Etape de minimisation

$$\begin{aligned} \alpha_k &= (r_k, z_k)/(Ap^k, p^k) \\ x_{k+1} &= x_k + \alpha_k p^k \\ r_{k+1} &= r_k - \alpha_k Ap^k \end{aligned}$$

- Etape de préconditionnement

$$\begin{cases} \text{Calculer } z_{k+1} \in \mathrm{R}^N, \text{ solution de} \\ Mz_{k+1} = r_{k+1} \end{cases}$$

- Calcul de la nouvelle direction de descente

$$\begin{aligned} \beta_k &= (r_{k+1}, z_{k+1})/(r_k, z_k) \\ p^{k+1} &= z_{k+1} + \beta_k p^k \end{aligned}$$

jusqu'à $\| r_{k+1} \| \leq \epsilon \| r_0 \|$

Remarque II.4 *Le G.C.P requiert trois types d'opérations élémentaires sur les vecteurs. La plus simple est la combinaison linéaire, la suivante le produit scalaire, enfin la dernière le produit matrice vecteur. Comme nous le verrons, par la suite ces opérations sont relativement bien adaptées à la CM-2. Le problème posé est donc bien celui du choix du préconditionnement.*

Pour une bonne efficacité de cette méthode la matrice M doit posséder plusieurs qualités :

(i) M proche de A, ou encore $\kappa(M^{-1}A)$ beaucoup plus petit que $\kappa(A)$,

(ii) le système $Mz = r$ doit être facile à résoudre,

(iii) la structure de la matrice M bien adaptée à l'architecture de la machine utilisée.

Remarque II.5 *Les conditions (ii) et (iii) dépendent de la machine utilisée. En effet, les conditions de résolution d'un système linéaire ne sont certes pas les mêmes sur différents ordinateurs (même s'ils sont tous parallèles). Par exemple, des méthodes efficaces sur ordinateurs séquentiels, telle la factorisation de Cholesky incomplète par points (ou par blocs pour les machines vectorielles), ne sont pas très adaptées à la CM-2. En effet, elles possèdent à des degrés divers des dépendances de données, qui engendrent un taux de parallélisme insuffisant pour la CM-2. Sur la CM-2, il faut donc se tourner vers d'autres types de préconditionnements.*

Remarque II.6 *La méthode du G.C.P est parmi les méthodes s'écrivant :*

$$x_{k+1} = x_0 + P_k(M^{-1/2}AM^{-t/2})z_0$$

celle qui minimise à chaque étape $\epsilon(x_{k+1})$, voir [GoMe83]. Il peut, donc, sembler en quelque sorte étrange voire inutile d'utiliser des préconditionnements polynômiaux pour améliorer le taux de convergence. Nous verrons dans cette thèse, que l'on parvient à atteindre, grâce aux caractéristiques de la CM-2, des performances tout à fait acceptables avec ce type de préconditionnements.

Remarque II.7 *Lors de la résolution effective, dans les chapitres III à VIII, nous avons pris comme initialisation* $x_0 = 0$ *et* $b = A \times xec$, *avec*

$$xec((j-1)*n+i) = ih*jh*(1-ih)*(1-jh)*exp(ih*jh)$$

$$et\ h = \frac{1}{n+1}.$$

Nous avons ainsi pu comparer la solution calculée par le G.C.P et la solution exacte. En fait dire que xec est la solution de (I.9) est équivalent à chercher un second membre de l'équation f tel que la fonction $u(x,y) = xy(1-x)(1-y)e^{xy}$ *soit solution du problème* $\mathcal{D}$.

II.4 Les supports informatiques

En conclusion de ce chapitre, nous allons présenter brièvement la Connection Machine et nous verrons, en outre les prémices des conditions à suivre pour obtenir des méthodes efficaces sur cette machine.

La création de la Connection Machine remonte à la thèse du Dr W. D. Hillis, [Hill85] et [Hill88]. Produite par TMC (Thinking Machine Corporation), la CM-1 était destinée à l'origine à l'intelligence artificielle et au traitement de l'image. Par la suite, afin d'étendre son spectre d'application à un domaine plus large incluant la simulation numérique la CM-1 a évolué pour devenir la CM-2 (1986-1992). La CM-200, crée en 1991, possède en réalité la même architecture que la CM-2, mais avec des performances plus élevées. Enfin le dernier supercalculateur né de la famille des Connection Machines est la CM-5. Il est apparu en 1992. Son architecture n'est certes plus aussi restrictive que celle de ses prédecesseurs.

Les résolutions numériques ont été effectuées, en premier lieu sur la CM-2. Puis dans un second temps, nous avons étudié ces méthodes de résolution sur la CM-200 pour d'une part confirmer les résultats obtenus sur la CM-2, et d'autre part pour suivre naturellement l'évolution du site qui nous a permis de réaliser nos études. Les calculateurs

utilisés dans cette thèse sont ceux du SEH (Site Expérimental en Hyperparallélisme, 16 bis avenue Prieur de la Côte d'Or, 94114 ARCUEIL CEDEX).

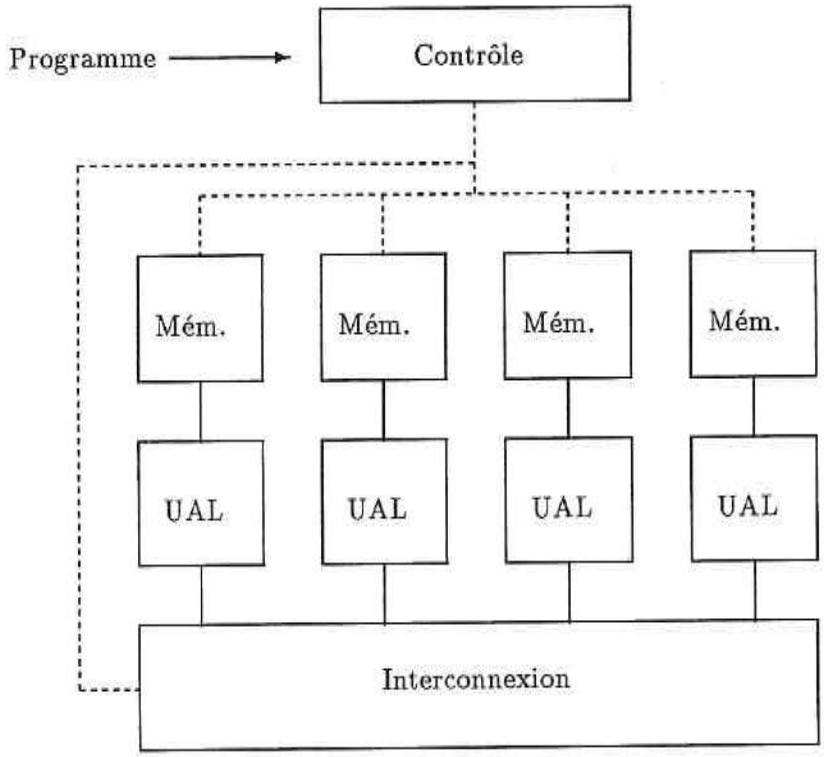

Figure II.1: Architecture de la CM-2

La CM-2 est une machine multiprocesseurs à mémoire locale. En fait, chaque UAL (Unité Arithmétique et Logique) possède une mémoire propre, ainsi une UAL plus une mémoire forme un processeur élémentaire (PE), voir figure II.1. La CM-2 comporte jusqu'à 65536 PEs, mais grâce à un partage statique de la mémoire elle peut être utilisée comme plusieurs machines comportant chacune 8192 PEs.

Remarque II.8 *En mode Slicewise, voir [Porc91], les PEs, qui sont physiquement des processeurs un bit (d'où les applications à l'intelligen-*

ce artificielle), sont regroupés par 64 pour accélérer les calculs flottants. Nous utiliserons uniquement ce mode pour nos calculs.

Remarque II.9 *Si lors de l'exécution, un programme génère un vecteur de taille plus grande que le nombre de processeurs disponibles, il y a alors création de processeurs virtuels. Détaillons ce phénomène sur un exemple plus précis, prennons quatre processeurs numérotés de un à quatre et un vecteur V à douze éléments, v_i, nous obtenons en schématisant le stockage suivant*

processeurs	1	2	3	4
vecteur V	v_1	v_2	v_3	v_4
	v_5	v_6	v_7	v_8
	v_9	v_{10}	v_{11}	v_{12}

Il est donc intéressant, afin d'obtenir des performances élevées, d'utiliser des vecteurs ou des tableaux ayant un nombre d'inconnues proportionel au nombre de processeurs. Sur l'exemple ci-dessus, que le vecteur V possède douze ou bien neuf inconnues, la CM-2 mettra le même temps pour effectuer des opérations sur ce vecteur.

D'après la classification de Flynn, voir [Flyn72], la CM-2 est une machine SIMD (Single Instruction Multiple Data) à mémoire locale, c'est à dire chaque PE effectue la même opération en même temps que tous les autres PEs ou rien du tout. Donc, la programmation s'effectue de manière séquentielle avec des vecteurs et des transferts de données synchrones entre processeurs (modèle de progammation data parallel). Grâce à sa mémoire locale, chaque PE peut effectuer immédiatement des opérations sur des données propres. Par contre, si un PE a besoin des données, qui ne se trouvent pas dans sa mémoire, il y a mise en place de liens de communications entre PEs grâce au réseau d'interconnexion (Les PEs sont reliés entre eux par un réseau hypercube[1] entre des nœuds

[1]le nœud i, qui comprend les PEs $16i$ à $16i$+15, est connecté au nœud $j \Leftrightarrow$ il existe k tel que $|i-j| = 2^k$. On dit alors que la connexion utilise la dimension k.

de 16 PEs). Mais ces communications diminuent les performances, notamment en mode Router (le mode des communications irrégulières). Ce mode est notamment utilisé lors d'un adressage indirect (par exemple lors d'un produit matrice vecteur avec un stockage morse d'une matrice à structure irrégulière). D'autres modes de communications sont néanmoins plus rapides, en particulier le mode News (directement lié à la structure de l'hypercube), utilisé pour des communications régulières telles que les shifts (par exemple x(i)=y(i+1)). De plus nous éviterons les communications entre PEs trop éloignés.

Remarque II.10 *On peut constater, par exemple, qu'une communication en mode News correspond à six additions alors qu'une communication en mode Router équivaut à 75 additions, voir [Porc91]. De plus, plus le nombre de processeurs virtuels est grand plus le mode News est rapide car dans ce cas il y a plus de transferts de données en mémoire locale. Nous pouvons donc constater dès maintenant que le coût de communication est très important sur la CM-2, notamment pour les communications sont irrégulières.*

Remarque II.11 *Les communications ont tout de même quelques avantages, il est possible en particulier d'effectuer des calculs au vol. C'est à dire que pendant un échange de données entre processeurs un calcul, une addition par exemple, est possible sur ces données échangées. Ceci permet d'obtenir un produit scalaire efficace.*

Remarque II.12 *Afin d'optimiser le produit matrice vecteur sur la CM-2, nous stockerons, dans les chapitres V à VIII, les diagonales de la matrice A comme des tableaux bidimensionnels, semblables au maillage de la figure I.1, voir [ChKT89] et [Robi91]. Ainsi, les shifts nécessaires au produit matrice vecteur s'effectuent dans deux directions au lieu d'une mais avec des longueurs de un au lieu de un et n. Nous espérons, dans ce cas, obtenir un produit matrice vecteur performant. Par contre, nous perdrons un peu d'efficacité avec le produit scalaire. En effet, alors qu'avec le stockage unidimensionnel le produit scalaire a été optimisé par TMC, la version bidimensionnelle se décompose en deux sommes successives (au lieu d'une seule) sur des vecteurs de taille n (au lieu de N). Malgré tout, d'après [Meur89b] le produit matrice vecteur étant l'opération la plus coûteuse en temps du G.C, il sera sans*

doute efficace d'optimiser le produit matrice vecteur au détriment du produit scalaire. Nous verrons au chapitre IX l'importance du produit matrice vecteur dans les performances des préconditionnements polynômiaux.

De plus, la CM-2 fonctionne sous le contrôle d'une machine hôte, une station SUN en général, la machine hôte transmettant au séquenceur de la CM-2 les instructions parallèles à exécuter.

Enfin, le langage de programmation utilisé est le CM-Fortran, voir [ThMC90] et [ThMC91]. Le CM-Fortran est un langage proche du Fortran 77 avec en plus des extensions, particulièrement des opérations sur les tableaux. Les performances et les résultats établis dans cette thèse ont été mesurés avec à l'horloge interne de la CM-2. Grâce à deux appels successifs, elle permet de déterminer deux caractéristiques importantes du temps : premièrement le temps total (et réel) d'exécution d'un programme TCPU, et deuxièmement le temps pendant lequel la CM-2 a été effectivement utilisée TCM2.

Définition II.1 *On appelle taux de parallélisme d'un programme le nombre τ défini par*

$$\tau = \frac{TCM2}{TCPU}.$$

Remarque II.13 *En fait, τ représente le taux d'utilisation de la CM-2, il est donc très important qu'il soit élevé pour ne pas sous-utiliser la CM-2. Dans l'absolu, avec un programme parfaitement massivement parallèle on pourrait atteindre $\tau = 1$. En réalité, nous n'atteindrons jamais cette valeur notamment à cause des communications entre la CM-2 et le SUN mais aussi à cause des calculs sur les scalaires effectués sur le SUN.*

Remarque II.14 *Hélas, TCPU n'est pas systématiquement fiable sur la CM-2. En effet, lors de nos nombreuses mesures, nous avons pu constater que même si nous étions connectés seul à la CM-2, il suffisait qu'une ou plusieurs personnes travaillent en même temps sur la station hôte pour que ces mesures ne soient plus représentatives des performances de la CM-2. Ainsi, toutes les mesures et toutes les performances exposées dans cette thèse ont été obtenues à partir de TCM2.*

Nous convenons, bien volontiers, que cette approximation n'est pas très juste, pourtant elle s'est avérée la seule possible à mettre en pratique. Ne pouvant relancer indéfiniment des programmes parfois très longs nous nous sommes résolus, comme beaucoup d'autres, à utiliser cette solution. Donc tous les temps de calculs présentés sont sous-évalués et toutes les vitesses (Mflops[2]) sont quelque peu surévaluées. Enfin, tous les résultats exposés sont en fait des moyennes prises sur au moins deux échantillons.

Remarque II.15 *A titre indicatif, nous avons tout de même établi quelques taux de parallélisme. Il était en effet intéressant de savoir si nous utilisions pleinement les capacités de calcul de la CM-2 . Dans ce cas, nous avons sélectionné des temps TCPU et TCM2, qui nous semblaient dignes de confiance, et dont nous possédions de nombreux résultats tous concordants.*

Nous allons clore ce chapitre en rappellant simplement que l'ordinateur séquentiel utilisé dans le chapitre IV est une station de travail Apollo (série 400) du laboratoire d'Analyse Numérique de l'Université Paris 6.

[2]Millions Floating Operations per Seconde

CHAPITRE III

LES PRÉCONDITIONNEMENTS UTILISÉS

III.1 Introduction

L'étude réalisée au chapitre précédent nous a permis de cerner les difficultés du sujet proposé. Nous allons donc, dans une première partie, exposer des conditions nécessaires pour obtenir des préconditionnements massivement parallèles. Ensuite ces contraintes nous permettront de décrire des préconditionnements a priori bien adaptés à l'architecture de la CM-2. Tout d'abord, le préconditionnenent Diagonal, puis des renumérotations Rouge-Noir de préconditionnements déjà connus et enfin des préconditionnements polynômiaux. D'autre part, nous rappellerons des préconditionnements non nécessairement bien adaptés à la CM-2, mais possédant des propriétés communes avec les précédents. Ces préconditionnements nous permettront de vérifier que les préconditionnements adaptés à l'architecture de la CM-2 restent tout de même des préconditionnements efficaces du point de vue du taux de convergence. Ainsi nous nous sommes intéressés à des approximations directes de l'inverse de la matrice, mais aussi à des expressions analytiques par blocs de l'inverse [CoGM85], [Meur89b], puis à des factorisations incomplètes par blocs [CoGM85], [Meur89b]. Pour finir ce chapitre nous avons considéré des préconditionnements locaux et puis des préconditionnements basés sur des compléments de Schur.

III.2 Conditions sur les préconditionnements

Sur la CM-2, la programmation se fait de manière séquentielle sur des tableaux. On peut donc assurément penser que les combinaisons

linéaires entre vecteurs sont efficaces. De plus, la CM-2 est capable d'effectuer des calculs au vol, on peut alors légitimement espérer que le produit scalaire est performant. Finalement, grâce au maillage régulier, à la numérotation classique et au stockage de la matrice par diagonales (voir remarques I.7 et II.11) nous n'utilisons que des communications régulières et courtes pour le produit matrice vecteur. Il semble donc que les opérations élémentaires du G.C.P soient relativement bien adaptées à l'architecture de la CM-2. Le problème posé est donc bien celui du choix du préconditionnement. Pour une bonne efficacité sur la CM-2, celui-ci doit posséder, dans la mesure du possible, plusieurs qualités, en particulier il devrait :

- évidemment être parallèle, mais de plus avec un degré de parallélisme[1] très élevé (N si possible), i.e être capable d'effectuer des opérations sur un grand nombre de données en même temps.
- ne posséder que peu de scalaires à initialiser ou à calculer, afin d'obtenir un bon taux de parallélisme.
- éviter les communications irrégulières et les communications entres PEs trop éloignés sur le réseau (nous nous fixerons donc par exemple comme condition de n'avoir que des communications entre un nœud et ses voisins immédiats).
- utiliser au maximum les opérations efficaces sur la CM-2 : combinaison linéaire, produit matrice multidiagonale vecteur, etc...

Toutes ces conditions sont destinées à obtenir un préconditionnement massivement parallèle, mais il ne faut tout de même pas oublier que le préconditionnement doit avant toute chose réduire le nombre d'itérations de la méthode du G.C.P. Ainsi, sous ses conditions nous avons réussi à dégager trois types de préconditionnements :

- le préconditionnement Diagonal

[1] On appellera degré de parallélisme d'un préconditionnement, la dimension minimum des vecteurs sur lequel on peut exécuter des instructions en parallèle lors de l'étape de préconditionnement mais aussi lors de l'initialisation de ce préconditionnement (mis à part les quelques opérations sur des réels). Il s'agit donc du nombre d'opérations calculable en parallèle.

- des renumérotations Rouge-Noir des inconnues à partir de préconditionnements classiques

- des approximations directes de l'inverse de A avec des polynômes en A

Nous allons maintenant présenter les préconditionnements retenus pour l'étude sur la CM-2. Nous expliciterons, souvent sous forme de remarques, quelques problèmes rencontrés lors de la mise en œuvre informatique de ces préconditionnements.

III.3 Le préconditionnement Diagonal

Le préconditionnement Diagonal (méthode notée DIAG) est certainement le préconditionnement le plus simple à imaginer et à utiliser. En effet il suffit de prendre M la matrice diagonale par points d'ordre N dont les éléments diagonaux sont les éléments diagonaux de A, c'est à dire :

$$M = diag_N(A).$$

Alors la résolution $Mz = r$ est éminemment parallèle. Ce choix est même certainement le meilleur possible au point de vue du parallélisme, mais il n'est certes pas très efficace du point de vue du nombre d'itérations (nous nous en apercevrons dans les experiences numériques). Par exemple pour le problème 1, le préconditionnement Diagonal ne change pas le conditionnement de la matrice.

Il ne faut tout de même pas négliger et sous-estimer ce préconditionnement, car il demeure pour beaucoup le préconditionnement capable de donner les temps de calcul les plus faibles sur la CM-2. Ce préconditionnement sera donc au cours de cette thèse le préconditionnement référence de la CM-2. L'un des objectifs est évidemment de trouver un préconditionnement plus efficace.

III.4 Renumérotation Rouge-Noir

Supposons, par exemple, que N soit pair. Alors nous répartissons les points du maillage en deux ensembles disjoints, les nœuds rouges (R) et les nœuds noirs (B). Ensuite nous numérotons tout d'abord les nœuds rouges de façon *naturelle*, puis les nœuds noirs de la même manière, comme indiqué sur la figure III.1 :

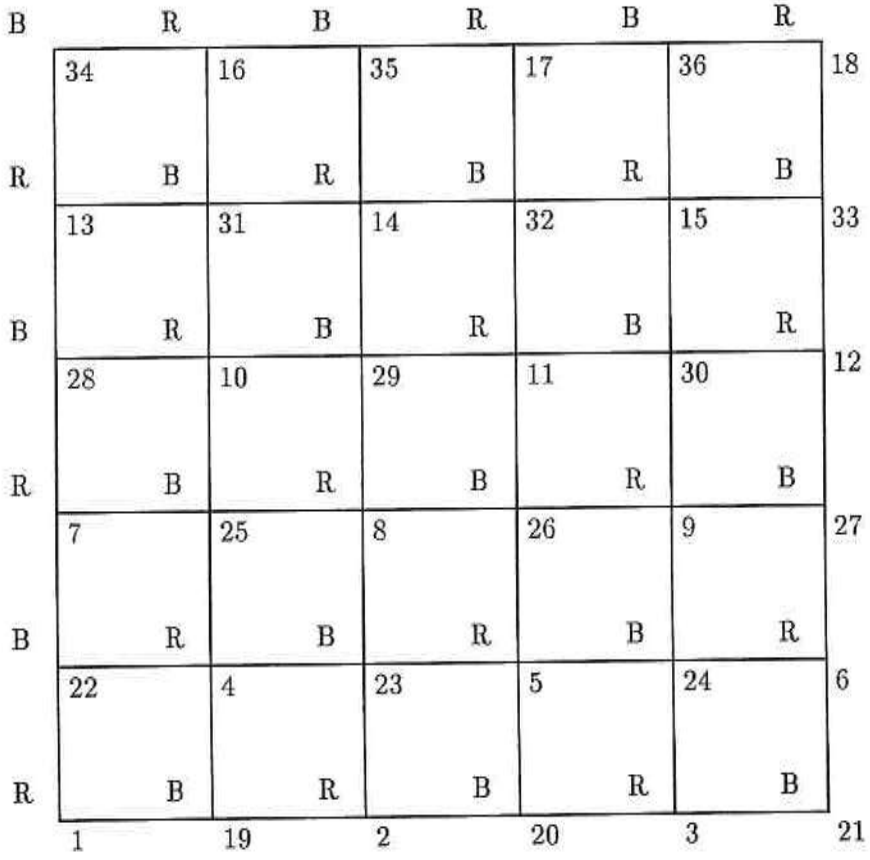

Figure III.1: Renumérotation Rouge-Noir des inconnues

Notons x_R et x_B les vecteurs des inconnues des nœuds rouges et

noirs respectivement, alors le système (I.9) se réecrit :

$$\overline{A}\begin{pmatrix} x_R \\ x_B \end{pmatrix} = \begin{pmatrix} b_R \\ b_B \end{pmatrix},$$

et

$$\overline{A} = \begin{pmatrix} R & C^T \\ C & B \end{pmatrix}.$$

R et B sont des matrices diagonales d'ordre $\frac{N}{2}$, et C est une matrice d'ordre $\frac{N}{2}$ avec au maximum quatre éléments non nuls par ligne. Pour la numérotation des inconnues, il est commode dans ce cas de se référer à la numérotation des nœuds du maillage par un couple d'indices (i,j). Nous noterons les éléments de C correspondants $(cb)_{(i,j)}, (cg)_{(i,j)}$, $(cd)_{(i,j)}, (ch)_{(i,j)}$.

A partir d'une factorisation incomplète de Cholesky

Nous allons chercher maintenant une décomposition de la forme :

$$M = LD^{-1}L^T,$$

avec

$$L = \begin{pmatrix} D_R & 0 \\ D_C & D_B \end{pmatrix} \text{ et } D = \begin{pmatrix} D_R & 0 \\ 0 & D_B \end{pmatrix}.$$

Alors

$$M = \begin{pmatrix} D_R & D_C^T \\ D_C & D_B + D_C D_R^{-1} D_C^T \end{pmatrix}.$$

Il est donc tout à fait logique de prendre $D_R = R$ et $D_C = C$ pour que M soit proche de $\overline{A}$. Lorsque nous calculons une décomposition complète D_B est une matrice de même structure que $D_C D_R^{-1} D_C^T$. Mais ici, nous voulons une décomposition incomplète efficace sur machine massivement parallèle, nous demanderons donc à D_B d'être diagonale.

Notre décomposition (RBIC) est donc définie par :

$$\begin{cases} D_R &= R \\ D_C &= C \\ D_B &= B - diag(CR^{-1}C^T) \end{cases}$$

Pour l'utilisation de cet algorithme, il nous faut donc calculer la diagonale de $CR^{-1}C^T$. Il est facile de voir que pour le point du maillage (i,j) nous avons :

$$\left(diag(CR^{-1}C^T)\right)_{(i,j)} = \frac{(cb)^2_{(i,j)}}{(R)_{(i,j-1)}} + \frac{(cg)^2_{(i,j)}}{(R)_{(i-1,j)}} + \frac{(cd)^2_{(i,j)}}{(R)_{(i+1,j)}} + \frac{(ch)^2_{(i,j)}}{(R)_{(i,j+1)}}.$$

Montrons que cette décomposition introduit du parallélisme dans la résolution de $Mz = r$ avec $M = LD^{-1}L^T$.

Nous résolvons tout d'abord

$$L\begin{pmatrix} y_R \\ y_B \end{pmatrix} = \begin{pmatrix} r_R \\ r_B \end{pmatrix},$$

par

$$\begin{cases} y_R &= R^{-1}r_R \\ y_B &= D_B^{-1}(r_B - Cy_R) \end{cases}.$$

Cette première étape est évidemment parallèle, car elle ne comporte que des inversions de matrices diagonales et un produit matrice vecteur.

Ensuite nous résolvons

$$D^{-1}L^T\begin{pmatrix} z_R \\ z_B \end{pmatrix} = \begin{pmatrix} y_R \\ y_B \end{pmatrix},$$

par

$$\begin{cases} z_B &= y_B \\ z_R &= y_R - R^{-1}C^T z_N \end{cases}.$$

Cette seconde étape est, pour les mêmes raisons, évidemment parallèle.

Les inconvénients de cette méthode sont, en premier lieu, que le degré de parallélisme n'est que de $\frac{N}{2}$ et ensuite, ainsi qu'il a été montré dans [DuMe89], que la vitesse de convergence du gradient conjugué avec ce préconditionnement n'est pas très bonne et en tout cas inférieure à celle obtenue avec la décomposition incomplète de Cholesky sur numérotation *naturelle*.

Remarque III.1 *Lors de la mise en œuvre informatique, nous avons crée un tableau bidimensionnel, semblable au maillage, contenant la valeur VRAI si le nœud est rouge et FAUX sinon. Alors à chaque calcul sur les nœuds rouges (respectivement noirs), nous avons effectué, grâce à une instruction CM-Fortran, un test pour ne travailler qu'avec les nœuds rouges (respectivement noirs). Nous sommes tout à fait conscient que cette opération a affaibli les performances de ces méthodes, notamment au niveau du temps total de résolution et des Mflops. D'autres modèles de programmation permettront sans doute d'optimiser les performances de ces préconditionnements, mais de toute façon il faudra éviter les communications irrégulières ou trop éloignés.*

A partir d'une décomposition SSOR

Ce préconditionnement (RBSSOR) est particulièrement simple, il consiste à poser :

$$M = LD^{-1}L^T$$

avec

$$L = \begin{pmatrix} R & 0 \\ C & B \end{pmatrix} \text{ et } D = \begin{pmatrix} R & 0 \\ 0 & B \end{pmatrix}.$$

La parallélisation s'effectue comme précédemment. Toutes les étapes sont, bien entendu, parallèles de degré $\frac{N}{2}$. De plus, contrairement à ce qui est fait pour la numérotation *naturelle*, nous n'avons pas introduit de paramètre d'optimisation ω, car il a été montré (cf [Meur89a]) que le paramètre optimal est $\omega = 1$.

III.5 Préconditionnements polynômiaux

III.5.1 Remarques sur les préconditionnements polynômiaux

Le principe des préconditionnements polynômiaux, dû à Rutishauser [Ruti59], consiste à trouver un préconditionnement M tel que

$$M^{-1}A = P_k(A)A,$$

avec P_k un polynôme de degré inférieur ou égal à k.

Définition III.1 *Nous noterons en lettres majuscules les polynômes appliqués à des matrices et en lettres minuscules correspondantes les polynômes appliqués à des scalaires.*

Cette idée peut paraître, à la fois, naturelle et étrange. Naturelle car d'après le théorème de Cayley-Hamilton, A^{-1} peut être exprimée comme un polynôme en A, mais aussi étrange car la méthode du G.C génère, dans un certain sens voir [GoMe83], un polynôme optimal. Ainsi le polynôme généré par le G.C, sans préconditionnement, après $m(k+1)$ itérations est meilleur que le polynôme généré par le G.C préconditionné par un polynôme de degré k après m itérations. En fait, en utilisant des préconditionnements polynômiaux nous désirons réduire suffisamment le nombre d'itérations, et ainsi supprimer quelques produits scalaires du G.C.P en les remplaçant par des produits matrice vecteur (opérations bien plus efficaces sur la CM-2).

Dans cette partie consacrée aux préconditionnements polynômiaux, nous allons étudier différents choix pour P_k. $P_k(A)$ doit être, d'une certaine manière, une approximation de A^{-1}. Bien sûr, nous avons plusieurs possibilités pour parvenir à ce résultat. En tout cas, comme l'ont prouvé Ashby, Manteuffel et Otto dans [AsMO92], il n'existe pas un unique polynôme optimal. Le choix du polynôme dépend en particulier de la distribution des valeurs propres, qui n'est pas connue a priori (mis à part les cas classiques).

Nous allons décrire, dans un premier temps, des préconditionnements basés sur des séries de Neumann tronquées [DuGR79]. L'avantage de ces préconditionnements polynômiaux est leur simplicité. En effet, ils n'ont aucun paramètre à estimer. Hélas, ils ne constituent pas une famille assez efficace de préconditionnements, du point de vue du gain d'itérations (voir les résultats numériques). De plus, il n'est pas intéressant d'utiliser des degrés plus élevés que un ou trois car l'amélioration du taux de convergence obtenue n'est pas du tout suffisante par rapport à l'augmentation du nombre de calculs à effectuer à chaque itération. Ainsi, avec ce type de préconditionnements polynômiaux, nous ne pouvons pas utiliser abondamment le produit matrice multidiagonale vecteur.

Donc, nous considérerons, dans un second temps, d'autres types de préconditionnements polynômiaux tels que $p_k(\lambda) = \sum_i \alpha_i \lambda^i$. Afin d'optimiser ces préconditionnements les α_i ne prendront plus seulement la valeur un, comme avec les séries de Neumann, mais ils seront calculés dans le but d'optimiser certaines normes. En variant les normes nous obtiendrons différents choix pour α_i. Ainsi nous exposerons un préconditionnement polynômial minimisant une généralisation du conditionnement (Minmax), voir entre autres [Adam82], [Adam85], [Ashb-87], [Ashb91], [AsMO92] et bien sûr [JoMP83]. Enfin nous nous interesserons à des préconditionnements minimisant une autre norme, voir [JoMP83], [Saad85] et [Saad87].

Remarque III.2 *Si on multiplie une matrice pentadiagonale par une autre matrice pentadiagonale on obtient une matrice avec treize diagonales. Alors, pour éviter ce remplissage, nous ne calculerons pas explicitement les différentes puissances de la matrice. Nous effectuerons la résolution du système $Mz = r$ (ou plutôt $z = P_k(A)r$) grâce à des schémas d'évaluation ne nécessitant que des produits matrice vecteur et des combinaisons linéaires. De plus, la phase d'initialisation des coefficients α_i n'étant pas du tout parallèle nous avons essayer de la réduire au maximum. Ainsi nous éviterons tout calcul de coefficients supplémentaires. En particulier nous ne calculons pas les points de Tchebycheff qui améliorent la stabilité du simple schéma de Hörner voir [StBu80]. Nous avons donc utilisé un schéma classique d'évaluation pour les polynômes le schéma de Hörner simple :*

supposons les α_i connus, alors $P_k(A)r$ est obtenu par les étapes suivantes

$$\begin{cases} b_k = \alpha_k r, \\ b_i = \alpha_i r + A b_{i+1}, \text{ pour } i = k-1, \ldots, 0 \\ \qquad \text{alors } z = P_k(A)r = b_0. \end{cases} \tag{III.1}$$

Ce schéma est optimal du point de vue du nombre d'opérations effectuées pour l'évaluation de $P_k(A)r$. Par contre, nous verrons au chapitre suivant qu'il peut conduire à des instabilités numériques, voir aussi [Gaut86], [Gaut90] et [StBu80].

III.5.2 Séries de Neumann tronquées

Comme proposé par Dubois, Greenbaum et Rodrigues [DuGR79], nous allons utiliser la propriété suivante :

$$si\ \rho(A) < 1\ alors\ (I - A)^{-1} \approx I + A. \tag{III.2}$$

Notons

$$A = D - L - L^T,$$

où D est la diagonale de A et L sa partie strictement triangulaire inférieure. De plus la matrice étudiée est symétrique définie positive donc ses éléments diagonaux sont positifs. Nous pouvons, dans ce cas, introduire la matrice $D^{\frac{1}{2}}$,

$$A = D^{\frac{1}{2}}(I - D^{-\frac{1}{2}}(L + L^T)D^{-\frac{1}{2}})D^{\frac{1}{2}}$$

et donc

$$A^{-1} = D^{-\frac{1}{2}}(I - D^{-\frac{1}{2}}(L + L^T)D^{-\frac{1}{2}})^{-1}D^{-\frac{1}{2}}$$

or

$$D^{-\frac{1}{2}}(L + L^T)D^{-\frac{1}{2}} = D^{-\frac{1}{2}}(D - A)D^{-\frac{1}{2}} = I - D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$$

et

$$\rho(D^{-\frac{1}{2}}(L + L^T)D^{-\frac{1}{2}}) = \rho(I - D^{-\frac{1}{2}}AD^{-\frac{1}{2}}) = \rho(I - D^{-1}A)$$

Utilisons le Lemme suivant :

Lemme III.1 *Soit A une matrice symétrique définie positive à diagonale dominante et soit $D = diag_N(a_i)$ alors*

$$\rho(I - D^{-1}A) < 1.$$

Preuve III.1 *Il suffit de montrer que les valeurs propres de la matrice $D^{-1}A$ sont comprises strictement entre 0 et 2.*

Or, A est définie positive donc $D^{-1}A$ aussi. Ainsi, les valeurs propres de $D^{-1}A$ sont strictement positives.

De plus, la matrice A est à diagonale dominante, donc $D^{-1}A$ aussi. Comme la diagonale de $D^{-1}A$ est constituée de 1, d'après le Théorème de Gershgörin les valeurs propres sont strictement[2] *inférieures à 2.*

Une autre démonstration de ce Lemme est proposée dans [Varg62], qui propose d'ailleurs une estimation plus fine des valeurs propres.

Maintenant en utilisant $\rho(I - D^{-1}A) < 1$, nous savons que dans ce cas la série de Neumann pour l'inverse de $I - D^{-\frac{1}{2}}(L+L^T)D^{-\frac{1}{2}}$ converge. Et donc l'approximation la plus simple consiste à prendre, tout comme (III.2), deux termes de cette serie de Neumann. Nous posons donc :

$$M^{-1} = D^{-\frac{1}{2}}(I + D^{-\frac{1}{2}}(L + L^T)D^{-\frac{1}{2}})D^{-\frac{1}{2}} = D^{-1} + D^{-1}(L + L^T)D^{-1}.$$

Cette méthode (NEUM1) est éminemment parallèle, son degré de parallélisme est N, chacune des multiplications matrice multidiagonale vecteur du calcul de $M^{-1}r$ pouvant s'effectuer de façon parallèle.

Remarque III.3 *Dans le chapitre V, nous noterons NEUM3 l'approximation qui consiste à utiliser les quatres premiers termes de la série de Neumann. Nous ne donnerons aucun résultat sur ce que l'on pourrait noter NEUM2, car il a été prouvé dans [DuGR79] et nous l'avons vérifié dans les expériences numériques que NEUM1 est plus efficace, voir également [Tong89].*

Remarque III.4 *On aurait pu inclure le préconditionnement DIAG dans la catégorie des préconditionnements polynômiaux, il s'agit en effet de NEUM0. Mais nous avons préféré lui consacrer un paragraphe entier pour bien marquer la référence qu'il constitue sur la CM-2.*

[2] L'inégalité est stricte car non seulement A est à diagonale dominante mais de plus il existe des lignes (et des colonnes) où cette dominance est stricte. Ainsi la plus grande valeur propre de $D^{-1}A$ ne peut atteindre la valeur 2.

III.5.3 Séries de Neumann tronquées sur une factorisation de Cholesky incomplète

Nous considérons tout d'abord une factorisation incomplète de Cholesky standard avec la même structure que la matrice A,

$$\mathcal{L}\mathcal{D}^{-1}\mathcal{L}^T \tag{III.3}$$

Notons $\overline{d_i}$ les éléments non nuls de la matrice diagonale $\mathcal{D}$ et $(\overline{c_i}, \overline{b_i}, \overline{d_i})$ les éléments non nuls de la i^{eme} ligne de $\mathcal{L}$. Alors

$$\begin{cases} \overline{c_i} &= c_i \\ \overline{b_i} &= b_i \\ \overline{d_i} &= d_i - \dfrac{(b_i)^2}{\overline{d}_{i-1}} - \dfrac{(c_i)^2}{\overline{d}_{i-n}} \end{cases} \tag{III.4}$$

avec, bien entendu, $A = penta_N(c_i, b_i, d_i, b_{i+1}, c_{i+n})$
Ensuite, nous cherchons une approximation de

$$\mathcal{L}^{-T}\mathcal{D}\mathcal{L}^{-1}$$

notons $\mathcal{L} = \mathcal{D} - \overline{\mathcal{L}}$. Comme précédemment nous utilisons les séries de Neumann, et nous avons :

$$\mathcal{L}^{-1} \approx \mathcal{D}^{-1} + \mathcal{D}^{-1}L\mathcal{D}^{-1}$$

Nous posons donc comme préconditionnement :

$$M^{-1} = (\mathcal{D}^{-1} + \mathcal{D}^{-1}L^T\mathcal{D}^{-1})\mathcal{D}(\mathcal{D}^{-1} + \mathcal{D}^{-1}L\mathcal{D}^{-1})$$

Il est clair que cette méthode (ICNEUM) est parallèle de degré N, chaque produit matrice vecteur se faisant en parallèle.

III.5.4 Le préconditionnement Minmax

Nous allons maintenant considérer une catégorie de préconditionnements polynômiaux plus élaborés que les précédents. Dans ce paragraphe, nous nous sommes intéressés à des polynômes minimisant une généralisation du nombre de conditionnement : le polynôme Minmax, voir [JoMP83].

Définition III.2 *Soit* $[\lambda_{min}\lambda_{max},] = [a,b]$ *un intervalle contenant les valeurs propres de* A, $0 \notin [a,b]$, *alors on note*

$$\mathcal{Q}_k = \{\textit{polynômes } q_k \textit{ de degré } \leq k | \forall \lambda \in [a,b], q_k(\lambda) > 0,$$

$$\textit{et } q_k(0) = 0\}.$$

Définition III.3 *Pour tout polynôme* $q \in \mathcal{Q}_{k+1}$ *on note*

$$cond(q) = \frac{sup_{\lambda \in [a,b]} q(\lambda)}{inf_{\lambda \in [a,b]} q(\lambda)}.$$

Alors nous cherchons q_{k+1} solution du problème suivant :

$$\begin{cases} Trouver\ q_{k+1} \in \mathcal{Q}_{k+1}\ tel\ que \\ \forall q \in \mathcal{Q}_{k+1}, cond(q_{k+1}) \leq cond(q) \end{cases} . \tag{III.5}$$

Pour nous ramener à l'intervalle $[-1,+1]$ nous posons :

$$\mu(\lambda) = \frac{2\lambda - b - a}{b - a}.$$

Alors la solution du problème (III.5) est donnée par exemple par Johnson, Michelli et Paul dans [JoMP83] :

$$q_{k+1}(\lambda) = 1 - \frac{T_{k+1}(\mu(\lambda))}{T_{k+1}(\mu(0))}. \tag{III.6}$$

où T_k est le polynôme de Tchebycheff de première espèce d'ordre k, voir [Davi75].

Nous obtenons ainsi p_k le polynôme Minmax avec :

$$p_k(\lambda) = \frac{1}{\lambda}\left(1 - \frac{T_{k+1}(\mu(\lambda))}{T_{k+1}(\mu(0))}\right). \tag{III.7}$$

Remarque III.5 *Pour les résultats numériques, présentés au chapitre suivant, nous avons d'abord calculé les coefficients* α_i *du polynôme* p_k *à partir des formules de récurrence des polynômes de Tchebycheff, puis nous avons résolu l'étape de préconditionnement grâce au schéma de Hörner (III.1). Nous noterons HMINMAX cette méthode.*

III.5.5 Les préconditionnements Mcarre et Leg

Une autre façon, d'obtenir des préconditionnements polynômiaux, est de dire que le polynôme p_k doit être dans un certain sens une approximation de la fonction $(\lambda \to 1/\lambda)$. C'est à dire $1 - \lambda p_k(\lambda)$ doit être proche de 0. Comme [JoMP83] et [Saad85] nous allons considérer les polynômes minimisant la norme quadratique suivante :

$$\int_a^b (1 - \lambda p(\lambda))^2 \omega(\lambda) d\lambda, \tag{III.8}$$

avec $\omega(\lambda)$ un poids capable d'accentuer certaines parties importantes du spectre. Pratiquement, nous avons pris des poids de Jacobi :

$$\omega(\alpha, \beta, \lambda) = (b - \lambda)^\alpha (\lambda - a)^\beta.$$

Notons $s_i(\lambda)$ les polynômes orthonormaux associés au poids $\omega(\lambda)$. Alors la solution du problème (III.8) est donnée par, voir [JoMP83] :

$$p_k(\lambda) = \sum_{j=0}^{k+1} b_j t_j(\lambda),$$

avec

$$b_j = \frac{s_j(0)}{\sum_{i=0}^{k+1} s_i^2(0)} \text{ et } t_j(\lambda) = \frac{s_j(0) - s_j(\lambda)}{\lambda}. \tag{III.9}$$

Enfin, nous avons pris, dans un premier temps, le poids de Tchebycheff $\alpha = \beta = -\frac{1}{2}$, puis le poids de Legendre $\alpha = \beta = 0$. Nous avons ainsi utilisé deux préconditionnements différents, associés au schéma de Hörner, notés respectivement HMCARRE et HLEG.

III.6 Les approximations de l'inverse exacte

Nous venons de décrire des préconditionnements qui semblent bien adaptés à l'architecture massivement parallèle de la CM-2. En outre, nous nous sommes beaucoup servis du fait qu'il est plus facile d'effectuer l'opération $z = M^{-1}r$ que de résoudre $Mz = r$ sur la CM-2. Néanmoins

on peut se demander si ces préconditionnements sont des préconditionnements efficaces au niveau du taux de convergence par rapport aux contraintes qu'ils respectent. En effet, à conditions égales (uniquement des commmunications entre un nœud et ses voisins immédiats) on peut se demander si d'autres approximations de l'inverse ne sont pas plus efficaces. A titre de comparaison, nous allons donc nous intéresser à des approximations directes de l'inverse, mais aussi à des expressions analytiques par blocs de l'inverse, puis à des factorisations incomplètes par blocs, et finalement à des préconditionnements locaux et pour généraliser à des préconditionnements basés sur des compléments de Schur.

III.6.1 Préconditionnement LAP

La méthode (LAP), que nous allons étudier, bien qu'elle ne soit pas très réaliste, consiste à prendre des éléments de l'inverse exact pour constituer M^{-1}.

Définition III.4 *On appelle squelette de la matrice A, l'ensemble d'indices $\mathcal{S}$ tel que $\mathcal{S} = \{(i,j) \ / \ 1 \leq i,j \leq N \ , \ A_{ij} \neq 0\}$.*

Posons

$$M^{-1}_{ij} = \begin{cases} A^{-1}_{ij} & si\ (i,j) \in \mathcal{S} \\ 0 & sinon \end{cases}$$

Il est évident que nous ne disposons pas de moyens simples et économiques de calculer ainsi certains éléments de l'inverse. Néanmoins sur le plan théorique, il est intéressant de voir ce que cela donne, puisqu'il a été montré dans [CoGM85] et [Kers70], pour des matrices tridiagonales, qu'une méthode analogue fournissait, dans certains cas, une bonne approximation de l'inverse. De plus, pour les problèmes qui nous intéressent, les éléments de A^{-1}_{ij} décroissent rapidement lorsque nous nous éloignons de la diagonale et des sous-diagonales (ie lorsque nous nous éloignons des diagonales non nulles de A), ou encore

$$A^{-1}_{(i,j)\notin\mathcal{S}} \leq A^{-1}_{(i,j)\in\mathcal{S}}$$

Toutefois, l'approximation pose dans notre cas un problème, dans la mesure où la matrice M^{-1} ainsi générée n'est pas nécessairement définie positive. En effet, A^{-1} est définie positive, mais lorsque nous annulons plusieurs termes d'une matrice définie positive, le résultat ne reste pas obligatoirement défini positif.

Une solution à ce problème consiste à renforcer la dominance de la diagonale de la matrice de départ.

Soit $\mathcal{N}$ la matrice diagonale définie par :

$$\mathcal{N}_{ii} = \sum_{j \neq i} \mid a_{ij} \mid$$

posons alors :

$$M_{ij}^{-1} = \begin{cases} (A + \theta \mathcal{N})_{ij}^{-1} & si\ (i,j) \in \mathcal{S} \\ 0 & sinon \end{cases}$$

avec θ un paramètre réel positif à déterminer, en pratique nous ferons varier θ entre 0 et 1 avec un pas de 0.1.

Remarque III.6 *Cette technique peut paraître inutile car trop peu réaliste (pour une seule résolution) mais aussi pas du tout parallèle lors de sa phase d'initialisation. Son but n'est pas du tout d'être efficace en temps, mais d'apporter une première pierre instructive pour la comparaison des nombres d'itérations entre les préconditionnements.*

III.6.2 Préconditionnement LIP

Maintenant pour obtenir un préconditionnement défini positif, nous pouvons partir d'une décomposition incomplète de Cholesky introduite en (III.3) :

$$\mathcal{L}\mathcal{D}^{-1}\mathcal{L}^T$$

avec les formules correspondantes (III.4).

Soit $\mathcal{S}_L = \{(i,j)\ /i \geq j\ et\ (i,j) \in \mathcal{S}\}$.

Posons

$$\mathcal{M}_{ij}^{-1} = \begin{cases} \mathcal{L}_{ij}^{-1} & si\ (i,j) \in \mathcal{S}_L \\ O & sinon \end{cases}$$

et

$$M^{-1} = \mathcal{M}^{-T}\mathcal{D}\mathcal{M}^{-1}.$$

Nous avons ainsi une matrice M^{-1} définie positive. De même que précédemment cette méthode (LIP) n'est pas utilisable dans la pratique. Le coût du calcul des éléments de l'inverse est bien évidemment trop important. Pour obtenir un rendement suffisant de ce type de méthodes, il faudrait par exemple résoudre plusieurs problèmes mais en conservant la même matrice. Ainsi, le surcoût d'initialisation serait quelque peu atténué par le nombre de résolution effectuées à l'aide de ce préconditionnement.

III.7 Expression analytique par blocs de l'inverse de A

En procédant de la même manière que dans [CoGM85], nous pouvons décrire une expression analytique de l'inverse, sous forme de blocs.

Nous définissons deux suites de matrices d'ordre n : Δ_i *et* Σ_i par

$$\begin{cases} \Delta_1 &= D_1 \\ \Delta_i &= D_i - A_i(\Delta_{i-1})^{-1}A_i^T \end{cases} \tag{III.10}$$

et

$$\begin{cases} \Sigma_n &= D_n \\ \Sigma_i &= D_i - A_{i+1}^T(\Sigma_{i+1})^{-1}A_{i+1} \end{cases} \tag{III.11}$$

A_i et D_i étant définies en (I.10).

Ensuite nous définissons Φ_i matrice d'ordre n par :

$$\Phi_i = D_i - A_i(\Delta_{i-1})^{-1}A_i^T - A_{i+1}^T(\Sigma_{i+1})^{-1}A_{i+1}. \tag{III.12}$$

Nota Lorsque les indices $i-1$ ou $i+1$ ne sont pas compris entre 1 et n, les termes correspondants sont pris nuls.

Dans ce cas, en notant L la matrice strictement triangulaire par blocs suivante

$$L = \begin{pmatrix} 0 & & & & \\ -A_2 & 0 & & & \\ & \ddots & \ddots & \ddots & \\ & & -A_{n-1} & 0 & \\ & & & -A_n & 0 \end{pmatrix},$$

nous pouvons écrire la factorisation de Cholesky de la manière suivante, voir [CoGM85] :

$$A = (\Sigma + L)\Sigma^{-1}(\Sigma + L^T).$$

Si nous considérons l'inverse A^{-1} par blocs, en notant $\{A^{-1}\}_{i,j}$ le bloc d'indice (i,j) de l'inverse, nous avons :

$$\begin{aligned} \{A^{-1}\}_{i,i} &= \Phi_i^{-1} \\ \{A^{-1}\}_{i,i-1} &= \Delta_{i-1}^{-1} A_i^T \Phi_i^{-1} \\ \{A^{-1}\}_{i,i+1} &= \Sigma_{i+1}^{-1} A_{i+1} \Phi_i^{-1}. \end{aligned}$$

Ensuite nous pouvons définir un inverse approché des matrices Δ_i et Σ_i par des matrices tridiagonales. Alors nous remplaçons (III.10) et (III.11) par :

$$\begin{cases} \Delta_1 &= D_1 \\ \Delta_i &= D_i - A_i trid_n((\Delta_{i-1})^{-1}) A_i^T \end{cases} \tag{III.13}$$

et

$$\begin{cases} \Sigma_n &= D_n \\ \Sigma_i &= D_i - A_{i+1}^T trid_n((\Sigma_{i+1})^{-1}) A_{i+1} \end{cases} \tag{III.14}$$

De plus, nous définissons Φ_i matrice d'ordre n par :

$$\Phi_i = D_i - A_i trid_n((\Delta_{i-1})^{-1}) A_i^T - A_{i+1}^T trid_n((\Sigma_{i+1})^{-1}) A_{i+1}. \tag{III.15}$$

Les matrices Δ_i, Σ_i et Φ_i sont tridiagonales et les approximations peuvent être facilement calculées.

Nous allons maintenant proposer deux types de méthodes, la première possède le même squelette (ou structure creuse) que la matrice initiale A. Tandis que la seconde possède plus de renseignements, et donc plus de dépendance de données soit moins de parallélisme.

III.7.1 Préconditionnement LAPINC

Si nous désirons un inverse approché de même squelette que A, nous posons :

$$\begin{aligned} \{M^{-1}\}_{i,i} &= trid_n(\Phi_i^{-1}) \\ \{M^{-1}\}_{i,i-1} &= diag_n(trid_n(\Delta_{i-1}^{-1})A_i^T trid_n(\Phi_i^{-1})) \\ \{M^{-1}\}_{i,i+1} &= \{M^{-1}\}_{i+1,i} \\ \{M^{-1}\}_{i,j} &= 0 \; sinon. \end{aligned}$$

Pour les mêmes raisons que la méthode LAP, ce préconditionnement (LAPINC) n'est pas non plus nécessairement défini positif, nous l'appliquerons donc, comme précédemment, à $A+\theta\mathcal{N}$ au lieu de A. Par contre, cette technique possède un degré de parallélisme de n.

III.7.2 Préconditionnement LAPINC1

Le préconditionnement précédant a le même squelette que A (c'est à dire uniquement des communications avec les nœuds voisins). Si nous sommes prêts à effectuer plus de travail (avec moins de parallélisme) nous pouvons définir la méthode suivante :

$$\begin{aligned} \{M^{-1}\}_{i,i} &= \Phi_i^{-1} \\ \{M^{-1}\}_{i,i-1} &= \Delta_{i-1}^{-1} A_i^T \Phi_i^{-1} \\ \{M^{-1}\}_{i,i+1} &= \{M^{-1}\}_{i+1,i} \\ \{M^{-1}\}_{i,j} &= 0 \; sinon. \end{aligned}$$

Apparemment, il faut calculer entièrement Φ_i^{-1} et Δ_i^{-1}, mais en réalité, ce qui nous intéresse, c'est $z = M^{-1}r$. Or avec les notations par blocs évidentes, nous avons :

$$z_i = \{M^{-1}\}_{i,i-1} r_{i-1} + \{M^{-1}\}_{i,i} r_i + \{M^{-1}\}_{i,i+1} r_{i+1}.$$

Soient $y_i = \Phi_i^{-1} r_i$ et $\omega_i = \Phi_i^{-1} r_{i-1}$ pour $i = 1, \ldots, n$, obtenus par $2n$ résolution de systèmes tridiagonaux,

alors

$$z_i = \Delta_{i-1}^{-1} A_i^T \omega_i + y_i + \Delta_i^{-1} A_{i+1}^T y_{i+1}.$$

Les premier et troisième termes sont calculés par résolution de systèmes tridiagonaux. Tous ces systèmes peuvent être résolus en parallèle, toutefois le degré de parallélisme n'est que de n. Enfin, cette approximation (LAPINC1) n'est pas non plus définie positive.

III.8 Approximation de l'inverse à l'aide d'une décomposition incomplète par blocs

Les deux préconditionnements précédents ont le défaut de ne pas être définis positifs, donc d'être des approximations de $A + \theta\mathcal{N}$ au lieu de A. Une autre façon d'obtenir une approximation de l'inverse est de considérer une décomposition incomplète par blocs, dont les blocs diagonaux sont les Δ_i.

Alors la décomposition incomplète par blocs est définie, comme précédemment pour la suite Σ_i, par :

$$(\Delta + L)\Delta^{-1}(\Delta + L^T),$$

Δ étant définie par (III.12).

Ensuite pour développer une approximation de l'inverse, regardons

$$(\Delta + L)^{-1}.$$

Il est clair que les blocs de la diagonale de cette matrice sont composés des Δ_i^{-1}, et la sous-diagonale par blocs est constituée par les éléments $\Delta_i^{-1} A_i \Delta_{i-1}^{-1}$. A partir de ces éléments nous allons pouvoir définir deux préconditionnements.

III.8.1 Préconditionnement LAPINV

Soit $\mathcal{M}^{-1}$ la matrice dont les éléments par blocs sont :

$$\begin{aligned} \{\mathcal{M}^{-1}\}_{i,i} &= trid_n(\Delta_i^{-1}) \\ \{\mathcal{M}^{-1}\}_{i,i-1} &= diag_n(trid_n(\Delta_i^{-1})A_i^T trid_n(\Delta_{i-1}^{-1})) \\ \{\mathcal{M}^{-1}\}_{i,j} &= 0 \; sinon. \end{aligned}$$

Nous posons ensuite :

$$M^{-1} = \mathcal{M}^{-T}\Delta\mathcal{M}^{-1}.$$

Par rapport aux précédentes, cette méthode (LAPINV) a l'avantage d'être définie positive, mais son degré de parallélisme n'est aussi que de n.

III.8.2 Préconditionnement LAPINV1

Comme précédemment, si nous acceptons de résoudre des systèmes tridiagonaux, nous pouvons poser (LAPINV1) :

$$\begin{aligned} \{\mathcal{M}^{-1}\}_{i,i} &= \Delta_i^{-1} \\ \{\mathcal{M}^{-1}\}_{i,i-1} &= \Delta_i^{-1}A_i^T\Delta_{i-1}^{-1} \\ \{\mathcal{M}^{-1}\}_{i,j} &= 0 \; sinon. \end{aligned}$$

Ensuite lors du calcul de $\mathcal{M}^{-1}r$, nous avons :

$$x_i = \{\mathcal{M}^{-1}\}_{i-1,i}r_{i-1} + \{\mathcal{M}^{-1}\}_{i,i}r_i,$$

et soit y_i donné par : $\Delta_i y_i = r_i$,
alors

$$x_i = \Delta_i^{-1}A_i^T y_{i-1} + y_i.$$

Le premier terme du second membre est calculé en résolvant des systèmes tridiagonaux. Enfin nous utilisons des formules similaires pour obtenir $\mathcal{M}^{-T}\Delta x$.

Remarque III.7 *Ces méthodes à base de factorisations par blocs, LAPINC, LAPINC1, LAPINV et LAPINV1 ont certes un degré de parallélisme insuffisant pour la CM-2, mais elles semblent bien adaptées à des calculateurs de type MIMD, voir remarque II.15. Une étude est donc à envisager.*

III.9 Les préconditionnements locaux

Dans cette partie nous allons étudier des préconditionnements basés sur le principe suivant : la solution du problème

$$Mz = r$$

est obtenue par les étapes suivantes : soit $\tilde{r}_i$ le second membre obtenu en annulant toutes les composantes de r sauf la i^{eme}. Notons aussi, $y|_{S_i}$ le vecteur de dimension n_i formé par les composantes de y dont les indices appartiennent à un ensemble S_i (correspondant à des nœuds voisins du nœud i sur le maillage).

Nous résolvons le problème

$$M_i z|_{S_i} = \tilde{r}_i|_{S_i},$$

ensuite en prolongeant $z|_{S_i}$ par zéro, nous obtenons $\tilde{z}_i$, et nous prenons

$$z = \sum_i \tilde{z}_i.$$

Ainsi pour chaque point du maillage , nous résolvons des problèmes locaux, sur un ensemble de points correspondant à S_i. Et enfin nous sommons les solutions obtenues.

Maintenant, avec chaque ensemble S_i choisi, nous obtenons une méthode différente avec des opérateurs locaux M_i différents.

Pour obtenir la solution locale, il faut évidemment expliciter M_i ou du moins calculer une certaine partie de son inverse, car à cause de la structure creuse de $\tilde{r}_i$, nous n'avons pas besoin de tout l'inverse de M_i. Une façon simple de définir M_i est de prendre la restriction de l'opérateur A.

L'ensemble S_i devant rester un ensemble de faible cardinalité, nous prendrons par exemple des ensembles comme les schémas à 5 et 9 points classiques.

Commençons par le schéma à 5 points, et numérotons les nœuds de façon *naturelle*.

La matrice M_i d'ordre 5 est donc :

$$M_i = \begin{pmatrix} a_{i-n} & 0 & c_i & 0 & 0 \\ 0 & a_{i-1} & b_i & 0 & 0 \\ c_i & b_i & a_i & b_{i+1} & c_{i+n} \\ 0 & 0 & b_{i+1} & a_{i+1} & 0 \\ 0 & 0 & c_{i+n} & 0 & a_{i+n} \end{pmatrix}.$$

Pour calculer la solution avec un second membre qui n'a qu'une seule composante non nulle (correspondant au nœud 3), nous n'avons besoin que de la troisième colonne de M_i^{-1}. Par exemple nous allons l'obtenir grâce à une décomposition de la forme suivante :

$$M_i = L_i D_i L_i^T,$$

avec

$$L_i = \begin{pmatrix} a_{i-n} & 0 & 0 & 0 & 0 \\ 0 & a_{i-1} & 0 & 0 & 0 \\ c_i & b_i & a_i & b_{i+1} & c_{i+n} \\ 0 & 0 & 0 & a_{i+1} & 0 \\ 0 & 0 & 0 & 0 & a_{i+n} \end{pmatrix},$$

et

$$D_i = diag_5(L_i),$$

en posant

$$d_i = a_i - \frac{c_i^2}{a_{i-n}} - \frac{b_i^2}{a_{i-1}} - \frac{b_{i+1}^2}{a_{i+1}} - \frac{c_{i+n}^2}{a_{i+n}}.$$

Alors les composantes de la troisième colonne sont données par

$$\left(-\frac{c_i}{a_{i-n}d_i}, -\frac{b_i}{a_{i-1}d_i}, \frac{1}{d_i}, -\frac{b_{i+1}}{a_{i+1}d_i}, -\frac{b_{i+n}}{a_{i+n}d_i} \right)^T.$$

Les solutions locales sont ensuite des multiples de cette troisième colonne.

Remarque III.8 *Par exemple, pour la discrétisation du problème 1 avec le schéma à 5 points nous obtenons :*

$$z_i|_{S_i} = \frac{r_i}{12} \begin{pmatrix} 1 \\ 1 \\ 4 \\ 1 \\ 1 \end{pmatrix}.$$

Cette méthode se généralise facilement aux autres ensembles S_i. Nous avons mis en œuvre cette approximation pour les ensembles à 5 points (LOC5) et à 9 points (LOC9).

III.10 Les préconditionnements basés sur des compléments de Schur

Une façon naturelle d'obtenir des approximations dans le cadre précédent, est de considérer le complément de Schur.

L'idée générale est la suivante : renumérotons les inconnues avec d'abord les inconnues n'appartenant pas à S_i, puis ceux de S_i. Le système (I.9) se récrit :

$$\begin{pmatrix} B_{11} & B_{12} \\ B_{12}^T & B_{22} \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} = r_i = \begin{pmatrix} 0 \\ \tilde{r} \end{pmatrix}. \tag{III.16}$$

Nous éliminons l'inconnue z_1 et nous obtenons :

$$(B_{22} - B_{12}^T B_{11}^{-1} B_{12}) z_2 = \tilde{r}. \tag{III.17}$$

La matrice de ce système est appelée complément de Schur de B_{22}. B_{11}^{-1} est une matrice pleine, dans notre cas, il n'est donc pas envisageable de calculer le complément de Schur. Nous allons essayer d'en trouver une approximation.

Décrivons notre méthode dans le cas où l'ensemble S_i représente le schéma à 9 points (SCHUR2). Nous éliminons d'abord les inconnues extérieures aux 9 points. Nous allons obtenir cela en deux étapes, tout d'abord nous considérons les trois lignes du maillage auxquelles

appartiennent les 9 points (ensemble 3). En renumérotant les inconnues dans des ensembles 1, 2 et 3 le système (III.16) se reécrit :

$$\begin{pmatrix} A_{11} & 0 & A_{13} \\ 0 & A_{22} & A_{23} \\ A_{12}^T & A_{23}^T & A_{33} \end{pmatrix} \begin{pmatrix} z'_1 \\ z'_2 \\ z'_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \tilde{r}_3 \end{pmatrix}.$$

Nous éliminons z'_1 et z'_2, nous obtenons :

$$S_{33}(i)z'_3 = (A_{33} - A_{13}^T A_{11}^{-1} A_{13} - A_{23}^T A_{22}^{-1} A_{23})z'_3 = \tilde{r}_3.$$

En principe, le complément de Schur est une matrice pleine, mais dans ce cas, les matrices A_{13} et A_{23} sont très creuses, et nous avons même :

$$S_{33}(i) = \begin{pmatrix} \Delta_{i-1} & -A_i^T & 0 \\ -A_i & D_i & -A_{i+1}^T \\ 0 & -A_{i+1} & \Sigma_{i+1} \end{pmatrix}$$

où Δ_i et Σ_i sont définies de la même manière que dans (III.10) et (III.11). Comme précédemment nous remplaçons Δ_i et Σ_i par leurs approximations tridiagonales définies par (III.13) et (III.14).

Avec ces approximations et une renumérotation, la matrice $S_{33}(i)$ est tridiagonale par blocs et d'ordre $3n$,

$$S_{33}(i) = \begin{pmatrix} G_1 & -F_2^T & & & \\ -F_2 & G_2 & -F_3^T & & \\ & \ddots & \ddots & \ddots & \\ & & -F_{n-1} & G_{n-1} & -F_n^T \\ & & & -F_n & G_n \end{pmatrix}.$$

Nous appliquons encore le principe de factorisation par blocs à cette matrice. Soit

$$\begin{cases} \Gamma_1 &= G_1 \\ \Gamma_i &= G_i - F_i(\Gamma_{i-1})^{-1} F_i^T \end{cases}$$

et

$$\begin{cases} \Lambda_n &= G_n \\ \Lambda_i &= G_i - F_{i+1}^T(\Lambda_{i+1})^{-1} F_{i+1} \end{cases}.$$

Ici, les Γ_i et Λ_i sont des matrices d'ordre trois, il est donc facile de calculer leurs inverses exacts.

Ensuite, nous ne gardons que les trois colonnes entourant le point i et nous arrivons finalement à une matrice d'ordre 9 :

$$\begin{pmatrix} \Gamma_{i-1} & -F_i^T & 0 \\ -F_i & G_i & -F_{i+1}^T \\ 0 & -F_{i+1} & \Gamma_{i+1} \end{pmatrix},$$

et le second membre étant $(0,0,0,0,r_i,0,0,0,0)^T$, nous ne nous intéressons qu'à la 5^{ieme} colonne de l'inverse. Nous pouvons la calculer de la même manière que précédemment, posons

$$\Omega_i = G_i - F_i\Gamma_{i-1}^{-1}F_i^T - F_{i+1}^T\Lambda_{i+1}^{-1}F_{i+1}.$$

Alors la troisième colonne par blocs est :

$$(\Gamma_{i-1}^{-1}F_i^T\Omega_i^{-1} \quad \Omega_i^{-1} \quad \Lambda_{i+1}^{-1}F_{i+1}\Omega_i^{-1})^T.$$

Nous ne calculons ainsi que des inverses de matrices d'ordre trois. Il est clair que cette méthode peut être facilement généralisée à des stencils 5×5 (SCHUR1) ou 7×7, nous calculons alors des inverses de matrices d'ordre 5 ou d'ordre 7.

Tous les préconditionnements étant maintenant explicités, nous allons pouvoir passer aux premiers résultats numériques.

CHAPITRE IV

ÉTUDE PRÉLIMINAIRE SUR ORDINATEUR SÉQUENTIEL

IV.1 Présentation

Ce chapitre présente les premiers résultats numériques sur les préconditionnements décrits au chapitre précédent. Afin de vérifier l'efficacité, en ce qui concerne le taux de convergence, des préconditionnements adaptés à l'architecture massivement parallèle de la CM-2, nous les avons étudiés sur une machine séquentielle pour pouvoir les comparer avec d'autres méthodes non nécessairement parallèles.

La majorité des calculs a été effectuée en réels simple précision (32 bits) sur un Apollo série 400. Le critère d'arrêt du G.C.P est $\epsilon = 10^{-6}$, ce qui semble raisonnable pour des réels simple précision. Le maillage est un maillage carré uniforme 40×40. Donc les matrices sont d'ordre 1600. Le problème effectivement mis en œuvre est le second problème, voir chapitre I, car il nécessite, à cause des sauts importants de ses coefficients, un nombre d'itérations beaucoup plus élevé que le problème 1. Ainsi, les différences de nombre d'itérations pour chaque préconditionnement sont plus grandes et bien entendu plus significatives, qu'avec le premier problème. Comme l'objectif de cette thèse est de résoudre des grands systèmes linéaires, donc de résoudre des systèmes ayant souvent besoin d'un grand nombre d'itérations pour converger, il est, sans aucun doute, plus utile de réaliser l'étude séquentielle sur un problème mal conditionné.

Par conséquent, nous allons d'abord présenter le nombre d'itérations du G.C.P avec les différents préconditionnements. Puis nous exposerons les temps de calcul pour quelques techniques, bien qu'ils ne nous apportent que peu d'informations sur leur efficacité réelle sur la CM-2.

IV.2 Nombre d'itérations

Nous constatons, en premier lieu, dans le tableau IV.1 que le nombre d'itérations de la méthode DIAG est relativement élevé (868) par rapport à l'ordre de la matrice (1600). Néanmoins, les préconditionnements à base de renumérotation Rouge-Noir parviennent à diminuer ce nombre d'itérations d'un facteur deux, ce qui est encourageant. On note, par ailleurs, que RBIC est plus efficace que RBSSOR, cela semble naturel d'après leurs définitions. De plus, le préconditionnement NEUM1 a, à peu de chose près, le même taux de convergence que RBIC et RBSSOR, alors que son degré de parallélisme est N, au lieu de $N/2$. Ce résultat paraît très positif. Toutefois, il faut espérer que ce gain d'itérations sera suffisant par rapport à l'augmentation du nombre de calculs à chaque itération (environ deux produits matrice vecteur au lieu d'un seul pour le préconditionnement Diagonal). Enfin, le nombre d'itérations de ICNEUM est équivalent à celui de DIAG. Comme à chaque itération ICNEUM effectue plus de calculs que DIAG, et puisque DIAG est éminemment parallèle, ICNEUM ne pourra que se révéler moins efficace que DIAG sur la CM-2. Ces premiers résultats nous montrent ainsi qu'il est inutile de tester la méthode ICNEUM sur la CM-2. Par contre, nous pouvons espérer atteindre de bonnes performances avec les autres techniques.

Préconditionnement utilisé	Nombre d'itérations
DIAG	868
RBIC	428
RBSSOR	437
NEUM	461
ICNEUM	858

Tableau IV.1: Nombre d'itérations pour le problème 2 avec un maillage 40×40.

Remarque IV.1 *Lors de la mise en œuvre des préconditionnements polynômiaux, nous avons besoin de connaitre la plus petite et la plus grande valeur propre (a et b) de la matrice, ou au moins une estimation. Ne possédant pas d'information sur la matrice du problème 2, nous avons ainsi normalisé symétriquement la matrice par sa diagonale. C'est à dire que au lieu de résoudre le problème (I.9), nous avons résolu, avec* $D = diag_N(a_i)$,

$$D^{-\frac{1}{2}} A D^{-\frac{1}{2}} y = D^{-1/2} b, \text{ puis } D^{1/2} x = y. \qquad \text{(IV.1)}$$

Et donc d'après le Lemme (III.1), nous savons que pour ce problème (IV.1) $b < 2$. *Ainsi nous avons pris comme approximation de la plus grande valeur propre* $b = 2$. *Enfin, pour la plus petite, sachant qu'elle est strictement positive et après une longue étude numérique (voir en particulier le chapitre VII), nous nous sommes apercus que la valeur* $a = 0.25\ 10^{-3}$ *permettait une utilisation satisfaisante des préconditionnements polynômiaux. Cette valeur n'est pas la valeur optimale pour les deux problèmes proposés. Néanmoins elle est suffisament représentative de l'efficacité des préconditionnements polynômiaux.*

Remarque IV.2 *Nous n'avons pas pris les valeurs propres exactes de la matrice du problème 2. Tout d'abord, contrairement au premier problème nous ne connaissons pas leur expression analytique. Tout de même, nous pouvions obtenir une estimation précise grâce à une méthode de recherche de valeurs propres. Mais le coût de ces techniques est relativement élevé par rapport à la résolution. Ainsi, nous nous sommes placés dans des conditions très réalistes d'utilisation des préconditionnements polynômiaux. Il est en effet très rare que l'utilisateur connaisse les valeurs propres de la matrice ou même qu'il essaie d'en obtenir une estimation.*

Nous présentons, dans les tableaux (IV.2) à (IV.4), les variations, en fonction du degré, du nombre d'itérations du G.C.P, munit des différents préconditionnements polynômiaux. Dans la suite, nous noterons *non déf. pos.* si le coefficient α_k du G.C.P (voir l'algorithme au chapitre II) devient inférieur ou égal à zéro, en d'autres termes la matrice de préconditionnement n'est plus, dans ce cas, définie positive.

Degré du polynôme HMINMAX	Nombre d'itérations
2	385
3	306
4	248
5	213
6	195
7	170
8	*non déf. pos.*

Tableau IV.2: Nombre d'itérations de HMINMAX pour le problème 2 avec un maillage 40×40.

Degré du polynôme HMCARRE	Nombre d'itérations
2	310
3	246
4	203
5	172
6	152
7	135
8	122
9	113
10	161

Degré du polynôme HLEG	Nombre d'itérations
2	332
3	254
4	209
5	178
6	156
7	137
8	128
9	118
10	153

Tableau IV.3 et Tableau IV.4: Nombre d'itérations de HMCARRE et HLEG pour le problème 2 avec un maillage 40×40.

On remarque, en premier lieu, que le nombre d'itérations est une fonction décroissante du degré du polynôme (pour des degrés pas trop élevés). En effet plus le degré du polynôme est grand plus l'estimation de l'inverse ainsi obtenue est précise. De façon similaire, on peut vérifier que si on assemble les matrices $P_k(A)$, avec P_k défini au chapitre III, leurs squelettes augmentent avec le degré du polynôme. Et donc,

l'étendue de nos informations sur la matrice inverse est une fonction croissante du degré du polynôme.

Heureusement, grâce au schéma de Hörner, nous évitons toute forme de remplissage (donc tout surcoût de stockage). Nous avons donc des préconditionnements, qui permettent d'atteindre des nombres d'itérations relativement faibles. De plus, leur degré de parallélisme est maximal et enfin ils utilisent au mieux les opérations adaptées à l'architecture de la CM-2 (combinaisons linéaires, produits matrice multidiagonale vecteur). Ces préconditionnements semblent donc très bien adaptés à l'architecture massivement parallèle de la CM-2.

Remarque IV.3 *Comme la factorisation incomplète de Cholesky les préconditionnements polynômiaux font varier le nombre d'itérations avec la structure plus ou moins creuse (ou bien le nombre de diagonales non nulles) qu'on lui impose. Ainsi, nous pouvons donc exhiber des approximations de l'inverse aussi précises que l'on veut, dans la mesure où le schéma reste convergent bien entendu.*

Par ailleurs, il semble que le polynôme Minmax soit moins efficace que les deux autres polynômes. Nous verrons, par la suite, que cette constatation n'est pas toujours vérifiée. En effet sur ce problème étudié, il semble que le conditionnement de la matrice ne soit pas encore assez important car l'ordre de la matrice n'est pas suffisamment élevé[1]. Or le polynôme Minmax minimise une généralisation du conditionnement. De plus, Minmax est bien plus sensible que Mcarre et Leg à une mauvaise estimation de la plus petite valeur propre, voir [AsMO92], [Saad85] et le chapitre VII. Etant donné que la valeur prise ici n'est qu'approximative, nous avons perdu énormément de précision. De toute façon il a été montré dans [AsMO92] qu'il n'existe pas de polynôme optimal, nous le confirmons donc grâce aux résultats numériques présents dans cette thèse.

Enfin, on constate, dès à présent, que pour des polynômes de degré supérieur ou égal à 8 ou 10, suivant le polynôme, le G.C.P ne con-

[1] Par exemple, le conditionnement de la matrice du problème 1, voir l'équation (I.14), est de l'ordre de 680, pour le maillage 40×40, alors que pour un maillage 512×512 il est proche de 10^5.

verge plus ou en tout cas moins bien qu'avec un degré inférieur. Ce phénomène peut s'expliquer par le schéma d'évaluation utilisé : le schéma de Hörner simple qui répercute les erreurs d'arrondi [Gaut86], [Gaut90] et [StBu80], mais aussi par le manque de précision dans le calcul des coefficients α_k des polynômes. Par exemple le coefficient α_{11} du polynôme Mcarre de degré 11, calculé en simple précision d'après les formules de récurrences des polynômes de Tchebycheff, est -165.654. Alors que si on effectue les mêmes calculs mais en double précision on obtient -165.654158123. Ainsi on atteint, lors du calcul du polynôme pour la valeur 2, une erreur de 0.3238. Donc pour un polynôme qui doit d'une certaine manière approcher la fonction $(x \to 1/x)$ nous pouvons nous retrouver assez loin de la valeur 0.5 que nous devrions atteindre pour $x = 2$ (pour le polynôme de degré 15 nous avons une erreur de l'ordre de $78,6432$!!!). A cause de ces problèmes de précision nous avons donc des préconditionnements qui ne sont pas très fiables.

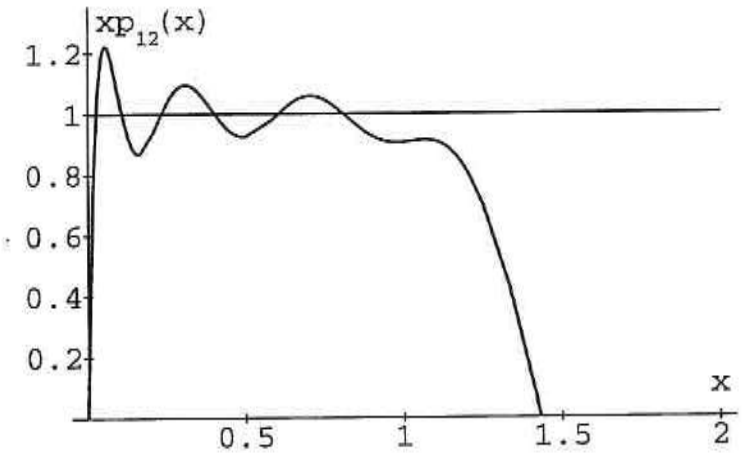

Figure IV.1: Polynôme Mcarre de degré 12.

De la même façon, pour un préconditionnement polynômial p_k donné, le polynôme $xp_k(x)$ doit osciller autour de un, tout en restant strictement positif (pour conserver un préconditionnement défini positif). Or on peut voir sur la figure (IV.1), que le polynôme $xp_k(x)$, avec p_k le polynôme Mcarre de degré 12, n'est même pas strictement positif, de plus mis à part au voisinage de zéro il n'oscille pas autour de un. En-

fin, la figure (IV.2) nous montre que même en utilisant des réels double précision le G.C, préconditionné par le polynôme Mcarre, ne converge plus à partir d'un certain degré (21), certes plus élevé. Par ailleurs, on a constaté les mêmes phénomènes pour les polynômes Minmax et Leg.

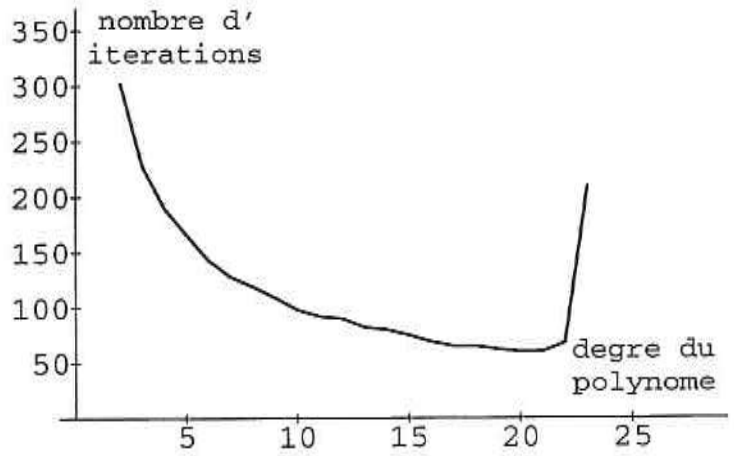

Figure IV.2: Nombre d'itérations de HMCARRE, en double précision, pour le problème 2 avec un maillage 40×40.

Remarque IV.4 *Les problèmes d'instabilité numérique des préconditionnements polynômiaux ont déja été maintes fois observés, en particulier dans [AsMO92], [Saad 85] et [CiMP92]. De plus, une certaine réponse a été apportée dans [AsMO92], [CiMP93]. Pour notre part, on peut se demander dès maintenant si ce problème de convergence n'est pas, dans une certaine mesure, un facteur limitant de l'efficacité des préconditionnements polynômiaux notamment sur la CM-2.*

Dans les tableaux suivants nous nous intéressons aux méthodes pas nécessairement bien adaptées à une architecture massivement parallèle. Nous ne présentons pas de résultats pour les méthodes LAPINV, LOC5 et LOC9 car soit elles ne convergent pas, soit les résultats obtenus nous paraissent sans réel intérêt. Dans le cas où les méthodes sont définies positives, et donc n'ont pas besoin de paramètre Θ, nous les avons quand même placées dans des tableaux semblables. Simplement, nous n'avons pas rempli la colonne de Θ correspondante.

Préconditionnement utilisé	Θ	Nombre d'itérations
LAP	0.	*non déf. pos.*
LAP	0.1	*non déf. pos.*
LAP	0.2	*non déf. pos.*
LAP	0.3	472
LAP	0.4	485
LAP	0.5	472
LAP	0.6	494
LAP	0.7	500
LIP		856

Tableau IV.5: Nombre d'itérations pour le problème 2 avec un maillage 40×40.

On note, sur le tableau (IV.5), que le nombre d'itérations pour les méthodes LAP et LIP n'est pas meilleur que celui des méthodes, de base, adaptées à la CM-2 : RBIC, RBSSOR et NEUM1. A contraintes identiques, nous avons donc des préconditionnements plus efficaces. D'autre part, il semble que la différence entrevue entre NEUM1 et ICNEUM au tableau (IV.1) soit semblable à celle entre LAP et LIP.

Préconditionnement utilisé	Θ	Nombre d'itérations
LAPINC	0.	*non déf. pos.*
LAPINC	0.1	*non déf. pos.*
LAPINC	0.2	*non déf. pos.*
LAPINC	0.3	483
LAPINC	0.4	472
LAPINC	0.5	483
LAPINC	0.6	492
LAPINC1	0.	*non déf. pos.*
LAPINC1	0.1	253
LAPINC1	0.2	297
LAPINC1	0.3	340
LAPINC1	0.4	379
LAPINV1		27

Tableau IV.6: Nombre d'itérations pour le problème 2 avec un maillage 40×40.

Préconditionnement utilisé	Θ	Nombre d'itérations
SCHUR1		1066
SCHUR2	0.	*non déf. pos.*
SCHUR2	0.1	*non déf. pos.*
SCHUR2	0.2	*non déf. pos.*
SCHUR2	0.3	468
SCHUR2	0.4	461
SCHUR2	0.5	469
SCHUR2	0.6	483
SCHUR2	0.7	494

Tableau IV.7: Nombre d'itérations pour le problème 2 avec un maillage 40×40.

D'après (IV.6), il semble, qu'à conditions égales, LAPINC est sensiblement aussi efficace que LAP, et donc moins performant que RBIC, RBSSOR et NEUM1. Par contre, LAPINC1 est plus précis, ce qui est normal car le squelette généré par ce type de préconditionnement est nettement plus dense que celui de la matrice initiale, on possède donc beaucoup plus d'informations. Il faudrait plutôt le comparer avec un préconditionnement polynômial ayant un squelette équivalent. Enfin, il semble que la méthode LAPINV1 soit très bien adaptée à ce problème.

D'autre part, il nous apparait, dans le tableau (IV.7), que SCHUR1 n'est pas très efficace (LOC5 et LOC9 ne le sont pas du tout). Par contre, avec un stencil un peu plus grand (9×9) SCHUR2 a des performances sensiblement meilleures que LAP.

Finalement, il semble, d'après les simulations numériques, qu'avec des conditions identiques (que des communications entre un nœud et ses voisins immédiats) les préconditionnements de base adaptés à l'architecture massivement parallèle de la CM-2 (RBIC, RBSSOR et NEUM1) sont des estimations de l'inverse relativement satisfaisantes par rapport aux autres préconditionnements proposés, sur ce type de problèmes. De plus, malgré les nombreuses contraintes liées à la machine, nous avons une catégorie de préconditionnements, les préconditionnements polynômiaux, qui permettent d'obtenir des nombres d'itérations assez faibles. Par contre, il faut quand même se rappeler qu'avec ces préconditionnements notamment avec des polynômes de degré élevé, nous effectuons beaucoup plus de calculs à chaque itération qu'avec le préconditionnement Diagonal. Or le but de cette étude n'est pas de trouver un préconditionnement possédant un meilleur taux de convergence que (DIAG), mais un préconditionnement capable de minimiser le temps de calcul. Il n'est donc pas du tout évident que ces préconditionnements soient plus efficaces que le préconditionnement Diagonal sur la CM-2.

Remarque IV.5 *A titre de comparaison, nous avons résolu ce problème avec des méthodes, plus classiques sur machines séquentielles, telles IC(1,1) et MIC(1,1) de Meijerink et Van der Vorst [MeVa77]. Dans ce cas, nous obtenons bien entendu des nombres d'itérations plus faibles* 43 *pour IC(1,1) et* 21 *pour MIC(1,1). Néanmoins, il est clair que, parmi*

les préconditionnements adaptés à la CM-2, seuls les préconditionnements polynômiaux pourront approcher ces taux de convergence.

IV.3 Temps de calcul

Nous présentons dans les tableaux (IV.8) à (IV.11) les temps de calcul CPU (en secondes) des méthodes, qui nous ont parus dignes d'intérêt. Ainsi, nous avons choisi les valeurs optimales, en temps, de Θ pour les méthodes non nécessairement définies positives. En outre, nous avons extrait à partir du temps de calcul (TTOT) le temps d'initialisation (TINIT), c'est à dire le temps nécessaire, entre autres, à la construction de la matrice mais aussi à l'initialisation de toutes les variables du G.C.P (TINIT est en réalité tout le temps de calcul avant les itérations du G.C.P).

Préconditionnement	TINIT	TTOT
DIAG	0.52	15.6
RBIC	0.55	11.6
RBSSOR	0.54	11.6
NEUM1	0.54	12.0
ICNEUM	0.55	24.3
LAP $\Theta = 0.3$	3024	5952
LIP	3181	6399
LAPINC $\Theta = 0.4$	0.70	22.8
LAPINC1 $\Theta = 0.1$	1.03	17.3
LAPINV1	1.12	2.7
SCHUR1	0.40	22.8
SCHUR2 $\Theta = 0.4$	4.69	35.9

Tableau IV.8: Temps de résolution (s) du problème 2 avec un maillage 40×40.

Le temps total n'est affiché que pour information. Il n'apporte, en effet, que peu d'informations sur les performances des préconditionnements sur des machines parallèles, car évidemment sur machine scalaire,

il n'est pas tenu compte des propriétés de parallélisme. En effet, alors que sur une machine séquentielle il suffit, bien souvent, de minimiser le nombre total de calculs pour minimiser le temps de résolution, sur une machine parallèle la vitesse avec laquelle on parvient à effectuer ces calculs prend une part au moins aussi importante dans l'efficacité de la méthode. Ainsi, une méthode plus efficace que DIAG sur machine séquentielle, mais pas parallèle (c'est à dire n'exploitant pas au maximum les capacités de la machine), n'est pas du tout sure de rester plus efficace que DIAG sur la CM-2, car DIAG est très bien adapté à son architecture. Pour toutes ces raisons nous avons jugé qu'il était inutile de présenter les temps pour toutes les méthodes, en particulier lorsque Θ varie.

Degré du polynôme HMINMAX	TINIT	TTOT
2	0.56	14.2
3	0.57	14.3
4	0.58	14.0
5	0.59	14.2
6	0.59	14.6

Degré du polynôme HMCARRE	TINIT	TTOT
2	0.57	11.7
3	0.59	11.6
4	0.60	11.6
5	0.60	11.7
6	0.61	11.7

Tableau IV.9 et Tableau IV.10: Temps de résolution (s) du problème 2 avec un maillage 40×40.

Degré du polynôme HLEG	TINIT	TTOT
2	0.56	12.4
3	0.57	12.0
4	0.58	11.9
5	0.59	11.9
6	0.60	12.0

Tableau IV.11: Temps de résolution (s) du problème 2 avec un maillage 40×40.

Néanmoins, on s'aperçoit qu'aucune technique adaptée à la CM-2 ne possède un temps d'initialisation beaucoup plus important que la méthode DIAG. Et donc, l'initialisation des paramètres supplémentaires, tels les α_i pour les polynômes, n'entraîne pas un surcoût de calcul trop important. Cette phase n'étant pas la plus parallèle de ces préconditionnements, il était préférable qu'elle ne soit pas trop pénalisante.

De plus, on remarque que les méthodes LAP et LIP ne sont pas du tout réalistes, notamment au niveau de TINIT. Enfin, les techniques à base de factorisation par blocs ou de compléments de Schur, se révèlent, sur ce type de problèmes, toutes (à part LAPINV1) plus lentes que les techniques appropriées à la CM-2, en particulier au moment de l'initialisation.

Il nous faut maintenant étudier ces préconditionnements (RBIC, RBSSOR, NEUM1 et les polynômes) sur la machine, qui leur semble appropriée, la CM-2 et les comparer avec le préconditionnement de référence de la CM-2 le préconditionnement Diagonal.

Remarque IV.6 *D'après les tableaux (IV.9) à (IV.11) il ne semble pas très utile de prendre des préconditonnements polynômiaux de degré supérieur à 3 ou 4 pour atteindre des temps de calcul minimum. En effet, rapidement et de la même manière que la factorisation de Cholesky incomplète, le surcoût de calcul provoqué par l'augmentation du degré du polynôme n'est pas suffisamment compensé par la diminution du nombre d'itérations. La divergence du G.C.P pour des polynômes de degré élevé ne constitue donc absolument pas, dans ce cas, un facteur limitant de l'efficacité des préconditionnements polynômiaux. Par contre sur une machine parallèle telle la CM-2, on peut s'attendre, grâce à la rapidité du produit matrice vecteur, à une augmentation du taux de parallélisme ainsi que des Mflops (en fait de la vitesse avec laquelle on parvient à effectuer les calculs) avec le degré du polynôme. Alors l'utilisation de polynômes de degré plus élevé, sera sans doute nécessaire pour atteindre un temps de résolution minimum.*

CHAPITRE V

ÉTUDE SUR SUPERCALCULATEURS MASSIVEMENT PARALLÈLES : CM-2 ET CM-200

V.1 Résultats numériques sur la CM-2

Après avoir constaté que, malgré les contraintes qu'on leur imposait, les préconditionnements, adaptés à une architecture massivement parallèle de type SIMD, semblent être de bonnes approximations de l'inverse, pour ce type de problèmes. Nous allons, dans ce chapitre, les étudier et les comparer sur la CM-2, puis sur la CM-200.

Nous avons réalisé tous les tests en réels simple précision sur 8192 PEs de la CM-2. Le langage de programmation est le CM-Fortran et nous ne sommes pas servis de la librairie scientifique CMSSL. Le critère d'arrêt des itérations (en particulier $\epsilon = 10^{-6}$) et toutes les initialisations du G.C.P sont les mêmes qu'au chapitre précédent. Les problèmes résolus sont, dans un premier temps, le problème 1 avec un maillage 512×512 (donc des matrices d'ordre 262 144), puis nous avons mis en œuvre le second problème avec un maillage 256×256. Finalement, nous présentons les résultats sous forme de tableaux contenant le nombre d'itérations du G.C.P, puis le temps de calcul (voir remarque II.14) nécessaire à la résolution (on ne comptabilise pas, cette fois, l'initialisation de la matrice) et pour finir les Mflops, c'est à dire les millions d'opérations exécutées en une seconde (pour simplifier quelque peu le décompte du nombre total d'opérations, nous avons retenu uniquement les calculs effectués sur les vecteurs d'ordre N).

Remarque V.1 *Malgré tous les efforts consentis pour obtenir des performances fiables, voir remarque II.14, il semble que la CM-2 associée à sa machine hôte soit tout de même sujette à des fluctuations de performances. Ainsi, certains résultats (Mflops ou temps) peuvent apparaître*

comme n'étant pas absolument fiables. En tout cas, nous pouvons estimer que l'erreur ne dépasse guère un Mflops pour les vitesses de calcul, et donc une proportion correspondante pour les temps de résolution. Néanmoins, tous ces problèmes ne modifieront nullement les observations que nous ne manquerons pas d'apporter après chaque étude.

V.1.1 Problème 1 avec un maillage 512×512

Avant de présenter les premiers résultats et afin de vérifier que les méthodes proposées ont une efficacité (en nombre d'itérations) relativement satisfaisante par rapport au préconditionnement Diagonal, nous posons la définition suivante :

Définition V.1 *Nous appelons taux de convergence de la méthode M par rapport à DIAG, le nombre $\overline{\tau}$ tel que*

$$\overline{\tau} = \frac{\textit{nombre d'itérations de la méthode } M}{\textit{nombre d'itérations de } DIAG}.$$

Préconditionnement utilisé	Nombre d' itérations	Temps de résolution	Mflops
DIAG	1521	70.5	124.5
RBIC	675	57.9	88.0
RBSSOR	683	62.1	88.2
NEUM1	761	54.1	114.4
NEUM3	537	58.1	119.0

Tableau V.1: Résultats pour le problème 1.

On constate, dans un premier temps, sur le tableau V.1 que les écarts entre les nombres d'itérations sont semblables à ceux enregistrés au chapitre précédent lors de l'étude séquentielle.

De plus, le préconditionnement Diagonal reste, dans ce cas et pour le moment, le préconditionnement capable d'atteindre les vitesses de calcul les plus élevées. Et donc, dans un certain sens, il demeure le préconditionnement le mieux adapté à une architecture massivement

parallèle SIMD. Néanmoins, les méthodes NEUM1 et NEUM3 ont des performances proches de DIAG. Donc, des polynômes de degré encore plus élevé dépasseront sans doute DIAG. D'autre part, à cause de leur degré de parallélisme ($N/2$), les préconditionnements basés sur des renumérotations Rouge-Noir ont des vitesses de calcul relativement faibles. Ce défaut serait sans doute bien trop pénalisant sur des machines plus puissantes et de même type (tout simplement sur une CM-2 avec 16 ou 32K PEs).

Enfin, sur ce problème, DIAG n'est pas la technique capable de résoudre le système (I.9) en un temps minimum. Avec NEUM1, nous parvenons à un gain de temps de l'ordre de 23%. Par contre, le temps de NEUM3 est relativement décevant, cela est dû notamment à son taux de convergence $\overline{\tau}$ insuffisant par rapport à son surcout de calculs. Enfin, on peut penser a priori que les préconditionnements polynômiaux les plus élaborés vont encore améliorer ces résultats.

Remarques sur le mode de stockage optimal en temps

En premier lieu, nous avions réalisé la même étude mais avec un stockage unidimensionnel des diagonales de la matrice, voir remarques I.7 et II.12. Par conséquent, nous proposons dans le tableau suivant, les résultats avec ce type de stockage ainsi que les résultats obtenus par le préconditionnement Diagonal avec un stockage bidimensionnel (en effet les résultats déjà présentés pour cette méthode ont été effectués à l'aide du stockage unidimensionnel), voir le chapitre IX pour une étude plus complète sur la CM-5.

Préconditionnement utilisé	Nombre d' itérations	Temps de résolution	Mflops
DIAG.2D	1345	71.0	109.2
RBIC.1D	675	80.1	67.4
RBSSOR.1D	689	81.4	67.7
NEUM1.1D	770	62.9	99.5

Tableau V.2: Résultats pour le problème 1.

Les performances sont évidemment moins bonnes que celles exposées au tableau V.1, car nous avons tout simplement choisi de ne retenir que les meilleurs résultats pour ce précédent tableau. Le préconditionnement Diagonal est, dans ce cas, plus efficace avec le stockage unidimensionnel qu'avec la version bidimensionnelle. Il semble, en effet, que l'efficacité du produit matrice vecteur recherchée avec le stockage bidimensionnel n'est pas suffisante par rapport à la perte de rendement du produit scalaire. En d'autres termes, avec le préconditionnement Diagonal et sur ce problème, le produit matrice vecteur n'est pas suffisament prépondérant par rapport au produit scalaire pour utiliser ce mode de stockage semblable au maillage.

De plus, pour les préconditionnements polynômiaux, sauf NEUM1, nous n'avons même pas essayé le stockage unidimensionnel car la rapidité du produit matrice vecteur est alors décisive.

Remarque V.2 *Dès à présent, on peut affirmer, et nous le vérifierons encore sur la CM-200, que l'efficacité des préconditionnements présentés dans cette thèse est, sur ce type de machine, fortement liée à la vitesse de calcul atteinte par le produit matrice vecteur. Dans ce but, nous avons utilisé un type de matrice avec un squelette relativement simple (multidiagonal), et nous l'avons associé à un mode de stockage bien adapté à la CM-2. En fait, les méthodes proposées conserveront en partie leur efficacité tant que nous serons capables d'obtenir un produit matrice vecteur suffisament rapide. Nous pouvons donc sans doute adapter avec efficacité ces méthodes à d'autres types de problèmes ne provenant pas uniquement d'une discrétisation par différences finies avec un maillage régulier d'équations aux dérivées partielles elliptiques.*

Enfin, on remarque que pour un même préconditionnement le mode de stockage varie le nombre d'itérations du G.C.P. Ces écarts sont dus aux erreurs d'arrondi de la machine. En effet, suivant le stockage, nous effectuons les calculs (par exemple les additions et les multiplications du produit matrice vecteur ou du produit scalaire) dans des ordres différents. Ainsi, les arrondis ne sont pas toujours les mêmes. De toute façon, sur les problèmes proposés il semble qu'aucune des deux formes de stockage ne se révèle plus précise qu'une autre. La précision de calcul ne peut donc pas constituer, dans ce cas, un critère de sélection.

Nous allons maintenant nous rendre compte, voir aussi [ChKT89], de l'efficacité de ce produit matrice vecteur associés aux préconditionnements polynômiaux de degré plus importants.

Remarque V.3 *Contrairement à l'étude précédente consacrée à la résolution du système linéaire induit par le second problème, nous connaissons une expression analytique des valeurs propres de la matrice, voir l'équation (I.13). Nous avons donc utilisé cette expression pour évaluer les différents préconditionnements polynômiaux. Ainsi, grâce aux différences d'exploitation des deux problèmes, nous pourrons vérifier l'influence de l'estimation des valeurs propres sur le rendement des préconditionnements polynômiaux, voir aussi le chapitre VII.*

Nous tenons, d'autre part, à préciser que nous n'avons présenté qu'une seule fois les Mflops des préconditionnements polynômiaux dans ces tableaux. En effet, comme nous évaluons chaque polynôme à l'aide du schéma de Hörner, HMINMAX, HMCARRE et HLEG exécutent le même type d'opérations à chaque itération. Tout de même, leur phase d'initialisation, notamment le calcul des coefficients α_k, diffère pour chaque polynôme. Mais cette étape reste négligeable (quelques dizaines de calculs en plus ou en moins) par rapport aux millions d'opérations de la résolution.

Degré du Polynôme HMINMAX	Nombre d' itérations	Temps de résolution	Mflops
2	447	41.7	123.9
3	342	38.8	127.3
4	272	36.5	129.3
5	230	35.4	131.5
6	197	34.4	132.9
7	173	33.7	134.0
8	182	39.3	134.4
9	234	55.2	135.0

Tableau V.3: Résultats pour le problème 1.

Nous pouvons constater, dans le tableau (V.3), que comme prévu la vitesse de calcul augmente avec le degré du polynôme, grâce à la rapidité du produit matrice vecteur. Ainsi, l'étape de préconditionnement, qui était une phase du G.C.P où il semblait difficile de concilier parallélisme et efficacité, atteint ici une vitesse de calcul comparable au produit matrice vecteur. Nous pouvons donc atteindre des vitesses de calcul très élevées, dans la limite supérieure du produit matrice vecteur évidemment. De ce fait, les vitesses de calcul HMINMAX, HMCARRE et HLEG sont toutes supérieures à celle de DIAG, dès que les degrés des polynômes sont strictement plus grand que 2.

Polynôme	HMCARRE		HLEG	
de degré	Nombre d' itérations	Temps de résolution	Nombre d' itérations	Temps de résolution
2	526	49.1	557	52.0
3	399	45.4	410	46.6
4	323	43.2	346	46.3
5	271	41.8	285	44.1
6	234	41.0	248	43.5
7	206	40.1	219	42.7
8	184	39.6	197	42.5
9	194	45.7	210	49.5
10	258	66.5	225	58.7

Tableau V.4: Résultats pour le problème 1.

De plus, comme au chapitre précédent nous sommes parvenus à diminuer considérablement le nombre d'itérations de DIAG. On remarque, en outre, que le nombre d'itérations de NEUM3 (537) est trop important par rapport aux autres préconditionnements polynômiaux de degré 3, par exemple HMINMAX (342), pour obtenir des performances acceptables. Ensuite, contrairement à l'étude sur machine séquentielle, voir chapitre IV, le polynôme Minmax est maintenant l'estimation de l'inverse la plus précise. Il semble que les valeurs propres utilisées, voir remarque V.3, ainsi que l'ordre plus élevé de la matrice ont eu un effet très bénéfique sur ce préconditionnement. Par suite, nous verrons

que cet ordre de grandeur (HMINMAX < HMCARRE < HLEG) des préconditionnements polynômiaux respecte les résultats théoriques esquissés à la section suivante. Par conséquent, comme ces techniques ont la même vitesse de calcul, HMINMAX est sur ce problème le préconditionnement polynômial le plus rapide en temps.

D'autre part pour les mêmes raisons qu'au chapitre précédent (voir remarque IV.4), nous constatons, sur les tableaux V.3 et V.4, qu'à partir d'un certain degré (7 ou 8) les méthodes ne réussissent plus à converger (ou en tout cas moins bien qu'avec un degré inférieur). Nous découvrons même cette divergence un peu plus tôt pour HMCARRE et HLEG, qu'au chapitre précédent. Cette nouvelle perte de précision de calcul est, sans doute, causée par le nombre d'itérations un peu plus important qu'à l'étude précédente, ainsi qu'à l'ordre plus élevé des matrices, le tout entraînant encore plus d'erreurs d'arrondis.

Finalement, nous sommes satisfaits d'avoir obtenu, au moins sur ce problème modèle, des préconditionnements, plus rapides (Mflops) que DIAG dès que $k \geq 3$, capables de réaliser la résolution dans des temps tous plus faibles que celui du préconditionnement Diagonal. Pour HMINMAX de degré 7, le gain de temps par rapport à DIAG est de l'ordre de 52%. Toutefois, le temps minimal est réalisé par le polynôme de degré k le plus élevé possible, c'est à dire dans la mesure où le schéma reste convergent[1]. En effet, par rapport à l'étude séquentielle le taux de convergence des préconditionnements polynômiaux s'est amélioré. De plus, sur les ordinateurs parallèles, la vitesse de calcul est souvent un facteur décisif dans l'efficacité des méthodes. Ainsi, alors que le degré optimal était d'environ 4 sur machine séquentielle, il est maintenant de 7 ou 8 sur la CM-2. Dans ce cas, l'instabilité des préconditionnements polynômiaux apparaît comme un facteur limitant. On se demande, en effet, si avec des polynômes de degré supérieur à 8, on ne pourrait pas encore réduire le temps de résolution de ces préconditionnements polynômiaux associés au schéma de Hörner simple.

[1]Si le nombre d'itérations du G.C.P préconditionné par un polynôme de degré $k+1$ est supérieur au nombre d'itérations avec la même méthode mais de degré k, nous considérerons que le préconditionnement polynômial de degré $k+1$ est divergent.

V.1.2 Problème 2 avec un maillage 256×256

Avant d'essayer de résoudre ces problèmes d'instabilité, nous allons délaisser le classique problème de Poisson pour aborder le problème 2 (avec un maillage 256×256, donc des matrices d'ordre 65 536), avec des initialisations du G.C.P identiques à celle de l'étude séquentielle (particulièrement pour l'approximation des valeurs propres).

Remarque V.4 *Pour ce second problème, nous n'avons pas utilisé des matrices d'ordre 262 144, car le problème est trop mal conditionné. Alors la résolution avec un maillage* 512×512 *aurait nécessité des temps de calcul beaucoup trop grands par rapport au temps machine disponible pour cette étude. De plus, il n'est pas inutile de regarder comment se comportent les performances en variant la taille des matrices.*

Préconditionnement utilisé	Nombre d' itérations	Temps de résolution	Mflops
DIAG	7368	129.0	82.3
RBIC	3683	128.2	57.6
RBSSOR	3685	128.1	57.8
NEUM1	3792	108.5	71.0
NEUM3	3238	126.3	82.4

Tableau V.5: Résultats pour le problème 2.

D'après le tableau V.5, DIAG reste une technique très compétitive sur la CM-2, que ce soit au niveau des Mflops mais aussi des temps de résolution. En effet, d'une part les taux de convergence de RBIC et RBSSOR sont moins avantageux que précédemment (environ 0.5 au lieu 0.45). D'autre part, la taille plus réduite (65 536 au lieu de 262 144) des matrices engendre un produit matrice vecteur moins rapide. Ainsi, mis à part NEUM1, ces préconditionnements de base ne sont pas réellement plus efficaces que le préconditionnement Diagonal. Donc, sur ce problème, on comprend mieux pourquoi le préconditionnement Diagonal demeure le préconditionnement de référence sur les machines massivement parallèle de type SIMD. Ainsi, on convient bien volontiers que les résultats des préconditionnements de base de cette thèse (RBIC,

RBSSOR et NEUM) ne sont pas toujours très encourageants par rapport à ceux de DIAG. Néanmoins leur étude a sans doute constitué une base solide pour mieux étudier les préconditionnements polynômiaux.

Finalement, les différences de Mflops constatées par rapport à l'étude du problème 1 sont dues à la taille des matrices, voir remarque II.9 et II.10.

Degré du Polynôme HMINMAX	Nombre d' itérations	Temps de résolution	Mflops
2	2236	77.7	83.1
3	1749	71.6	88.1
4	1407	66.1	92.1
5	1181	62.6	95.3
6	1026	60.2	98.4
7	962	62.2	100.5
8	1338	94.9	101.7

Tableau V.6: Résultats pour le problème 2, avec HMINMAX.

Polynôme	HMCARRE		HLEG	
de degré	Nombre d' itérations	Temps de résolution	Nombre d' itérations	Temps de résolution
2	2329	80.9	2447	84.9
3	1778	72.8	1817	74.4
4	1451	68.1	1544	72.5
5	1238	65.6	1301	69.0
6	1077	63.1	1139	66.7
7	950	61.4	1011	65.4
8	856	61.2	921	65.3
9	893	69.2	906	70.2

Tableau V.7: Résultats pour le problème 2, avec HMCARRE et HLEG.

Les tableaux V.6 et V.7 nous montrent que les taux de convergence des préconditionnements polynômiaux restent intéressants pour ce second problème. Ils se sont même un peu amélioré, à cause du nombre

important d'itérations de DIAG (qui se révèle en général moins performant quand les problèmes sont mal conditionnés). De plus, l'écart entre HMINMAX et les autres s'est quelque peu atténué, à cause de l'estimation des valeurs propres bien moins précise que pour le premier problème. En effet, voir [Saad85] et le chapitre VII, le polynôme Minmax est beaucoup plus sensible à une mauvaise estimation de sa plus petite valeur propre que les autres préconditionnements polynômiaux. Néanmoins, nous ne pouvons pas dire, voir [AsMO92], que le polynôme Minmax est le plus efficace dans tous les cas. En effet, il n'existe pas de polynôme optimal, ils dépendent tous de la distribution de toutes les valeurs propres (et pas uniquement de la plus grande et de la plus petite). Ainsi, à chaque matrice il existe, sans doute, un polynôme, parmi les trois proposés ici ou bien parmi d'autres, mieux adapté à sa distribution des valeurs propres. Malgré tout, HMINMAX demeure, dans ce cas, le préconditionnement polynômial apportant les nombres d'itérations les plus faibles (conformément aux résultats théoriques de la partie suivante de ce chapitre, mais contrairement à l'étude séquentielle). Et ainsi, le gain de temps réalisé se révèle une nouvelle fois correct, 53% pour HMINMAX de degré 6 par rapport à DIAG.

Enfin, nous constatons à nouveau l'instabilité des préconditionnements polynômiaux, dès que le degré devient grand. Malheureusement, à cause du nombre élevé d'itérations, ce problème surgit encore plus rapidement (pour des degrés 6 ou 7). Tout ceci, nous amène à penser que l'efficacité des préconditionnements polynômiaux est limitée par le manque de précision dans les calculs. L'utilisation de réels double précision [Saad85] ou l'introduction de schémas d'évaluations plus stables [AsMO92] et [CiMP93] pour ces polynômes sont donc des possibilités à envisager afin de diminuer encore les temps de résolution.

En conclusion de ces premiers résultats numériques, nous pouvons affirmer que, sur les deux problèmes étudiés, les préconditionnements de base ne surpassent pas en tout point le préconditionnement Diagonal. Par contre, l'utilisation des préconditionnements polynômiaux peut s'avérer très avantageuse. Ainsi, pour les problèmes considérés le préconditionnement Diagonal n'est plus le préconditionnement le plus

efficace sur la CM-2, en temps total de résolution mais aussi en Mflops. Il serait intéressant d'étudier leur efficacité sur des maillages moins réguliers, voir chapitre IX, ou sur d'autres types d'équations.

V.2 Un résultat de convergence

Parmi les méthodes s'écrivant :

$$x_{k+1} = x_0 + P_k(A)z_0$$

le G.C est celle qui minimise $\epsilon(x_{k+1})$, voir [GoMe83] et chapitre II. Il pouvait donc sembler étrange d'utiliser des préconditionnements polynômiaux. Dans un certain sens, pour obtenir des préconditionnements polynômiaux efficaces, il faudrait que le G.C.P préconditionné par un polynôme de degré k effectue $k+1$ fois moins d'itérations que le G.C (en fait que DIAG). En effet, lors d'une itération du G.C.P préconditionné par un polynôme de degré k, nous effectuons comme opérations l'équivalent de $k+1$ produits matrice vecteur alors que le G.C, même préconditionné par la diagonale, n'en nécessite qu'un seul. Nous pouvons donc comparer les taux de convergence des préconditionnements polynômiaux avec le taux (noté G.C) qu'ils devraient essayer d'atteindre.

Degré du polynôme	G.C	Polynôme Minmax	Polynôme Mcarre	Polynôme Leg
2	0.3333	0.3323	0.3911	0.4141
3	0.25	0.2543	0.2967	0.3048
4	0.2	0.2022	0.2401	0.2572
5	0.1667	0.1710	0.2015	0.2119
6	0.1429	0.1465	0.1740	0.1844
7	0.125	0.1286	0.1532	0.1628

Tableau V.8: Taux de convergence pour le problème 1.

Sur le tableau V.8, nous constatons que les taux réalisés par HMINMAX sont remarquables. Grâce aux calculs (erreurs d'arrondis), ils

sont même parfois inférieurs aux taux théoriques. Les taux de HMIN-MAX, HMCARRE et HLEG sont même encore plus intéressants pour le second problème à cause de la perte d'efficacité de DIAG sur les problèmes mal conditionnés.

Nous avons vu au deuxième chapitre que la vitesse de convergence du G.C.P dépend de $\kappa(M^{-1}A)$ nombre de conditionnement de la matrice $M^{-1}A$, avec M la matrice de préconditionnement. Donc pour un polynôme P_k donné, nous allons nous intéresser à $\kappa(P_k(A)A) = \kappa(Q_{k+1}(A))$. Malheureusement, même si nous connaissons les valeurs propres de A (et donc facilement celles de $Q_{k+1}(A)$), il est difficile, en général, de retrouver la plus petite et la plus grande parmi celles de $Q_{k+1}(A)$. Donc, pour simplifier ce problème nous utiliserons, en premier lieu et pour chaque polynôme, la généralisation du nombre de conditionnement $cond(q_{m+1})$ introduite en III.3, qui est en fait une borne supérieure du nombre de conditionnement $\kappa(A)$. Puis finalement, nous regarderons une généralisation du facteur de convergence ν.

Dans ce but, nous allons reprendre les résultats obtenus par Johnson, Michelli et Paul dans [JoMP83], mais de manière un peu moins restrictive puis nous étendrons ces résultats au cas du polynôme Mcarre. Mais, procédons d'abord à quelques calculs préliminaires.

Notons

$$\kappa(A) = \kappa = \frac{\lambda_{max}}{\lambda_{min}} = \frac{b}{a}$$

le nombre de conditionnement de la matrice A, voir fin du chapitre I. Dans la suite nous ne supposons donc pas comme dans [JoMP83] que $\kappa(A) = (1+\beta)/(1-\beta)$.

Notons également

$$\nu = \left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)$$

le facteur de réduction de l'erreur à chaque itération du G.C, voir II.1. Remarquons que $1 > \nu > 0$.

Inversement, on obtient

$$\sqrt{\kappa} = \frac{1+\nu}{1-\nu}.$$

Ensuite, nous rappelons l'application linéaire bijective, qui va de $[a, b]$ dans $[-1, 1]$, définie au chapitre III par :

$$\mu(\lambda) = \frac{2\lambda - b - a}{b - a}.$$

Donc

$$\mu(0) = \frac{-b-a}{b-a} = \frac{1+b/a}{1-b/a} = \frac{1+\kappa}{1-\kappa} = \frac{(1-\nu)^2+(1+\nu)^2}{(1-\nu)^2-(1+\nu)^2} = \frac{1+\nu^2}{-2\nu},$$

ainsi $\nu^2 + 2\mu(0)\nu + 1 = 0$. Notons ν_1 et ν_2 les solutions de cette équation, alors on a

$$\begin{aligned} \nu_1 &= -\mu(0) - \sqrt{\mu^2(0)-1}, \\ \nu_2 &= -\mu(0) + \sqrt{\mu^2(0)-1}, \end{aligned}$$

or $\nu_2 > 1 > \nu_1 > 0$.

On a donc

$$\nu = -\mu(0) - \sqrt{\mu^2(0)-1}, \tag{V.1}$$

puis

$$\nu^{-1} = -\mu(0) + \sqrt{\mu^2(0)-1}. \tag{V.2}$$

De plus, comme $|\mu(0)| > 1$, nous connaissons l'expression des polynômes de Tchebycheff et

$$T_k(\mu(0)) = \frac{1}{2}\left(\left(\mu(0) - \sqrt{\mu^2(0)-1}\right)^k + \left(\mu(0) + \sqrt{\mu^2(0)-1}\right)^k\right),$$

c'est à dire, d'après (V.1) et (V.2)

$$T_{k+1}(\mu(0)) = \frac{(-1)^{k+1}}{2}\left(\nu^{k+1} + \nu^{-k-1}\right). \tag{V.3}$$

En particulier

$$|T_{k+1}(\mu(0))|^{-1} \leq 2\nu^{k+1}. \tag{V.4}$$

Maintenant, regardons, pour le polynôme Minmax (voir sa définition en III.4), la généralisation du nombre de conditionnement définie pour les polynômes en III.3

$$
\begin{aligned}
cond(q_{k+1}) &= \frac{sup_{\lambda\in[a,b]}\left(q_{k+1}(\lambda)\right)}{inf_{\lambda\in[a,b]}\left(q_{k+1}(\lambda)\right)} \\
&= \frac{|T_{k+1}\left(\mu(0)\right)| - inf_{\lambda\in[a,b]}\left((-1)^{k+1}T_{k+1}\left(\mu(\lambda)\right)\right)}{|T_{k+1}\left(\mu(0)\right)| - sup_{\lambda\in[a,b]}\left((-1)^{k+1}T_{k+1}\left(\mu(\lambda)\right)\right)} \\
&= \frac{|T_{k+1}\left(\mu(0)\right)| + 1}{|T_{k+1}\left(\mu(0)\right)| - 1} \\
&= \frac{\nu^{-k-1} + \nu^{k+1} + 2}{\nu^{-k-1} + \nu^{k+1} - 2}, \text{ d'après (V.3)} \\
&= \left(\frac{1+\nu^{k+1}}{1-\nu^{k+1}}\right)^2.
\end{aligned}
$$

Enfin, de la même manière que nous avons généralisé le nombre de conditionnement, nous généralisons ν le facteur de réduction l'erreur du G.C par

$$
\frac{\sqrt{cond(q_{k+1})}-1}{\sqrt{cond(q_{k+1})}+1} = \frac{1+\nu^{k+1}-1+\nu^{k+1}}{1+\nu^{k+1}+1-\nu^{k+1}},
$$

et donc

$$
\frac{\sqrt{cond(q_{k+1})}-1}{\sqrt{cond(q_{k+1})}+1} = \nu^{k+1}. \tag{V.5}
$$

En d'autres termes, une étape du G.C.P préconditionné par le polynôme Minmax de degré k a environ la même convergence que $(k+1)$ étapes du G.C. Bien entendu, il faut relativiser ce résultat car il ne s'agit que de généralisation du nombre de conditionnement et du facteur de réduction de l'erreur. Néanmoins, sur les deux problèmes proposés sur la CM-2, voir en particulier le tableau V.8, nous constatons que ces approximations sont assez bien respectées par Minmax. Par contre, il semble que pour le problème sur machine séquentielle (problème 2

et maillage 40×40) l'estimation soit moins juste (à cause des valeurs propres utilisées pour définir le polynôme et du conditionnement pas assez élevé).

Nous allons, maintenant, étudier la seconde catégorie des préconditionnements polynômiaux. Nous regarderons, en premier lieu, le polynôme Leg, comme dans [JoMP83], puis nous étendrons les résultats à Mcarre.

Définition V.2 *On note $\mathcal{P}_k$ l'espace vectoriel sur R, des polynômes à coefficients réels de degré inférieur ou égal à k.*

On défini ensuite un produit scalaire sur cet espace.

Définition V.3 *Soit $(p,q) \in \mathcal{P}_k \times \mathcal{P}_k$, on note*

$$(p,q)_\omega = \int_a^b p(t)q(t)\omega(t)dt$$

le produit scalaire sur $\mathcal{P}_k$ associé au poids ω (en pratique nous prendrons des poids de Jacobi, voir [Davi75], de la forme $\omega(\alpha,\beta,\lambda) = (b-\lambda)^\alpha(\lambda-a)^\beta$).

Lemme V.1

$$\forall p \in \mathcal{P}_k, \forall \lambda \in [a,b], |p(\lambda)|^2 \leq C_k \int_a^b |p(t)|^2 dt$$

avec

$$C_k = \frac{(k+1)^2}{b-a}.$$

Preuve V.1 *Soit $(s_i)_{0\leq i\leq k}$ une base orthonormée de $\mathcal{P}_k$ associée au poids de Legendre $\omega(0,\bar{0},.)$ et donc au produit scalaire $(.,.)_1$. Alors, pour tout p dans $\mathcal{P}_k$ et tout λ dans $[a,b]$*

$$p(\lambda) = \sum_{i=0}^{k} p_i s_i(\lambda), \text{ avec } p_i = \int_a^b p(t)s_i(t)dt.$$

Par conséquent

$$|p(\lambda)|^2 = \left|\sum_{i=0}^{k}\left(\int_a^b p(t)s_i(t)dt\right)s_i(\lambda)\right|^2$$

$$= \left|\int_a^b\left(\sum_{i=0}^{k}s_i(t)s_i(\lambda)\right)p(t)dt\right|^2$$

puis d'après Cauchy-Schwarz

$$|p(\lambda)|^2 \leq \int_a^b\left(\sum_{i=0}^{k}s_i(t)s_i(\lambda)\right)^2 dt \int_a^b |p(t)|^2 dt$$

$$\leq \sum_{i=0}^{k}s_i^2(\lambda)\int_a^b |p(t)|^2 dt$$

Nous remarquons que jusqu'ici, nous ne nous sommes pas servi des propriétés du poids $\omega(0,0,.) \equiv 1$, nous aurions donc pu faire cette démonstration pour un poids ω quelconque (en particulier avec le poids de Tchebycheff ($\alpha = \beta = -1/2$)).

Exprimons, maintenant, les polynômes s_i. En fait

$$s_i(\lambda) = \left(\frac{2i+1}{b-a}\right)^{\frac{1}{2}} L_i(\mu(\lambda)),$$

avec L_i le polynôme de Legendre de degré i, voir par exemple [Davi75]. Ainsi

$$s_i^2(\lambda) \leq \frac{2i+1}{b-a}, \text{ car } \forall x \in [-1,1], |L_i(x)| \leq 1.$$

Donc finalement

$$C_k = \sum_{i=0}^{k}\frac{2i+1}{b-a} = \frac{(k+1)^2}{b-a}.$$

Comme précédemment, nous nous intéressons à la généralisation du conditionnement mais cette fois pour le polynôme Leg.

Nous avons

$$cond(q_{k+1}) = \frac{\sum_{i=0}^{k+1} s_i^2(0) - inf_{\lambda\in[a,b]}\left(\sum_{i=0}^{k+1} s_i(0)s_i(\lambda)\right)}{\sum_{i=0}^{k+1} s_i^2(0) - sup_{\lambda\in[a,b]}\left(\sum_{i=0}^{k+1} s_i(0)s_i(\lambda)\right)},$$

puis, en appliquant le Lemme V.1 au polynôme de degré $k+1$ suivant $\sum_{i=0}^{k+1} s_i(0)s_i(\lambda)$ on a :

$$cond(q_{k+1}) \leq \frac{\sum_{i=0}^{k+1} s_i^2(0) + C_{k+1}^{1/2}\left(\int_a^b \left(\sum_{i=0}^{k+1} s_i(0)s_i(t)\right)^2 dt\right)^{1/2}}{\sum_{i=0}^{k+1} s_i^2(0) - C_{k+1}^{1/2}\left(\int_a^b \left(\sum_{i=0}^{k+1} s_i(0)s_i(t)\right)^2 dt\right)^{1/2}},$$

comme

$$\int_a^b \left(\sum_{i=0}^{k+1} s_i(0)s_i(t)\right)^2 dt = \sum_{i=0}^{k+1} s_i^2(0),$$

alors

$$\begin{aligned} cond(q_{k+1}) &\leq \frac{\left(\sum_{i=0}^{k+1} s_i^2(0)\right)^{1/2} + (b-a)^{-1/2}(k+2)}{\left(\sum_{i=0}^{k+1} s_i^2(0)\right)^{1/2} - (b-a)^{-1/2}(k+2)} \\ &\leq \frac{\sqrt{2}|T_{k+1}(\mu(0))| + (k+2)}{\sqrt{2}|T_{k+1}(\mu(0))| - (k+2)}, \end{aligned} \tag{V.6}$$

car

$$\left(\sum_{i=0}^{k+1} s_i^2(0)\right)^{1/2} \geq |s_{k+1}(0)| \geq \left(\frac{2}{b-a}\right)^{1/2} |T_{k+1}(\mu(0))|.$$

Enfin

$$\begin{aligned} \frac{\sqrt{cond(q_{k+1})} - 1}{\sqrt{cond(q_{k+1})} + 1} &\leq \frac{cond(q_{k+1}) - 1}{cond(q_{k+1}) + 1} \\ &\leq \frac{(k+2)/\sqrt{2}}{|T_{k+1}(\mu(0))|}, \text{ grâce à l'inégalité (V.6).} \end{aligned}$$

D'après V.4, nous obtenons une borne supérieure de la généralisation du facteur de convergence pour le préconditionnement polynômial Leg

$$\frac{\sqrt{cond(q_{k+1})}-1}{\sqrt{cond(q_{k+1})}+1} \leq \sqrt{2}(k+2)\ \nu^{k+1}. \qquad (V.7)$$

Procédons de la même manière, mais en remplaçant le poids précédent, $\omega(0,0,.)$, par le poids de Tchebycheff, $\omega(-1/2,-1/2,.)$. Ainsi, nous obtiendront une estimation pour le polynôme Mcarre. Nous avons, tout d'abord un lemme similaire au lemme V.1 :

Lemme V.2

$$\forall p \in \mathcal{P}_k, \forall \lambda \in [a,b], |p(\lambda)|^2 \leq \overline{C}_k \int_a^b |p(t)|^2 \omega(t)dt$$

avec

$$\overline{C}_k = \frac{2k+1}{\pi},\ \textit{et}\ \ \omega(t) = (b-t)^{-1/2}(t-a)^{-1/2}.$$

Preuve V.2 *Le début de la démonstration est semblable à celle du lemme V.1, seule diffère la définition des s_i et par conséquent la valeur du coefficient $\overline{C}_k$. En effet, la base orthonormée sur $[a,b]$ associée à $(.,.)_\omega$ est :*

$$s_0(\lambda) = \frac{1}{\sqrt{\pi}},$$

$$\textit{et}\ s_i(\lambda) = \sqrt{\frac{2}{\pi}} T_i(\mu(\lambda)),$$

avec T_i le polynôme de Tchebycheff de première espèce de degré i.

Donc, cette fois

$$\sum_{i=0}^{k} s_i^2(\lambda) \ \leq\ \frac{1}{\pi} + \frac{2}{\pi}\sum_{i=1}^{k} 1,\ \textit{car}\ |T_i(\mu(\lambda))| < 1,$$

$$\leq\ \frac{2k+1}{\pi}.$$

Ensuite, dans ce cas

$$\begin{aligned} cond(q_{k+1}) &\leq \frac{\left(\sum_{i=0}^{k+1} s_i^2(0)\right)^{1/2} + \overline{C}_{k+1}{}^{1/2}}{\left(\sum_{i=0}^{k+1} s_i^2(0)\right)^{1/2} - \overline{C}_{k+1}{}^{1/2}}, \\ &\leq \frac{\sqrt{2}|T_{k+1}(\mu(0))| + (k+3/2)^{1/2}}{\sqrt{2}|T_{k+1}(\mu(0))| - (k+3/2)^{1/2}}, \end{aligned}$$

car

$$\sum_{i=0}^{k+1} s_i^2(0) = \frac{1}{\pi} + \frac{2}{\pi}\sum_{i=0}^{k+1} T_i^2(\mu(0)) \geq \frac{2}{\pi} T_{k+1}^2(\mu(0)).$$

Finalement, nous obtenons un majorant pour la généralisation du facteur de convergence du G.C.P préconditionné par MCARRE :

$$\frac{\sqrt{cond(q_{k+1})} - 1}{\sqrt{cond(q_{k+1})} + 1} \leq \sqrt{4k+6}\ \nu^{k+1}. \qquad \text{(V.8)}$$

A priori, les estimations V.7 et V.8 peuvent paraître imprécises voire même inutiles (quand le majorant est supérieur à 1). Néanmoins, si ν est petit ces bornes supérieures deviennent des approximations plus justes de la généralisation du facteur de convergence. De toute façon, nous ne possédons que très peu de résultats théoriques sur les préconditionnements polynômiaux, c'est pourquoi nous nous sommes contentés de ces majorations V.5, V.7 et V.8. De plus, comme elles reflètent bien la plupart des nombres d'itérations réalisés dans les expériences numériques de cette thèse (MINMAX < MCARRE < LEG), nous avons estimé que ces quelques résultats théoriques pouvaient constituer un début d'explication pour ces différences de taux de convergence.

V.3 Résultats numériques sur la CM-200

Pour clore ce chapitre, et afin de confirmer les résultats de la CM-2, nous allons maintenant présenter quelques expériences réalisées sur la CM-200. En fait, la CM-200 est une machine dérivée de la CM-2. Ces deux supercalculateurs ont des caractéristiques assez semblables, notamment

au niveau de l'architecture (SIMD à mémoire locale). Toutefois, à nombre égal de processeurs la CM-200 est plus puissante car ces PEs ont une fréquence de 10 Mhz au lieu de 7 pour la CM-2. Malheureusement, la CM-200 du S.E.H ne possède que 4096 PEs, contre 8192 à la CM-2. Il semble que les performances atteintes par ces deux machines soient comparables. Enfin, la CM-200 bénéficie encore de quelques avantages, en particulier certaines instructions sont plus robustes et de plus nous pouvons, contrairement à la CM-2 8K du S.E.H, effectuer les calculs avec des réels double précision.

Afin d'alléger un peu ce chapitre, et comme les résultats observés sur la CM-200 confirment ceux de la CM-2, nous n'avons présenté que des résultats sur le second problème. De plus, comme il est clair maintenant que, sur les problèmes étudiés dans cette thèse, les préconditionnements polynômiaux sont les techniques les mieux adaptées à une architecture massivement parallèle de type SIMD à mémoire locale, nous n'avons pas utilisé les méthodes RBIC, RBSSOR, NEUM1 et NEUM3. Nous avons, quand même, inclus DIAG dans le tableau V.9 pour mesurer les différences entre la référence qu'il constitue, et les préconditionnements polynômiaux.

Degré du Polynôme	Nombre d' itérations	Temps de résolution	Mflops
DIAG			
0	7368	117.4	90.5
HMINMAX			
2	2238	76.4	84.5
3	1751	72.3	87.3
4	1407	67.6	90.1
5	1181	65.0	91.8
6	1026	63.6	93.1
7	961	66.4	94.0
8	1359	103.3	94.9

Tableau V.9: Résultats pour le problème 2 sur la CM-200, en simple précision.

Polynôme	HMCARRE		HLEG	
de degré	Nombre d' itérations	Temps de résolution	Nombre d' itérations	Temps de résolution
2	2333	81.2	2443	84.9
3	1778	73.5	1817	74.4
4	1451	70.3	1547	72.5
5	1225	67.6	1301	69.0
6	1072	66.5	1139	66.7
7	955	66.0	1009	69.6
8	857	65.2	921	69.9
9	897	74.4	908	75.3

Tableau V.10: Résultats pour le problème 2 sur la CM-200, en simple précision.

On remarque, dans les deux tableaux, que les performances réalisées par les préconditionnements sur la CM-200 sont semblables à celles sur la CM-2. Tout de même, les Mflops des préconditionnements polynômiaux de degré élevé sont plus faibles que précédemment (à cause sans doute du nombre de PEs relativement faible de cette CM-200). Ceci entraîne un gain de temps un peu moins important. Nous diminuons, malgré tout, le temps de résolution d'environ 46%. Finalement, cette étude sur la CM-200 ne fait que confirmer les résultats obtenus sur la CM-2. En outre, on remarque que plus la machine est capable d'atteindre des vitesses de calcul élevées, plus les gains en temps de résolution des préconditionnements polynômiaux deviennent importants. Donc, avec de nouveaux ordinateurs encore plus rapides, nous pouvons espérer adapter ces techniques avec encore plus d'efficacité.

Enfin, nous observons encore le problème d'instabilité des préconditionnements polynômiaux de degré supérieur à 6 ou 8. Mais cette fois nous avons, dans un second temps, effectué les calculs à l'aide de réels double précision. Comme nous avons présenté le polynôme Mcarre

en double précision lors de l'étude séquentielle et comme le polynôme Leg a le même comportement que Mcarre, nous exposons les tests, sur la CM-200, du préconditionnement HMINMAX en double précision. Ainsi, nous pourrons confirmer qu'aucun des trois préconditionnements polynômiaux de cette thèse n'évite les problèmes d'instabilité.

Alors, nous pouvons voir sur la figure V.1 que le nombre d'itérations de HMINMAX est décroissant jusqu'au degré 19, après les préconditionnements proposés ne sont plus définis positifs. Donc, même avec des réels double précision on trouve à nouveau les problèmes d'instabilité, mais certes avec des degrés plus grands. Le polynôme Minmax a donc un comportement identique à celui de Mcarre, voir figure IV.1.

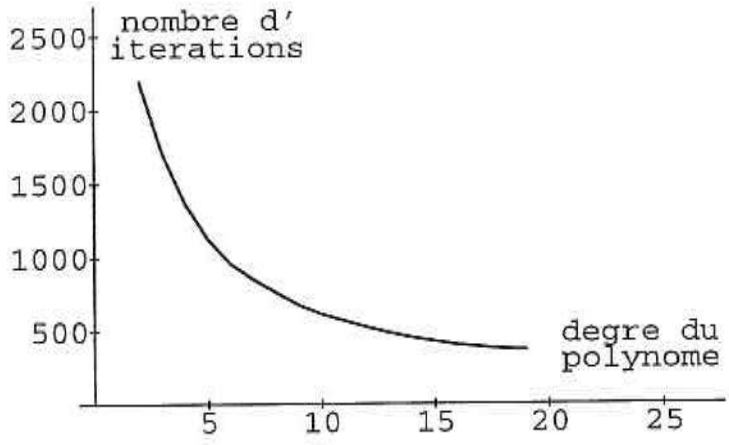

Figure V.1: Nombre d'itérations de HMINMAX en double précision, pour le problème 2.

Toutefois, contrairement aux calculs en simple précision nous avons atteint dans ce cas le degré où le temps de résolution est minimal, voir figure V.2. Donc ces problèmes de stabilité ne sont plus ici un fac-

teur limitant à l'efficacité des préconditionnements polynômiaux. Malheureusement, le surcoût de calcul produit par l'utilisation de réels double précision est trop important. En effet, pour le degré 15 HMINMAX résout le système en 97.0 secondes, au lieu de 63.6 pour HMINMAX de degré 6 en simple précision. Donc l'augmentation de la précision de calcul n'apparaît pas comme un moyen efficace sur la CM-200 de diminuer le temps de résolution du G.C.P préconditionné par des préconditionnements polynômiaux.

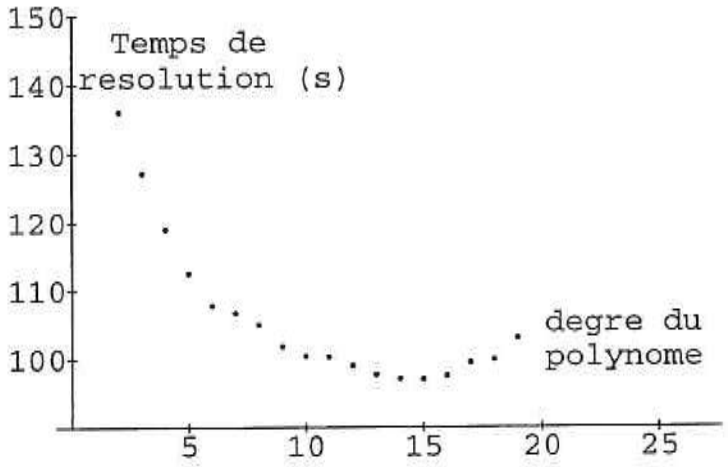

Figure V.2: Temps de résolution de HMINMAX en double précision, pour le problème 2.

Néanmoins, la figure V.2 montre bien que le degré optimal pour la minimisation du temps de calcul est supérieur à ceux trouvés jusqu'à présent. Donc, il semble tout à fait judicieux d'introduire des schémas d'évaluation des polynômes plus stables que le classique schéma de Hörner afin d'utiliser des degrés plus élevés, dans le but d'optimiser l'efficacité des préconditionnements polynômiaux sur les ordinateurs massivement parallèles SIMD à mémoire locale.

Remarque V.5 *Nous avons introduit au chapitre II, voir la définition II.1, la notion de taux de parallélisme. Cette valeur est en réalité destinée à mesurer si nos programmes exploitent complétement les capacités de calculs de la machine. Pour vérifier que nos programmes sont bien adaptés à la CM-200, nous allons présenter quelques taux de parallélisme (avec les notations évidentes HMINk est la méthode HMIN-MAX de degré k) dans le tableau suivant.*

Préconditionnement	*DIAG*	*NEUM1*	*HMIN4*	*HMIN9*
Taux de parallélisme	*0.973*	*0.971*	*0.987*	*0.992*

Tableau V.11: Taux de parallélisme pour le problème 1.

Nous constatons dans ce tableau que les taux de parallélisme proposés sont tout à fait honorables, surtout si on les compare avec celui du préconditionnement Diagonal. Ces techniques utilisent donc convenablement les possibilités de la machine.

De plus, et de manière identique aux Mflops, nous avons enregistré des différences de taux de parallélisme entre les deux problèmes. Ils sont dus aux tailles respectives de ces problèmes. Finalement, on constate comme pour les Mflops que le taux de parallélisme est une fonction croissante du degré du polynôme, cela constitue une raison supplémentaire d'utiliser des préconditionnements polynômiaux de degrés plus importants. Nous n'avons pas présenté de résultats plus détaillés sur les taux de parallélisme, notamment un graphique pour les préconditionnements polynômiaux, car comme nous l'avons déjà souligné dans la remarque II.12 à II.14 les résultats obtenus ne sont hélas pas très fiables.

CHAPITRE VI

UTILISATION DE SCHÉMAS STABLE POUR L'ÉVALUATION DES PRÉCONDITIONNEMENTS POLYNÔMIAUX

VI.1 Présentation

Nous avons constaté à plusieurs reprises, dans le chapitre précédent, que l'instabilité des préconditionnements polynômiaux semble limiter leur efficacité sur des calculateurs du type de la CM-2. Alors, sur la CM-200 nous avons effectué les calculs avec des réels double précision. Et nous nous sommes aperçus que le surcoût de calcul, provoqué par l'augmentation de la précision, était trop important pour diminuer le temps de résolution. Par contre, cette étude complémentaire nous a montré que le degré optimal (en temps) était supérieur à ceux obtenus à l'aide du schéma de Hörner.

Dans ce chapitre, nous allons essayer de réduire le temps de résolution du système (I.9) en utilisant des schémas d'évaluation des polynômes plus stables que le schéma de Hörner. Ainsi, nous serons capables d'atteindre des degrés bien plus élevés. Malheureusement et parce que ces schémas stables effectuent nécessairement plus d'opérations à chaque itération que le précédent schéma, il n'est pas évident que nous parvenions à réduire le temps de calcul.

Dans une première partie, nous nous servirons des relations de récurrence des polynômes de Tchebycheff, pour évaluer le polynôme Minmax, comme dans [AsMO92] et [CiMP93]. Puis, nous généraliserons la formule de Clenshaw, voir [BjDa74], afin d'obtenir des schémas stables pour les polynômes Mcarre et Leg, [CiMP93]. Enfin, nous testerons et vérifierons l'efficacité de ces méthodes, sur la CM-2, sur les deux problèmes proposés dans cette thèse.

VI.2 Expression de la récurrence à trois termes de Minmax

De manière semblable à [AsMO92], mais en explicitant toutes les relations [CiMP93], cherchons des récurrences pour le polynôme Minmax p_k, nous aurons ainsi une nouvelle technique pour calculer l'étape de préconditionnement $z = P_k(A)r$.

Théorème VI.1 *le polynôme p_k vérifie la relation de récurrence*

$$p_k(\lambda) = \frac{4}{a-b}\frac{c_k}{c_{k+1}} + 2\mu(\lambda)\frac{c_k}{c_{k+1}}p_{k-1}(\lambda) - \frac{c_{k-1}}{c_{k+1}}p_{k-2}(\lambda),$$

avec les conditions initiales

$$p_0(\lambda) = \frac{2}{a+b},$$

$$et \quad p_1(\lambda) = \frac{8(a+b-\lambda)}{a^2+b^2+6ab}.$$

En notant $c_k = T_k(\mu(0))$ (T_k polynôme de Tchebycheff de première espèce degré k).

Preuve VI.1 *Exprimons, d'abord la relation de récurrence des polynômes de Tchebycheff, notés T_k*

$$T_{k+1}(\mu(\lambda)) = 2\mu(\lambda)T_k(\mu(\lambda)) - T_{k-1}(\mu(\lambda)). \tag{VI.1}$$

Pour les mêmes raisons les c_k vérifient

$$c_{k+1} = 2\mu(0)c_k - c_{k-1}. \tag{VI.2}$$

Ensuite, la définition du polynôme Minmax est

$$p_k(\lambda) = \frac{1}{\lambda}\left(1 - \frac{T_{k+1}(\mu(\lambda))}{T_{k+1}(\mu(0))}\right),$$

et inversement

$$\frac{T_j(\mu(\lambda))}{\lambda} = c_j\left(\frac{1}{\lambda} - p_{j-1}(\lambda)\right).$$

Ainsi avec (VI.1) et la définition du polynôme Minmax on obtient

$$p_k(\lambda) = \frac{1}{\lambda} - \frac{1}{c_{k+1}}\left(2\mu(\lambda)\frac{T_k(\mu(\lambda))}{\lambda} - \frac{T_{k-1}(\mu(\lambda))}{\lambda}\right),$$

puis en substituant T_k et T_{k-1} par leurs valeurs, nous avons

$$p_k(\lambda) = \frac{1}{\lambda}\left(1 - 2\mu(\lambda)\frac{c_k}{c_{k+1}} + \frac{c_{k-1}}{c_{k+1}}\right) + 2\mu(\lambda)\frac{c_k}{c_{k+1}}p_{k-1}(\lambda) - \frac{c_{k-1}}{c_{k+1}}p_{k-2}(\lambda).$$

Finalement, avec (VI.2) nous calculons le premier terme

$$\frac{1}{\lambda}\left(1 - 2\mu(\lambda)\frac{c_k}{c_{k+1}} + \frac{c_{k-1}}{c_{k+1}}\right) = \frac{2}{\lambda}\frac{c_k}{c_{k+1}}(\mu(0) - \mu(\lambda)) = \frac{4}{a-b}\frac{c_k}{c_{k+1}}.$$

Regardons maintenant les conditions initiales

$$p_0(\lambda) = \frac{1}{\lambda}\left(1 - \frac{T_1(\mu(\lambda))}{T_1(\mu(0))}\right) = \frac{1}{\lambda}\left(1 - \frac{\mu(\lambda)}{\mu(0)}\right) = \frac{2}{a+b},$$

et

$$p_1(\lambda) = \frac{1}{\lambda}\left(1 - \frac{T_2(\mu(\lambda))}{T_2(\mu(0))}\right) = \frac{1}{\lambda}\left(1 - \frac{2\mu(\lambda)^2 - 1}{2\mu(0)^2 - 1}\right) = \frac{8(a+b-\lambda)}{a^2+b^2+6ab}.$$

Donc, pour évaluer $z = P_k(A)r$ nous effectuons les itérations du schéma suivant (les c_k sont supposés connus, en fait nous les calculons lors de la phase d'initialisation à l'aide par exemple de (VI.2))

$$\begin{cases} z_0 = \dfrac{2}{a+b} r, \\ z_1 = \dfrac{8}{a^2+b^2+6ab}((a+b)I - A)r, \\ z_j = \dfrac{4}{a-b}\dfrac{c_j}{c_{j+1}} r + \dfrac{2}{b-a}\dfrac{c_j}{c_{j+1}}(2A-(a+b)I)z_{j-1} - \dfrac{c_{j-1}}{c_{j+1}} z_{j-2}, \\ \quad \textit{pour } j = 2, \ldots, k. \end{cases}$$

Alors la solution de $z = M^{-1}r = P_k(A)r$ est $z = z_k$. Nous noterons MINMAX cette méthode.

Précisons les notations courantes de cette thèse afin d'éviter toutes confusions entre un polynôme et le préconditionnement polynômial associé à un schéma stable ou pas.

Notation

Nous notons Minmax (respectivement Mcarre et Leg) le polynôme défini en III.7 (respectivement III.9). D'autre part, nous notons avec les majuscules correspondantes (par exemple MINMAX) le préconditionnement polynômial correspondant (içi Minmax), associé à son schéma stable d'évaluation. Enfin, nous ajoutons le préfixe H (HMINMAX), lorsque le préconditionnement polynômial est associé au schéma de Hörner.

VI.3 Formule de Clenshaw étendue à une base donnée

Dans cette partie, nous recherchons des schémas stables pour les polynômes Mcarre et Leg. Ces deux polynômes ont beaucoup de propriétés similaires. Donc, nous avons décrit une manière commune à ces deux polynômes d'obtenir des méthodes stables. Tout de même, pour une utilisation pratique plus rapide, nous expliciterons chaque technique d'évaluation.

Soit $(U_i)_{0\leq i\leq k}$ une base orthonormée de l'espace $\mathcal{P}_k$ associé à un produit scalaire $(.,.)_\omega$.

Alors la suite $(U_i)_{0\leq i\leq k}$ vérifie une récurrence à trois termes de la forme, voir [Davi75], pour $j \geq 1$

$$U_{j+1}(\lambda) = (a_j + b_j\lambda)U_j(\lambda) - c_jU_{j-1}(\lambda), \qquad \text{(VI.3)}$$

Supposons (et nous le vérifierons plus tard, pour différents choix de poids de Jacobi) que VI.3 s'écrive de la manière suivante

$$\text{pour } j \geq 1,\ U_{j+1}(\lambda) = \alpha_j\nu(\lambda)U_j(\lambda) - \gamma_{j-1}U_{j-1}(\lambda), \qquad \text{(VI.4)}$$

avec de plus $U_{-1} \equiv 0$ et $U_0 \equiv A_0$

Dans ce cas, pour calculer les valeurs d'un polynôme p_k, défini par un développement suivant la base orthonormée $(U_i)_{0\leq i\leq k}$ c'est à dire

$$p_k(\lambda) = \sum_{j=0}^{k} p^jU_j(\lambda),$$

il suffit d'utiliser la formule de Clenshaw, décrite par Björck et Dahlquist [BjDa74], et donc d'effectuer les itérations suivantes

$$\begin{cases} z_{k+2} = z_{k+1} = 0, \\ z_j = \alpha_j\nu(\lambda)z_{j+1} - \gamma_jz_{j+2} + p^j, \\ \qquad \text{pour } j = k, \ldots, 0, \end{cases}$$

alors $p_k(\lambda) = U_0z_0 = A_0z_0$.

Or, dans le cas des polynômes des moindres carrés associés aux différents poids de Jacobi, voir III.8 et III.9, la solution est donnée par le polynôme

$$p_k(\lambda) = \sum_{j=0}^{k+1} b_jt_j(\lambda),$$

avec

$$b_j = \frac{s_j(0)}{\sum_{j=0}^{k+1} s_j^2(0)} \text{ et } t_j(\lambda) = \frac{s_j(0) - s_j(\lambda)}{\lambda}$$

Malheureusement, nous ne pouvons pas appliquer immédiatement la formule de Clenshaw, car d'après le Lemme suivant la suite $(t_i)_{0\leq i\leq k+1}$ ne vérifie pas la relation VI.4.

Par contre, les s_j vérifient une telle relation (VI.4) avec de plus

$$\frac{(\nu(0)-\nu(\lambda))}{\lambda} = K, \ K \text{ étant une constante réelle.}$$

Alors on a

Lemme VI.1

pour $j \geq 1$, $t_{j+1}(\lambda) = \delta_{j+1} + \alpha_j \nu(\lambda) t_j(\lambda) - \gamma_{j-1} t_{j-1}(\lambda)$,

en notant $\delta_{j+1} = K\alpha_j s_j(0)$.

Preuve VI.2 *pour* $j \geq 1$,

$$\begin{aligned} t_{j+1}(\lambda) &= (s_{j+1}(0) - s_{j+1}(\lambda))/\lambda \\ &= (\alpha_j \nu(0) s_j(0) - \gamma_{j-1} s_{j-1}(0) - \alpha_j \nu(\lambda) s_j(\lambda) \\ &\quad + \gamma_{j-1} s_{j-1}(\lambda))/\lambda \\ &= \alpha_j \frac{\nu(0)-\nu(\lambda)}{\lambda} s_j(0) + \alpha_j \nu(\lambda) \frac{s_j(0)-s_j(\lambda)}{\lambda} \\ &\quad - \gamma_{j-1} \frac{s_{j-1}(0)-s_{j-1}(\lambda)}{\lambda} \\ &= K\alpha_j s_j(0) + \alpha_j \nu(\lambda) t_j(\lambda) - \gamma_{j-1} t_{j-1}(\lambda). \end{aligned}$$

Ensuite pour évaluer $p_k(\lambda)$, nous généralisons, dans le théorème suivant, la formule de Clenshaw à ce type de polynôme,

---

Théorème VI.2

$$p_k(\lambda) = t_1(\lambda)\eta_1(\lambda) + \omega_2(\lambda),$$

avec d'une part

$$\eta_j(\lambda) = b_j + \alpha_j \nu(\lambda) \eta_{j+1}(\lambda) - \gamma_j \eta_{j+2}(\lambda), \ \textit{pour } j = k+1, \ldots, 1,$$

$$et\ \eta_{k+3} \equiv \eta_{k+2} \equiv 0,$$

et d'autre part

$$\omega_j(\lambda) = \delta_j \eta_j(\lambda) + \omega_{j+1}(\lambda),\ pour\ j = k+1, \ldots, 2,$$

$$et\ \omega_{k+2} \equiv 0.$$

Preuve VI.3 *Notons* $A_k(\lambda) = \sum_{j=0}^{k} b_j t_j(\lambda)$ *et exprimons* p_k *suivant* A_{k-l}.

Posons(H_l) *l'hypothèse suivante*

$$p_k(\lambda) = A_{k-l}(\lambda) + \omega_{k-l+2}(\lambda) + \eta_{k-l+1}(\lambda) t_{k-l+1}(\lambda) - \gamma_{k-l}\eta_{k-l+2}(\lambda) t_{k-l}(\lambda).$$

En premier lieu, $p_k(\lambda) = A_{k+1}(\lambda)$ *donc* H_{-1} *est vérifiée.*
Ensuite, supposons (H_l) *vraie (avec* $l < k$ *évidemment) et montrons* H_{l+1}. *D'après* (H_l) *on a*

$$\begin{aligned}
p_k(\lambda) &= A_{k-l-1}(\lambda) + b_{k-l} t_{k-l}(\lambda) + \omega_{k-l+2}(\lambda) \\
&\quad -\gamma_{k-l}\eta_{k-l+2}(\lambda) t_{k-l}(\lambda) + \eta_{k-l+1}(\lambda) t_{k-l+1}(\lambda)
\end{aligned}$$

puis en utilisant le Lemme (VI.1)

$$\begin{aligned}
&= A_{k-l-1}(\lambda) + b_{k-l} t_{k-l}(\lambda) + \omega_{k-l+2}(\lambda) \\
&\quad -\gamma_{k-l}\eta_{k-l+2}(\lambda) t_{k-l}(\lambda) \\
&\quad +\eta_{k-l+1}(\lambda)(\delta_{k-l+1} + \alpha_{k-l}\nu(\lambda) t_{k-l}(\lambda) - \gamma_{k-l-1} t_{k-l-1}(\lambda)) \\
&= A_{k-l-1}(\lambda) + (b_{k-l} + \alpha_{k-l}\nu(\lambda)\eta_{k-l+1}(\lambda) \\
&\quad -\gamma_{k-l}\eta_{k-l+2}(\lambda)) t_{k-l}(\lambda) + \omega_{k-l+2}(\lambda) + \eta_{k-l+1}(\lambda)\delta_{k-l+1} \\
&\quad -\gamma_{k-l-1}\eta_{k-l+1}(\lambda) t_{k-l-1}(\lambda) \\
&= A_{k-l-1}(\lambda) + \eta_{k-l}(\lambda) t_{k-l}(\lambda) + \omega_{k-l+1}(\lambda) \\
&\quad -\gamma_{k-l-1}\eta_{k-l+1}(\lambda) t_{k-l-1}(\lambda)
\end{aligned}$$

d'après les définitions de $\eta_j(\lambda)$ *et* $\omega_j(\lambda)$.

Donc, (H_{l+1}) *est vérifiée.*

Et finalement pour $l = k$

$$p_k(\lambda) = A_0(\lambda) + \omega_2(\lambda) + \eta_1(\lambda)t_1(\lambda) - \gamma_0\eta_2(\lambda)t_0(\lambda) = \omega_2(\lambda) + \eta_1(\lambda)t_1(\lambda),$$

car $A_0 \equiv 0$ *et* $t_0 \equiv 0$ *(en fait* $t_0(\lambda) = (s_0(0) - s_0(\lambda))/\lambda$ *et* s_0 *est un polynôme de degré* 0*).*

Particulièrement, nous avons utilisé ces relations pour deux sortes de polynômes associées chacune à un poids de Jacobi. La première, Mcarre, est associée au poids de Tchebycheff ($\alpha = \beta = -\frac{1}{2}$) sur $[a, b]$, et la seconde, Leg, au poids de Legendre ($\alpha = \beta = 0$).

VI.3.1 Algorithme pour Mcarre

Pour appliquer le théorème (VI.2), à ces polynômes, il faut d'abord exprimer les s_j, c'est à dire les bases orthonormées correspondantes à ces poids. Or, ce travail a déjà été réalisé au chapitre précédent dans sa partie théorique. Nous avons donc décrit uniquement les relations de récurrence, semblables à VI.4, de ces bases.

Ainsi, pour le polynôme Mcarre, regardons la base orthonormée associée au produit scalaire $(.,.)_\omega$ avec $\omega = (b - \lambda)^{-1/2}(\lambda - a)^{-1/2}$. Elle vérifie la relation suivante

$$\text{pour } j \geq 2,\ s_{j+1}(\lambda) = 2\mu(\lambda)s_j(\lambda) - s_{j-1}(\lambda),$$

donc, dans ce cas, pour $j \geq 1$, $\alpha_j = 2$, $\nu(\lambda) = \mu(\lambda)$, $\gamma_j = 1$ et $\gamma_0 = \sqrt{2}$.

Ainsi, pour le polynôme Mcarre, nous résolvons $P_k(A)r = z$ avec le schéma suivant. Nous initialisons, d'abord, quelques variables

$$s_0(0) = \frac{1}{\sqrt{\pi}},$$

$$s_1(0) = \sqrt{\frac{2}{\pi}}\frac{a+b}{a-b},$$

$$\text{et } s_2(0) = \sqrt{\frac{2}{\pi}}\left(2\left(\frac{a+b}{a-b}\right)^2 - 1\right),$$

ensuite

$$\text{pour } j = 3, \ldots, k+1,\ s_j(0) = 2\mu(0)s_{j-1}(0) - s_{j-2}(0),$$

enfin

$$b_j = \frac{s_j(0)}{\sum_{i=0}^{k+1} s_i^2(0)},\ \text{pour } j = 1, \ldots, k+1.$$

Puis, à chaque étape de préconditionnement, nous effectuons les itérations suivantes

soient z_j et u_j tels que

$$\begin{cases} z_{k+1} &= b_{k+1}r, \\ z_k &= b_k r + \dfrac{2}{b-a}(2A - (a+b)I)z_{k+1}, \\ z_j &= b_j r + \dfrac{2}{b-a}(2A - (a+b)I)z_{j+1} - z_{j+2}, \\ & \text{pour } j = k-1, \ldots, 1, \end{cases}$$

puis

$$\begin{cases} u_{k+1} &= \dfrac{4}{a-b}s_k(0)z_{k+1}, \\ u_{j+1} &= \dfrac{4}{a-b}s_j(0)z_{j+1} + u_{j+2},\ \text{pour } j = k-1, \ldots, 1. \end{cases}$$

Finalement

$$z = \sqrt{\frac{2}{\pi}}\frac{2}{a-b}z_1 + u_2.$$

Notons MCARRE cette méthode.

VI.3.2 Algorithme pour Leg

Ensuite, pour le polynôme Leg, regardons la relation de récurrence de la base orthonormée associée au produit scalaire $(.,.)_1$, voir [Maro92], et donc pour $j \geq 1$,

$$s_j(\lambda) = \frac{(4j^2-1)^{1/2}}{j}\mu(\lambda)s_{j-1}(\lambda) - \frac{j-1}{j}\left(\frac{2j+1}{2j-3}\right)^{1/2} s_{j-2}(\lambda),$$

On retrouve, ainsi les valeurs correspondantes de α_j, ν et γ_{j-1}.

Donc, de la même manière que pour Mcarre et grâce au théorème (VI.2) pour résoudre $P_k(A)r = z$ avec P_k le polynôme Leg, nous procédons de la façon suivante. En premier lieu, on calcule

$$\phi_j = \frac{(4j^2-1)^{1/2}}{j}, \text{ pour } j = 2, \ldots, k+1,$$

et

$$\psi_j = \frac{j-1}{j}\left(\frac{2j+1}{2j-3}\right)^{1/2}, \text{ pour } j = 2, \ldots, k+1.$$

Puis

$$s_0(0) = \sqrt{\frac{1}{b-a}},$$

et

$$s_1(0) = \sqrt{\frac{3}{b-a}}\frac{a+b}{a-b},$$

ensuite

$$\text{pour } j = 2, \ldots, k+1,\ s_j(0) = \phi_j\mu(0)s_{j-1}(0) - \psi_j s_{j-2}(0),$$

enfin

$$b_j = \frac{s_j(0)}{\sum_{i=0}^{k+1} s_i^2(0)}, \text{ pour } j = 1, \ldots, k+1.$$

Ces calculs préliminaires terminés, nous effectuons à chaque étape de préconditionnement les itérations suivantes

soient z_j et u_j tels que

$$\begin{cases} z_{k+1} = b_{k+1}r, \\ z_k = b_k r + \dfrac{1}{b-a}\phi_{k+1}(2A-(a+b)I)z_{k+1}, \\ z_j = b_j r + \dfrac{1}{b-a}\phi_{j+1}(2A-(a+b)I)z_{j+1} - \psi_{j+2}z_{j+2}, \\ \quad \text{pour } j = k-1,\ldots,1, \end{cases}$$

et

$$\begin{cases} u_{k+1} = \dfrac{2}{a-b}\phi_{k+1}s_k(0)z_{k+1}, \\ u_{j+1} = \dfrac{2}{a-b}\phi_{j+1}s_j(0)z_{j+1} + u_{j+2}, \text{ pour } j = k-1,\ldots,1. \end{cases}$$

Pour finir

$$z = \frac{2}{a-b}\sqrt{\frac{3}{b-a}}z_1 + u_2.$$

Notons LEG cette technique.

VI.4 Résultats numériques sur la CM-2

VI.4.1 Problème 1 avec un maillage 512×512

Avant de passer aux résultats numériques, nous pouvons remarquer que les schémas proposés semblent bien adaptés à l'architecture massivement parallèle de la CM-2. En effet, ils n'utilisent que des produits matrices vecteurs et des combinaisons linéaires (comme le schéma de Hörner d'ailleurs). De plus, la phase d'initialisation des coefficients des polynômes est réduite. Seulement quelques coefficients de la base orthonormée sont calculés (comme pour Hörner), nous n'explicitons plus en revanche les coefficients α_k des polynômes. Malheureusement, MINMAX, MCARRE et LEG nécessitent, comme nous nous en apercevrons dans le tableau VI.2, tous plus d'opérations que le schéma de Hörner à chaque itération. Nous ne sommes donc pas certains de diminuer le temps de calcul avec ces schémas.

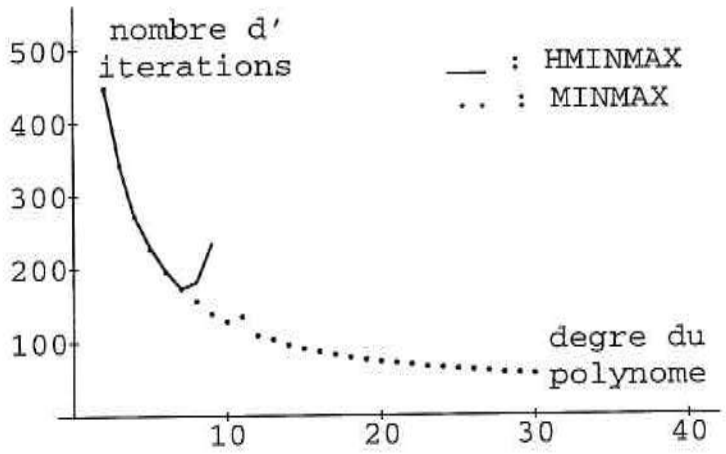

Figure VI.1: Nombre d'itérations de MINMAX et HMINMAX, pour le problème 1 sur la CM-2.

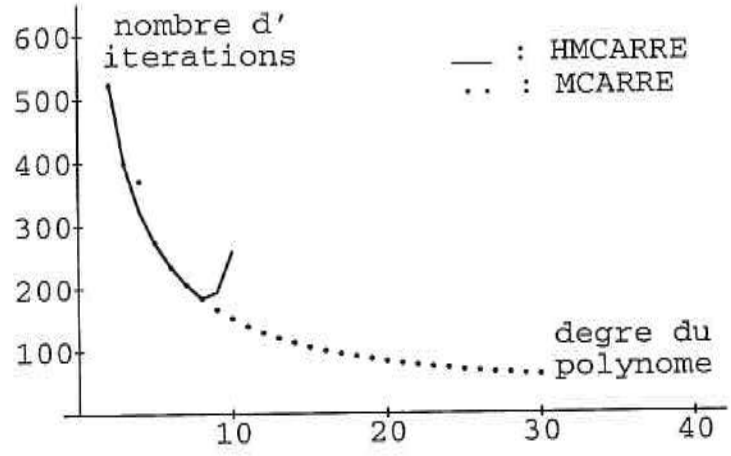

Figure VI.2: Nombre d'itérations de MCARRE et HMCARRE, pour le problème 1 sur la CM-2.

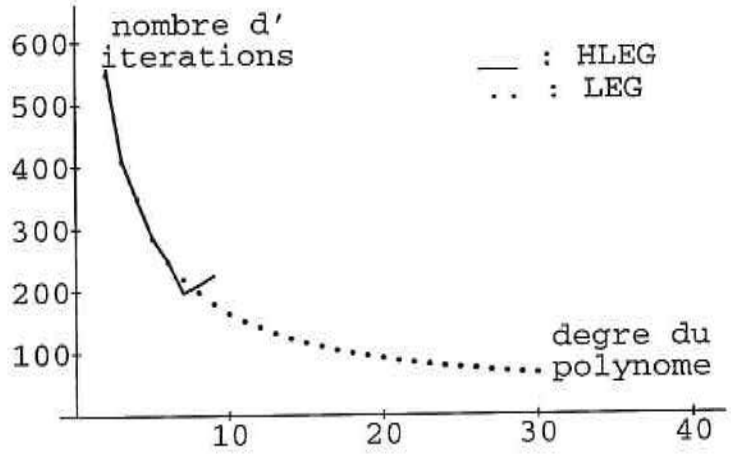

Figure VI.3: Nombre d'itérations de LEG et HLEG, sur le problème 1 sur la CM-2.

Nous avons testé les méthodes MINMAX, MCARRE et LEG sur la CM-2 8K PEs du SEH, d'abord sur le problème 1 avec un maillage 512×512 (et bien entendu toujours avec les mêmes conditions initiales qu'aux chapitres précédents, notamment pour le calcul des valeurs propres indispensables dans la définition des polynômes). Sur les figures VI.1 à VI.3, nous présentons les nombres d'itérations de ces méthodes mais aussi ceux de HMINMAX, HMCARRE et HLEG, correspondants aux schémas de Hörner. Ensuite sur les figures VI.4 à VI.6, nous regardons les temps de résolution de ces nouvelles techniques. Enfin, le tableau VI.1 récapitule les temps optimaux pour chaque méthodes.

Les figures VI.1 à VI.3 montrent que les différences entre les schémas proposés et le schéma de Hörner correspondant aux mêmes polynômes ne sont guère importantes (pas plus d'une ou deux itérations) jusqu'à un certain degré (8 ou 9). Par contre, et comme nous le souhaitions, au-delà des limites de stabilité du schéma de Hörner, les méthodes MINMAX, MCARRE et LEG continuent à converger et restent stables même pour des degrés très élevés, sur ce premier problème. En effet,

ces figures n'exposent des résultats que jusqu'au degré 30, mais nous avons testé des degrés beaucoup plus grands (500). Ces schémas nous permettent donc d'utiliser avec efficacité (du point de vue du nombre d'itérations) des préconditionnements polynômiaux avec des degrés nettement supérieurs à ceux des schémas de Hörner correspondants. Tout de même, nous constatons quelques irrégularités sur ces courbes (MINMAX de degré 11 et MCARRE de degré 4 par exemple). Mais, ces petites imprécisions semblent trop isolées et trop peu nombreuses pour être réellement significatives. Enfin, la résolution des problèmes d'instabilité va sans doute nous permettre d'atteindre des temps de calculs minimum pour ces schémas.

Alors qu'avec le schéma de Hörner le temps minimal de résolution dépendait de la stabilité du schéma, ici le temps minimum de MINMAX, MCARRE et LEG n'est plus limité par la précision de calcul. Ainsi, les degrés optimaux sont respectivement 12, 16 et 10. Ils sont donc tous plus élevés qu'au chapitre précédent. Hélas, pour ces préconditionnements les temps de résolution minimum (respectivement 36.8, 45.4 et 49.0, voir le tableau VI.1) sont, dans ce cas, tous supérieurs à ceux réalisés par les schémas de Hörner correspondants (respectivement 33.7, 39.6 et 42.5, voir le tableau VI.1 et les résultats du chapitre précédent). En effet, à chaque étape de préconditionnement nous effectuons avec ces méthodes plus d'opérations qu'avec Hörner pour un même préconditionnement polynômial, voir remarque VI.1. Ainsi sur ce problème, nous ne sommes pas parvenus à optimiser les temps de calculs des préconditionnements polynômiaux à l'aide des schémas introduits.

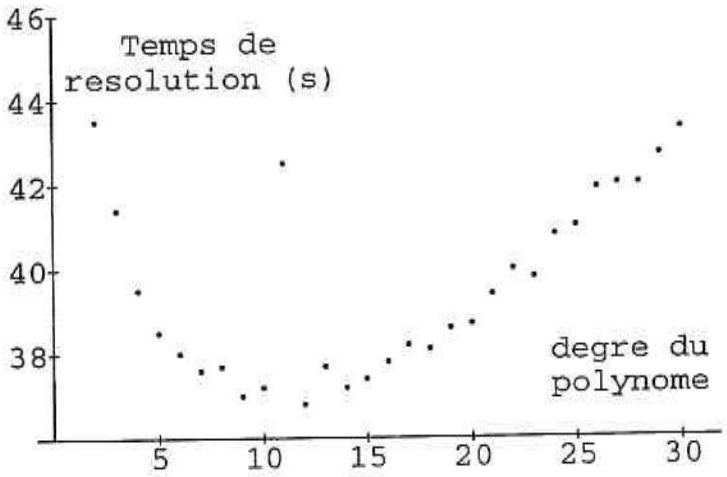

Figure VI.4: Temps de résolution de MINMAX, pour le problème 1 sur la CM-2.

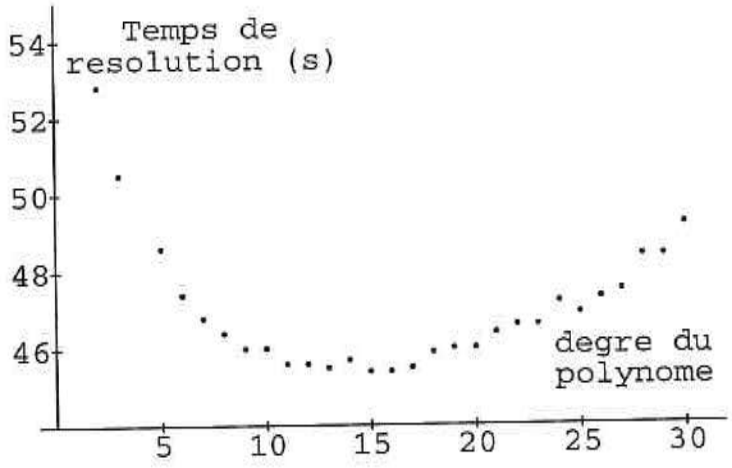

Figure VI.5: Temps de résolution de MCARRE, pour le problème 1 sur la CM-2.

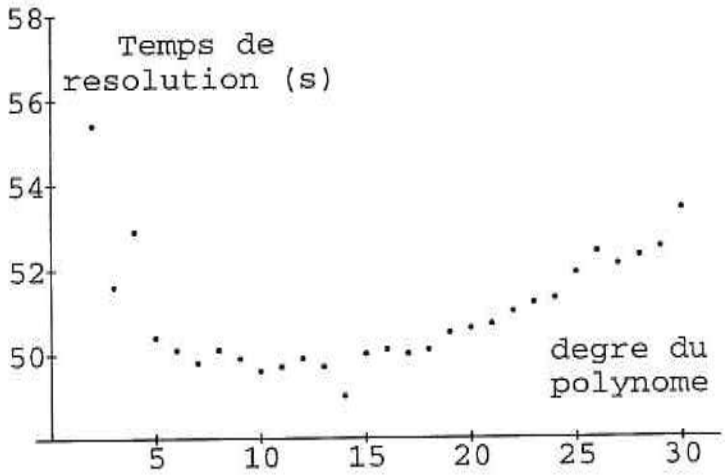

Figure VI.6: Temps de résolution de LEG, pour le problème 1 sur la CM-2.

Preconditionnement	Degré optimal	Nombre d' itérations	Temps total (s)	Mflops
DIAG	0	1521	70.5	124.5
NEUM1	1	761	54.1	114.4
NEUM3	3	537	58.1	119.0
MINMAX	12	109	36.8	164.4
HMINMAX	7	173	33.7	134.0
MCARRE	16	100	45.4	171.6
HMCARRE	8	184	39.6	134.4
LEG	10	163	49.6	174.4
HLEG	8	197	42.5	134.4

Tableau VI.1: Résultats optimaux pour toutes les méthodes, pour le problème 1 sur la CM-2.

Nous allons apercevoir sur le second problème que les résultats des préconditionnmenets polynômiaux obtenus avec ces techniques plus stables peuvent améliorer les résultats obtenus à l'aide du schéma de Hörner. Et même sur ce problème, avec une machine plus puissante et de même type, voir seconde annexe, nous sommes parvenus à diminuer les temps de résolution des préconditionnements polynômiaux associés au schéma de Hörner par l'utilisation de ces schémas plus stables.

Remarque VI.1 *Dans le tableau VI.2, nous donnons le nombre d'opérations nécessaires, dans le cas d'une matrice pentadiagonale, à l'étape de préconditionnement du G.C.P pour chaque schéma d'évaluation des polynômes, de degré k ($k \geq 2$), en fonction de la taille N du problème étudié.*

Schéma d'évaluation	*Hörner*	*MINMAX*	*MCARRE*	*LEG*
Nombre d' opérations	*(1+11k)N*	*(-3+16k)N*	*(1+17k)N*	*18kN*

Tableau VI.2: Nombre d'opérations du préconditionnement.

Evidemment, le schéma de Hörner possède le nombre d'opérations le moins important. Ainsi, à degré égal et pour le même type de polynôme son temps de résolution (tant qu'il reste convergent évidemment) est inférieur à celui du schéma stable correspondant. Et donc, comme sur ce problème le degré optimal des schémas stables n'est pas encore suffisamment grand, les préconditionnements polynômiaux associés au schéma de Hörner sont, dans ce cas, les méthodes les plus rapides. De plus, les différences de temps entre MINMAX et MCARRE (et aussi LEG) se sont accrues par rapport au schéma de Hörner car d'une part MINMAX effectue moins d'itérations (d'ailleurs ces trois polynômes respectent à nouveau la hiérarchie énoncée au chapitre V et pour des degrés plus élevés), et d'autre part cette méthode a besoin de moins de calculs pour résoudre l'étape de préconditionnement. Finalement, le surcoût de calcul provenant de l'introduction des schémas stables nous a non seulement empêché de réduire le temps de calcul du schéma de

Hörner, mais il a aussi favorisé le polynôme Minmax par rapport aux autres préconditionnements polynômiaux.

Remarque VI.2 *Les irrégularités remarquées sur les nombres d'itérations se répercutent évidemment de manière encore plus conséquente sur les temps de résolution. Ainsi, nous pouvons constater, par exemple, que pour MINMAX de degré 11 son temps de calcul est équivalent à des degrés 2 ou 3 (même remarque pour Mcarre de degré 4). Néanmoins, ces petites irrégularités ne nous empêchent nullement d'atteindre les temps optimaux pour ces méthodes.*

VI.4.2 Problème 2 avec un maillage 256×256

Testons maintenant les schémas stables sur la matrice provenant du problème 2 avec un maillage 256×256. En outre, nous avons conservé la même évaluation des valeurs propres qu'aux chapitres IV et V, voir les remarques IV.1 et IV.2, pour définir ces préconditionnements polynômiaux. Le nombre d'itérations relativement plus élevé, engendré par ce problème, va sans doute nous permettre d'obtenir des résultats positifs pour des degrés plus conséquents. Ainsi, nous espérons diminuer, grâce aux schémas d'évaluation stables MINMAX, MCARRE et LEG, les temps de calculs réalisés à l'aide du schéma de Hörner.

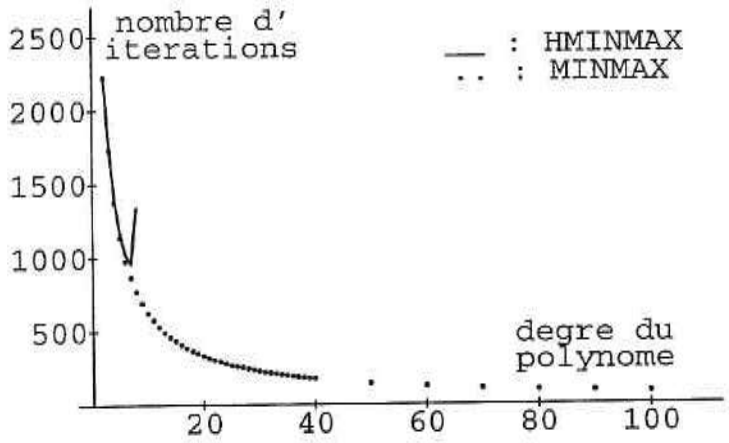

Figure VI.7: Nombre d'itérations de MINMAX et HMINMAX, pour le problème 2 sur la CM-2.

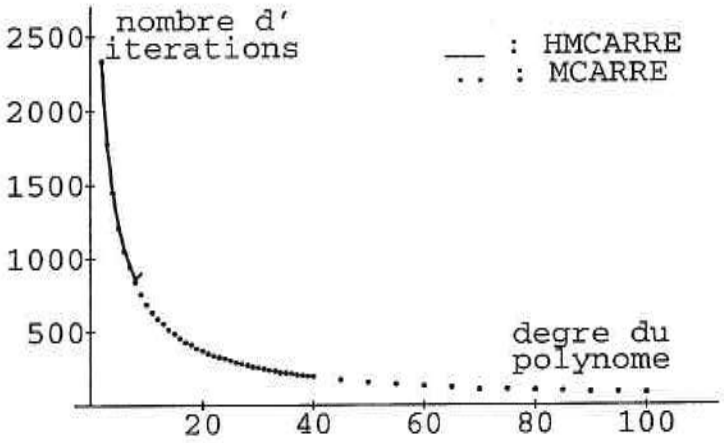

Figure VI.8: Nombre d'itérations de MCARRE et HMCARRE, pour le problème 2 sur la CM-2.

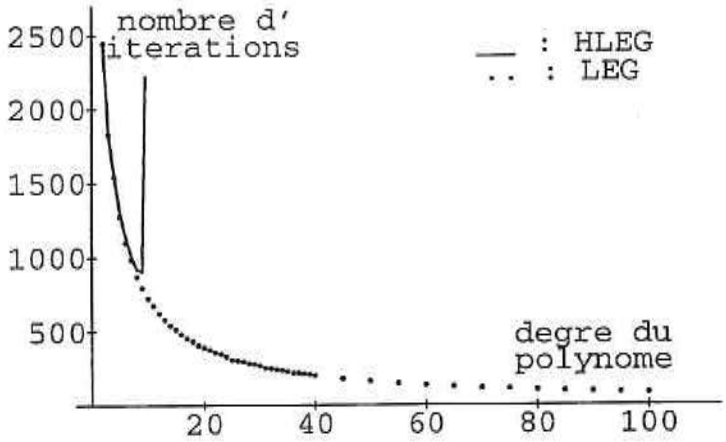

Figure VI.9: Nombre d'itérations de LEG et HLEG, sur le problème 2 sur la CM-2.

D'après les figures VI.7 à VI.9, nous constatons, comme au précédent problème, que les schémas proposés sont très stables. Nous montrons même sur ces figures des degrés bien plus élevés. De plus, même les petites irrégularités ont disparu. La stabilité de ces schémas semble donc être indépendante du conditionnement de la matrice, sur ce

type de problème. Contrairement au schéma de Hörner où l'erreur sur le calcul des coefficients α_k des polynômes s'amplifiait d'autant plus que le nombre d'itérations était important, ce qui entraînait une apparition encore plus rapide des instabilités, les méthodes MINMAX, MCARRE et LEG semblent rester efficaces même sur des problèmes mal conditionnés. Enfin, la hiérarchie proposée est une nouvelle fois respectée, néanmoins pour des degrés très grands les écarts deviennent moins sensibles (5 itérations pour le degré 100).

Maintenant, comme ce problème est mal conditionné (donc le degré optimal en temps va surement augmenter) et comme le schéma de Hörner est instable plus rapidement que pour le premier problème, nous espérons que nous allons réussir à diminuer les temps de calculs réalisés par les préconditonnements polynômiaux associés au schéma de Hörner.

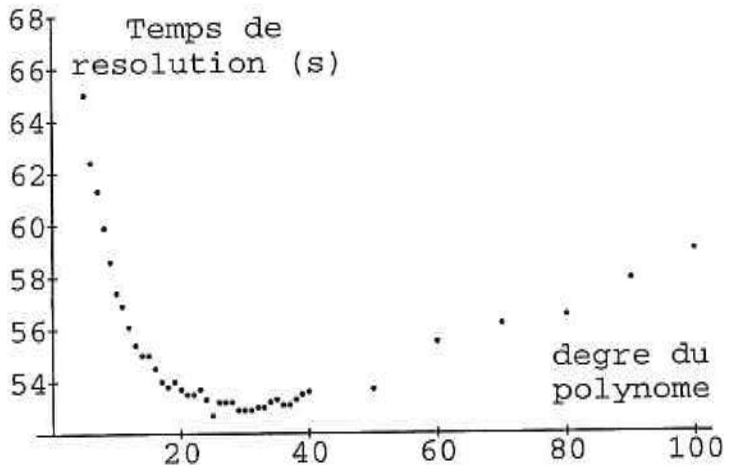

Figure VI.10: Temps de résolution de MINMAX, pour le problème 2 sur la CM-2.

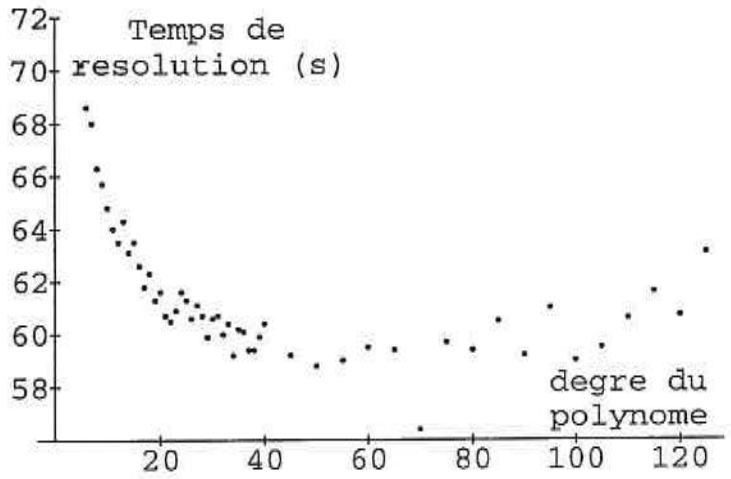

Figure VI.11: Temps de résolution de MCARRE, pour le problème 2 sur la CM-2.

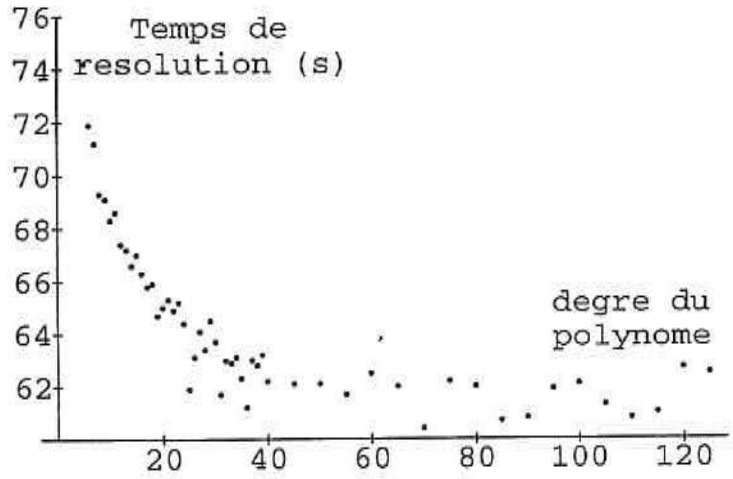

Figure VI.12: Temps de résolution de LEG, pour le problème 2 sur la CM-2.

Preconditionnement	Degré optimal	Nombre d' itérations	Temps total (s)	Mflops
DIAG	0	7368	129.0	82.3
NEUM1	1	3792	108.5	71.0
NEUM3	3	3238	126.3	82.4
MINMAX	31	223	52.9	142.7
HMINMAX	6	1026	60.2	98.4
MCARRE	70	108	56.4	153.4
HMCARRE	8	863	61.2	101.7
LEG	70	115	60.4	161.3
HLEG	8	921	65.7	101.7

Tableau VI.3: Résultats optimaux pour toutes les méthodes, pour le problème 2 sur la CM-2.

Sur les figures VI.10 à VI.12, voir également le tableau VI.3, les temps de résolution de MINMAX, MCARRE et LEG sont atteints avec des degrés (respectivement 25, 70 et 70) encore plus grands qu'avec le problème 1. En effet, ce problème est mal conditionné, donc à chaque augmentation du degré du polynôme la diminution du nombre d'itérations reste plus conséquente plus longtemps que pour le premier problème. On explique d'ailleurs de la même manière le degré optimal plus élevé des méthodes MCARRE et LEG. D'autre part, les différences de temps de résolution, pour les polynômes de très hauts degrés, découlent en partie maintenant du nombre d'opérations des schémas utilisés car les nombres d'itérations sont sensiblement équivalents.

De plus, grâce au degré optimal plus élevé et à la divergence plus rapide du schéma de Hörner, nous sommes parvenus à obtenir des temps de résolution inférieurs à ceux des préconditionnements polynômiaux évalués par le schéma de Hörner, voir tableau VI.3. Ainsi, le gain de MINMAX de degré 25 par rapport à HMINMAX 6 est de l'ordre de 12%, ce qui nous amène sur ce problème à une diminution de 59% par rapport à DIAG. De même, MCARRE et LEG de degré 70 diminuent les

temps de calculs de HMCARRE et HLEG de degré 8 de respectivement 8% et 7.5%. Les résultats de MCARRE et LEG sont moins bons que ceux de MINMAX certainement à cause du surcoût de calculs de ces méthodes, mais aussi parce que HMINMAX diverge plus rapidement que HMCARRE et HLEG sur ce problème.

Enfin, maintenant que nous avons la possiblité de choisir, indépendamment de la précision de calcul, le degré du préconditionnement polynômial, il semble que la recherche du degré optimal en temps soit un problème assez difficile à résoudre. L'expérience des préconditionnements polynômiaux et des machines massivement parallèles acquise au cours de cette thèse, nous permet néanmoins de donner quelques conseils pour l'estimation de ce degré optimal. En premier lieu, ce degré dépend du conditionnement de la matrice. En fait, plus le problème est mal conditionné plus le degré optimal est élevé (voir les différences entre les problèmes 1 et 2), et donc plus nous avons la possibilité de réaliser un gain de temps substantiel avec les schémas stables. Il est fortement lié aussi à la puissance de calculs de la machine utilisée. En fait le degré optimal dépend fortement de la vitesse du produit matrice vecteur, voir chapitre IX. D'autre part, l'un des avantages des préconditionnements polynômiaux associés à un schéma plus stable est qu'il n'est pas nécessaire de choisir avec précision le degré du polynôme pour obtenir un temps de résolution proche du temps optimal. En effet contrairement au schéma de Hörner où des écarts de un degré entrainent des écarts de temps de résolution importants, voir le chapitre précédent, nous constatons sur les figures VI.4 à VI.6 et VI.10 à VI.12 que la pente de la courbe des temps de résolution n'est pas très élevé pour des degrés compris dans un certain intervalle. Les bornes et la longueur de cet intervalle dépendent, comme le degré optimal, de la vitesse du produit matrice vecteur, du problème résolu et du polynôme utilisé. Nous pouvons ainsi considérer que pour des degrés compris dans les intervalles suivants : [5,20] pour le problème 1 avec MINMAX, [7,20] pour le problème 1 avec MCARRE et LEG et [10,60] pour le problème 2 avec MINMAX, [30,100] pour le problème 2 avec MCARRE et LEG, les méthodes correspondantes ont un temps de résolution relativement satisfaisant.

En conclusion, ces schémas paraissent bien adaptés aux problèmes mal conditionnés, mais aussi aux supercalculateurs tels que la CM-2 (également à la future génération de calculateurs SIMD ou plus simplement à des supercalculateurs proposant un produit matrice vecteur efficace). Mais n'oublions, pas tout de même, que la structure très régulière de la matrice étudiée, nous a permis de construire un produit matrice vecteur très rapide. Donc sur des problèmes moins réguliers, l'efficacité de ces schémas, ainsi que des préconditionnements polynômiaux en général, dépendra fortement de la vitesse du produit matrice vecteur disponible, voir chapitre IX. Néanmoins, les nombreuses études réalisées avec les séries de Neumann tronquées sur des problèmes moins réguliers [BPRS91], [PeWe92], [Peti93], [EdPe93], [FPSW93] et [Weil94] laissent présager de bons résultats pour les préconditionnements polynômiaux sur des structures moins régulières avec la CM-2.

Avant de clore ce chapitre, nous présentons sur la figure VI.13 les Mflops réalisées par ces méthodes. Nous constatons ainsi que la vitesse de calcul est une nouvelle fois une fonction croissante du degré du polynôme et que cette vitesse est limitée par la rapidité du produit matrice vecteur (et donc par la puissance de calculs de la machine). Par ailleurs, l'ordre des vitesses est en fait établi par le nombre d'opérations de l'étape de préconditionnement. En effet, par exemple à degré égal LEG effectue plus de calculs que MINMAX. Comme ces calculs supplémentaires n'ont pas besoin de communications (saxpy), LEG atteint une vitesse plus élevée.

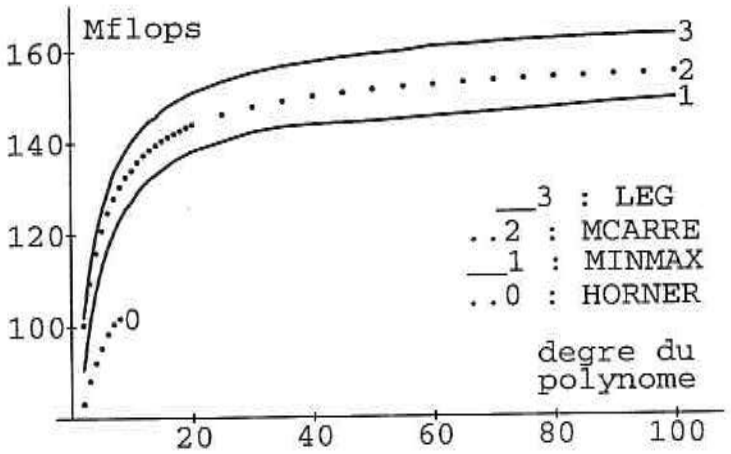

Figure VI.13: Comparaison de Mflops, pour le problème 2.

Remarque VI.3 *Nous avons présenté dans la remarque V.5 des taux de parallélisme pour divers préconditionnements sur la CM-200. Pour les schémas stables nous avons réalisé aussi quelques mesures aussi précises que possibles sur la CM-2. Alors, pour le problème 1 le taux de parallélisme de MINMAX est 0.895 pour le degré 12 et 0.985 pour le degré 100. Nous considérons que ces taux sont relativement satisfaisants surtout par rapport à celui de DIAG (de l'ordre de 0.97). Ils sont toutefois moins intéressants que sur la CM-200. Mais, nous rappelons que ces taux ne sont donnés qu'à titre indicatif car ils ne sont hélas guère fiables, surtout sur la CM-2.*

VI.5 Conclusions

Nous venons de voir dans ce chapitre que des schémas plus stables permettaient d'utiliser des préconditionnements polynômiaux de degrés élevés, alors qu'avec le schéma de Hörner simple nous ne pouvions

dépasser le degré 8. Néanmoins, le surcoût de calcul occasionné par ces méthodes nous a empêché de réduire le temps de calculs pour le classique problème de Poisson. Par contre, pour le second problème nous nous sommes placés dans des conditions beaucoup plus réalistes d'utilisation, en estimant de manière succincte les valeurs propres de la matrice. Dans ce cas, les résultats sont positifs. Donc, pour des problèmes mal conditionnés, ils semblent que ces schémas puissent diminuer les temps de résolution. Or, on recherche souvent des préconditionnements efficaces lorsque les problèmes sont mal conditionnés, car alors le G.C préconditionné par la Diagonale perd en partie son efficacité. En tout cas, sur des machines telles que la CM-2, les préconditionnements polynômiaux associés à un schéma stable ou pas (suivant le conditionnement de la matrice et suivant la puissance de la machine) améliorent énormément le rendement du G.C.P, pour le type de problème étudié (régulier). De plus l'utilisation de schémas stables permet une estimation moins précise du degré optimal (pour le temps de résolution) sans perte importante de temps (contrairement à Hörner).

Nous pouvons signaler que certaines simulations numériques, sur CM-2 ou CM-5, durent parfois plusieurs jours, la résolution de systèmes linéaires prenant une part importante du temps, voir [Uebe94]. Il devient alors très important de réduire les temps de résolution de manière conséquente, aifn de pouvoir rentabiliser au mieux des machines qui coutent très chères. Et dans ce cas, les gains en temps réalisés dans cette thèse prennent toute leur importance. Il faudrait, néanmoins, tester ces préconditionnements sur d'autres types de matrices, notamment avec des squelettes moins réguliers, pour vérifier leur efficacité dans tous les cas.

Finalement, le plus gros défaut de ces préconditionnements reste qu'ils ont besoin d'une estimation de la plus petite et de la plus grande valeur propre de la matrice étudiée. D'autre part, à cause de leurs schémas d'évaluation trop coûteux en nombres d'opérations, les polynômes Mcarre et Leg de degré élevé ont été fortement pénalisés par rapport à Minmax. Par conséquent, nous proposons dans le chapitre suivant un préconditionnement polynômial ne nécessitant aucune estimation des valeurs propres, plus précis que les séries de Neumann

tronquées et possédant un schéma d'évaluation stable dont le nombre d'opérations est équivalent à celui de MINMAX.

CHAPITRE VII

UN PRÉCONDITIONNEMENT POLYNÔMIAL INDÉPENDANT DES VALEURS PROPRES:NORM

VII.1 Introduction

Nous venons de nous rendre compte, lors des deux précédents chapitres, de l'efficacité des préconditionnements polynômiaux sur les machines massivement parallèles SIMD à mémoire locale. Malheureusement, l'un des principaux défauts de ces préconditionnements est qu'ils nécessitent une estimation de la plus petite et de la plus grande valeur propre de la matrice étudiée. Pour le classique problème de Poisson nous avons utilisé une expression analytique des valeurs propres. Par contre, pour le second problème nous possédions moins d'informations. Ainsi, grâce à une normalisation symétrique de la matrice par sa diagonale et grâce au Théorème de Gershgörin, voir le Lemme III.1, nous avons évalué la plus grande valeur propre par $b = 2$. C'est à dire, qu'au lieu de résoudre $Ax = b$ nous avons résolu avec $D = diag_N(a_i)$

$$\tilde{A}y = D^{-1/2}AD^{-1/2}y = D^{-1/2}b \; puis \; D^{1/2}x = y. \qquad \text{(VII.1)}$$

Ensuite, nous avons pris une valeur a suffisamment petite pour que l'intervalle $[a, b]$ contienne l'ensemble des valeurs propres. On peut se demander pourquoi nous n'avons pas, comme [Saad85] avec le polynôme Mcarre, tout simplement essayé $a = 0$? Le préconditionnement ainsi obtenu aurait été alors indépendant des valeurs propres de la matrice, de la même manière que les préconditionnements polynômiaux basés sur les séries de Neumann tronquées. Il faut tout de même se rappeler que le polynôme Minmax avait perdu en partie son efficacité entre le premier et le second problème, voir chapitre V. Dans le cas $a = 0$, les préconditionnements ne vont-ils pas devenir totalement inutilisables?

Nous allons donc voir dans une première partie, de manière pratique mais aussi théorique, les raisons qui entraînent l'inefficacité du polynôme Minmax, quand $a = 0$. Puis, après avoir constaté que Mcarre avait un comportement différent [Saad85], nous utiliserons une propriété de ce nouveau préconditionnement polynômial pour définir un autre schéma stable [AsMO92]. Finalement, nous testerons ce préconditionnement sur la CM-2.

VII.2 Polynôme Minmax, avec $a = 0$

Dans un premier temps, nous avons cherché la solution de (VII.1) pour le problème 1 avec un maillage 512×512, sur la CM-2, pour des valeurs de a autour de sa valeur exacte (de l'ordre de 0.025) calculée à l'aide de (I.13).

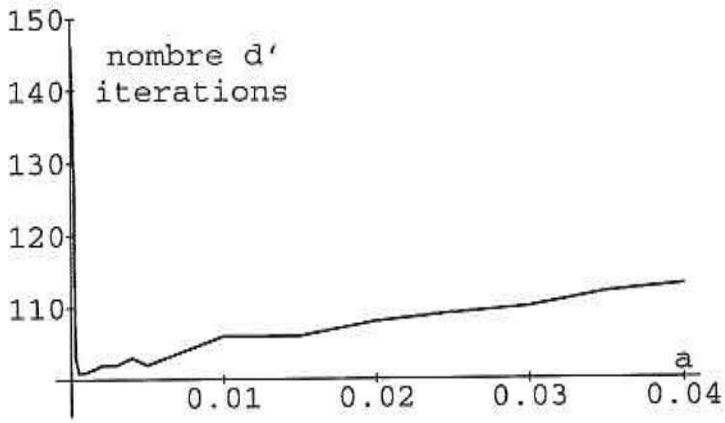

Figure VII.1: Nombre d'itérations de MINMAX de degré 12 en variant a.

Ainsi, d'après la figure VII.1 nous effectuons plusieurs remarques : tout d'abord la valeur propre exacte n'est pas toujours pour ce problème la valeur optimale de a, du point de vue du nombre d'itérations. Sur ce problème, il semble que jusqu'a environ $0.24\ 10^{-3}$ les préconditionnements ainsi définis soient plus efficaces qu'avec la valeur exacte. Mal-

heureusement, passée cette valeur, le préconditionnement devient moins rentable, il est même totalement inutilisable quand $a = 0$. D'autre part, dès que l'on utilise une valeur supérieure, le préconditionnement perd une partie de son efficacité. L'estimation de la plus petite valeur propre a donc des conséquences très importantes sur l'utilisation de Minmax.

Nous allons regarder graphiquement le comportement du polynôme Minmax sur l'intervalle $[0, 2]$, quand a tend vers 0. Notons p le polynôme Minmax de degré 6. Alors, pour obtenir une approximation satisfaisante de A^{-1}, il faut, dans un certain sens, que le polynôme $xp(x)$ soit proche de la constante 1, tout en restant strictement positif (pour le préconditionnement soit défini positif), sauf en 0 évidemment.

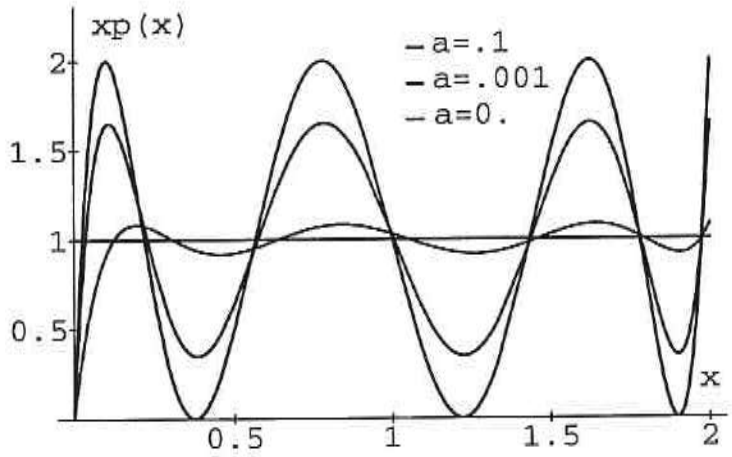

Figure VII.2: Polynômes Minmax de degré 6, avec des variations de a.

La figure VII.2 montre que les oscillations autour de 1 du polynôme $xp(x)$ s'accroissent au fur et à mesure que a se rapproche de zéro. Par conséquent, la précision du préconditionnement Minmax décroit quand a est près de zéro. De plus, il apparaît que si $a = 0$ l'équation p(x)=0 a des solutions, et donc dans ce cas le préconditionnement n'est plus inversible.

soit p_k le polynôme Minmax défini en III.7 par

$$p_k(\lambda) = \frac{1}{\lambda}\left(1 - \frac{T_{k+1}(\mu(\lambda))}{T_{k+1}(\mu(0))}\right).$$

avec

$$\mu(\lambda) = \frac{2\lambda - b - a}{b - a}.$$

Cherchons, pour b quelconque, les solutions dans $[a, b]$ de

$$p_k(\lambda) = 0 \tag{VII.2}$$

Théorème VII.1 *si $a = 0$ alors l'équation (VII.2) admet, dans l'intervalle $[a, b]$, p solutions, avec k=2p-1 ou 2p, qui sont donnés par*

si $k = 2p - 1$

$$\text{pour } i = 0, \ldots, p-1,\ \lambda_i = \frac{b}{2}\left(cos\left(\frac{i}{p}\pi\right) + 1\right), \tag{VII.3}$$

si $k = 2p$

$$\text{pour } i = 0, \ldots, p-1,\ \lambda_i = \frac{b}{2}\left(cos\left(\frac{2i+1}{2p+1}\pi\right) + 1\right). \tag{VII.4}$$

Enfin, si $a \neq 0$ alors (VII.2) n'admet aucune solution dans $[a, b]$.

Preuve VII.1 *Regardons d'abord quand a=0.*

Dans ce cas, $\mu(0) = -1$ et alors $T_{k+1}(\mu(0)) = (-1)^{k+1}$.

Donc λ_i est solution de III.1 si et seulement si

$$(-1)^{k+1} T_{k+1}(\mu(\lambda_i)) = 1.$$

Or T_k possède $k + 1$ extrema, voir par exemple [LaTh86], atteint pour

$$x_i = cos\left(\frac{i\pi}{n}\right),\ i = 0, \ldots, k, \tag{VII.5}$$

pour lesquels $T_k(x_i) = (-1)^i$.

Si k=2p, il faut donc $T_{k+1}(\mu(\lambda_i)) = -1$, c'est à dire d'après VII.5

$$\mu(\lambda_i) = cos\left(\frac{2i+1}{2p+1}\pi\right).$$

Ensuite, on procède de même pour k=2p-1, et on retrouve ainsi les valeurs de VII.3. On remarque, enfin, que toutes les solutions de VII.2, décrite par VII.3 ou VII.4, sont dans $[a, b]$.

Maintenant pour $a \neq 0$, *il est clair que* $|\mu(0)| > 1$,

donc $|T_{k+1}(\mu(0))| > 1$.

Or, $\forall \lambda \in [a, b]$, $|T_{k+1}(\mu(\lambda))| \leq 1$,
donc le polynôme p_k *ne peut s'annuler sur l'intervalle* $[a, b]$.

Nous pouvons donc conclure que le polynôme Minmax devient totalement inefficace lorsque $a = 0$. Néanmoins, nous avons appris, de la même manière que [AsMO92], par cette étude que sur ce problème modèle, voir en particulier la figure VII.1, la connaissance de la valeur propre exacte n'est pas absolument indispensable. Une approximation un peu inférieure peu même rendre le préconditionnement un peu plus rentable, pour ce problème. Par contre, il vaut mieux éviter non seulement que cette évaluation soit trop proche de zéro, mais également qu'elle soit supérieure à la valeur propre exacte.

VII.3 Polynôme Mcarre, avec $a = 0$

Dans [Saad85], voir aussi [AsMO92], Y. Saad a déja noté que contrairement au polynôme Minmax, il semble que le polynôme Mcarre soit relativement insensible à une mauvaise estimation de a. Il propose même de prendre directement $a = 0$. Alors, pour comparer les comportements, suivant a, de ces deux préconditionnements polynômiaux, nous avons étudié le préconditionnement Mcarre de la même manière que Minmax.

D'après la figure VII.3, nous déduisons que, sur ce problème, le polynôme Mcarre est relativement insensible à une mauvaise évaluation de a. De plus, pour $a = 0$ le préconditionnement est aussi efficace qu'avec la valeur propre exacte, dans ce cas le nombre d'itérations est en effet identique. Par ailleurs, alors que pour le polynôme Minmax une estimation de a inférieure à la valeur propre exacte permettait d'obtenir un nombre d'itérations plus petit (huit itérations sur ce problème), pour

le polynôme Mcarre l'écart est infime (pas plus de une itération). Enfin, si a est supérieur à la valeur propre exacte, le préconditionnement perd en partie son efficacité.

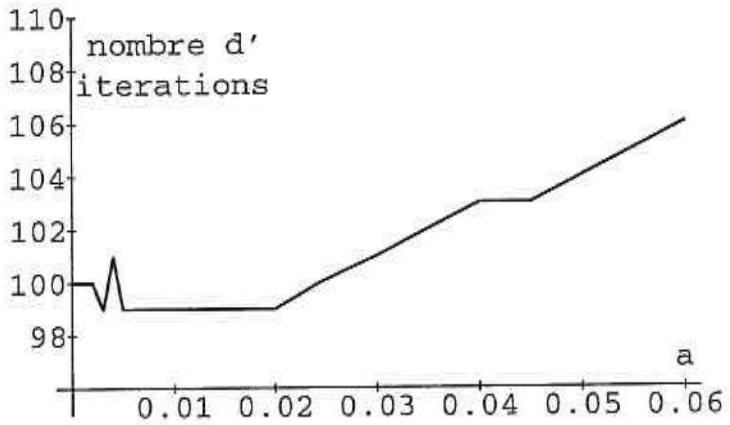

Figure VII.3: Nombre d'itérations de MCARRE de degré 16 en variant a, pour le problème 1.

Regardons graphiquement, dans un second temps, le comportement du polynôme Mcarre quand a tend vers zéro.

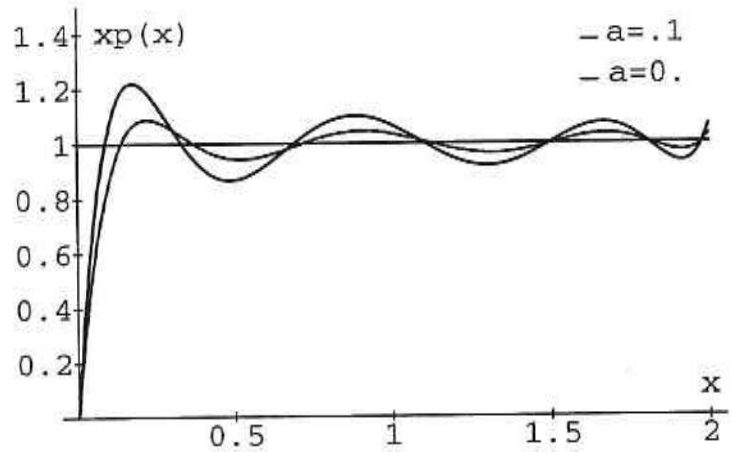

Figure VII.4: Polynômes Mcarre de degré 16, avec des variations de a.

Sur la figure VII.4, nous constatons que contrairement au polynôme Minmax, voir figure VII.2, la dépendance en a du polynôme Mcarre est moins forte. Mieux même, lorsque $a = 0$ le polynôme ne s'annule plus. Montrons donc que le polynôme Mcarre ne peut s'annuler sur $[0, b]$.

soit p_k le polynôme Mcarre défini en III.9 par

$$p_k(\lambda) = \sum_{j=0}^{k+1} b_j t_j(\lambda),$$

avec

$$b_j = \frac{s_j(0)}{\sum_{i=0}^{k+1} s_i^2(0)} \text{ et } t_j(\lambda) = \frac{s_j(0) - s_j(\lambda)}{\lambda}.$$

et plus pour Mcarre nous avons,

$$\textit{pour } j \neq 0,\ s_j(\lambda) = \sqrt{\frac{2}{\pi}} T_j(\mu(\lambda)),$$

recherchons, pour b quelconque et $a = 0$, les solutions, dans $[0, b]$, de

$$p_k(\lambda) = 0. \tag{VII.6}$$

Or, nous avons le théorème suivant

Théorème VII.2 *L'équation VII.6 n'admet aucune solution dans l'intervalle* $[0, b]$.

Preuve VII.2 *Pour* $a = 0$, $s_j(0) = \sqrt{2/\pi}(-1)^j$, *avec* $j \neq 0$.

Ainsi,

$$\sum_{i=0}^{k+1} s_i^2(0) = \frac{1}{\pi} + \frac{2}{\pi} \sum_{i=0}^{k+1} 1 = \frac{2k+3}{\pi},$$

puis

$$p_k(\lambda) = \sum_{j=1}^{k+1} b_j t_j(\lambda) = \frac{2}{2k+3} \frac{1}{\lambda} \sum_{j=1}^{k+1} \left(1 - (-1)^j T_j(\mu(\lambda))\right).$$

Or,

$$\forall \lambda \in [0, b], |T_j(\mu(\lambda))| \leq 1,$$

donc pour que λ soit solution de VII.6, il faut que

$$\forall 1 \leq j \leq k+1, 1-(-1)^j T_j(\mu(\lambda)) = 0.$$

D'après la définition des extrema des polynômes de Tchebycheff, voir VII.5, il est clair qu'il n'existe aucun λ vérifiant ces dernières égalités. Donc, VII.6 n'a pas de solution dans $[0, b]$. Cette propriété du polynôme Mcarre provient du fait que deux polynômes de Tchebycheff n'atteignent pas leurs extrema aux mêmes points (sauf en zéro).

Remarque VII.1 *Nous pouvons réaliser des raisonnements identiques à celui du polynôme Mcarre, pour le polynôme Leg. Mais, comme dans les études précédentes, voir chapitres V et VI, les nombres d'itérations du préconditionnement Leg sont supérieurs à ceux de Mcarre, nous ne testerons pas le préconditionnement Leg pour $a = 0$.*

VII.4 Schéma stable pour Norm

Non seulement Mcarre semble très peu sensible quand a est proche de 0, mais dans le cas $a = 0$, on peut alors le définir de manière différente. Ainsi, la solution de III.8 est donnée, voir [Saad85] et [AsMO92] pour plus de détails, par

$$p_k(\lambda) = \frac{1}{\lambda}\left(1 - \frac{J_{k+1}(1-\lambda)}{J_{k+1}(1)}\right), \qquad \text{(VII.7)}$$

avec $J_{k+1}(\lambda)$ le polynôme de Jacobi sur $[-1, 1]$ associé au poids $\omega(\frac{1}{2}, -\frac{1}{2}, \lambda)$, voir [Davi75] ou [Maro92].

Ce résultat provient de la λ−orthogonalité du polynôme résiduel, voir [Stie58],

$$R_{k+1}(\lambda) = \frac{\sum_{j=0}^{k+1} s_j(0)s_j(\lambda)}{\sum_{j=0}^{k+1} s_j^2(0)}.$$

De plus, comme les polynômes de Tchebycheff, les polynômes de Jacobi J_k vérifient une relation de récurrence à trois termes, voir [Maro92]

$$J_{k+1}(x) = \frac{2k+1}{k+1} x J_k(x) - \frac{(k+1/2)(k-1/2)}{k(k+1)} J_{k-1}(x). \qquad \text{(VII.8)}$$

Par conséquent, au lieu de réaliser l'étape de préconditionnement à l'aide du schéma MCARRE, nous allons utiliser, de la même manière que pour le polynôme Minmax, la relation VII.8, pour décrire un algorithme d'évaluation plus stable que le simple schéma de Hörner et nécessitant moins d'opérations que MCARRE.

Théorème VII.3 *Posons* $d_k = J_k(1)$,
alors le polynôme p_k *vérifie la relation de récurrence suivante*

$$p_k(\lambda) = \frac{(2k+1)d_k}{(k+1)d_{k+1}}(1 + (1-\lambda)p_{k-1}(\lambda)) - \frac{(k^2-1/4)d_{k-1}}{k(k+1)d_{k+1}} p_{k-2}(\lambda),$$

avec les conditions initiales

$$p_0(\lambda) = \frac{2}{3},$$

$$\text{et} \quad p_1(\lambda) = -\frac{4}{5}\lambda + 2.$$

Preuve VII.3 *La démonstration est en tout point semblable à celle du théorème VI.1, il suffit d'utiliser correctement les relations de récurrence VII.8.*

Donc, de la même façon qu'avec le schéma stable MINMAX, pour résoudre $p_k(\tilde{A})r = z$, nous calculons d'abord d_k avec $d_0 = 1$, $d_1 = 3/2$ et

$$d_{j+1} = \frac{2j+1}{j+1} d_j - \frac{(j+1/2)(j-1/2)}{j(j+1)} d_{j-1}, \text{ pour } j = 1, \ldots, k,$$

puis nous effectuons les itérations suivantes

$$\begin{cases} z_0 = \frac{2}{3}r, \\ z_1 = -\frac{4}{5}\tilde{A}r + 2r, \\ z_j = \frac{2j+1}{j+1}\frac{d_j}{d_{j+1}}(r + (I - \tilde{A})z_{j-1}) - \frac{(j^2 - 1/4)}{j(j+1)}\frac{d_{j-1}}{d_{j+1}}z_{j-2}, \\ \qquad \text{pour } j = 2, \ldots, k. \end{cases}$$

Alors la solution de $z = M^{-1}r = P_k(\tilde{A})r$ est $z = z_k$. Nous noterons NORM cette méthode.

Remarque VII.2 *Evidemment, nous pouvons également associer ce préconditionnement polynômial, noté Norm, au classique schéma de Hörner.*

VII.5 Tests sur la CM-2

Nous présentons les résultats de NORM uniquement pour le second problème avec un maillage 256×256, car en fait les préconditionnements polynômiaux Norm et Mcarre effectuent approximativement le même nombre d'itérations. Il nous a donc semblé inutile de présenter deux problèmes différents (surtout que sur le premier problème, les schémas stables n'optimisent pas les temps de calcul) mais aussi d'exposer les nombres d'itérations de NORM (il suffit de se référer à la figure VI.8). Nous proposons donc sur la figure VII.5 une comparaison des temps de résolution entre NORM et MCARRE pour le problème 2 avec un maillage 256 × 256.

Ainsi, nous pouvons constater sur la figure suivante que les temps de résolution de NORM sont, en général, inférieurs à ceux de MCARRE. En effet, ces deux méthodes nécessitent à peu près le même nombre d'itérations, par contre à chaque itération NORM réalise l'étape de préconditionnement avec moins de calculs. Enfin, comme le taux de

parallélisme de NORM est semblable au taux de MINMAX (et donc de peu inférieur au taux de MCARRE), à nombre d'itérations égal, NORM est plus rapide (en temps de résolution) que MCARRE.

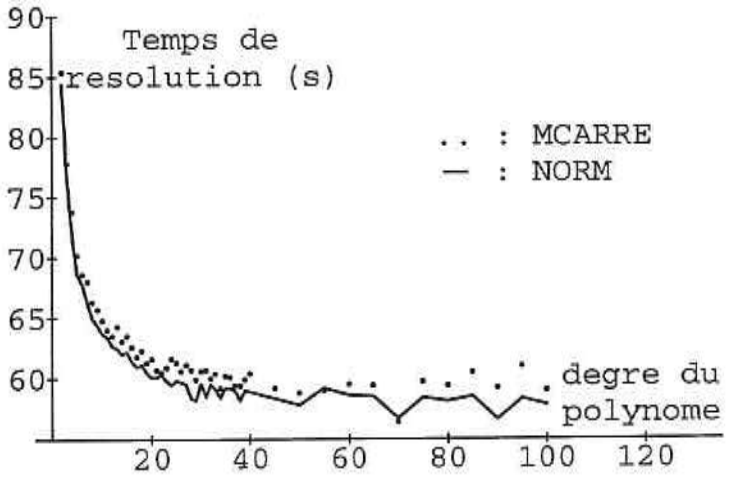

Figure VII.5: Temps de résolution de NORM et MCARRE, pour le problème 2.

En conclusion, il nous apparaît que le préconditionnement NORM est une technique au moins aussi efficace que MCARRE en nombre d'itérations et plus rentable du point de vue du temps de calcul, sur ce type de problèmes. Il possède deux principaux avantages par rapport aux autres préconditionnements polynômiaux, tout d'abord le nombre d'opérations nécessaires à son schéma stable est inférieur à ceux de MCARRE et de LEG et est équivalent à celui de MINMAX. Mais surtout, ce préconditionnement, sans perdre beaucoup de l'efficacité de Mcarre, n'a besoin d'aucune estimation des valeurs propres. Ce préconditionnement a donc les mêmes qualités de simplicité des préconditionnements polynômiaux basés sur les séries de Neumann, mais avec un taux de convergence nettement plus intéressant. Enfin, si son nombre d'itérations reste certes supérieur à celui de Minmax, son utilisation nous semble bien plus *pratique*.

CHAPITRE VIII

UN
DOUBLE
PRÉCONDITIONNEMENT

VIII.1 Présentation

Nous avons déjà vu au second chapitre que le G.C préconditionné par un préconditionnement M consiste à résoudre le système suivant par le G.C

$$M^{-1/2}AM^{-t/2}y = M^{-1/2}b, \; puis \; M^{t/2}x = y, \tag{VIII.1}$$

ou bien, si M commute avec A

$$M^{-1}Ax = M^{-1}b. \tag{VIII.2}$$

Ainsi, le G.C préconditionné par un préconditionnement polynômial P_k est équivalent à la résolution par le G.C du système suivant

$$P_k(A)Ax = P_k(A)b, \tag{VIII.3}$$

car un polynôme en A commute avec A.

Alors, si la distribution initiale des valeurs propres de A ne nous semble pas suffisament favorable au préconditionnement polynômial choisi (en particulier pour Minmax si la plus petite valeur propre est trop proche de zéro), ou bien si nous désirons améliorer le rendement des préconditionnements polynômiaux, nous avons la possibilité de préconditionner une première fois le système puis de le résoudre à l'aide d'un G.C préconditionné par un préconditionnement polynômial, c'est à dire de combiner VIII.1 (ou VIII.2, si le préconditionnement commute) avec VIII.3, et donc de résoudre à l'aide du G.C le système

$$P_k(M^{-1/2}AM^{-t/2})M^{-1/2}AM^{-t/2}y = P_k(M^{-1/2}AM^{-t/2})M^{-1/2}b, \tag{VIII.4}$$

$$puis\ M^{t/2}x = y$$

Ensuite, il ne reste plus qu'à choisir le premier préconditionnement M, par exemple une factorisation incomplète, par points ou par blocs, de Cholesky ou un premier préconditionnement polynômial, et à déterminer la famille du préconditionnement polynômial P_k. Nous aurons obtenu un double préconditionnement. Enfin, si nous parvenons à obtenir une estimation assez précise du conditionnement de $M^{-1}A$, le préconditionnement $P_k(A)$ deviendra d'autant plus efficace (notamment dans le cas du polynôme Minmax).

Dans un premier temps nous allons étudier un double préconditionnement polynôme-polynôme, puis nous testerons un double préconditionnement factorisation de Cholesky incomplète-polynôme.

Remarque VIII.1 *Lors de la résolution effective du second problème à l'aide des préconditionnements polynômiaux, nous avons normalisé symétriquement la matrice par sa diagonale, voir chapitre IV. Nous avons donc de cette manière déjà étudié un double préconditionnement. Par conséquent, lors de la résolution du second problème par un double préconditionnement, nous utiliserons en réalité un triple préconditionnement.*

VIII.2 Polynôme-polynôme

Lors de la résolution, au chapitre VI, du premier problème le surcoût de calcul des schémas stables était trop important pour diminuer les temps de calcul des méthodes associées au schéma de Hörner. Nous avons donc cherché un moyen d'utiliser des polynômes de degré plus élevé tout en conservant le coût minimum d'opérations du schéma de Hörner (qui ne permettait pas jusqu'à présent de dépasser le degré

8). Donc comme un polynôme en A commute avec A nous avons posé comme premier préconditionnement

$$M^{-1} = P_l(A) \text{ et } M^{-1}A = Q_{l+1}(A).$$

Puis, nous avons utilisé le G.C préconditionné par un polynôme P_k sur ce système. Ainsi, nous avons résolu

$$P_k(P_l(A)A)P_l(A)Ax = P_k(P_l(A)A)P_l(A)b,$$

ou encore

$$Q_{k+1}(Q_{l+1}(A))x = P_k(Q_{l+1}(A))P_l(A)b.$$

Ainsi, nous verrons qu'en utilisant des polynômes (de degré peu élevé) de polynômes (de degré peu élevé) nous obtenons des résultats équivalents à ceux de polynômes simples mais de degré bien plus élevé. D'autre part, afin de clarifier la situation supposons dès maintenant que les polynômes P_k et P_l appartiennent à la famille des polynômes minimisant la généralisation du conditionnement (Minmax). Nous noterons alors MINMAXL-MINMAXK ces méthodes. En particulier, il est clair que MINMAX0-MINMAXK correspond à HMINMAX de degré k. De plus nous allons voir que les résultats (en nombre d'itérations) de MINMAXL-MINMAXK et MINMAX((K+1)(L+1)-1) sont à peu près égaux et que grâce à l'utilisation du schéma de Hörner pour l'évaluation de MINMAXL-MINMAXK (car se sont en fait en fait deux polynômes de faible degré) nous parvenons à diminuer les temps de calculs réalisé par les méthodes MINMAX (pénalisée par le surcoût de calcul important des récurrences à trois termes).

Remarque VIII.2 *Nous avons exposé dans cette thèse plusieurs familles de préconditionnements polynômiaux. Nous aurions donc pu les associer de diverses manières dans ce double préconditionnement. D'ailleurs, nous avons essayé de combiner Norm et Minmax, mais nous n'exposerons que les résultats de MINMAXL-MINMAXK car ils sont sensiblement meilleurs. Toutefois pour une utilisation plus pratique de ces doubles préconditionnements polynômiaux il serait sans doute plus judicieux de tester un double préconditionnement NORM-NORM. D'autre part, cette notion de double préconditionnement se généralise*

aisément à un préconditionnement multiple. Il suffit de réaliser des polynômes de polynômes de polynômes etc ...

Ensuite, si nous connaissons la plus grande et la plus petite valeur propre de A (respectivement b et a) ou bien si nous avons normalisé symétriquement A par sa diagonale (et donc estimé les valeurs propres par $a = 0.25\ 10^{-3}$ et $b = 2$, voir remarques IV.1 et IV.2 mais aussi l'introduction du chapitre précédent), nous pouvons, car nous connaissons l'expression du polynôme Q_{l+1}, évaluer les valeurs propres de $Q_{l+1}(A)$ ou au moins utiliser la généralisation du conditionnement (introduite dans la définition III.3) comme au chapitre V, et donc estimer le maximum et le minimum du polynôme[1] q_{l+1} sur l'intervalle $[a, b]$. Le polynôme q_{l+1} étant généralement un polynôme de faible degré (à cause des instabilités numériques liées au schéma de Hörner, $l \leq 8$), nous pouvons directement calculer son mimimum et son maximum sur $[a, b]$ à partir de son expression explicite (nous avons procédé ainsi pour $l \leq 2$). Nous pouvons également utiliser les résultats démontrés au chapitre V, voir aussi [AsMO92], de la manière suivante,

posons

$$m = inf_{\lambda \in [a,b]} (q_{l+1}(\lambda)) \ et \ M = sup_{\lambda \in [a,b]} (q_{l+1}(\lambda)) .$$

Alors

$$\begin{aligned} m &= inf_{\lambda \in [a,b]} (1 - T_{l+1}(\mu(\lambda))/T_{l+1}(\mu(0))) \\ m &= 1 - sup_{\lambda \in [a,b]} (T_{l+1}(\mu(\lambda))/T_{l+1}(\mu(0))) \\ &= 1 - 1/|T_{l+1}(\mu(0))| \\ &= 1 - 2/(\nu^{l+1} + \nu^{-l-1}), \text{ d'après (V.3),} \end{aligned}$$

avec

$$\nu = \left(\frac{\sqrt{b} - \sqrt{a}}{\sqrt{b} + \sqrt{a}} \right) .$$

[1] Rappelons, voir la définition III.1, que nous notons en lettres majuscules les polynômes appliqués à des matrices et en lettres minuscules correspondantes les mêmes polynômes mais appliqués cette fois à des scalaires.

De la même manière, on obtient

$$M = 1 + 1/|T_{l+1}(\mu(0))| = 2 - m.$$

Remarque VIII.3 *Dans un premier temps, nous avons utilisé cette estimation des valeurs propres, mais devant les résultats pas toujours performants (en particulier à cause d'une estimation trop imprécise de la plus petite valeur propre m) nous avons remplacé l'évaluation de m par $q_{l+1}(a)$, qui s'est avérée bien plus efficace. Finalement, les résultats exposés dans les figures suivantes sont les résultats optimaux.*

Le conditionnement de $Q_{l+1}(A)$ ainsi calculé, nous avons construit le second préconditionnement polynômial P_k à partir de ces nouvelles valeurs propres m et M. Enfin, si ces valeurs propres ont été correctement estimées, nous pouvons espérer que ces doubles préconditionnements MINMAXL-MINMAXK correspondent à un seul préconditionnement polynômial mais de degré $(k+1)(l+1)-1$, du point de vue du nombre d'itérations.

D'autre part, afin de ne pas encombrer la mémoire du calculateur mais aussi afin de conserver un produit matrice vecteur efficace, nous n'avons pas assemblé la matrice $Q_{l+1}(A)$. En effet, le produit d'une matrice pentadiagonale par une matrice pentadiagonale engendre une matrice avec 13 diagonales, d'où un surcoût de stockage. De plus, les nouvelles diagonales sont encore plus éloignées et donc les temps de communications entre les processeurs deviennent plus longs. Donc pour éviter ces pertes de performances, lors de la résolution à chaque appel d'un produit de $Q_{l+1}(A)$ par un vecteur y, nous avons utilisé le schéma de Hörner pour calculer $Q_{l+1}(A)y$. Dans un premier temps nous avions tout de même essayer d'assembler $Q_{l+1}(A)$ pour $l = 1$, mais devant les résultats peu performants par rapport à la version non assemblée, nous avons décidé de ne plus tester de versions assemblées.

Finalement, nous espérons que les instabilités numériques liées au schéma de Hörner n'interviendront pas trop rapidement pour que l'on puisse atteindre avec ces doubles préconditionnements des degrés supérieurs à ceux réalisés par les simples préconditionnements polynômiaux associés au schéma de Hörner.

VIII.2.1 Problème 1 avec un maillage 512×512

Nous proposons sur les figures VIII.1 à VIII.4 et dans les tableaux VIII.1, VIII.2, VIII.4 et VIII.5 les nombres d'itérations et les temps de résolution sur la CM-200 4K PEs du SEH de ces doubles préconditionnements MINMAXL-MINMAXK, tant que ces méthodes restent convergentes[2], d'abord sur le problème 1 (avec un maillage 512×512) puis sur le second problème (avec un maillage 256×256). Nous les comparons à HMINMAX et à MINMAX.

D'autre part, nous ne présentons pas de résultats de ces doubles préconditionnements sur une CM-2 8K PEs, car suivant l'évolution du site de calcul (le SEH), nous n'avons plus accès à la CM-2 depuis MAI 1993. Néanmoins, l'essentiel est de pouvoir comparer les différentes méthodes, la machine référence de ce chapitre sera donc une CM-200 4K PEs.

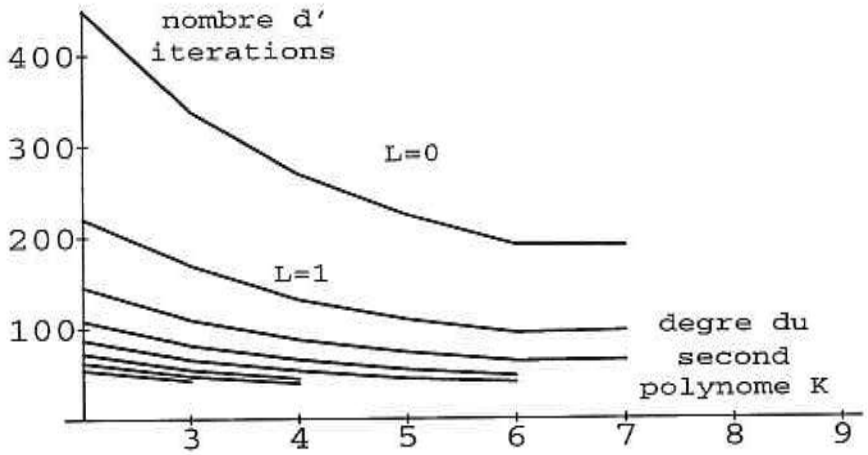

Figure VIII.1: Comparaison des nombres d'itérations des doubles préconditionnements MINMAXL-MINMAXK, pour le problème 1.

Etant donné que nous n'avons conservé que les résultats décroissants, les méthodes sont ordonnées sur la figure VIII.1 de la manière

[2]On considèrera que la méthode MINMAXL-MINMAXK reste convergente, si à K fixé le nombre d'itérations de MINMAX(L+1)-MINMAXK est inférieur à celui de MINMAXL-MINMAXK, et de même pour L fixé.

suivante le nombre d'itérations de MINMAXL-MINMAXK est supérieur à celui de MINMAXL'-MINMAXK si $L < L'$, pour $L = 0, \ldots, 7$.

Alors, nous constatons sur cette figure VIII.1 et sur le tableau VIII.2 que nous sommes parvenus à utiliser des doubles préconditionnements polynômiaux de degrés assez élevés (correspondant au degré 34 d'un simple préconditionnement polynômial) en tout cas plus élevés que le degré de HMINMAX (7 sur ce problème), voir aussi le tableau VIII.1. Les temps réalisés par les préconditionnements polynômiaux simples associés au schéma de Hörner vont donc sans aucun doute être diminués par ces doubles préconditionnements. Il apparait également que, pour K fixé, l'instabilité numérique, liée au schéma de Hörner, de ces techniques MINMAXL-MINMAXK est une fonction croissante de L (même remarque pour K fixé). Ainsi, sur le tableau VIII.2 on remarque que la divergence[3] de MINMAXL-MINMAXK apparait nettement plus vite pour L=5 que pour L=3 (mêmes conclusions pour L=0 avec le tableau VIII.1).

Degré K	HMINMAXK (L=0)	
	Nombre d' itérations	Temps de résolution (s)
2	448	46.6
3	338	43.8
4	269	41.8
5	224	40.4
6	191	39.5
7	190	39.9

Tableau VIII.1: Résultats de HMINMAXK, pour le problème 1 sur la CM-200.

[3]On parle de divergence d'une méthode MINMAXL-MINMAXK si son nombre d'itérations est supérieur ou égal à celui de MINMAXL-MINMAX(K-1), dans ce cas on note non conv. son nombre d'itérations dans les tableaux VIII.2 et VIII.5.

Degré K	L=3		L=5	
	Nombre d' itérations	Temps de résolution (s)	Nombre d' itérations	Temps de résolution (s)
2	108	37.0	72	35.6
3	81	36.0	54	35.1
4	65	35.6	non conv.	inutile
5	54	35.2	non conv.	inutile
6	47	35.5	non conv.	inutile

Tableau VIII.2: Comparaison des résultats de méthodes MINMAXL-MINMAXK, pour le problème 1 sur la CM-200.

Degré K	MINMAX		
	Nombre d' itérations	Temps de Résolution (s)	Couple (L,K) correspondant
15	91	45.6	(3,3)
17	83	46.7	(5,2)
19	76	47.3	(3,4)
23	66	47.9	(3,5) ou (5,3)
27	60	48.2	(3,6)

Tableau VIII.3: Nombre d'itérations et temps de résolution (s) de MINMAX de degré correspondant à un couple (L,K) ((K+1)(L+1)-1), pour le problème 1 sur la CM-200.

D'autre part, on remarque que les résultats de MINMAXL-MINMAXK et de MINMAXK-MINMAXL sont sensiblement identiques. De plus, la précision (du point de vue du nombre d'itérations) de ces doubles préconditionnements est supérieure, pour ce problème, à celle du préconditionnement MINMAX de degré correspondant, voir les tableaux VIII.2 et VIII.3. En effet, grâce à la seconde estimation des valeurs propres (en particulier de la plus petite a) nous proposons une valeur de a moins proche de zéro pour la définition du second préconditionnement polynômial. Le polynôme Minmax étant très sensible à l'estimation de a, voir [AsMO92], [Saad85] et le chapitre 7, nous

améliorons de cette manière l'efficacité du second préconditionnement. La double estimation des valeurs propres $Q_{l+1}(A)$ se révèle donc être assez efficace sur ce problème.

Etudions maintenant les temps de calcul de ces doubles préconditionnements sur le problème 1.

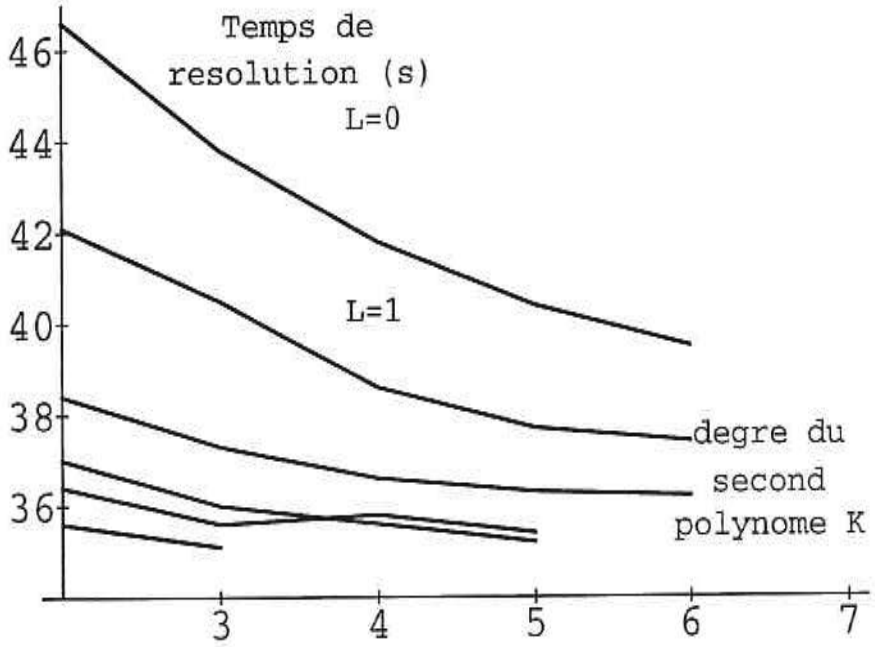

Figure VIII.2: Comparaison des temps de résolutions des doubles préconditionnements MINMAXL-MINMAXK, pour le problème 1 sur la CM-200.

Dans le cas du problème 1, nous avons vu au chapitre VI que la méthode HMINMAX était la technique la plus rapide pour la résolution du sytème (I.3) sur la CM-2. Après avoir renouvelé ces expériences sur la CM-200, nous nous sommes rendus aux mêmes conclusions. Par contre sur la figure VIII.2 ainsi que sur les tableaux VIII.1 et VIII.2, il est clair que les doubles préconditionnements parviennent à diminuer sensiblement ces temps de calcul. Donc même sur ce problème plutôt bien conditionné nous avons réussi à optimiser le temps de résolution de

HMINMAX, en utilisant en fait des préconditionnements polynômiaux mais de degré plus élevé mais aussi d'une certaine manière plus précis, voir tableau VIII.3. Le gain de temps réalisé par MINMAX5-MINMAX3 sur ce problème n'est pas négligeable, il est de l'ordre de 11% par rapport à HMINMAX6 et 55% par rapport à DIAG.

Par contre, le choix du couple (L,K) optimal en temps ne semble pas a priori un problème facile à résoudre. En effet, contrairement aux schémas stables où une petite variation autour du degré optimal n'entraine qu'une perte relativement peu importante de temps, ces méthodes semblent limitées p[ar les instabilités numériques liées au schéma de Hörner. Nous pouvons donner au moins un critère afin de mieux déterminer ce couple optimal. Sur ce problème, le couple (L,K) est au bord du domaine de convergence[4] des méthodes MINMAXL-MINMAXK. Et donc de la même manière que les préconditionnements polynômiaux simples associés à un schéma de Hörner simple ces méthodes semblent limitées par la précision de calcul. Mais dans ce cas, il est tout à fait inutile d'introduire des schémas plus stables car ces schémas (plus stables) permettent déjà d'atteindre de degrés élevés avec des préconditionnements polynômiaux simples.

VIII.2.2 Problème 2 avec un maillage 256×256

Regardons maintenant les résultats sur la CM-200 4K PEs de ces doubles préconditionnements sur le second problème avec un maillage 256×256 et des simples préconditionnements associés à un schéma stable ou pas.

Cette nouvelle étude montre que les problèmes d'instabilité numérique (liés au schéma de Hörner simple) de ces doubles préconditionnements apparaissent d'autant plus vite que le problème résolu est mal conditionné (comportement identique à ceux des simples préconditionnements polynômiaux associés au schéma de Hörner simple, voir le chapitre 5).

[4]On appellera domaine de convergence des méthodes MINMAXL-MINMAXK, l'ensemble des couples (L,K) tels que MINMAXL-MINMAXK soient convergentes. Donc, un couple (L,K) appartient au bord du domaine si la technique MINMAX(L+1)-MINMAXK ou MINMAXL-MINMAX(K+1) ne converge pas.

A cause de ces instabilités, les degrés maximums atteints à l'aide du schéma de Hörner simple sont inférieurs à ceux du problème 1. De plus, contrairement au premier problème les nombres d'itérations des méthodes MINMAXL-MINMAXK sont supérieurs à ceux réalisés par un préconditionnement MINMAX de degré correspondant, voir le tableau VIII.6, sans doute à cause d'une mauvaise estimation de la plus petite valeur propre mais aussi du conditionnement de ce second problème. Un double préconditionnement NORM-NORM serait sans doute moins sensible à ce changement de problème.

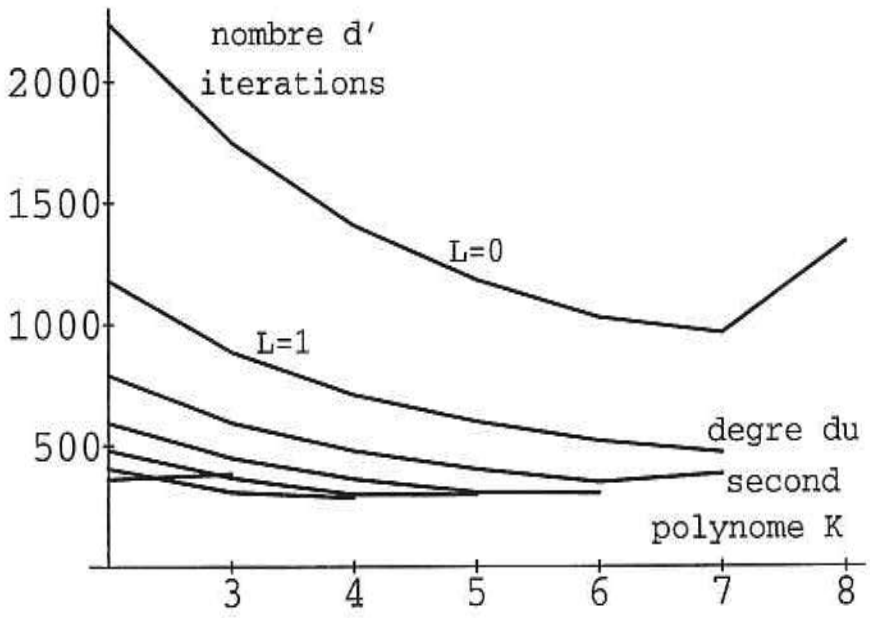

Figure VIII.3: Comparaison des nombres d'itérations des doubles préconditionnements MINMAXL-MINMAXK, pour le problème 2 sur la CM-200.

Sur une machine plus puissante que la CM-200 4K PEs, par exemple une CM-2 8K PEs, nous savons que le degré optimal, pour un temps de

résolution minimum, des préconditionnements polynômiaux associés à un schéma stable était sur ce problème très élevé (31 pour MINMAX et 70 pour MCARRE, voir le tableau VI.3). Donc, sur un tel calculateur les instabilités numériques de ces doubles préconditionnements limiteraient sans doute leur efficacité. Par contre, sur cette CM-200 4K PEs le degré optimal (25 pour MINMAX) est suffisamment petit pour que ces doubles préconditititionnements diminuent les temps de résolution des schémas les plus stables, voir les tableaux VIII.4 et VIII.5.

HMINMAXK			MINMAXK		
K	Nombre d' itérations	Temps de résolution (s)	K	Nombre d' itérations	Temps de résolution (s)
2	2238	76.4	10	624	63.3
3	1751	72.3	15	435	62.1
4	1407	67.6	20	333	61.4
5	1181	65.0	25	269	60.7
6	1026	63.6	30	230	61.5
7	961	66.4	35	201	62.0

Tableau VIII.4: Comparaison des résultats de méthodes HMINMAXK et MINMAXK, sur le problème 2 avec la CM-200.

	L=3		L=5	
K	Nombre d' itérations	Temps de résolution (s)	Nombre d' itérations	Temps de résolution (s)
2	593	59.6	405	58.1
3	449	58.0	308	57.1
4	362	57.3	285	65.2
5	307	58.2	non conv.	inutile

Tableau VIII.5: Comparaison des résultats des méthodes MINMAXL-MINMAXK, sur le problème 2 avec la CM-200.

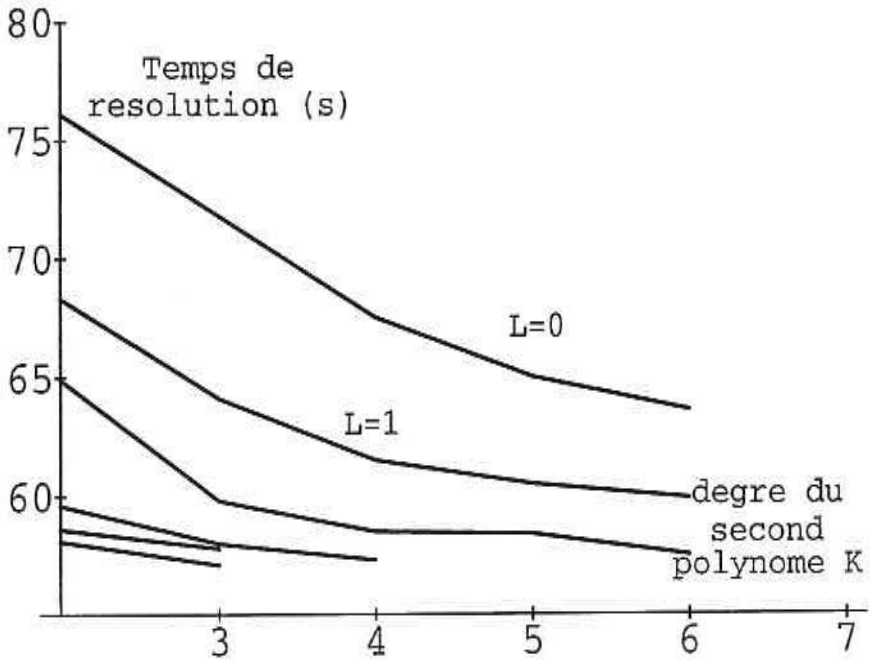

Figure VIII.4: Comparaison des temps de résolutions des doubles préconditionnements MINMAXL-MINMAXK, pour le problème 2 sur la CM-200.

	L=3		L=5	
K	Nombre d' itérations	Temps de résolution (s)	Nombre d' itérations	Temps de résolution (s)
2	593	59.6	405	58.1
3	449	58.0	308	57.1
4	362	57.3	285	65.2
5	307	58.2	non conv.	inutile

Tableau VIII.6: Comparaison des résultats des méthodes MINMAXL-MINMAXK, sur le problème 2 avec la CM-200.

Degré K	MINMAX		
	Nombre d' itérations	Temps de Résolution (s)	Couple (L,K) correspondant
15	435	62.1	(3,3)
17	386	61.5	(5,2)
19	351	61.8	(3,4)
23	295	61.7	(3,5) ou (5,3)
27	246	61.6	(3,6)

Tableau VIII.7: Nombre d'itérations et temps de résolution (s) de MINMAX de degré correspondant à un couple (L,K) ((K+1)(L+1)-1), pour le problème 2 sur la CM-200.

D'autre part, le gain de temps maximum réalisé par MINMAXL-MINMAXK sur ce problème est moins important que pour le premier problème, tout de même 5.9% par rapport à MINMAX25 et 51.1% par rapport à DIAG. Mais il faut rappeler que sur ce second problème MINMAX diminuait déjà sensiblement les temps obtenus avec HMINMAX. Ces résultats restent néanmoins intéressants par rapport à HMINMAX 10.2% de mieux.

Par contre, le choix du couple (L,K) optimal semble réellement constituer un problème difficile à résoudre. Malgré tout, nous pouvons

encore appliquer le conseil cité au problème 1, mais cette fois les bords du domaine de convergence apparaissent moins nettement. Et donc ce problème se complique d'autant plus.

VIII.3 Factorisation de Cholesky incomplète-polynôme

Afin d'étendre quelque peu l'idée du double préconditionnement nous avons étudié un double préconditionnement associant une factorisation de Cholesky incomplète par points ($\mathcal{M}^{1/2}\mathcal{M}^{t/2}$) à un préconditionnement polynômial. Ainsi nous avons résolu

$$P_k(\mathcal{M}^{-1/2}A\mathcal{M}^{-t/2})\mathcal{M}^{-1/2}A\mathcal{M}^{-t/2}y = P_k(\mathcal{M}^{-1/2}A\mathcal{M}^{-t/2})\mathcal{M}^{-1/2}b,$$

$$puis\ \mathcal{M}^{t/2}x = y$$

avec $\mathcal{M}^{1/2}$ la matrice triangulaire inférieure de même squelette que $A = penta_N(c_i, b_i, d_i, b_{i+1}, c_{i+n})$ définie par

$$\mathcal{M}^{1/2} = penta_N(\overline{c}_i, \overline{b}_i, \overline{d}_i, 0, 0)$$

et

$$\begin{cases} \overline{c}_i &= c_i/\overline{d}_{i-n} \\ \overline{b}_i &= b_i/\overline{d}_{i-1} \\ \overline{d}_i &= d_i - \overline{c}_i{}^2 - \overline{b}_i{}^2 \end{cases}$$

Ensuite, à chaque appel de $\mathcal{M}^{-1/2}$ (respectivement $\mathcal{M}^{-t/2}$) lors de l'étape de préconditionnement nous avons résolu un système triangulaire inférieur (respectivement supérieur). Nous n'avons donc pas assemblé la matrice $\mathcal{M}^{-1/2}A\mathcal{M}^{-t/2}$.

D'autre part ce type de récurrence par points (voire même par blocs) ne sont pas très efficaces sur une machine telle que la CM-200, voir [Weil94] pour une factorisation ILU. Nous n'avons donc pas testé ce double préconditionnement sur la CM-200 mais uniquement dans les mêmes conditions qu'au chapitre IV c'est à dire sur une station de

travail Apollo (série 3500 cette fois). Le problème résolu est le second problème avec un maillage 40×40, voir chapitre IV.

Enfin nous avons associé cette factorisation de Cholesky incomplète à deux types de préconditionnements polynômiaux, dans un premier temps Minmax puis, devant la difficulté à trouver une valeur optimale de la plus petite valeur propre a, Norm. En fait nous avons constaté au chapitre précédent que l'efficacité du polynôme Minmax n'est pas toujours optimale même lorsque l'on connait la valeur exacte de la plus petite valeur propre a. Alors nous n'avons même pas cherché pour ce problème à évaluer a, nous nous sommes contentés de faire varier cette valeur au-delà de zéro. Les polynômes sont évalués à l'aide du schéma de Hörner simple, mais pour atteindre des nombres d'itérations encore plus faibles on pourrait utiliser des récurrences à trois termes.

Degré du polynôme	1	2	3	4	5	6	7	8	9
CHOLESKY-MINMAX									
$a = 0.25$	24	17	14	12	11	10	10	9	11
$a = 0.13$	23	17	14	12	10	10	9	11	16
$a = 0.025$	22	18	17	14	11	11	10		
$a = 0.0025$	22	30	43	31	25	21	49		
CHOLESKY-NORM	25	18	14	12	10	9	9	11	

Tableau VIII.8: Nombres d'itérations de doubles préconditionnements, pour le problème 2 avec un maillage 40×40.

Sur le tableau VIII.8 nous constatons que ce type de double préconditionnement est efficace au point de vue du nombre d'itérations, sur ce problème. En effet, alors que DIAG nécessite 876 itérations, HMINMAX de degré 7 170, factorisation de Cholesky incomplète seule 46, ces doubles préconditionnements réussissent à converger en 9 itérations, tout en conservant la même structure creuse que la matrice initiale. Malheureusement le surcout de calculs n'est pas compensé, comme sur une machine massivement parallèle, par une augmentation de la vitesse de calcul. Ainsi le temps optimal pour cette machine est atteint avec des degrés très faibles (2 ou 4), voir le tableau VIII.9. Néanmoins

ces temps optimaux sont inférieurs à ceux réalisé par une factorisation de Cholesky incomplète seule (23.6 s). Il serait donc relativement intéressant de coupler les préconditionnements polynômiaux avec d'autres préconditionnements et ce sur machine séquentielle (avec par exemple un préconditionnement MIC(1,1)) ou parallèle (avec des factorisations par blocs).

Degré du polynôme	1	2	3	4
CHOLESKY-MINMAX $a = 0.13$	20.7	20.8	21.8	22.7
CHOLESKY-NORM	22.2	21.9	21.8	22.7

Tableau VIII.9: Temps de résolution de doubles préconditionnements, pour le problème 2 avec un maillage 40×40 sur Apollo series 3500.

Remarque VIII.4 *Pour la résolution du problème 2 à l'aide de préconditionnements polynômiaux, nous normalisons symétriquement la matrice A par sa diagonale. Nous avons donc résolu le système initial à l'aide du G.C muni en fait d'un triple préconditionnement (Diagonal, factorisation de Cholesky incomplète et polynômial).*

CHAPITRE IX

EFFICACITÉ DES SCHÉMAS STABLES SUIVANT LA RÉGULARITÉ DU MAILLAGE

Chapitre IX

Efficacité des schémas stables suivant la régularité du Maillage

IX.1 Présentation

Nous nous sommes rendu compte dans cette thèse que l'utilisation de schémas stables, voir chapitre VI, pour l'évaluation de préconditionnements polynômiaux de degrés élevés, est efficace sur CM. Néanmoins, ces bonnes performances en temps de calcul sont liées au bon rendement du produit matrice vecteur sur ce type de numérotation sur machine SIMD, voir remarque V.2. En effet, la structure pentadiagonale des matrices, provenant de la régularité du maillage carré uniforme et de la numérotation *naturelle*, permet d'écrire un produit matrice vecteur très rapide (grâce aux communications régulières, telles que les shifts).

Ainsi, si on perturbe la numérotation régulière, on diminue l'efficacité du produit matrice vecteur et donc on augmente le rôle des produits scalaires. Nous allons donc introduire plusieurs produits matrice vecteur puis nous mesurerons les performances des schémas stables (comportement du degré optimal) suivant la rapidité de ces produits matrice vecteur. Nous conclurons par quelques résultats essayant d'encadrer a priori le degré optimal.

IX.2 Trois produits matrice vecteur

Le premier produit matrice vecteur proposé est déjà présenté dans les deux premiers chapitres, voir I.7, II.12 et [ChKT89]. Dans ce cas, nous stockons les cinq diagonales de la matrice, ainsi que les vecteurs du problème, dans des tableaux bidimensionnels (de taille $n \times n$ semblables au maillage uniforme (de pas $1/(n+1)$) du carré unité). Ensuite en

notant CB, CG, CC, CD et CH les cinq diagonales de la matrice A et X, Y deux vecteurs semblables aux diagonales, nous pouvons écrire le produit matrice vecteur Y=AX en CM-Fortran à l'aide de communications shifts de la manière suivante :

```
Y=CB*EOSHIFT(X,2,-1)+CG*EOSHIFT(X,1,-1)+CC*X
. +CD*EOSHIFT(X,1,1)+CH*EOSHIFT(X,2,1)
```

Nous notons 2D ce produit matrice vecteur.

Ce mode de stockage des vecteurs optimise la vitesse du produit matrice vecteur au détriment du produit scalaire. Ainsi, pour le préconditionnement Diagonal on lui préfère, voir remarque V.2, le stockage unidimensionnel. Le stockage unidimensionnel consiste à stocker les diagonales de la matrice dans des tableaux unidimensionnels de longueur $N = n^2$. Nous écrivons alors le produit matrice vecteur de la manière suivante, avec X et Y deux tableaux de longueur N :

```
Y=CB*EOSHIFT(X,1,-n)+CG*EOSHIFT(X,1,-1)+CC*X
. +CD*EOSHIFT(X,1,1)+CH*EOSHIFT(X,1,n)
```

Nous notons 1D ce produit matrice vecteur.

Remarquons que ce produit matrice vecteur utilise des shifts de longueur 1 et n au lieu de 1 pour le précédent. Nous avons ainsi constaté dans les expériences numériques que le produit 1D est moins rapide que le produit 2D. Néanmoins comme le produit scalaire, du stockage 1D, est optimisé il peut donner des temps de résolution plus faible pour des préconditionnements polynômiaux de degré peu élevé et notamment pour le préconditionnement Diagonal.

Enfin, afin de pouvoir tester des numérotations beaucoup plus irrégulières nous avons utilisé une sorte de stockage condensé, dérivé du stockage Ellpack-Itpack format (ELL) de [FPSW93]. Dans ce cas, les éléments non nuls de la matrice sont stockés dans un tableau de réels A à N lignes et 5 colonnes. Ensuite, dans un tableau d'entiers IA semblable à A nous stockons les numéros de colonnes correspondants aux éléments non nuls. Enfin, les vecteurs X et Y sont stockés dans des

tableaux de taille N. Alors, nous écrivons le produit matrice vecteur, noté CONDENSE, à l'aide d'une variable temporaire TEMP semblable à A :

```
DO j=1,IC
TEMP(:,j)=X(IA(:,j))
ENDDO
Y=SUM(A*TEMP,2)
```

Remarque IX.1 *Le stockage CONDENSE n'est pas un véritable stockage condensé (avec des pointeurs). De plus, vu la structure très régulière du maillage, le tableau d'adresses IA est très régulier. Malgré tout, nous allons voir dans les résultats numériques que CONDENSE est beaucoup plus lent que 2D et 1D tout en conservant un produit scalaire optimal, grâce au mode de stockage des vecteurs. Nous sommes donc dans un cas de figure très défavorable aux préconditionnements polynômiaux par rapport au préconditionnement Diagonal. Pour plus de détails sur des stockages pour structures irrégulières sur CM-5 consultez [FPSW93].*

IX.3 Influence de la vitesse du produit matrice vecteur sur l'efficacité des schémas stables

Tous les résultats présentés dans ce paragraphe ont été réalisés sur une CM-5. En fait la CM-5 est un ordinateur massivement parallèle MIMD et non plus SIMD comme les autres Connection Machines, voir [ThMC90b]. Ses Processeurs Elémentaires ne sont plus des processeurs un bit mais des processeurs SPARC. Ainsi la CM-5 utilisée ne possède que 32 PNs. D'autre part, la CM-5 est aisément synchronisable on peut donc l'utiliser tout de même comme une machine SIMD telle que la CM-2 (on parle alors d'un modèle SPMD : Single Program Multiple Data, pour cette machine). Donc pour comparer ses performances avec celles de la CM-2 nous utiliserons ce mode SIMD.

D'autre part, dans ce chapitre nous résolvons le problème 2, que nous avons normalisé symétriquement par sa diagonale, ainsi nous avons estimé la plus petite valeur propre par $0.25\ 10^{-3}$ et la plus grande par 2.

Mesurer l'efficacité des schémas stables d'évaluation des polynômes revient à regarder le comportement du degré optimal en temps de ces schémas. En effet, ces schémas sont d'autant plus efficaces, notamment par rapport au préconditionnement Diagonal et également par rapport au schéma de Hörner, que le degré optimal en temps est grand.

Dans les trois tableaux suivants nous présentons le degré optimal et le temps de résolution, pour trois tailles de matrices et pour les trois produits matrice vecteur, du G.C préconditionné par le polynôme Minmax associé à son schéma stable (MINMAX).

Problème de taille 64×64=4096

Mat-Vec	degré optimal	Temps Total (s)
2D	46	1.07
1D	18	1.2
CONDENSE	1	7.3

Tableau IX.1: Evolution du degré optimal suivant le Mat-Vec, sur CM-5.

Problème de taille 128×128=16384

Mat-Vec	degré optimal	Temps Total (s)
2D	55	3.1
1D	20	3.6
CONDENSE	1	25.1

Tableau IX.2: Evolution du degré optimal suivant le Mat-Vec, sur CM-5.

Problème de taille 256×256=65536

Mat-Vec	degré optimal	Temps Total (s)
2D	25	13.6
1D	25	14.9
CONDENSE	2	117.5

Tableau IX.3: Evolution du degré optimal suivant le Mat-Vec, sur CM-5.

Sur les tableaux IX.1 à IX.3 nous constatons que le degré optimal en temps du schéma MINMAX est une fonction croissante de la rapidité du produit matrice vecteur. Ainsi, avec 2D le degré peut devenir très grand, jusqu'à 55, alors qu'avec CONDENSE un polynôme de faible degré (1 ou 2) est le préconditionnement le plus efficace. Le rendement des schémas stables et par conséquent des préconditionnements polynômiaux est donc fortement dépendant du produit matrice vecteur utilisé (et donc dans ce cas de la structure régulière du maillage).

Malgré tout, même avec le stockage CONDENSE les préconditionnements polynômiaux de degré 1 ou 2 diminuent les temps de résolution du préconditionnement diagonal correspondant de 5 à 15%, voir le tableau IX.4. Ces temps de résolution sont par ailleurs très élevés par rapport à 2D. Enfin les écarts de temps entre 2D et 1D sont relativement importants compte tenu du peu de différences entre ces deux stockages (ceci montre encore l'importance du produit matrice vecteur).

Taille	Temps de résolution	
$N = n^2$	DIAGONAL	MINMAX
4096	7.7	7.3
16384	28.4	25.1
65536	138.4	117.5

Tableau IX.4: Temps de résolution (s) avec le stockage CONDENSE, sur CM-5.

Remarque IX.2 *Nous n'avons pas perturbé plus la numérotation, car au vu des résultats de CONDENSE, il ne nous a pas semble utile de diminuer encore l'efficacité du produit matrice vecteur pour obtenir des résultats plus significatifs.*

IX.4 Recherche d'encadrement du degré optimal

La recherche a priori du degré optimal en temps est une question très difficile. Néanmoins il serait intéressant de lui trouver au moins une borne supérieure. En effet, il ne parait guère envisageable pour un utilisateur de tester les préconditionnements polynômiaux pour un degré variant, par exemple comme pour le problème 2 de 0 à 100. Dans ce paragraphe nous allons donc essayer de délimiter le degré optimal. Par hypothèse, nous nous plaçons dans le cas d'un produit matrice vecteur très rapide, comme 2D, lors de la résolution d'un problème mal conditionné (un problème nécessitant beaucoup d'itérations). Dans un cas contraire, nous n'avons guère besoin de toute façon de délimiter l'intervalle du degré optimal. En effet, si le produit matrice vecteur n'est pas suffisament rapide nous venons de voir que des degrés très faibles sont optimaux. D'autre part, si le problème converge en peu d'itérations, très rapidement le préconditionnement ne diminuera plus le nombre d'itérations, donc l'intervalle de recherche du degré optimal est naturellement plus limité.

Nous allons partir de l'idée suivante : lorsque l'on augmente le degré du polynôme le nombre d'itérations décroît. Donc, pour que le temps de résolution diminue il faut que ce gain d'itérations (accompagné d'une augmentation de la vitesse de calcul) soit suffisament conséquente pour compenser le surcoût de calcul par itération (environ un produit matrice vecteur en plus). En d'autres termes, si entre le degré k et le degré $k+1$ on ne gagne que "peu" d'itérations, il est inutile de tester le degré $k+1$.

Soit $\mathcal{P}$ une famille de préconditionnements polynômiaux (par exemple Minmax, Mcarre ou Leg).

Soit $\mathcal{S}$ un système linéaire $Ax = b$.

Notons N_k le nombre d'itérations pour résoudre $\mathcal{S}$ avec p_k dans $\mathcal{P}$ de degré k.

Notons N le nombre d'itérations pour résoudre $\mathcal{S}$ avec le préconditionnement Diagonal.

Alors d'après [JoMP83], (k+1) itérations avec le préconditionnement Diagonal est équivalent à une itération avec un préconditionnement polynômial Minmax de degré $k+1$. Donc d'après [JoMP83], on a pour la famille Minmax

$$N_k = \frac{N}{k+1}. \tag{IX.1}$$

Evidemment dans cette égalité N_k doit rester un entier. D'autre part, (IX.1) n'étant pas toujours vérifiée d'un point de vue numérique et comme de toute façon elle ne se vérifie pas pour les autres familles de polynômes on lui préfère

$$N_k = C\frac{N}{k+1}. \tag{IX.2}$$

avec C représentant la précision de la famille de polynômes. C dépend évidemment de la famille de polynômes choisie et des valeurs propres utilisées dans l'évaluation des polynômes.

Remarque IX.3 *Il semble au vu des résulats numériques que $C = 1$ soit une bonne estimation pour les polynômes Minmax. D'autre part, il semble que malheureusement C dépende de k et que C soit une fonction croissante de k. Malgré tout cette croissance reste faible, nous supposerons donc dans nos calculs que C est indépendante de k.*

Alors si on veut que $N_k - N_{k+1} = p$, avec p un nombre d'itérations fixé a priori (en fonction de la taille de la matrice et de la vitesse du produit matrice vecteur), il suffit d'utiliser (IX.2) et on a

$$C\frac{N}{k+1} - C\frac{N}{k+2} = p$$

d'où

$$k^2 + 3k + (2 - \frac{CN}{p}) = 0.$$

Donc si on note E la fonction partie entière, ie si $n \leq x < n+1$ alors $E(x) = n$, on a

$$\text{dès que }, k \geq E\left(\frac{-3 + \sqrt{1 + 4CN/p}}{2}\right), \qquad \text{(IX.3)}$$

$$\text{alors on a, } N_k - N_{k+1} \leq p.$$

On vérifie aisément que cette formule est vérifiée sur les premiers termes. Par exemple, il est clair que pour la famille Minmax, on a d'après [JoMP83] $N_0 - N_1 = N - N/2 = N/2$, d'où en remplaçant p par $N/2$ dans (IX.3), avec $C = 1$, on obtient

$$k \geq E\left(\frac{-3 + \sqrt{1 + 4N/(N/2)}}{2}\right) = E\left(\frac{-3 + \sqrt{9}}{2}\right) = 0.$$

Quelques exemples numériques

Nous avons déjà vu, voir la fin du chapitre VI, qu'avec l'utilisation de schémas stables une petite erreur sur le degré optimal n'entraîne qu'une perte de temps relativement faible sur la CM-2. Ainsi, grâce à l'estimation obtenue avec (IX.3) nous espérons réaliser des temps de résolution proche du temps minimum.

Nous présentons dans les trois tableaux suivants, correspondants aux trois tailles du problème du paragraphe précédent, le degré k estimé en fonction de p à l'aide de (IX.3), le temps de résolution de MINMAX de degré k (MINMAXk) enfin la perte de temps par rapport au temps minimum.

Problème 2, taille 64×64=4096

p	degré k estimé	Temps (s) MINMAXk	Perte par rapport à MINMAX26
1	39	1.106	3.6 %
2	27	1.098	2.8 %
3	22	1.103	3.3 %

Tableau IX.5: Perte de temps suivant le degré estimé a priori, sur CM-5.

Problème 2, taille 128×128=16384

p	degré k estimé	Temps (s) MINMAXk	Perte par rapport à MINMAX55
1	58	3.115	2.0 %
2	40	3.092	1.3 %
3	33	3.068	0.7 %

Tableau IX.6: Perte de temps suivant le degré estimé a priori, sur CM-5.

Problème 2, taille 256×256=65536

p	degré k estimé	Temps (s) MINMAXk	Perte par rapport à MINMAX25
1	84	14.59	7.6 %
2	59	14.09	3.9 %
3	48	13.88	2.4 %

Tableau IX.7: Perte de temps suivant le degré estimé a priori, sur CM-5.

Grâce au schémas stables et à la vitesse du produit matrice vecteur sur la CM-5, la perte de temps enregistrée par rapport au temps minimum n'excède pas 4%, si on prend p=2 ou 3. Donc, sur ce problème 2 et sur la CM-5 l'estimation obtenue avec (IX.3) est satisfaisante. Par contre, nous allons voir que sur la CM-2 les résultats sont un peu moins bons, voir le tableau IX.8, à cause de la différence de vitesse entre la CM-2 et la CM-5 qui entraîne des écarts plus importants lorsque l'on s'éloigne du degré optimal. De plus nous ne proposons pas de résultats sur le problème 1 car d'une part l'estimation réalisée à partir de (IX.3) est nettement moins efficace et de plus comme ce problème nécessite moins d'itérations la recherche du degré optimal est moins fastidieuse. Donc en dehors des problèmes possédant un grand nombre d'itérations couplé avec un produit matrice vecteur très rapide, l'utilisation de (IX.3) est plus compliquée. Le choix de p devient alors un problème aussi difficile que la découverte du degré optimal.

Problème 2, taille 256×256=65536

p	degré k estimé	Temps (s) MINMAXk	Perte par rapport à MINMAX31
2	59	59.5	5.5 %
3	48	59.0	4.6 %
4	41	59.4	3.9 %

Tableau IX.8: Perte de temps suivant le degré estimé a priori, sur CM-2.

En conclusion, il semble que (IX.3) ne nous permette pas de trouver directement le degré optimal. Néanmoins, cette égalité permet de réduire la recherche du degré optimal, et de s'approcher du temps minimum (surtout pour le problème 2).

Remarque IX.4 *Ce type de raisonnement pourrait se révéler très efficace sur des machines scalaires. En effet, sur ce type de machine l'augmentation du degré du polynôme (nombre de produits matrice vecteur) ne s'accompagne pas d'une augmentation de la vitesse de calcul. Il suffit donc dans ce cas de compter le nombre d'opérations par itération pour un polynôme de degré k puis d'optimiser en fonction de k le nombre total d'opérations.*

CHAPITRE X

INTÉRÊT DU G.C.P DANS UN PROBLÈME D'ÉVOLUTION, SUR CM-5

X.1 Introduction

Les systèmes linéaires provenant de problèmes d'évolution se résolvent souvent à l'aide de méthodes directes. En effet, si la matrice du système linéaire est indépendante du temps, alors on factorise une fois pour toute la matrice puis on résout aux temps suivants des systèmes triangulaires. Malheureusement, sur des machines de type SIMD les méthodes directes sont peu efficaces à cause de leur faible degré de parallélisme, ainsi sur ce type de machine on résout les systèmes linéaires à l'aide du G.C.P.

Alors dans ce chapitre nous allons essayer d'améliorer l'efficacité du G.C.P pour les problèmes d'évolution de la manière suivante. Nous allons stocker quelques directions de descente lors d'une première résolution avec le G.C.P d'un système $Ax = f(t)$ puis nous utiliserons ces directions stockées pour résoudre le même système mais avec un second membre $f(t + \Delta t)$.

Dans un premier temps nous donnerons quelques rappels sur l'équation de la chaleur en dimension deux. Puis nous comparerons les résidus du G.C.P avec ceux du G.C.P avec directions préstockées. Enfin, nous comparerons les temps de résolution de ces deux méthodes sur quelques exemples.

X.2 Rappels sur la résolution de l'équation de la chaleur

Nous avons résolu l'équation de la chaleur en dimension deux dans le carré $\Omega = [0, \pi]^2$. On note $u(t, x, y)$ la température à l'instant t au

point d'abscisse x et d'ordonnée y. Alors si les constantes physiques sont supposées égales à 1, u vérifie le système suivant, voir [RavT83]

$$\begin{cases} \dfrac{\partial u}{\partial t} - \Delta u = f, \ dans \ \Omega \\ \\ u_{|\partial\Omega} = 0 \\ \\ u(0,x,y) = u^0(x,y), \ pour \ (x,y) \in \Omega \end{cases} \tag{X.1}$$

avec f la source de chaleur (dans les expériences numériques nous avons pris f constante) et u^0 l'état initial de la température. Donc pour connaitre l'évolution de la chaleur il faut résoudre (X.1).

Nous avons ainsi utilisé une discrétisation uniforme du carré de pas $h = 1/(n+1)$. Notons Δt le pas de discrétisation en temps ($\Delta t > 0$) alors on note $U_{i,j}^m$ la valeur approchée de u au temps $m\Delta t$ et au point (ih, jh) (avec m positif et i et j de 1 à n). Enfin on note $f_{i,j} = f(ih, jh)$, pour i et j de 1 à n.

Alors en discrétisant suivant la Θ-méthode avec des schémas centrés, voir [RavT83], on obtient

$$\frac{U_{i,j}^{m+1} - U_{i,j}^m}{\Delta t} - \frac{1}{h^2}\Big(\Theta(U_{i+1,j}^{m+1} + U_{i,j+1}^{m+1} - 4U_{i,j}^{m+1} + U_{i-1,j}^{m+1} + U_{i,j-1}^{m+1})$$

$$+(1-\Theta)(U_{i+1,j}^m + U_{i,j+1}^m - 4U_{i,j}^m + U_{i-1,j}^m + U_{i,j-1}^m)\Big) = f_{i,j}.$$

En notant L la matrice tridiagonale par bloc d'ordre $N = n^2$

$$L = \begin{pmatrix} D_n & -I_n & & & \\ -I_n & D_n & -I_n & & \\ & \ddots & \ddots & \ddots & \\ & & -I_n & D_n & -I_n \\ & & & -I_n & D_n \end{pmatrix}$$

avec I_n la matrice identité de dimension n et

$$\forall i = 1, \ldots, n, D_n = trid_n(-1, 4, -1).$$

alors à chaque pas de temps nous résolvons le système linéaire

$$\left(I_N + \Theta\frac{\Delta t}{h^2}L\right) x_{m+1} = \left(I_N - (1-\Theta)\frac{\Delta t}{h^2}L\right) x_m$$

avec $x_m(k) = U^m_{i,j}$ avec $k = (j-1)n + i$.

Ce schéma est inconditionellement stable pour $1/2 \leq \Theta \leq 1$, voir [RavT83]. Pour $\Theta = 1/2$ nous retrouvons la méthode de Crank-Nicolson, dans les expériences numériques nous avons pris $\Theta = 2/3$.

Donc, finalement nous résolvons à chaque étape un système linéaire symétrique (car I_N et L le sont) défini positif (car I_N et L le sont et de plus Θ ainsi que Δt sont positifs) dont la matrice est indépendante du temps. Par conséquent nous pouvons appliquer le G.C.P à ce système. Dans la suite on note A et B les matrices d'ordre N $(I_N + \Theta\Delta t/h^2 L)$ et $(I_N - (1-\Theta)\Delta t/h^2 L)$ respectivement.

X.3 Résidus du G.C.P

Précisons dans un premier temps la méthode de résolution utilisée. A l'étape 1 nous résolvons le système linéaire $Ax_1 = Bx_0$ avec le G.C.P. Lors de cette résolution nous stockons dans des vecteurs semblables à x_1 les nd premiers vecteurs p^k, Ap^k et z_k ainsi que les scalaires (Ap^k, p^k), voir l'algorithme du G.C.P au chapitre II. Ensuite, lors de l'étape suivante nous résolvons le système $Ax_2 = Bx_1$ de la manière suivante. Tout d'abord nous utilisons les vecteurs préstockés comme suit

soit $\epsilon > 0$ la précision

$$r_2^0 = Bx_1 (\textit{on prend donc } x_1^0 = 0)$$

On note $\epsilon_0^2 = \epsilon \parallel r_0^2 \parallel$

Faire pour $k = 0$ **à** $nd - 1$

$$\begin{aligned} \alpha_k^2 &= (r_2^k, z_k)/(Ap^k, p^k) \\ r_2^{k+1} &= r_2^k - \alpha_k^2 Ap^k \\ x_{k+1}^2 &= x_k^2 + \alpha_k^2 p^k \end{aligned}$$

Nous notons préGCP cette procédure.

Enfin, nous terminons la résolution avec le G.C.P où nous utilisons la solution approchée à l'aide de cette première étape et dont le critère d'arrêt est $\| r_2^{k+1} \| \leq \epsilon_0^2$.

Comparons les opérations d'une itération de préGCP avec celles d'une itération d'un G.C.P. En fait, comme p^k, Ap^k, z_k et (Ap^k, p^k) sont connus nous effectuons un produit scalaire (r_2^k, z_k) et deux combinaisons linéaires par itération de préGCP. Alors que lors d'une itération du G.C.P nous calculons trois combinaisons linéaires, trois produits scalaires, un produit matrice vecteur et l'étape de préconditionnement. Le produit matrice vecteur et l'étape de préconditionnement (sauf pour le préconditionnement Diagonal) sont les phases de calcul les plus longues de l'itération du G.C.P Il est donc évident que le temps par itération de préGCP sera nettement plus faible que celui d'une itération du G.C.P. Par contre il n'est pas évident du tout que les itérations de préGCP vont permettre de s'approcher de la solution (et donc de diminuer $\| r_2^k \|$). Néanmoins les α_k^2 sont calculés pour diminuer la norme $(r_2^{k+1}, A^{-1} r_2^{k+1})$. De plus, si les seconds membres Bx_0 et Bx_1 sont suffisament "proches" on peut penser que les directions de descente ne seront pas trop "éloignées".

Nous allons donc regarder, sur les figures X.1 et X.2, le comportement au cours des premières itérations du logarithme en base 10 de $\left(\| r_2^k \| / \| r_2^0 \| \right)^2$ suivant que nous conservons ou pas des directions de descente. Nous avons testé deux préconditionnements : le préconditionnement Diagonal, avec nd=0 ou 100, et le préconditionnement polynômial Minmax de degré 4 associé au schéma de Hörner (HMINMAX4), avec nd=0 ou 14.

Enfin le problème résolu est le problème de la chaleur décrit précédemment, sur le carré $[0, \pi]^2$ avec un maillage uniforme de pas h =

$1/(n+1)$ avec $n = 512$, les matrices sont donc d'ordre 262144. Le pas de temps est $\Delta t = 5\ 10^{-3}$. La solution initiale est

$$u^0(x,y) = sin(\sqrt{x\pi})sin(\sqrt{y\pi})$$

pour $(x,y) \in [0,\pi]^2$. Et tous les calculs on été réalisés sur CM-5 32 PNs en double précision.

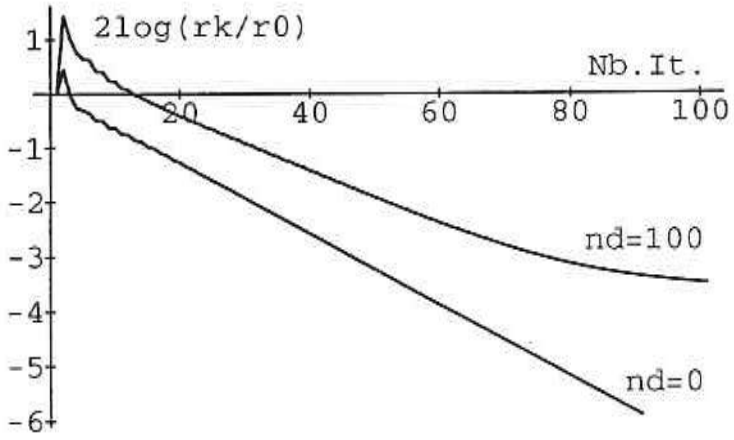

Figure X.1: Logarithme de $\left(\| r_2^k \| / \| r_2^0 \|\right)^2$ calculé avec préGCP, pour nd=0 ou 100, avec Diagonal.

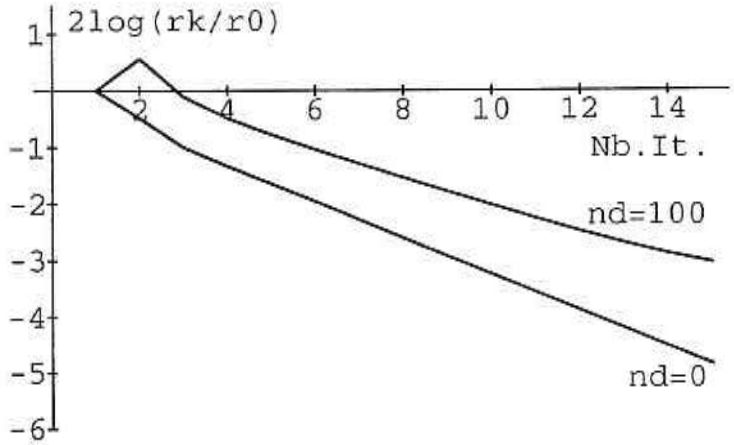

Figure X.2: Logarithme de $\left(\| r_2^k \| / \| r_2^0 \|\right)^2$ calculé avec préGCP, pour nd=0 ou 14, avec HMINMAX4.

A chaque itération du G.C.P on calcule une direction descente qui diminue de façon optimal la norme suivante $(r_2^{k+1}, A^{-1}r_2^{k+1})$. Or sur les figures X.1 et X.2 on constate que les résidus calculés à l'aide des directions stockées sont supérieurs à ceux calculés avec le G.C.P. Ces résultats numériques concordent donc avec la théorie.

D'autre part, bien que ces directions stockées ne soient pas optimales, les itérations de préGCP permettent de réduire le résidu de manière importante.

Néanmoins sur la figure suivante nous constatons qu'au temps $t = 5\ 10^{-2}$ (après donc plusieurs itérations en temps) les itérations de préGCP ne permettent plus d'atteindre des résidus aussi intéressants. Et donc pour éviter qu'au bout d'un certain temps préGCP ne soit plus efficace on pourrait relancer le stockage des directions à partir d'un certain nombre de pas de temps.

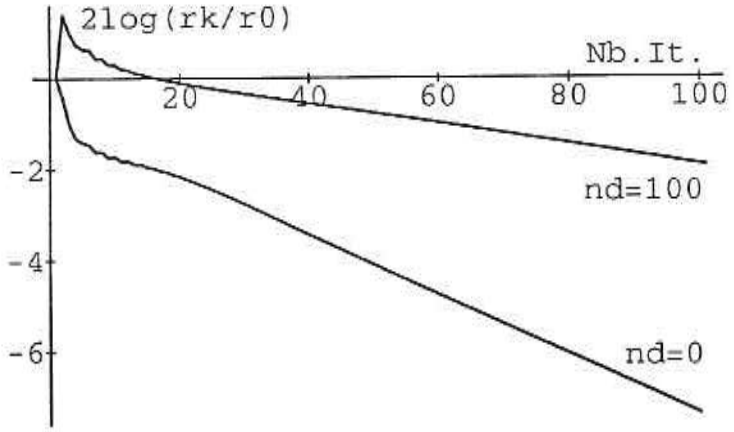

Figure X.3: Logarithme de $\left(\| r_{10}^k \| / \| r_{10}^0 \|\right)^2$ calculé avec préGCP, pour nd=0 ou 100, avec Diagonal.

Regardons maintenant le comportement du G.C.P après avoir effectué préGCP.

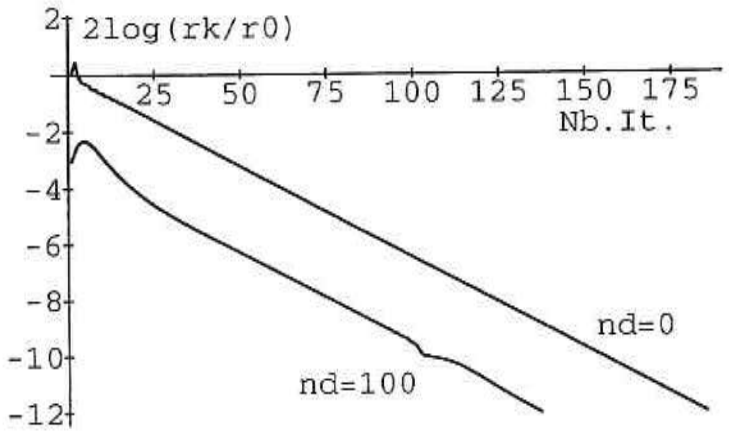

Figure X.4: Logarithme de $\left(\| r_2^k \| / \| r_2^0 \| \right)^2$ du GCP après préGCP, pour nd=0 ou 100, avec Diagonal.

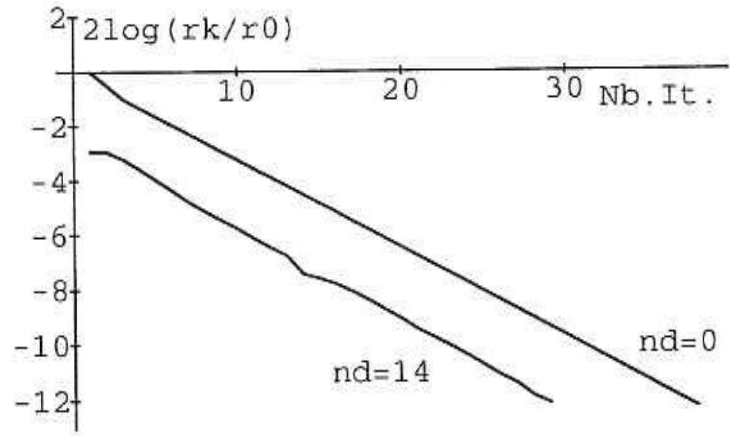

Figure X.5: Logarithme de $\left(\| r_2^k \| / \| r_2^0 \| \right)^2$ du GCP après préGCP, pour nd=0 ou 14, avec HMINMAX4.

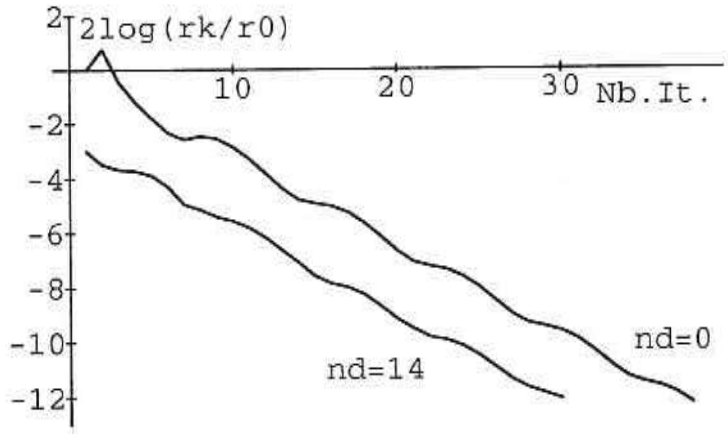

Figure X.6: Logarithme de $\left(\| r_2^k \| / \| r_2^0 \| \right)^2$ du GCP après préGCP, pour nd=0 ou 14, avec HMINMAX4 et une source nulle.

Les figures X.4, X.5 et X.6 nous montrent qu'après nd itérations de préGCP, le G.C.P retrouve une vitesse de convergence semblable à celle du G.C.P sans préGCP. A l'aide de préGCP nous réalisons donc un certain gain d'itérations. Pour les figures X.4 et X.5 nous avons pris un terme source constant non nul relativement grand, ainsi le logarithme des résidus est quasi-linéaire, alors que pour la dernière figure nous avons pris une source nulle. Dans les deux cas on constate néanmoins que les courbes sont "parallèles", et donc que les vitesses de convergence sont identiques.

X.4 Temps de résolution

Nous présentons maintenant les temps de calculs (en secondes) du problème précédent, avec une source constante non nulle, pour les préconditionnements Diagonal et HMINMAX4. Dans les deux tableaux suivants nous présentons en fonction du nombre de directions stockées (nd) : le temps de stockage de ces directions (Temps Stock.), le temps

passé dans préGCP (Temps préGCP), le nombre d'itérations du G.C.P (It.) après les nd itérations de préGCP, le temps de calculs pour les It. itérations du G.C.P (Temps G.C.P), le temps total de résolution (Temps Résol.=Temps préGCP+ Temps G.C.P) enfin le gain de temps par rapport à une résolution sans stockage (nd=0).

nd	Temps Stock.	Temps préGCP	It. G.C.P	Temps G.C.P	Temps Résol.	Gain (%)
0	-	-	184	2.670	2.670	-
70	0.094	0.246	144	2.093	2.339	12.4
95	0.123	0.332	138	2.003	2.335	12.5
100	0.130	0.350	137	1.992	2.342	12.3

Tableau X.1: Résultats du G.C.P Diagonal avec préstockage des directions.

nd	Temps Stock.	Temps préGCP	It. G.C.P	Temps G.C.P	Temps Résol.	Gain (%)
0	-	-	37	1.478	1.478	-
14	0.041	0.052	29	1.153	1.205	18.5
19	0.046	0.069	28	1.112	1.182	20.0
38	0.068	0.135	28	1.112	1.247	15.6

Tableau X.2: Résultats du G.C.P HMINMAX4 avec préstockage des directions.

D'après les tableaux X.1 et X.2 les gains de temps réalisés avec l'utilisation de préGCP sont appréciables (de 12 à 20%). De plus, les directions étant déjà stockées on peut les utiliser lors d'une nouvelle résolution. Ainsi le surplus de temps dû au stockage, pas trop important d'ailleurs, est d'autant moins pénalisant que nous réutilisons ces directions.

Ensuite, il semble inutile de stocker toutes les directions calculées lors de la première résolution. En effet, à partir d'un certain nd la diminution du résidu réalisée à l'aide de préGCP est insuffisante pour diminuer le nombre d'itérations du G.C.P. Néanmoins, pour atteindre des gains de temps maximum il faut tout de même stocker assez de directions. En effet, le temps par itération du G.C.P étant relativement important, le gain de une itération grâce à préGCP peut s'avérer suffisante pour diminuer le temps total de résolution.

Enfin, cette méthode de préstockage est tout de même très coûteuse en place mémoire (nd=90 pour Diagonal), néanmoins avec des préconditionnements polynômiaux de degrés élevés nous avons moins de directions à stocker.

CONCLUSION

Dans [McBr88] et [McBr89] il apparaît que le G.C est beaucoup mieux adapté à une architecture massivement parallèle que les méthodes multigrilles. De plus le préconditionnement Diagonal est souvent considéré comme la méthode capable d'atteindre les temps de résolution minimum. Le but de cette thèse était donc de montrer, au moins sur des problèmes modèles, qu'il existe des préconditionnements plus efficaces (au point de vue temps de résolution) que le préconditionnement Diagonal sur des machines massivement parallèles de type SIMD, telles que CM-2.

Notre étude sur la résolution, sur CM-2, CM-200 ou CM-5, de systèmes linéaires provenant de la discrétisation par le schéma à cinq points d'équations aux dérivées partielles elliptiques, nous permet de donner quelques conclusions.

En premier lieu, les préconditionnements de base de cette thèse (séries de Neumann tronquées ou renumérotation Rouge-Noir) ne sont pas plus rentables en vitesse (Mflops) que le préconditionnement Diagonal. Néanmoins certains parviennnent à diminuer le temps total de résolution.

Par contre, les gains de temps (bien plus importants que ceux déjà publiés, voir notamment [ChKT89] et [Weil94]) ainsi que la rapidité des préconditionnements polynômiaux Minmax, Mcarre, Leg et Norm sont très prometteurs, sur des problèmes réguliers. En effet, le temps de calcul du G.C préconditionné par la diagonale est tout de même, sur les deux problèmes, divisé par deux avec ces techniques. De plus, même avec des structures moins régulières nous avons vu au chapitre IX que des préconditionnements polynômiaux certes de faible degré diminuent les temps de résolution de la méthode Diagonale. Par ailleurs, grâce à l'introduction de schémas plus stables, [AsMO92] et [CiMP93], ou bien grâce à l'utilisation de doubles préconditionnements nous parvenons à optimiser les temps de résolution réalisés à l'aide du

simple schéma de Hörner. De plus, l'utilisation des schémas stables permet une évaluation, a priori, moins précise du degré du polynôme sans augmentation trop importante du temps de résolution. D'autre part, la comparaison entre les préconditionnements polynômiaux et les préconditionnements proposés dans [Lore95] réalisée sur la CM-5 dans l'annexe 2 nous confirme l'efficacité des préconditionnements polynômiaux sur ce type de problèmes réguliers. Ensuite, il est clair d'après le chapitre X qu'il existe encore d'autres moyens (le préstockage des directions de descente) efficaces de réduire les temps de résolution du G.C.P pour les problèmes d'évolution.

Enfin, il est clair que Norm constitue un préconditionnement rentable sur la CM-2, et néanmoins pratique car il ne nécessite aucune estimation des valeurs propres. Dans cette thèse nous proposons donc deux familles de préconditionnements polynômiaux Minmax et Norm (ainsi que des schémas) très efficaces en temps et également en vitesses, capables de donner sur la CM-2 des temps de résolution très inférieurs à ceux réalisés par le préconditionnement Diagonal sur des problèmes réguliers. DIAG n'est donc plus la méthode donnant les temps de résolution les plus faibles sur ce type de machine et même pour des structures moins régulières.

Si les problèmes résolus dans cette thèse paraissent trop académiques, ils restent quand même d'actualité en particulier en mécanique des fluides. Par exemple dans la résolution avec des domaines fictifs, les succès obtenus par Boeing dans le cadre tridimensionnel sur des problèmes d'écoulements transsoniques [BBJMSY91] et des problèmes électromagnétiques [BBJSY91], ou encore récemment les travaux sur des écoulements instationnaires autour d'obstacles mobiles [He94] prouvent qu'il reste important de rechercher (sur tout type de machines) des solveurs rapides pour l'équation de Poisson. Les problèmes proposés ne sont donc pas si éloignés des problèmes industriels. Il serait en outre intéressant d'étudier le couplage de méthodes de domaines fictifs avec des préconditionnements polynômiaux sur des machines parallèles. De toute façon l'efficacité des préconditionnements polynômiaux a déjà été étudié sur de nombreux problèmes [HoVB90].

La suite naturelle de ces travaux est l'application de ces préconditionnements au cas de problèmes définis pour des maillages quelconques bidimensionnels, avant de passer aux problèmes définis dans un domaine tridimensionnel. Dans ce cas, les nombreuses études réalisées sur des structures plus irrégulières [BPRS91], [PeWe92], [Peti93], [EdPe93], [FPSW93], [Weil94] et chapitre IX, et sur le placement des données [Feau91] et [Feau93] nous laissent optimistes sur le rendement des préconditionnements polynômiaux de degré moins élevé sur des structures moins régulières. Il serait également très intéressant de tester les préconditionnements polynômiaux sur des problèmes plus concrets tels que la résolution des équations de Navier-Stokes.

De plus, Freund et Fischer suggèrent dans [FrFi94] que les préconditionnements polynômiaux adaptés à la distribution initiale des valeurs propres sont plus efficaces (en nombre d'itérations) que Minmax et Leg. Alors si l'étape d'initialisation de ce type de polynômes ne s'avère pas trop importante, le gain de temps réalisé à l'aide de ces préconditionnements polynômiaux pourrait se révéler intéressant.

D'autre part, l'efficacité des préconditionnements polynômiaux repose en partie sur la vitesse du produit matrice vecteur disponible. Il serait donc bon d'adapter ces méthodes à d'autres machines parallèles que la CM-2, comme dans [Meur89b]. En effet, il existe d'autres types d'ordinateurs massivement parallèles. Par exemple, le Paragon, l'IPSC 860, le Ncube, le Suprenum,... sont des calculateurs massivement parallèles à mémoire locale MIMD (Multiple Instruction Multiple Data). Sur cette classe de machines, les PEs sont certes moins nombreux mais ils sont beaucoup plus puissants. Surtout ils peuvent effectuer des opérations différentes en même temps. Les produits matrice-vecteur efficaces sont déjà nombreux. Il aurait été également intéressant d'étudier la famille des préconditionnements basée sur des approximations par blocs sur ce type de machine MIMD. Enfin, grâce aux résultats enregistrés, au chapitre VIII, sur les doubles préconditionnements une étude plus approfondie et notamment le couplage des préconditionnements polynômiaux avec d'autres types de préconditionnements (IC, MIC ou approximations par blocs) pourrait se révéler fructueuse sur des machines séquentielles ou massivement parallèles MIMD.

Finalement, nous proposons une dernière utilisation des préconditionnements polynômiaux. En effet, les préconditionnements polynômiaux ne nécessitent que trois opérations élémentaires : combinaison linéaire de vecteurs, produit matrice vecteur et produit scalaire. Nous n'avons donc pas besoin de connaître explicitement la matrice à préconditionner. Ainsi, nous pourrions éviter le coût (souvent important) d'assemblage d'une matrice provenant de la discrétisation par éléments finis d'équations aux dérivées partielles. Il suffirait de créer un produit matrice vecteur à partir des matrices élémentaires, puis de résoudre le système linéaire à l'aide du G.C préconditionné par un polynôme.

ANNEXE 1

Dans le chapitre VII, nous proposons un préconditionnement polynômial, Norm, ne nécessitant aucune estimation des valeurs propres. En effet, grâce à une normalisation symétrique de la matrice par sa diagonale, c'est à dire en résolvant

$$D^{-1/2}AD^{-1/2}y = \tilde{A}x = D^{-1/2}b, \ puis \ D^{1/2}x = y \qquad (.4)$$

$$avec \ D = diag_N(a_i),$$

nous avons estimé la plus grande valeur propre de $\tilde{A}$ par $b = 2$, voir le Lemme II.1, et la plus petite par $a = 0$, comme dans [Saad85]. Ensuite en injectant ces évaluations dans le polynôme Mcarre nous avons obtenu le polynôme Norm. Ainsi la mise en œuvre informatique de Norm s'est trouvée très simplifiée par rapport à Mcarre. D'autre part, dans le chapitre VII, nous avons testé Norm uniquement avec son schéma stable, néanmoins il est clair que comme les taux de convergence de Norm et de Mcarre sont sensiblement égaux nous pouvons également utilisé le préconditionnement polynômial Norm associé au schéma de Hörner.

Alors, nous proposons dans cette annexe d'exprimer les vingts premiers polynômes Norm (au delà il faudra sans doute utiliser le schéma stable du chapitre VII) pour simplifier quelque peu le travail des utilisateurs. En effet, ceux-ci disposeront ainsi d'un préconditionnement directement utilisable (de la même manière que les séries de Neumann tronquées), après avoir bien entendu normalisé symétriquement la matrice par sa diagonale.

Reprenons la définition de Norm, voir le paragraphe III.5.5,

$$p_k(\lambda) = \sum_{j=0}^{k+1} b_j t_j(\lambda),$$

avec, voir (III.9),

$$b_j = \frac{s_j(0)}{\sum_{i=0}^{k+1} s_i^2(0)},\ t_j(\lambda) = \frac{s_j(0) - s_j(\lambda)}{\lambda}.$$

Comme les s_j sont les polynômes orthonormaux associés au poids de Tchebycheff sur [0,2] on a

$$s_0(\lambda) = \frac{1}{\sqrt{\pi}},\ \text{et pour tout } i \neq 0,\ s_i(\lambda) = \sqrt{\frac{2}{\pi}} T_i(\lambda - 1),$$

avec T_i le polynôme de Tchebycheff de première espèce de degré i.

Car dans ce cas ($a = 0$ et $b = 2$), l'application linéaire bijective $\mu : [a, b] \to [-1, 1]$ se réduit à

$$\mu(\lambda) = \lambda - 1.$$

Ainsi

$$\sum_{i=0}^{k+1} s_i^2(0) = \frac{1}{\pi} + \frac{2}{\pi} \sum_{i=1}^{k+1} (-1)^{2i} = \frac{2k+3}{\pi}.$$

Comme $t_0 \equiv 0$, finalement

$$\begin{aligned} p_k(\lambda) &= \sum_{i=1}^{k+1} ((-1)^i (2\pi)^{1/2}/(2k+3)) \cdot (2/\pi)^{1/2} \frac{(-1)^i - T_i(\lambda - 1)}{\lambda} \\ &= \frac{2}{(2k+3)\lambda} \left((k+1) - \sum_{i=1}^{k+1} T_i(1 - \lambda) \right). \end{aligned}$$

Nous avons également vu au chapitre VII que le polynôme Norm peut s'écrire de la manière suivante

$$p_k(\lambda) = \frac{1}{\lambda} \left(1 - \frac{J_{k+1}(1-\lambda)}{J_{k+1}(1)} \right),$$

avec $J_{k+1}(\lambda)$ le polynôme de Jacobi sur $[-1, 1]$ associé au poids $\omega(\frac{1}{2}, -\frac{1}{2}, \lambda)$. Nous avons préféré dans cette annexe exprimer le polynôme Norm en fonction des polynômes de Tchebycheff T_i, car l'expression des T_i est bien plus connue que celle des J_i.

Alors voici les vingts premiers polynômes Norm

$$
\begin{aligned}
p_0(\lambda) &= 2/3 \\
p_1(\lambda) &= -4/5\ \lambda + 2 \\
p_2(\lambda) &= 8/7\ \lambda^2 - 4\lambda + 4 \\
p_3(\lambda) &= -16/9\ \lambda^3 + 8\lambda^2 - 12\lambda + 20/3 \\
p_4(\lambda) &= 32/11\ \lambda^4 - 16\lambda^3 + 32\lambda^2 - 28\lambda + 10 \\
p_5(\lambda) &= -64/13\ \lambda^5 + 32\lambda^4 - 80\lambda^3 + 96\lambda^2 - 56\lambda + 14 \\
p_6(\lambda) &= 128/15\ \lambda^6 - 64\lambda^5 + 192\lambda^4 - 880/3\ \lambda^3 + 240\lambda^2 \\
&\quad -504/5\ \lambda + 56/3 \\
p_7(\lambda) &= -256/17\ \lambda^7 + 128\lambda^6 - 448\lambda^5 + 832\lambda^4 - 880\lambda^3 + 528\lambda^2 \\
&\quad -168\lambda + 24 \\
p_8(\lambda) &= 512/19\ \lambda^8 - 256\lambda^7 + 1024\lambda^6 - 2240\lambda^5 + 2912\lambda^4 \\
&\quad -2288\lambda^3 + 1056\lambda^2 - 264\lambda + 30 \\
p_9(\lambda) &= -1024/21\ \lambda^9 + 512\lambda^8 - 2304\lambda^7 + 17408/3\ \lambda^6 - 8960\lambda^5 \\
&\quad +8736\lambda^4 - 16016/3\ \lambda^3 + 13728/7\ \lambda^2 - 396\lambda + 110/3 \\
p_{10}(\lambda) &= 2048/23\ \lambda^{10} - 1024\lambda^9 + 5120\lambda^8 - 14592\lambda^7 + 26112\lambda^6 \\
&\quad -30464\lambda^5 + 23296\lambda^4 - 11440\lambda^3 + 3432\lambda^2 - 572\lambda + 44 \\
p_{11}(\lambda) &= -4096/25\ \lambda^{11} + 2048\lambda^{10} - 11264\lambda^9 + 35840\lambda^8 - 72960\lambda^7 \\
&\quad +496128/5\ \lambda^6 - 91392\lambda^5 + 56576\lambda^4 - 22880\lambda^3 + 5720\lambda^2 \\
&\quad -4004/5\ \lambda + 52
\end{aligned}
$$

puis

$$
\begin{aligned}
p_{12}(\lambda) &= 8192/27\ \lambda^{12} - 4096\lambda^{11} + 24576\lambda^{10} - 259072/3\ \lambda^9 \\
&\quad +197120\lambda^8 - 306432\lambda^7 + 330752\lambda^6 - 248064\lambda^5 \\
&\quad +127296\lambda^4 - 388960/9\ \lambda^3 + 9152\lambda^2 - 1092\lambda + 182/3
\end{aligned}
$$

$$
\begin{aligned}
p_{13}(\lambda) &= -16384/29\ \lambda^{13} + 8192\lambda^{12} - 53248\lambda^{11} + 204800\lambda^{10} \\
&\quad -518144\lambda^9 + 906752\lambda^8 - 1123584\lambda^7 + 992256\lambda^6 \\
&\quad -620160\lambda^5 + 268736\lambda^4 - 77792\lambda^3 + 14144\lambda^2 - 1456\lambda \\
&\quad +70
\end{aligned}
$$

$$
\begin{aligned}
p_{14}(\lambda) &= 32768/31\ \lambda^{14} - 16384\lambda^{13} + 114688\lambda^{12} - 479232\lambda^{11} \\
&\quad +1331200\lambda^{10} - 2590720\lambda^9 + 3627008\lambda^8 - 3691776\lambda^7 \\
&\quad +2728704\lambda^6 - 1447040\lambda^5 + 537472\lambda^4 - 134368\lambda^3 \\
&\quad +21216\lambda^2 - 1904\lambda + 80
\end{aligned}
$$

$$
\begin{aligned}
p_{15}(\lambda) &= -65536/33\ \lambda^{15} + 32768\lambda^{14} - 245760\lambda^{13} + 3325952/3\ \lambda^{12} \\
&\quad -3354624\lambda^{11} + 7188480\lambda^{10} - 33679360/3\ \lambda^9 \\
&\quad +12953600\lambda^8 - 11075328\lambda^7 + 20920064/3\ \lambda^6 \\
&\quad -3183488\lambda^5 + 11286912/11\ \lambda^4 - 7390240/3\ \lambda^3 \\
&\quad +31008\lambda^2 - 2448\lambda + 272/3
\end{aligned}
$$

$$
\begin{aligned}
p_{16}(\lambda) &= 131072/35\ \lambda^{16} - 65536\lambda^{15} + 524288\lambda^{14} - 2539520\lambda^{13} \\
&\quad +8314880\lambda^{12} - 97284096/5\ \lambda^{11} + 33546240\lambda^{10} \\
&\quad -123114240/7\ \lambda^9 + 42099200\lambda^8 - 30764800\lambda^7 \\
&\quad +83680256/5\ \lambda^6 - 6656384\lambda^5 + 1881152\lambda^4 - 361760\lambda^3 \\
&\quad +310080/7\ \lambda^2 - 15504/5\ \lambda + 102
\end{aligned}
$$

$$
\begin{aligned}
p_{17}(\lambda) &= -262144/37\ \lambda^{17} + 131072\lambda^{16} - 1114112\lambda^{15} + 5767168\lambda^{14} \\
&\quad -20316160\lambda^{13} + 51552256\lambda^{12} - 97284096\lambda^{11} \\
&\quad +138977280\lambda^{10} - 151557120\lambda^9 + 126297600\lambda^8 \\
&\quad -79988480\lambda^7 + 38036480\lambda^6 - 13312768\lambda^5 + 3328192\lambda^4 \\
&\quad -568480\lambda^3 + 62016\lambda^2 - 3876\lambda + 114
\end{aligned}
$$

enfin

$$\begin{aligned} p_{18}(\lambda) = \; & 524288/39\ \lambda^{18} - 262144\lambda^{17} + 2359296\lambda^{16} \\ & -38993920/3\ \lambda^{15} + 49020928\lambda^{14} - 134086656\lambda^{13} \\ & +824836096/3\ \lambda^{12} - 430829568\lambda^{11} + 521164800\lambda^{10} \\ & -488350720\lambda^{9} + 353633280\lambda^{8} - 196335360\lambda^{7} \\ & +247237120/3\ \lambda^{6} - 332819200/13\ \lambda^{5} + 5705472\lambda^{4} \\ & -2615008/3\ \lambda^{3} + 85272\lambda^{2} - 4788\lambda + 380/3 \end{aligned}$$

$$\begin{aligned} p_{19}(\lambda) = \; & -1048576/41\ \lambda^{19} + 524288\lambda^{18} - 4980736\lambda^{17} \\ & +29097984\lambda^{16} - 116981760\lambda^{15} + 343146496\lambda^{14} \\ & -759824384\lambda^{13} + 1296171008\lambda^{12} - 1723318272\lambda^{11} \\ & +1795123200\lambda^{10} - 1465052160\lambda^{9} + 932305920\lambda^{8} \\ & -458115840\lambda^{7} + 171164160\lambda^{6} - 47545600\lambda^{5} \\ & +9509120\lambda^{4} - 1307504\lambda^{3} + 115368\lambda^{2} - 5852\lambda + 140 \end{aligned}$$

Ensuite il ne reste plus qu'à résoudre le système (1) à l'aide du G.C préconditionné par Norm, associé au schéma de Hörner évidemment.

ANNEXE 2

Comparaison de préconditionnements Diagonal, Minmax et Loreaux

Dans cette annexe nous proposons une comparaison entre le préconditionnement Diagonal,le préconditionnement polynômial Minmax associé à un schéma stable, voir chapitre VI, et le préconditionnement présenté dans [Lore95] par P. Loreaux en collaboration avec F. Alouges et C. Labourdette. Plus précisement, nous utilisons le préconditionnement décrit au chapitre 4 de [Lore95], la méthode 2. Cette méthode 2 consiste à approcher A^{-1} comme suit

$$A^{-1} \approx (I - E_1^T)(I - E_2^T)...(I - E_L^T)D^{-1}(I - E_1)(I - E_2)...(I - E_L)$$

avec D une matrice diagonale. Précisons cette factorisation de A. En fait, à chaque étape on annule une diagonale de A. On définit ainsi $A_1 = A$, puis on construit E_1 une matrice sous-diagonale pour annuler une diagonale de A_1 de la manière suivante

$$A_2 = (I - E_1)A_1(I - E_1^T).$$

Ensuite les E_i sont définies par récurrence, pour plus de détails sur la factorisation de A et donc la définition des E_i voir [Lore95].

Nous avons comparé les trois préconditionnements : Diagonal, Minmax et Loreaux sur le problème suivant

$$\begin{cases} -div(\lambda \nabla u) = f, \text{ dans } \Omega \\ u_{|\partial\Omega} = 0 \end{cases},$$

où

$$\lambda = \begin{pmatrix} c(x,y) & 0 \\ 0 & d(x,y) \end{pmatrix},$$

et avec $\Omega =]0,1[^2$. Le schéma de discrétisation est le schéma aux différences finies décrit au chapitre I. Le maillage est un maillage uniforme de pas $1/(n+1)$ dans les deux directions, numéroté de façon naturelle.

Nous avons résolu effectivement trois problèmes. Les deux premiers sont déjà présentés dans cette thèse. Le premier problème est le problème de Poisson classique. Le second problème est plus difficile à résoudre numériquement car il présente des sauts de coefficients importants dans la première direction, il possède une structure tridiagonale dominante. Enfin, le troisième possède également des sauts de coefficients importants mais dans les deux directions.

Problème 1

$$\begin{cases} c \equiv 1 \\ d \equiv 1 \end{cases}$$

Problème 2

$$\begin{cases} c(x,y) = \begin{cases} 1000, \text{ si } x \in]\frac{1}{4},\frac{3}{4}[\\ 1, \text{ sinon} \end{cases} \\ \\ d \equiv 1 \end{cases}$$

Problème 3

$$\begin{cases} c(x,y) = \begin{cases} 1000, \text{ si } x \in]\frac{1}{4},\frac{3}{4}[\\ 1, \text{ sinon} \end{cases} \\ \\ d(x,y) = \begin{cases} 1000, \text{ si } y \in]\frac{1}{4},\frac{3}{4}[\\ 1, \text{ sinon} \end{cases} \end{cases}$$

Enfin, afin de bien résoudre les mêmes systèmes que dans [Lore95], nous avons pris un second membre b où

$$b(i) = sin(\sqrt{i}),\ pour\ i = 1,\ldots,n^2$$

Nous présentons dans les trois tableaux suivants les résultats réalisés par Diagonal, MINMAX et Loreaux sur les trois problèmes proposés,

sur une CM-5 32 PNs en mode multiutilisateur. Précisons les notations dans ces tableaux, Taille n correspond au nombre de points de maillage sur une direction donc les matrices sont d'ordre n^2. Puis Préc. est évidemment le préconditionnement utilisé, on note MINMAXk le préconditionnement Minmax de degré k associé à son schéma stable et Loreaux$L$ la méthode 2 de Loreaux avec L diagonales. It. correspond au nombre d'itérations du G.C.P. Ensuite, tous les temps mesurés sont en secondes (il s'agit du busy time de la CM-5 et non de l'elapsed comme dans [Lore95], voir remarque II.14). Temps Fact. est le temps de factorisation (nul pour Diagonal et MINMAX, pas négligeable pour Loreaux). Enfin Temps It. correspond au temps passé pour la résolution avec le G.C.P hors factorisation et Temps Total est le temps total de résolution, il s'agit donc la somme de Temps Fact. avec Temps It..

Taille n	Préc.	It.	Temps Fact.	Temps It.	Temps Total
256	Diagonal	600	-	2.0	2.0
	Loreaux3	275	2.6	2.4	5.0
	MINMAX33	23	-	1.6	1.6
512	Diagonal	1333	-	11.0	11.0
	Loreaux3	516	10.8	12.0	22.8
	MINMAX38	33	-	8.1	8.1

Tableau .1: Comparaison de préconditionnements pour le problème 1, sur CM-5.

Les résultats, en particulier les Temps de calcul, ne semblent pas concorder en tout point avec ceux proposés dans [Lore95]. En effet, sur la CM-5 on enregistre des écarts entre le busy time et l'elapsed time plus importants que sur la CM-2. Mais pour utiliser l'elapsed time il faut tester les préconditionnements sur la CM-5 en mode monoutilisateur et même réserver la machine pour plusieurs heures (ou jours!), ce qui n'était évidemmment pas envisageable. Tout de même, si on compare quelques elapsed times mesurés en mode multiutilisateur (et donc

peu fiable) alors le préconditionnement Loreaux a un Temps It. (3.5 pour n=256 et 13.6 pour n=512) plus faible que le préconditionnement Diagonal (respectivement 4.0 et 21.8) mais plus élevé que MINMAX (respectivement 3.2 et 12.0). Donc sur ce problème il semble bien que MINMAX soit la méthode capable d'atteindre les temps de calcul les plus faibles.

Taille n	Préc.	It.	Temps Fact.	Temps It.	Temps Total
256	Diagonal	8225	-	28.0	28.0
	Loreaux9	295	11.6	4.6	16.2
	MINMAX34	243	-	17.2	17.2
512	Diagonal	16275	-	157.2	157.2
	Loreaux9	584	45.2	24.7	69.9
	MINMAX35	469	-	106.3	106.3

Tableau .2: Comparaison de préconditionnements pour le problème 2, sur CM-5.

Taille n	Préc.	It.	Temps Fact.	Temps It.	Temps Total
256	Diagonal	4524	-	15.4	15.4
	Loreaux9	732	11.6	21.0	32.6
	MINMAX40	116	-	11.0	11.0
512	Diagonal	8985	-	86.6	86.6
	Loreaux11	838	57.6	96.7	154.3
	MINMAX34	265	-	68.7	68.7

Tableau .3: Comparaison de préconditionnements pour le problème 3, sur CM-5.

Les Temps It. réalisés par la méthode Loreaux sur le problème 2 sont très intéressants. Ces bons résultats sont dûs en partie à une très

grande accélération de la convergence sur ce problème. En effet, on voit sur les tableaux .2 et .3 que sur le problème 2 de taille 256 Loreaux avec 9 diagonales réduit le nombre d'itérations de Diagonal d'un facteur 28 alors que sur le problème 3 le facteur de réduction de la même méthode n'est plus que de 6. Le caractère tridiagonal dominant de la matrice du problème 2 semble donc très bien adapté au préconditionnement Loreaux. D'autre part, les Temps It. de MINMAX sont également inférieurs à ceux de Diagonal et cette méthode reste intéressante par rapport à Loreaux surtout si on compare le Temps Total, notamment pour la taille 256.

Sur le dernier problème nous avons testé le préconditionnement polynômial Minmax à l'aide de réels double précision. Il semble en effet que la grandeur des coefficients de la matrice du problème 2, engendre des erreurs de calculs qui empêche MINMAX de converger convenablement. Néanmoins grâce aux unités vectorielles de la CM-5 l'utilisation de réels double précision n'a pas entraîné pas de surcoût de temps de calcul trop important.

Enfin nous remarquons sur le tableau .3 que les Temps It. de MINMAX sont inférieurs à ceux de Diagonal et de Loreaux. De plus, même si nous avions comparé des elapsed times nous aurions constaté que, comme pour le problème 1, MINMAX est la méthode capable de réaliser les temps de résolution minimum devant Loreaux puis Diagonal.

Conclusions

Il apparaît dans notre étude que le préconditionnement Loreaux est un préconditionnement efficace sur le problème 2. Il serait encore plus intéressant pour des problèmes d'évolution. Par contre sur les deux autres problèmes le rendement de MINMAX est bien meilleur, et même sur le problème 2 il diminue les temps de résolution de Diagonal. En tout cas, il est clair que sur ce type de problèmes très réguliers Diagonal n'est pas la méthode la plus rapide d'un point de vue temps de résolution.

ANNEXE 3

Influence des préconditionnements polynômiaux sur les valeurs propres

Nous proposons dans cette annexe de regarder l'influence des préconditionnements polynômiaux sur les valeurs propres. Ce travail a été réalisé avec l'aide de P. Joly (CNRS/Laboratoire d'Analyse Numérique, Université P. et M. CURIE). Ainsi nous présentons dans les figures suivantes les valeurs propres calculées des matrices $P_k(A)A$, avec k variant de 0 à 7 et $P_k(A)$ étant évalué à l'aide du classique schéma de Hörner. De plus, nous avons normalisé symétriquement A par sa diagonale, les valeurs propres de A sont donc comprises entre 0 et 2. Nous avons pris alors dans la définition des polynômes la plus grande valeur propre à 2 ensuite nous avons fait varier la plus petite a ($a = 0.016$ puis 0.1).

La matrice étudiée provient de la discrétisation du problème suivant, il s'agit du problème 2

Problème 2

$$\begin{cases} c(x,y) = \begin{cases} 1000, \text{ si } x \in]\frac{1}{4}, \frac{3}{4}[\\ 1, \text{ sinon} \end{cases} \\ \\ d \equiv 1 \end{cases}$$

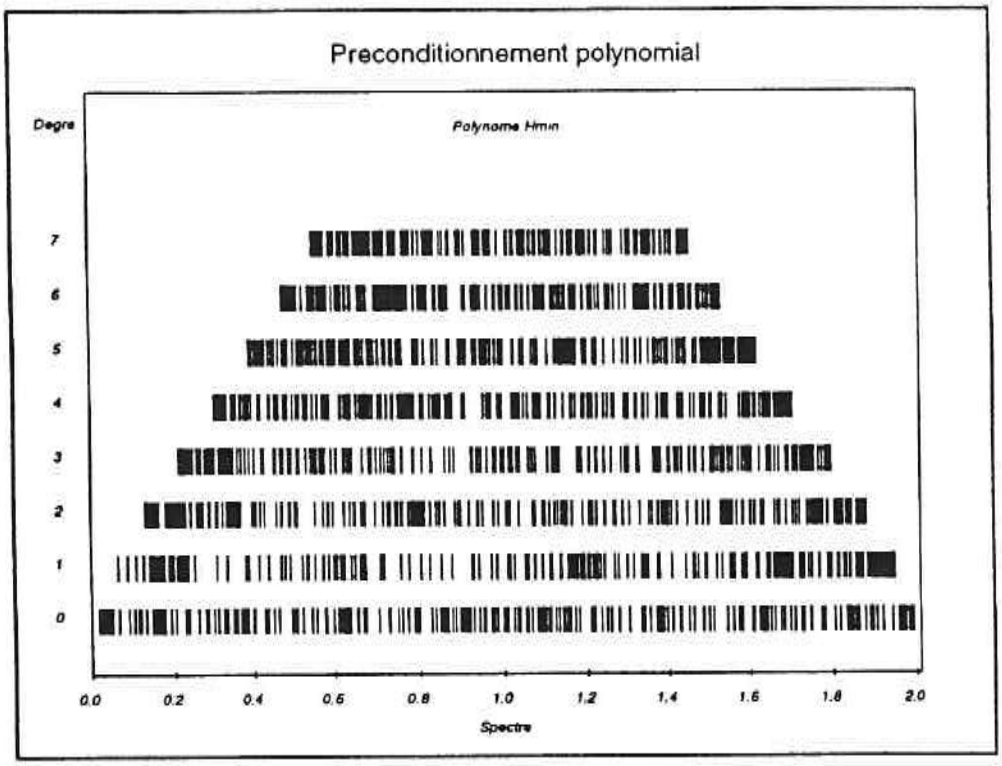

Figure 1 : Préconditionnement par MinMax, avec $a = 0.016$

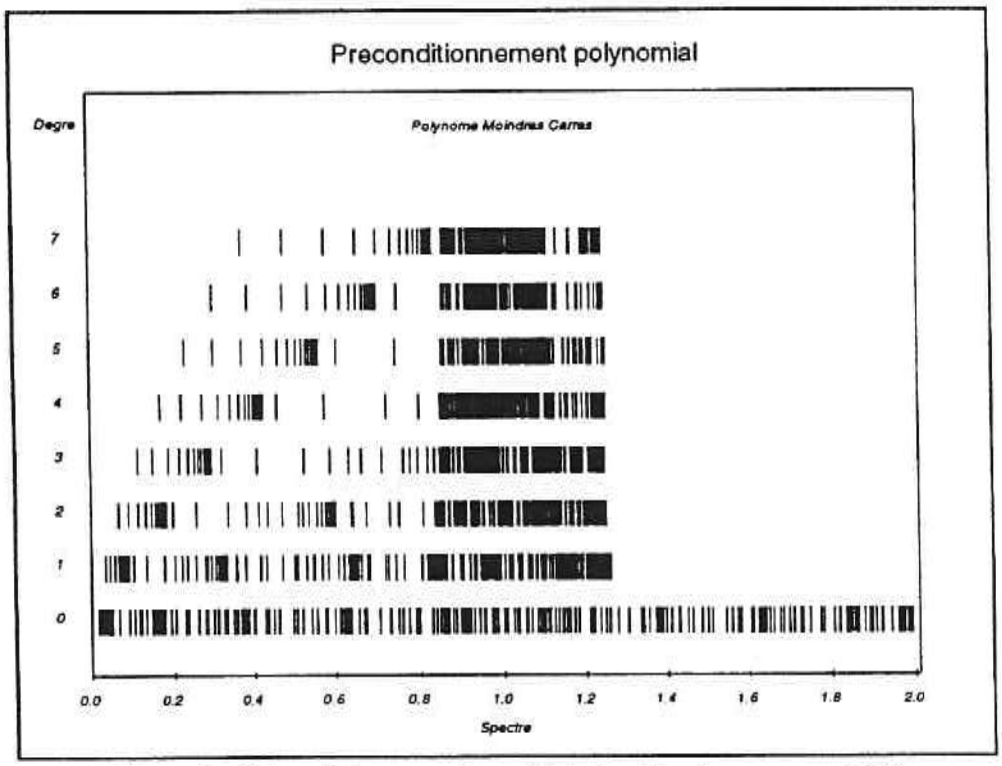

Figure 2 : Préconditionnement par Moindres Carrés, avec $a = 0.016$

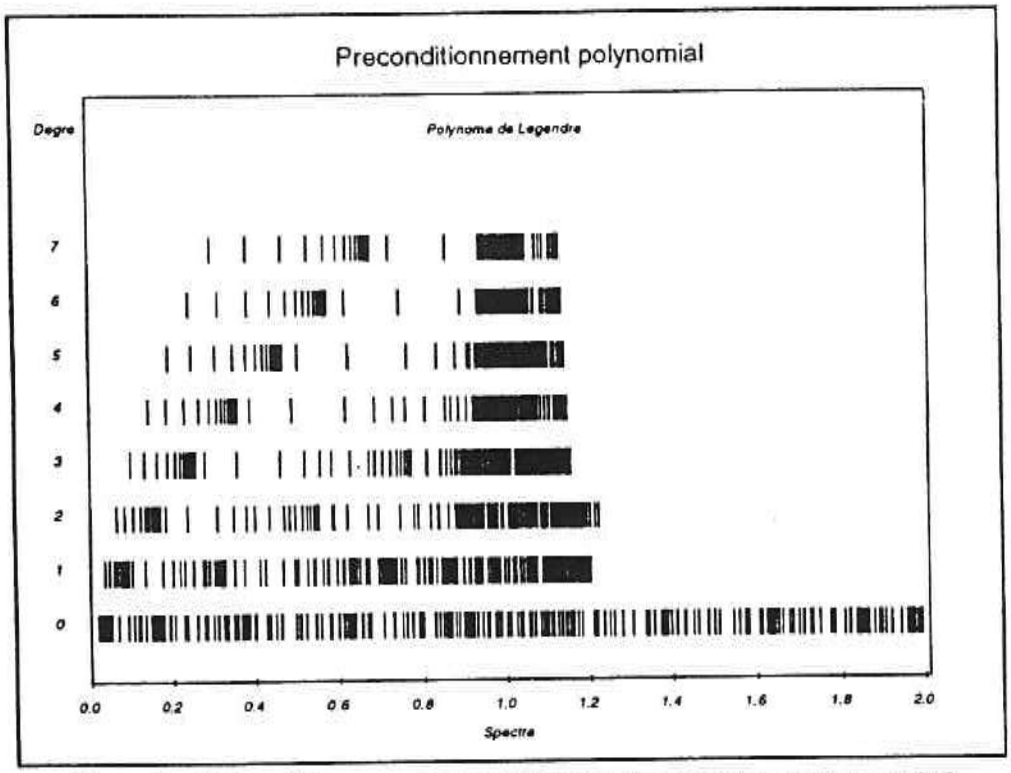

Figure 3 : Préconditionnement par polynômes de Legendre, avec $a = 0.016$

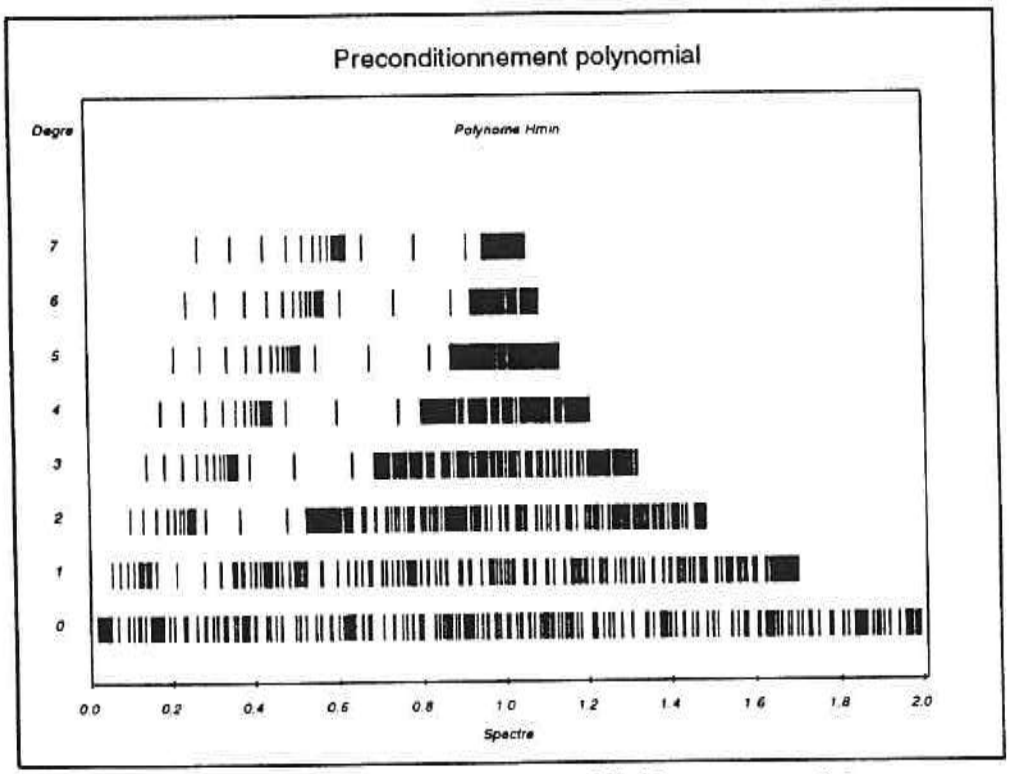

Figure 4 : Préconditionnement par MinMax, avec $a = 0.1$

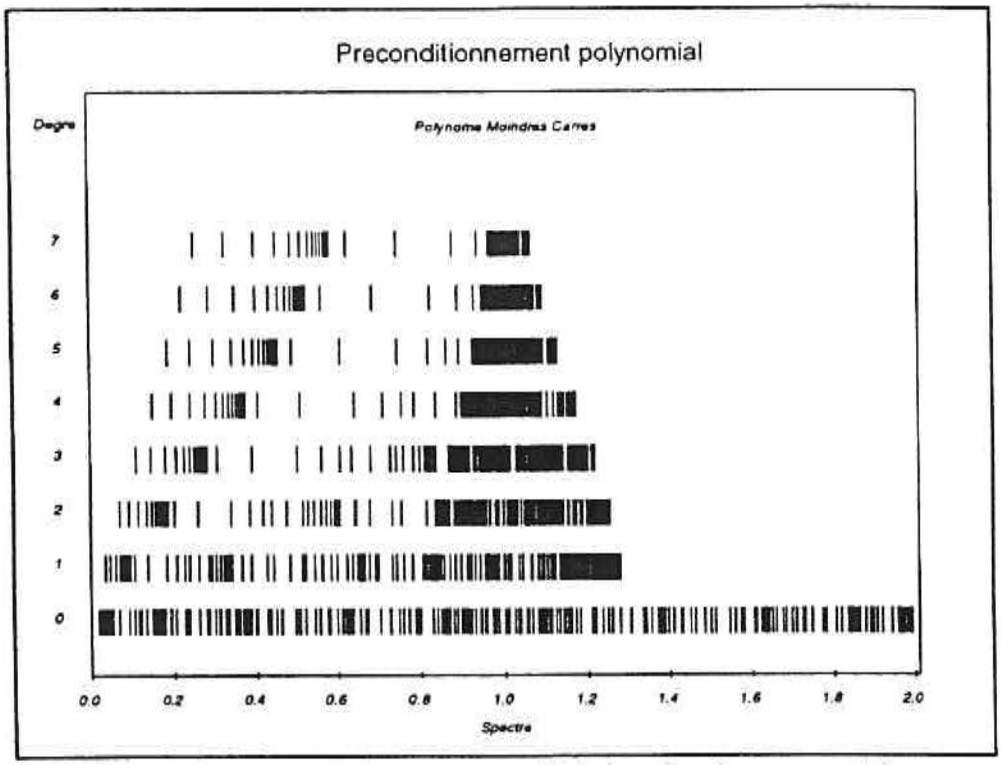

Figure 5 : Préconditionnement par Moindres Carrés, avec $a = 0.1$

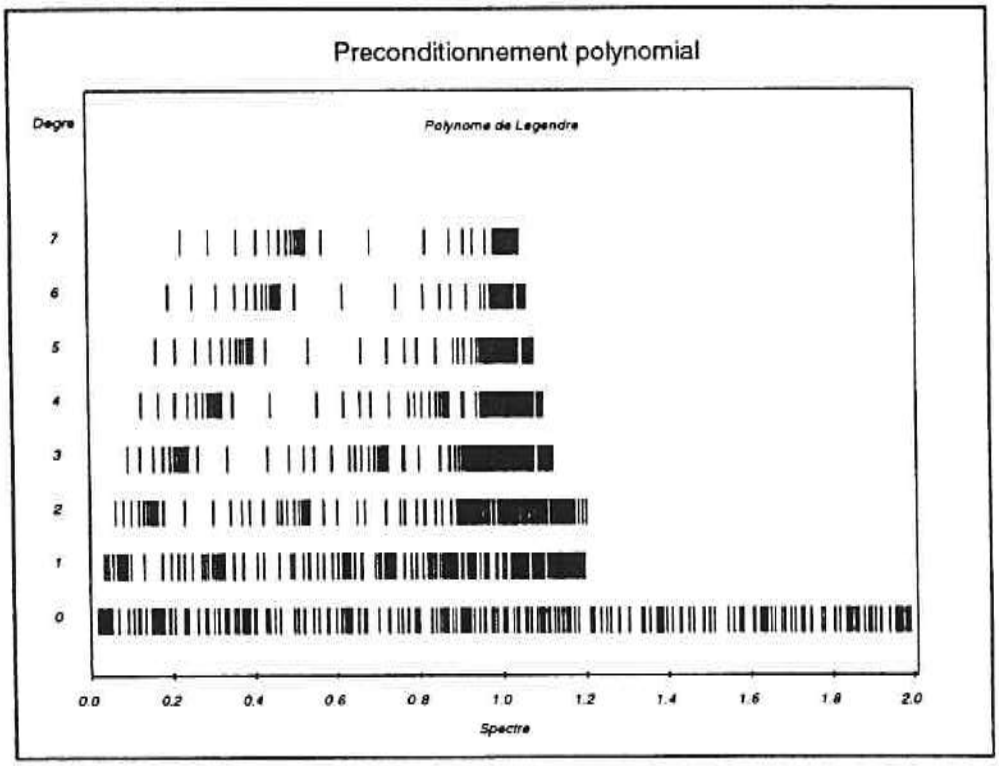

Figure 6 : Préconditionnement par polynômes de Legendre, avec $a = 0.1$

Nous constatons sur les trois premières figures que le spectre dépend fortement de la famille de polynôme choisie. Ainsi, le polynôme Minmax distribue les valeurs propres de façon uniforme, alors que les polynômes des Moindres carrés (Mcarre) et de Legendre rapprochent plus rapidement les plus grandes valeurs propres de la valeur 1. D'autre part, Minmax semble plus efficace sur les petites valeurs propres que Mcarre et Legendre.

Pour les premières figures nous avons pris une valeur de a très satisfaisante ($a = 0.016$, voir le tableau .1). En effet, la plus petite valeur propre calculée est 0.0170269. Par contre pour les figures suivantes nous avons pris une valeur de a ($a = 0.1$) bien plus éloignée de la plus valeur propre réelle. Alors nous constatons sur ces figures que l'estimation a priori de a est très importante pour Minmax, et un peu moins pour Mcarre et Legendre. Ainsi si on prend a supérieur à la valeur exacte de la plus petite valeur propre, alors il semble que les préconditionnements polynômiaux s'occupent assez mal des valeurs propres inférieures à a.

Enfin nous présentons, pour les trois familles de préconditionnements polynômiaux, dans les deux tableaux suivants la plus petite et la plus grande valeur propre (respectivement Val. p. Min. et Val. p. Max.) ainsi que le conditionnement de $P_k(A)A$ ($\kappa(A)$), pour les deux valeurs de a précédentes.

Nous constatons ainsi que la famille de préconditionnement Minmax est plus efficace que Mcarre et Legendre sur le conditionnement, pour les deux valeurs de a. De plus, il est clair, une nouvelle fois, que le choix de la valeur a a des conséquences importantes avec Minmax et un peu moins avec les autres polynômes.

Préc. pol.	Degré	Val. p. Min.	Val. p. Max	$\kappa(A)$
Minmax	0	0.170269E-01	0.198297E+01	0.116461E+03
	1	0.649508E-01	0.193890E+01	0.298518E+02
	2	0.129150E+00	0.187072E+01	0.144849E+02
	3	0.211846E+00	0.178820E+01	0.844102E+01
	4	0.300618E+00	0.169934E+01	0.565281E+01
	5	0.389202E+00	0.161080E+01	0.413873E+01
	6	0.472962E+00	0.152705E+01	0.322870E+01
	7	0.549450E+00	0.145071E+01	0.264029E+01
Mcarre	0	0.170269E-01	0.198297E+01	0.116461E+03
	1	0.341661E-01	0.125732E+01	0.368001E+02
	2	0.682718E-01	0.124539E+01	0.182417E+02
	3	0.113285E+00	0.124410E+01	0.109820E+02
	4	0.168243E+00	0.124526E+01	0.740154E+01
	5	0.231509E+00	0.124538E+01	0.537941E+01
	6	0.300809E+00	0.124322E+01	0.413292E+01
	7	0.373444E+00	0.123777E+01	0.331446E+01
Legendre	0	0.170269E-01	0.198297E+01	0.116461E+03
	1	0.337996E-01	0.119976E+01	0.354962E+02
	2	0.626938E-01	0.121808E+01	0.194290E+02
	3	0.988668E-01	0.115037E+01	0.116355E+02
	4	0.141534E+00	0.114271E+01	0.807378E+01
	5	0.189715E+00	0.113721E+01	0.599433E+01
	6	0.242261E+00	0.113231E+01	0.467392E+01
	7	0.297904E+00	0.112730E+01	0.378411E+01

Tableau .1: Conditionnement du problème 2, avec a=0.016.

Préc. pol.	Degré	Val. p. Min.	Val. p. Max	$\kappa(A)$
Minmax	0	0.170269E-01	0.198297E+01	0.116461E+03
	1	0.544592E-01	0.169289E+01	0.310855E+02
	2	0.981931E-01	0.147962E+01	0.150684E+02
	3	0.139421E+00	0.131588E+01	0.943818E+01
	4	0.176796E+00	0.120351E+01	0.680735E+01
	5	0.211014E+00	0.112996E+01	0.535489E+01
	6	0.242904E+00	0.108267E+01	0.445721E+01
	7	0.273026E+00	0.105251E+01	0.385498E+01
Mcarre	0	0.170269E-01	0.198297E+01	0.116461E+03
	1	0.350222E-01	0.127573E+01	0.364263E+02
	2	0.687106E-01	0.124945E+01	0.181842E+02
	3	0.107704E+00	0.121275E+01	0.112600E+02
	4	0.147119E+00	0.116773E+01	0.793732E+01
	5	0.184383E+00	0.112422E+01	0.609718E+01
	6	0.218932E+00	0.108814E+01	0.497020E+01
	7	0.251067E+00	0.106075E+01	0.422499E+01
Legendre	0	0.170269E-01	0.198297E+01	0.116461E+03
	1	0.333893E-01	0.119245E+01	0.357136E+02
	2	0.605809E-01	0.119619E+01	0.197454E+02
	3	0.922130E-01	0.111979E+01	0.121435E+02
	4	0.126047E+00	0.109606E+01	0.869567E+01
	5	0.160307E+00	0.107477E+01	0.670444E+01
	6	0.193899E+00	0.105628E+01	0.544758E+01
	7	0.226300E+00	0.104123E+01	0.460112E+01

Tableau .2: Conditionnement du problème 2, avec a=0.1.

RÉFÉRENCES BIBLIOGRAPHIQUES

- [Adam82] L M Adams, *Iterative algorithms for large sparse linear systems on parallel computers*, Ph. D. thesis, Dept. of Applied Mathematics, University of Virginia, Charlottesville, VA, 1982.

- [Adam85] L M Adams, *m-step preconditioned conjugate gradient methods*, SIAM J. Sci. Statis. Comput., 6 (1985), pp 452-463.

- [Ashb87] S F Ashby, *Polynomial preconditioning for conjugate gradient method*, Ph. D. thesis, Dept. of Computer Science, University of Illinois, Urbana, IL, December 1987.

- [Ashb91] S F Ashby, *Minimax polynomial preconditioning for hermitian linear systems*, SIAM J. Mat. Anal. Appl., 12 (1991), pp 766-789.

- [AsMO92] S F Ashby, T A Manteuffel and J S Otto, *A comparison of adaptative Chebyshev and least squares polynomial preconditioning for hermitian positive definite systems*, SIAM J. Sci. Statis. Comput., 13 (1992), pp 1-29.

- [BBJMSY91] M B Bieterman, J E Bussoletti, F T Johnson, R G Melvin, S S Samanth and D P Young, *A locally refined finite rectangular grid finite element method. Application to computational physics*, J. Comp. Physics, (1991), pp 1-66.

- [BBJSY91] R H Burkhart, J E Bussoletti, F T Johnson, S S Samanth and D P Young, *Emtranair : Steps toward solution of general 3d Maxwell's equations*, In R Glowinski, editor, in Computers Methods in Applied Sciences and Engineering, Commack, NY 1991, pp 49-72.

- [BjDa74] A Björck and G Dahlquist, *Numerical methods*, Prentice-Hall (1974).

- [BPRS91] H Berryman, S Petiton, A Rifkin and J Saltz, *Performance effects of irregular communications patterns on massively parallel multiprocessors*, Journal of parallel and distributed computing, (1991), 13, pp 202-212.

- [ChKT89] T F Chan, C J Kuo and C Tong, *Parallel elliptic preconditioners : Fourier analysis and performance on the Connection Machine*, Comput. Phys. Comm., 53 (1989), pp 237-252.
- [Ciar82] P Ciarlet, *Introduction à l'analyse numérique matricielle et à l'optimisation*, Masson, Paris 1982.
- [CiMP93] P Ciarlet Jr, G Meurant et O Perlot, *Préconditionnements massivement parallèles*, journées du SEH, Janvier 1993.
- [CoGM85] P Concus, G H Golub and G Meurant, *Block preconditioning for the conjugate gradient method* , SIAM J. Sci. Statis. Comput., 6 (1985), pp 220-252.
- [CoGO76] P Concus, G H Golub and D P O'Leary, *A generalized conjugate gradient method for the numerical solution of elliptic partial differential equations*, in Sparse Matrix Computations, J. R. Bunch and D. J. Rose, eds., Academic Press, New York, 1976, pp 309-332.
- [Davi75] P J Davis, *Interpolation and approximation*, Dover, New York, 1975.
- [DuGR79] P F Dubois, A Greenbaum and G H Rodrigue, *Approximating the inverse of a matrix for use in iterative algorithms on vectors processors*, Computing, 22 (1979), pp 257-268.
- [DuMe89] I S Duff and G Meurant, *The effect of ordering on preconditioned conjugate gradients*, BIT v 29 (1989) pp 635-657.
- [EdPe93] G Edjlali and S Petiton, *Data parallel structures and algorithms for sparse matrix computation*, in Advances in Parallel Computing, North-Holland, Amsterdam, 1993.
- [Feau91] P Feautrier, *Dataflow Analysis of Array and Scalar References*, International Journal of Parallel Programming, 20 (1991), pp 23-53.
- [Feau93] P Feautrier, *Toward automatic distribution*, in ACM International Conference on Supercomputing, Tokyo (1993).

- [Flyn72] M J Flynn, *Some computer organizations and their effectiveness*, IEEE Transactions on Computers, vol. C-21, (1972), pp 948-960.
- [FPSW93] W Ferng, S Petiton, Y Saad and K Wu, *Basic sparse matrix computation on the CM-5*, International Journal of Modern Phycics C., 4 (1993), pp 65-83.
- [Freu89] R W Freund, *Polynomial preconditioners for Hermitian and certain Nonhermitian Matrices*, paper presented at SIAM Annual Meeting, San Diego, CA, July 1989.
- [FrFi92] R W Freund and B Fischer, *On adaptative polynomial preconditioning for Hermitian positive definite matrices*, Proceedings of the Copper Mountain Conference on Iterative Methods, Copper Mountain, April 1992.
- [FrFi94] R W Freund and B Fischer, *On Adaptative Weighted Polynomial Preconditioning for Hermitian Positive Definite Matrices*, SIAM J. Sci. Comput., 15 (1994), pp 408-4111.
- [Gaut86] W Gautschi, *On the Sensitivity of Orthogonal Polynomials to Pertubations in the Moments*, Numer. Math. 48, 369-382 (1986).
- [Gaut90] W Gautschi, *Some applications and numerical methods for orthogonal polynomials*, Num. Anal. and Math. Mod., Banach Center Pub., 24, PWN-Polish Scientific Publishers, Warsaw 1990.
- [GiRa86] V Girault and P A Raviart, *Finite elements methods for Navier-Stokes equations*, Springer-Verlag, Berlin Heidelberg 1986.
- [GoMe83] G H Golub et G Meurant, *Résolution numérique des grands systèmes linéaires*, Eyrolles, Paris, 1983.
- [He94] J He, *Méthodes de domaines fictifs en mécanique des fluides, Applications aux écoulements potentiels instationnaires autour d'obstacles mobiles*, Doctorat de Mathématiques appliquées, Université P et M Curie, 1994.

- [HeSt52] M R Hestenes and E Stiefel, *Methods of conjugate gradients for solving linear systems*, J. Res. Nat. Bur. Standards, 49 (1952), pp 409-435.

- [Hill85] W D Hillis, *The Connection Machine*, MIT Press, Cambridge, MA, 1985.

- [Hill88] W D Hillis, *La Machine à Connexions*, Masson, 1988.

- [HoVB90] B Holter and G Van den Berghe, *A comparison of vectorized methods for solving the two-dimensional diffusion equation: multigrids versus polynomial preconditioned conjugate gradient*, Applied Mathematics and Computation, 40, pp 77-103 (1990).

- [Jenn77] A Jennings, *Influence of the eigenvalue spectrum on the convergence rate of the conjugate gradient method*, J. Inst. Math. Applics, 20 (1977), pp 307-316.

- [Joly90] P Joly, *Résolution numérique des grands systèmes linéaires*, Publications du Laboratoire d'Analyse Numérique, Université P et M Curie, nov. 1990.

- [JoMP83] O G Johnson, C A Micchelli and G Paul, *Polynomial preconditioning for conjugate gradient calculations*, SIAM J. Numer. Anal.,20 (1983), pp 362-376.

- [Kers70] D Kershaw, *Inequalities on the elements of the inverse of a certain tridiagonal matrix*, Math. Comp., 24 (1970), pp. 155-158.

- [LaTh86] P Lascaux et R Théodor, *Analyse numérique matricielle appliquée à l'art de l'ingénieur*, Masson, Paris, 1986.

- [Lore95] P Loreaux, *Simulation numérique d'instabilités de mélange sur calculateur massivement parallèle et schémas numériques d'ordre élevé*, Doctorat de mathématiques appliquées, Université P et M Curie, 1995.

- [McBr88] O A Mac Bryan, *New Architecture : Performances highlights and new algorithms*, Parallel Computing, 7, 477-499, 1988.

- [McBr89] O A Mac Bryan, *The Connection Machine : PDE Solution on 65536 processors*, Parallel Computing, 9, 1-24, 1988/1989.

- [Maro92] P Maroni, *Une théorie algébrique des polynômes orthogonaux. Application aux polynômes orthogonaux semi-classiques*, Publications du Laboratoire d'Analyse Numérique, Université P et M Curie, dec. 1992.

- [Meur89a] G Meurant, *Iterative methods for multiprocessor vector computers*, Computer Physics Reports 11 (1989), pp 51-80.

- [Meur89b] G Meurant, *Practical use of the C.G.M on parallel supercomputers*, Computer Physics Communications 53 (1989), pp 467-477.

- [MeVa77] J A Meijerink and H Van Der Vorst, *An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix*, Math. Comp., 31 (1977), pp. 148-162.

- [Peti93] S G Petiton, *Contribution à une méthodologie Globale pour le Calcul Scientifique parallèle*, rapport d'habilitation à diriger des recherches, Université Paris VI, Octobre 1993.

- [PeWe92] S G Petiton and C J Weill-Duflos, *Very sparse preconditioned conjugate gradient on massively parallel architectures*, SEH publications, 1992.

- [Porc91] T Porcher, *Architecture de la CM-2 CM-200*, journées du SEH 1991, tome 1.

- [PSWF93] S Petiton, Y Saad, K Wu et W Ferng, *Basic sparse matrix computations on the CM-5*, International Journal of Modern Physics C., vol. 4, Fev. 1993, pp 65-83, World Scientific Publishing Company.

- [RaTh83] P A Raviart et J M Thomas, *Introduction à l'analyse numérique des équations aux dérivées partielles*, Masson, 1983.
- [Reid71] J K Reid, *On the method of conjugate gradients for the solution of large sparse systems of linear equations*, In : Large sparse Sets of Linear equations, éd. par J K Reid, pp 231-254, Academic Press.
- [Robi91] F Robin, *Le langage CM-Fortran*, journées du SEH 1991, tome 1.
- [Ruti59] H Rutishauser, *Theory of gradient methods in refined iterative methods for computation of the solution and the eigenvalues of a self-adjoint boundary value problem*, Institute of Applied Mathematics, Zurich, Basel-Stuttgart, 1959, pp24-49.
- [Saad85] Y Saad, *Practical use of polynomial preconditioning for the conjugate gradient method*, SIAM J. Sci. Statis. Comput., 6 (1985), pp 865-881.
- [Saad87] Y Saad, *Least squares polynomials in the complex plane and their use for solving nonsymmetric linear systems*, SIAM J. Numer. Anal., 24 (1987), pp 155-169.
- [Stie58] E L Stiefel, *Kernel polynomials in linear algebra and their applications*, U.S. NBS Aplied Math. Series, 49 (1958), pp 1-24.
- [StBu80] J Stoer and R Bulirsch, *Introduction to Numerical Analysis*, Springer Verlag, New York (1980).
- [ThMC90a] Thinking Machines Corporation, *Connection Machine model CM-2 technical summary*, version 6.0, nov. 1990.
- [ThMC90b] Thinking Machines Corporation, *Connection Machine CM-5 technical summary*, version 6.0, jan. 1990.
- [ThMC91] Thinking Machines Corporation, *Getting started in CM Fortran*, jan. 1991.

- [Tong89] C Tong, *The preconditioned conjugate gradient method on the Connection Machine*, in proceedings of the conference on scientific applications of the Connection Machine, éd. par Simon (Horst D), World Scientific.

- [Uebe94] P Ueberholz, *Lattice gauge theory on the Connection Machine*, journées du SEH 1994.

- [Varg62] R S Varga, *Matrix iterative analysis*, Prentice-Hall (1962).

- [Weil94] C J Weill-Duflos, *Optimisation de méthodes de résolution itératives de grands systèmes linéaires creux sur machines massivement parallèles*, Doctorat d'Informatique, Université P et M Curie, 1994.