Université des Sciences et Technologies de Lille
U.F.R. de Mathématiques Pures et Appliquées
Laboratoire Paul Painlevé - U.M.R.-C.N.R.S 8524

Méthodes d'Accélération de Convergence en Analyse Numérique et en Statistique

THÈSE

présentée et soutenue publiquement le 27 juin 2005
pour l'obtention du

Doctorat de l'Université de Lille I
(Spécialité : Mathématiques Appliquées)

par

Christophe ROLAND

Composition du jury

Président :	Albert Cohen	Professeur Univ. P. et M. Curie, Paris.
Rapporteurs :	Alain Berlinet	Professeur, Univ. Montpellier II, Montpellier.
	Gérard Meurant	Directeur de Recherches, CEA, Bruyères-Le-Chatel.
	Marcos Raydan	Professeur, Univ. Centrale du Vénézuela, Caracas.
Examinateurs :	Bernhard Beckermann	MdC Habileté Univ. de Lille I, Villeneuve d'Ascq.
	Claude Lemaréchal	Directeur de Recherches INRIA Rhône-Alpes, Saint-Ismier.
Directeur de Thèse :	Claude Brezinski	Professeur, Univ. de Lille I, Villeneuve d'Ascq.

A mes grands-parents maternels

Madeleine et Omer DUCHATELLE-MADRAGORE

A mes grands-parents paternels

Florine et Omer ROLAND-PAMART

Remerciements

Je remercie *Jean-Paul Morillon* et *Serge Nicaise* de m'avoir initié à l'analyse numérique et persuadé de poursuivre ma maîtrise de mathématiques par le D.E.A.

Je remercie *Bernhard Beckermann* pour son accueil chaleureux dès mon arrivée à Lille en tant qu'étudiant en DEA. Depuis ce moment, il m'a toujours accueilli dans son bureau avec enthousiasme. Il a su être d'une disponibilité totale pour répondre à mes questions mathématiques et administratives.

Je remercie mon directeur de thèse, *Claude Brezinski*, pour ses encouragements aux initiatives personnelles et son respect d'une grande liberté d'action tout en répondant à mes questions dès qu'il le fallait. Je lui suis particulièrement reconnaissant de m'avoir permis de travailler sur l'algorithme EM en me mettant en contact avec Ravi Varadhan. Pour tout cela, je lui adresse un grand merci.

Je remercie *Jean-Paul Chehab* pour ses discussions mathématiques et pour les nombreux articles qu'il m'a conseillés et commentés.

Je remercie *tous mes collaborateurs* qui m'ont permis incontestablement d'évoluer. En particulier, un grand merci à *Ravi Varadhan* qui, par ses mails incessants, me fait partager sa passion pour la recherche, son expérience et ses idées.

Je remercie *Thierry Goudon* de m'avoir incité à participer au Cemracs 2003, et guidé dans des choix judicieux pour l'avenir.

Je remercie *tous les membres du jury* pour le temps qu'ils ont passé à juger mon travail, j'ai été sensible à leurs remarques qui ont permis d'améliorer la qualité de ce manuscrit, et leurs encouragements pour la suite.

Je remercie *tous les membres de l'équipe AN-EDP du laboratoire Paul Painlevé* de l'université de Lille pour leurs discussions, leurs conseils et le partage de leurs expériences.

Je remercie *l'équipe de l'IRMA* de Strasbourg, plus particulièrement *Eric Sonnendrücker*, pour leur accueil à Luminy lors du Cemracs 2003 et à Strasbourg lors de mon bref séjour.

Je remercie *les thésards et ex-thésards de Lille et de Valenciennes*, en particulier *Grégory Boutry et Abdelatif Tinzefte* pour les moments de détente qu'ils m'ont offerts mais aussi pour le partage de leurs connaissances sur les méthodes itératives et les problèmes de valeurs propres. Merci aussi à *Delphine Jennequin* de n'avoir cessé de me répéter de nombreuses commandes Linux.

Je remercie *toutes les personnes* qui m'ont accordé du temps et de l'intérêt au cours de conférences, groupes de travail, séminaires et séjours en centre de recherche auxquels j'ai participé.

Je remercie *mes amis cyclotouristes du club de Fontaine-au-Piré*. Ils m'ont régulièrement permis, grâce au sport, d'oublier mes préoccupations mathématiques et d'évacuer la tension qu'elles suscitent.

Je remercie aussi *toute ma famille* et *ma belle-famille* pour leur présence et leur soutien. Je n'oublie pas *tous mes amis* qui me supportent depuis des années. Et enfin, je remercie ma femme *Audrey* du fond du coeur pour son soutien si précieux...

Introduction

En analyse numérique et en mathématiques appliquées, nous devons souvent utiliser des suites. Elles sont, par exemple, utilisées pour résoudre un système linéaire, un problème de point fixe ou pour trouver la solution d'une équation aux dérivées partielles. En pratique, ces suites peuvent converger lentement ce qui entraîne une certaine réticence à les utiliser. C'est pour cette raison que des méthodes d'accélération de convergence sont étudiées depuis de nombreuses années. Ces méthodes sont basées généralement sur l'idée très naturelle d'extrapolation qui a mené en outre à la théorie de la transformation des suites. Dans le cas scalaire, ces transformations sont basées sur des estimations de l'erreur [24] ; un outil important pour leur construction est le complément de Schur [26]. La compréhension de la méthodologie de telles transformations dans le cas vectoriel a progressé [23] et certains algorithmes basés sur des projections sont apparus [25]. En 1980, Delahaye et Germain-Bonne [51] ont prouvé sous certaines conditions qu'une transformation universelle de suites pour accélérer la convergence de toutes les suites convergentes ne peut exister. D'un point de vue optimiste, ce résultat fondamental en théorie de transformation de suites pourrait signifier qu'il semble toujours intéressant de trouver et étudier de nouvelles transformations de suites, puisqu'en fait chacune d'entre elles est *seulement* capable d'accélérer la convergence de *certaines* classes de suites. Il faudrait alors classifier les suites et proposer pour chaque classe une transformation de suites qui accélére la convergence de toutes les suites appartenant à cette classe. Un autre point de vue est, non pas de s'intéresser à la suite, mais de remettre en cause le processus ou la méthode qui a produit cette suite. Dans ce contexte, il faut essayer de comprendre la méthode utilisée et en proposer une amélioration. Dans cette thèse, je m'intéresse à l'accélération de la convergence, au moyen des transformations de suites (chapitre 2) mais plus généralement par une amélioration des méthodes itératives (les autres chapitres). Je me consacre à trois domaines différents : résolution de systèmes linéaires, résolution de problèmes de point fixe (problèmes de bifurcation, statistiques) et enfin à un problème de physique des plasmas. C'est pour cette raison que cette thèse est décomposée en trois parties distinctes.

Dans le chapitre 1, je m'intéresse à deux méthodes différentes proposées par Altman [3, 4] pour résoudre un système linéaire $Ax = b$, où la matrice est supposée hermitienne définie positive d'ordre p. Ces méthodes peuvent être considérées comme des méthodes de sous-espaces de Krylov pour résoudre un système projeté du système initial $Ax = b$. Le lien avec les méthodes classiques de sous-espaces de Krylov est précisé et des résultats à la fois théoriques et numériques sur le comportement de la convergence sont donnés. Ce

travail effectué en collaboration avec B. Beckermann et C. Brezinski a donné lieu à une publication [103].

Dans le chapitre 2, je considère un système linéaire $Ax = b$ où la matrice A est supposée non singulière d'ordre p et $\tilde{x}$ une approximation du vecteur x obtenue soit par une méthode directe, soit par une méthode itérative. Auchmuty [10] puis Brezinski [27] ont obtenu des estimations de la norme de l'erreur $e = x - \tilde{x}$ qui sont indépendantes de la méthode utilisée et sont valables pour n'importe quelle norme vectorielle. Ces estimées peuvent être obtenues directement ou par une procédure d'extrapolation. Dans ce chapitre, l'approche donnée par Brezinski [27] est utilisée afin d'obtenir des estimations du vecteur erreur lui-même. Le but est d'utiliser cette nouvelle approche afin d'obtenir plusieurs méthodes connues de projection pour résoudre un système d'équations linéaires. Habituellement, ces méthodes sont obtenues par des approches différentes : projection sur différents plans, minimisation du résidu correspondant, ou, quand la matrice est symétrique et définie positive, une minimisation d'une fonctionnelle quadratique. Je vais donc montrer à partir d'une estimation du vecteur erreur grâce à la décomposition en valeurs singulières, en abrégé S.V.D., que ces méthodes (la méthode de la plus profonde descente, le gradient conjugué, les méthodes multiparamètres) habituellement présentées indépendamment peuvent découler de cette approche. Cela clarifie la connection entre les méthodes étudiées et simplifie leurs présentations. De cette idée découlent des procédures pour accélérer la convergence d'une méthode itérative.

Dans le chapitre 3, je propose une nouvelle méthode qui peut être considérée comme une modification des méthodes Δ^k introduites par Brezinski et Chehab [28] pour résoudre des problèmes non linéaires de point fixe $X = T(X)$, où $T : \mathbb{R}^p \to \mathbb{R}^p$. A chaque itération du nouveau schéma, le pas de descente de la méthode Δ^k est évalué une fois et utilisé deux fois. Des résultats numériques variés illustrent l'efficacité des nouveaux schémas. Ils concernent la solution d'un problème de réaction-diffusion avec bifurcations. Un autre exemple, impliquant une distribution de Poisson suggère que le nouveau schéma peut être adapté avec succès à un problème important en statistique qui concerne le problème d'accélération de convergence de l'algorithme EM [52]. A noter que ce chapitre fait l'objet d'une publication [104].

Les chapitres 4, 5 et 6 sont la suite logique du chapitre 3. En effet, le résultat numérique concernant la distribution de Poisson obtenu dans le chapitre 3 motive l'adaptation du nouveau schéma au problème de l'accélération de l'algorithme EM.

Ainsi, dans le chapitre 4, j'effectue quelques rappels nécessaires pour la compréhension et l'implémentation de l'algorithme EM. En particulier, il y est justifié que l'algorithme EM peut être considéré comme la méthode classique de Picard [44, Eqn 8] appliquée à un problème de point fixe. Un exemple simple concernant une loi multinomiale illustre la théorie présentée.

Puis, dans le chapitre 5, une nouvelle classe de schémas itératifs est donnée afin d'accélérer la convergence de l'algorithme EM. En particulier, j'exploite la connection entre les transformations de suites [36] et les méthodes de point fixe basées sur la stratégie de

cyclage [75] qui consiste à définir des cycles : à l'intérieur de chaque cycle, à partir d'un vecteur initial, des itérations de Picard [44, Eqn 8] sont calculées, puis une transformation de suites est appliquée à ces itérations de Picard pour obtenir un vecteur dit extrapolé qui sera le vecteur initial au prochain cycle. Il en résulte une méthode itérative pour calculer la solution du problème de point fixe. L'exemple le plus connu qui illustre cette stratégie est la connection entre le processus Δ^2 d'Aitken et la méthode de Steffensen, dans le cas scalaire [36]. Dans un premier temps, j'effectue ainsi quelques rappels sur les transformations de suites scalaires et vectorielles et sur la stratégie de *cyclage*. Dans ce chapitre, je m'intéresse uniquement aux transformations de suites d'ordre un, c'est à dire à un pas de descente, pour deux raisons : (1) leur simplicité et (2) leur coût de calcul relativement faible. En particulier, j'étudie deux méthodes d'extrapolation d'ordre 1, la *Reduced Rank Extrapolation* [55] et la *minimal polynomial extrapolation* [39] notées respectivement RRE1 et MPE1. Puis, je définis une nouvelle technique appelée *squaring* afin d'obtenir une nouvelle classe de méthodes appelées *squarem*. Cette stratégie *squaring* consiste à appliquer deux fois à l'intérieur de chaque cycle la transformation de suites. Les méthodes squarem convergent linéairement, comme l'algorithme E.M., mais elles ont un taux de convergence plus rapide que l'algorithme EM et que leurs homologues avec une seule application par cycle d'une transformation de suites. Trois exemples numériques différents démontrent l'efficacité des méthodes squarem d'ordre 1, notées SqRRE1 et SqMPE1. Mais les schémas d'extrapolation sont souvent assujettis à des problèmes numériques de stagnation ou/et de division par zéro. Un nouveau schéma itératif hybride est présenté : il combine les schémas RRE1 et MPE1 de telle manière à éviter ces problèmes de stagnation et de division par zéro. Ce schéma hybride noté Sqhyb1 émerge des expériences numériques. Il conserve la rapidité de convergence de la SqMPE1 et la stabilité de la SqRRE1, en évitant les divisions par zéro et la stagnation. Les méthodes squarem peuvent être intégrées facilement dans l'algorithme EM. Elles requièrent seulement quelques itérations de l'algorithme EM pour leurs implémentations. Aucune autre quantité telle que le logarithme de la fonction de densité de probabilité des données complètes, ou son gradient ou hessienne est nécessaire. Elles sont donc une option attractive dans les problèmes avec un nombre important de paramètres et dans les problèmes où le modèle statistique est complexe puisque l'algorithme EM est lent. Ce chapitre 5 présente les premiers résultats obtenus en collaboration avec Ravi Varadhan. Pour une version statistique complète, voir le rapport technique [125]. A noter que ce rapport est soumis en version raccourcie [126].

Le chapitre 6 illustre l'efficacité des méthodes squarem sur un problème important : la reconstruction d'image en tomographie. La tomographie est une technique très utilisée pour représenter l'activité d'un organe, par exemple le cerveau, dans le but de détecter les régions dont l'activité est anormale (tumeurs). L'algorithme EM joue un rôle important pour ce problème, mais l'inconvénient est que sa convergence est lente. Les méthodes squarem sont alors testées sur ce problème et leur efficacité est une fois de plus montrée numériquement.

Le dernier chapitre traite d'un problème issu de la physique des plasmas. Le but est d'améliorer l'efficacité des codes Particles In Cell (PIC) à l'aide d'une reconstruction de la densité basée sur une méthode d'ondelettes. Les codes PIC sont très largement utilisés pour la simulation de particules chargées dans des accélérateurs pour la physique des plasmas.

Ils consistent, à chaque pas de temps, à résoudre l'équation de Vlasov par une méthode particulaire, puis à déposer la densité de charge sur un maillage de l'espace physique et enfin à résoudre les équations de Poisson sur ce maillage. Le dépôt de la densité de charge sur le maillage correspond à la reconstruction d'une densité de probabilité à partir d'un échantillon. Le coût du calcul dans une méthode PIC est en général largement dominé par le déplacement des particules car il en faut un très grand nombre. Ainsi, si à partir d'un échantillon moins important, la densité de charge est reconstruite avec une bonne précision, l'efficacité de la méthode sera considérablement améliorée. L'approche utilisée pour reconstruire cette densité est basée sur une technique d'ondelettes : la densité est premièrement estimée dans une base appropriée d'ondelettes comme une fonction de distribution à partir des données empiriques, puis débruitée par une technique de seuillage. Des résultats numériques concernant le problème de l'amortissement Landau sont présentés pour valider la méthode. Ce chapitre est le fruit du projet APICIB du Cemracs 2003, proposé par Eric Sonnendrücker (Université de Strasbourg) et Albert Cohen (Université Paris VI). Ce projet effectué en collaboration avec J.P. Chehab, A. Cohen, D. Jennequin, J.J. Nieto et J. Roche, a donné lieu à une publication [45].

Table des matières

Première partie

Résolution de Systèmes Linéaires

Chapitre 1

Analyse des méthodes d'Altman

1.1 Introduction

Considérons un système d'équations linéaires

$$Ax = b, \tag{1.1}$$

où A est supposée hermitienne définie positive d'ordre p, et (sans perte de généralité) le second membre est supposé de norme euclidienne $||b|| = 1$. Dans une série d'articles [3, 4, 5, 6, 7, 8], Altman considère le problème associé

$$Ay = (Ay, b)b \tag{1.2}$$

que nous réécrivons $PAy = 0$, où $P = I - bb^*$ est le projecteur orthogonal sur le complément orthogonal de b. Remarquons que les solutions de (1.2) sont l'ensemble des vecteurs $\alpha A^{-1}b$ avec $\alpha \in I\!R$, et ainsi une solution de (1.1) est donnée par $x = y/(Ay, b)$, où y est une solution non triviale de (1.2). Pour le système (1.2), Altman considère l'opérateur linéaire

$$r(y) := PAy = -P(b - Ay),$$

coïncidant au signe près au résidu projeté de (1.1).

Dans les articles mentionnés ci-dessus, Altman propose essentiellement deux méthodes itératives donnant les solutions approximatives non triviales de (1.2) (et ainsi de (1.1) après normalisation). Etant donné un vecteur y_0 tel que $(y_0, b) \neq 0$, la première méthode présentée dans [3, Eqns (5) et (6)], analysée et généralisée dans [8], minimise la norme du résidu projeté

$$y_{n+1} = y_n + \alpha_n r(y_n), \quad \alpha_n = \arg\min\{||r(y_n + \alpha r(y_n))|| : \alpha \in \mathbb{R}\}, \tag{1.3}$$

menant à une généralisation de la méthode itérative de Richardson. Pour la seconde méthode, (voir aussi [21, pp. 132-139]), Altman considère une forme quadratique [4, Eqn. (7)] étroitement liée à la forme quadratique

$$G(y) := \left(\left(A - \frac{bb^*}{(A^{-1}b, b)}\right) y, y\right) \tag{1.4}$$

(la forme quadratique F considérée par Altman, implique r et son inverse, défini sur le complément orthogonal de b, voir l'annexe A pour plus de détails). Il n'est pas difficile de vérifier en utilisant la formule (1.13) donnée ci-après que $G(y) \geq 0$ et $G(y) = 0$ si et seulement si y est une solution de (1.2). Alors une généralisation de la méthode de la plus profonde descente, introduite dans [4, Eqns (10) et (11)] et analysée dans [7], est définie par

$$y_{n+1} = y_n + \alpha_n r(y_n), \quad \alpha_n = \arg\min\{G(y_n + \alpha r(y_n)) : \alpha \in \mathbb{R}\}. \tag{1.5}$$

Précisons que les approches de relaxation pour ces méthodes ont été discutées dans [5, 6] et des projections complémentaires sur d'autres sous-espaces ont été examinées dans [7, 8].

Les deux méthodes (1.3) et (1.5) peuvent être considérées comme des versions avec redémarrage (après une itération) de méthodes généralisées de Krylov : ici nous minimisons sur des sous-espaces de Krylov

$$K_n(A, c) := \text{vect}(c, Ac, ..., A^{n-1}c)$$

avec le vecteur c défini par $c = b - Ay_0$, où les éléments de K_n, c'est à dire A et c, sont projetés par P. En effet, (1.3) est la version à un pas de la méthode

$$y_n = y_0 + \arg\min\{||r(y_0 + u)|| : u \in K_n(PA, P(b - Ay_0))\}, \tag{1.6}$$

notée dans ce qui suit AMinRes, et (1.5) est la version à un pas de la méthode

$$y_n = y_0 + \arg\min\{G(y_0 + u) : u \in K_n(PA, P(b - Ay_0))\}, \tag{1.7}$$

notée dans ce qui suit ACG. À notre connaissance, les méthodes d'Altman n'ont pas été considérées de cette façon auparavant.

Le but de ce chapitre est de montrer que les deux approches (1.6) et (1.7) (et de ce fait le travail d'Altman) sont mathématiquement équivalentes aux approches classiques de la méthode itérative de Richardson (MinRes) et du gradient conjugué (CG), appliquées à la matrice hermitienne définie semi-positive

$$\widetilde{A} := PAP. \tag{1.8}$$

En conséquence, nous trouvons des formules de récurrences pour calculer les vecteurs y_n de (1.6) et (1.7), et donnons des estimées d'erreur impliquant le conditionnement de la matrice $\widetilde{A}$, question déjà traitée d'une certaine manière par Altman [3, Eqn (16)], [4, Eqn (12)]. En particulier, nous établissons des propriétés d'entrelacement des valeurs propres correspondantes et concluons que les deux méthodes (1.6) et (1.7) se comportent toujours au moins aussi bien que les méthodes classiques correspondantes MinRes et CG, avec une amélioration de la vitesse de convergence seulement pour des second membres b particuliers. Le reste du chapitre est organisé comme suit : dans la section 1.2 nous prouvons l'équivalence entre l'approche d'Altman et l'approche classique et discutons du comportement de MinRes et CG appliqués à des systèmes hermitiens et consistants mais singuliers. Dans la section 1.3, nous étudions le comportement de convergence des méthodes d'Altman, à la fois linéaire et super-linéaire. La section 1.4 est consacrée à la programmation récursive

des itérés des méthodes d'Altman. Puisque l'analyse pour (1.6) et (1.7) est similaire, nous nous concentrons dans cette partie seulement à (1.7). Dans la section 1.5, nous présentons des expériences numériques confirmant les observations théoriques de la section 1.3. Finalement, dans les annexes, nous discutons dans la section A des deux formes quadratiques F et G et nous dédicaçons la section B à Mieczysław Altman en y ajoutant sa biographie obtenue avec l'aide de son fils Tom Altman.

1.2 L'équivalence

Etant donnée la matrice hermitienne définie semi-positive $\widetilde{A} = PAP$ (formule (1.8)), considérons le système d'équations linéaires

$$\widetilde{A}\widetilde{x} = \widetilde{b} := -r(y_0) = -PAy_0. \tag{1.9}$$

Dans ce qui suit, nous donnons le lien exact entre les algorithmes d'Altman, en particulier leurs versions multi-pas (1.6) et(1.7), et les algorithmes classiques MinRes et CG appliqués à (1.9).

Théorème 1. *La suite $(y_n - y_0)_{n\geq 0}$ avec y_n comme dans (1.6) est obtenue en appliquant l'algorithme MinRes au système (1.9) avec comme vecteur initial $x_0 = 0$.*

Similairement, la suite $(y_n - y_0)_{n\geq 0}$ avec y_n comme dans (1.7) est obtenue en appliquant l'algorithme CG au système (1.9) avec comme vecteur initial $x_0 = 0$.

Avant de donner la preuve du théorème 1, regardons précisément le comportement de MinRes et CG appliqués au système (1.9). Puisque $b^*\widetilde{b} = 0$, nous obtenons que

$$\widetilde{b} \in < b >^{\perp} = \mathrm{Ker}(\widetilde{A})^{\perp} = \mathrm{Im}(\widetilde{A}),$$

et de ce fait (1.9) est consistant, bien que sa matrice soit singulière. Précisons que la performance des méthodes de sous-espaces de Krylov appliquées à des systèmes singuliers mais inconsistants a été discutée par plusieurs auteurs, voir par exemple [60] et ses citations. En tout cas, pour des systèmes hermitiens singuliers et consistants, le comportement est facilement prévisible : il est facile de voir que

$$K_n(\widetilde{A}, \widetilde{b}) \subset \mathrm{Ker}(\widetilde{A})^{\perp}.$$

D'où tous les itérés de MinRes/CG avec comme vecteur initial 0 appliqués à (1.9) sont des éléments de $\mathrm{Ker}(\widetilde{A})^{\perp}$. Remarquons aussi que (1.9) a une solution unique dans $\mathrm{Ker}(\widetilde{A})^{\perp}$, notée $\widetilde{A}^{\dagger}\widetilde{b}$, où $\widetilde{A}^{\dagger}$ correspond au pseudo-inverse de $\widetilde{A}$. Donc, si un des deux algorithmes se termine (après au plus $\dim(K_n(\widetilde{A}, \widetilde{b})) \leq p - 1$ itérations), alors l'itéré correspondant coïncide avec $\widetilde{A}^{\dagger}\widetilde{b}$. De plus, le taux de convergence des deux algorithmes (soit exprimé en terme de norme du résidu ou de "norme" énergie) peut être borné de la même manière que pour les systèmes hermitiens non singuliers.

Afin de montrer le théorème énoncé précédemment, nous considérons les opérateurs de projection $P = I - bb^*$ et

$$Q = I - \frac{A^{-1}bb^*}{(A^{-1}b, b)}.$$

Les propriétés suivantes sont facilement vérifiées.

Lemme 1. *Nous avons*

$$PQ = Q, \tag{1.10}$$
$$QP = P, \tag{1.11}$$
$$PA = PAQ, \tag{1.12}$$
$$A - \frac{bb^*}{(A^{-1}b, b)} = AQ = Q^*AQ, \tag{1.13}$$
$$\widetilde{b} = -PAQy_0, \qquad \widetilde{A}^\dagger \widetilde{b} = -Qy_0, \tag{1.14}$$
$$K_n(PA, P(b - Ay_0)) = K_n(\widetilde{A}, \widetilde{b}). \tag{1.15}$$

Démonstration.
Par définition des matrices P et Q, et le fait que $||\, b\,|| = 1$, nous obtenons facilement les quatre propriétés suivantes

$$PQ = Q - bb^*Q = Q - bb^* + \frac{b(b^*A^{-1}b)b^*}{(A^{-1}b, b)} = Q,$$
$$QP = P - \frac{A^{-1}bb^*}{(A^{-1}b, b)}P = P - \frac{A^{-1}bb^*}{(A^{-1}b, b)} + \frac{A^{-1}b(b^*b)b^*}{(A^{-1}b, b)} = P,$$
$$PAQ = PA - \frac{Pbb^*}{(A^{-1}b, b)} = PA - \frac{(bb^* - b(b^*b)b^*)}{(A^{-1}b, b)} = PA,$$
$$Q^*AQ = AQ - \frac{bb^*Q}{(A^{-1}b, b)} = AQ - \left(\frac{bb^*}{(A^{-1}b, b)} - \frac{b(b^*A^{-1}b)b^*}{(A^{-1}b, b)}\right) = AQ.$$

Les propriétés (1.10), (1.11), (1.12) et (1.13) sont ainsi montrées. Par (1.12), nous avons $\widetilde{b} = -PAy_0 = -PAQy_0$. De ce fait, avec (1.10), nous déduisons que

$$\widetilde{A}^\dagger \widetilde{b} = -\widetilde{A}^\dagger PAQy_0 = -\widetilde{A}^\dagger PAPQy_0 = -\widetilde{A}^\dagger \widetilde{A}Qy_0 = -PQy_0 = -Qy_0.$$

Comme le vecteur $\widetilde{b}$ vérifie $P\widetilde{b} = \widetilde{b}$, nous avons $K_n(PA, P(b - Ay_0)) = K_n(PA, \widetilde{b}) = K_n(PAP, \widetilde{b}) = K_n(\widetilde{A}, \widetilde{b})$. Ainsi le lemme est prouvé. □

Démonstration. (*du Théorème 1*)
Appliquant (1.12), (1.14), et (1.10), nous avons, pour n'importe quel vecteur u,

$$||r(y_0 + Pu)|| = ||PAy_0 + PAPu|| = ||\widetilde{A}(Qy_0 + u)|| = ||\widetilde{b} - \widetilde{A}u||.$$

Prenant en compte (1.15) et le fait que $K_n(PA, P(b - Ay_0)) = PK_n(PA, P(b - Ay_0))$, nous obtenons avec y_n comme dans (1.6)

$$y_n - y_0 = \arg\min\{||r(y_0 + Pu)|| : u \in K_n(PA, P(b - Ay_0))\} = \arg\min\{||\widetilde{b} - \widetilde{A}u|| : u \in K_n(\widetilde{A}, \widetilde{b})\},$$

ce dernier étant la $n^{\text{ième}}$ itéré de MinRes avec comme vecteur initial 0 appliqué à (1.9).

Similairement, appliquant (1.13), (1.10), (1.11), et (1.14), nous avons pour n'importe quel vecteur u

$$\begin{aligned} G(y_0+Pu) &= \big((A-\frac{bb^*}{(A^{-1}b,b)})(y_0+Pu),(y_0+Pu)\big) = \big(AQ(y_0+Pu),Q(y_0+Pu)\big) \\ &= \big(\widetilde{A}Q(y_0+Pu),Q(y_0+Pu)\big) = \big(\widetilde{A}(Qy_0+u),(Qy_0+u)\big) \\ &= \big(\widetilde{A}(u-\widetilde{A}^\dagger\widetilde{b}),(u-\widetilde{A}^\dagger\widetilde{b})\big) =: \widetilde{G}(u), \end{aligned}$$

et de ce fait avec y_n comme dans (1.7)

$$y_n - y_0 = \arg\min\{||G(y_0+Pu)|| : u \in K_n(PA, P(b-Ay_0))\} = \arg\min\{\widetilde{G}(u) : u \in K_n(\widetilde{A},\widetilde{b})\},$$

ce dernier étant la $n^{\text{ième}}$ itéré de CG avec comme vecteur initial 0 appliqué à (1.9). Le théorème 1 est alors montré. □

1.3 Convergence

Avant de discuter du taux de convergence des méthodes d'Altman, discutons de la propriété de terminaison finie des itérations. Comme mentionné après le théorème 1, MinRes/CG avec comme vecteur initial 0 appliqué à (1.9) se termine (avec valeur du minimum étant égal à 0) si et seulement si les itérés correspondants $y_n - y_0$ coïncident avec $\widetilde{A}^\dagger\widetilde{b}$, c'est à dire (en utilisant (1.14)),

$$y_n = y_0 + \widetilde{A}^\dagger\widetilde{b} = y_0 - Qy_0 = \frac{(y_0,b)}{(A^{-1}b,b)}A^{-1}b.$$

Par le théorème 1, la propriété bien connue de terminaison finie des itérations pour MinRes/CG apporte ainsi une propriété similaire pour AMinRes/ACG.

Cependant, comme pour les autres méthodes de Krylov, nous sommes plus intéréssés par le taux de convergence que par la propriété de terminaison finie des itérations. Si nous traçons la norme euclidienne de l'erreur comme une fonction du nombre d'itérations en échelle semi-logarithmique, alors selon Nevanlinna [94] nous pouvons observer trois étapes différentes lesquelles sont plus ou moins prononcées selon les exemples : en général la courbe sera d'abord convexe, puis linéaire et finalement concave, correspondant ainsi aux étapes de convergence sublinéaire, linéaire, et superlinéaire. Dans cette description, les erreurs dues à l'arithmétique de l'ordinateur ne sont pas prises en compte engendrant en pratique des courbes plus compliquées. Dans le cas d'un système symétrique (1.1) avec des second membres généraux, le comportement de convergence linéaire peut être décrit par un certain nombre de conditions sur les matrices sous-jacentes, voir par exemple [106, Théorème 6.6, Eqn. (6.105), et Corollaire 6.1] pour MinRes et CG. Par contre, la convergence superlinéaire dépend essentiellement de la distribution des valeurs propres de la matrice, voir [12, 13, 14] pour plus d'informations à ce sujet.

Étudions ici le comportement de ACG et AMinRes. Utilisant le fait que, pour $u \in \mathrm{Ker}(\widetilde{A})^\perp$

$$||u-\widetilde{A}^\dagger\widetilde{b}||^2 \le ||\widetilde{A}^\dagger||\,\widetilde{G}(u), \quad ||u-\widetilde{A}^\dagger\widetilde{b}|| \le ||\widetilde{A}^\dagger||\,||\widetilde{b}-\widetilde{A}u||,$$

nous obtenons comme conséquence à nos résultats de la section précédente

$$\left\| y_n - \frac{(y_0,b)}{(A^{-1}b,b)} A^{-1}b \right\|^2 = ||y_n - y_0 - \widetilde{A}^\dagger \widetilde{b}||^2 \leq \begin{cases} ||\widetilde{A}^\dagger||^2 \, ||r(y_n)||^2 & \text{pour AMinRes,} \\ ||\widetilde{A}^\dagger|| \, G(y_n) & \text{pour ACG.} \end{cases}$$

La décroissance de $||r(y_n)||$ ou $G(y_n)$ découle de la décroissance connue de MinRes/CG [106, Théorème 6.6, Eqn. (6.105), et Corollaire 6.1]. D'où le résultat suivant

Corollaire 1 (Convergence linéaire)**.** *Avec* $\kappa(\widetilde{A}) = ||\widetilde{A}|| \, ||\widetilde{A}^\dagger||$ *et*

$$s := (\kappa(\widetilde{A}) - 1)/(\kappa(\widetilde{A}) + 1) < 1 \quad \text{et} \quad q := (\sqrt{\kappa(\widetilde{A})} - 1)/(\sqrt{\kappa(\widetilde{A})} + 1) < 1$$

nous avons pour les itérés de AMinRes

$$\frac{||r(y_n)||}{||r(y_0)||} \leq \frac{2}{s^n + s^{-n}} \leq 2s^n,$$

et pour les itérés de ACG

$$\sqrt{\frac{G(y_n)}{G(y_0)}} \leq \frac{2}{q^n + q^{-n}} \leq 2q^n.$$

En particulier, pour $n = 1$ nous obtenons

$$\frac{2}{q^1 + q^{-1}} = \frac{\kappa(\widetilde{A}) - 1}{\kappa(\widetilde{A}) + 1} \leq \frac{\kappa(A) - 1}{\kappa(A) + 1}.$$

A noter que la dernière inégalité provient du lemme 2 énoncé ci-dessous. En particulier, nous retrouvons les estimées d'erreur d'Altman [3, Eqn (16)] et [4, Eqn (12)] pour les méthodes (1.3) et (1.5).

Pour la convergence super-linéaire, les courbes de convergence des deux méthodes (1.6) et (1.7) (ou par équivalence MinRes/CG pour (1.9)) sont déterminées par la distribution des valeurs propres de $\widetilde{A}$, laquelle, selon le résultat qui suit, est essentiellement la même que pour A. Définissons $\lambda_1(B) \geq \lambda_2(B) \geq \ldots \geq \lambda_p(B)$ les valeurs propres d'une matrice hermitienne B d'ordre p.

Lemme 2 (Convergence Superlinéaire)**.** *Pour les matrices A de (1.1) et* $\widetilde{A}$ *de (1.8) nous avons*

$$\lambda_1(A) \geq \lambda_1(\widetilde{A}) \geq \lambda_2(A) \geq \lambda_2(\widetilde{A}) \geq \ldots \geq \lambda_p(A) > \lambda_p(\widetilde{A}) = 0.$$

Démonstration. Il est clair que b est un vecteur propre de $\widetilde{A}$ correspondant à la valeur propre 0. Notant H_b le complément orthogonal du sous-espace vectoriel engendré par b ($< b >$), définissons la base orthonormale de vecteurs propres $v_1, \ldots, v_{p-1} \in H_b$, $v_p \in < b >$ de la matrice $\widetilde{A}$ tel que $\widetilde{A}v_i = \lambda_i(\widetilde{A})v_i$. Il reste donc à montrer que, pour tout $i \in \{1, \ldots, p-1\}$

$$\lambda_i(A) \geq \lambda_i(\widetilde{A}) \geq \lambda_{i+1}(A).$$

Définissons $S_i =< v_1, \ldots, v_i >$. En utilisant le théorème du MinMax de Courant-Fischer, nous obtenons

$$\begin{aligned}
\lambda_i(A) &= \max_{dim(S)=i} \min_{\substack{y\in S \\ y\neq 0}} \frac{y^*Ay}{y^*y} \geq \min_{\substack{y\in S_i \\ y\neq 0}} \frac{y^*Ay}{y^*y} \\
&= \min_{\substack{y\in S_i \\ y\neq 0}} \frac{y^*P^*APy}{y^*y} \quad \text{car } S_i \subseteq H_b \\
&= \min_{\substack{y\in S_i \\ y\neq 0}} \frac{y^*\widetilde{A}y}{y^*y} = v_i^*\widetilde{A}v_i = \lambda_i(\widetilde{A}),
\end{aligned}$$

et de ce fait, $\lambda_i(A) \geq \lambda_i(\widetilde{A})$. Similairement, avec $V_{p-i} =< v_i, \ldots, v_{p-1} >$ de dimension $p-i$

$$\begin{aligned}
\lambda_{i+1}(A) &= \min_{dim(S)=p-i} \max_{\substack{y\in S \\ y\neq 0}} \frac{y^*Ay}{y^*y} \leq \max_{\substack{y\in V_i \\ y\neq 0}} \frac{y^*Ay}{y^*y} \\
&= \max_{\substack{y\in V_i \\ y\neq 0}} \frac{y^*P^*APy}{y^*y} \quad \text{car } V_i \subseteq H_b \\
&= \max_{\substack{y\in V_i \\ y\neq 0}} \frac{y^*\widetilde{A}y}{y^*y} = v_i^*\widetilde{A}v_i = \lambda_i(\widetilde{A}).
\end{aligned}$$

□

Ces deux derniers résultats nous permettent de comparer ACG et CG pour le système (1.1) : selon le lemme 2, le comportement de convergence dans la phase super-linéaire devrait être similaire, mais, selon le corollaire 1, il pourrait y avoir un comportement différent pour la phase linéaire si $\kappa(A) = \lambda_1(A)/\lambda_p(A)$ est plus grand que $\kappa(\widetilde{A}) = \lambda_1(\widetilde{A})/\lambda_{p-1}(\widetilde{A})$. Nous confirmerons ceci par des expériences numériques reportées dans la section 1.5.

1.4 L'algorithme ACG

Pour l'algorithme CG appliqué à (1.9), nous construisons récursivement des bases $\widetilde{A}$-conjuguées $p_0, ..., p_{n-1}$ de $K_n(\widetilde{A}, \widetilde{b})$, menant à des problèmes de minimisation unidimensionnelle et de ce fait à des relations de récurrence simples avec peu de termes (voir [22, p.41]). Remarquant que

$$\widetilde{r}_n := \widetilde{b} - \widetilde{A}(y_n - y_0) = \widetilde{b} - PA(y_n - y_0) = -r(y_n),$$

nous obtenons à partir du théorème 1 les relations suivantes de récurrence (nous gardons les notations de CG utilisées dans [106, Algorithme 6.17]) pour les itérés y_n de ACG

Initialisons $\widetilde{r}_0 = p_0 = -r(y_0)$,

Calculons pour $n = 0, 1, ...$ jusqu'à ce que $||\widetilde{r}_n||$ soit suffisamment petite

$$\alpha_n = \frac{(\widetilde{r}_n, \widetilde{r}_n)}{(\widetilde{A}p_n, p_n)} = \frac{(\widetilde{r}_n, \widetilde{r}_n)}{(r(p_n), p_n)} = \frac{(\widetilde{r}_n, \widetilde{r}_n)}{(Ap_n, p_n)},$$

$$y_{n+1} = y_n + \alpha_n p_n, \quad \widetilde{r}_{n+1} = \widetilde{r}_n - \alpha_n \widetilde{A}p_n = \widetilde{r}_n - \alpha_n r(p_n),$$

$$\widetilde{\beta}_n = \frac{(\widetilde{r}_{n+1}, \widetilde{r}_{n+1})}{(\widetilde{r}_n, \widetilde{r}_n)}, \quad p_{n+1} = \widetilde{r}_{n+1} + \widetilde{\beta}_n p_n.$$

Par conséquence, pour la suite $x_n = y_n/(Ay_n, b)$ approchant la solution de (1.1) nous obtenons le résidu

$$r_n := b - Ax_n = \frac{1}{(Ay_n, b)}((Ay_n, b)b - Ay_n) = -r(y_n)/(Ay_n, b) = \widetilde{r}_n/(Ay_n, b),$$

et en posant $z_n = p_n/(Ay_n, b)$ nous avons

$$x_{n+1} = \frac{(Ay_n, b)}{(Ay_{n+1}, b)}(x_n + \alpha_n z_n) = \frac{x_n + \alpha_n z_n}{1 + \alpha_n(Az_n, b)}$$

et

$$z_{n+1} = r_{n+1} + \widetilde{\beta}_n \frac{(Ay_n, b)}{(Ay_{n+1}, b)} z_n = r_{n+1} + \beta_n(1 + \alpha_n(Az_n, b))z_n, \quad \beta_n = \frac{(r_{n+1}, r_{n+1})}{(r_n, r_n)}.$$

Nous obtenons ainsi la formulation suivante pour ACG

Initialisons $r_0 = z_0 = b - Ax_0$

Calculons pour $n = 0, 1, ...$ jusqu'à ce que $||r_n||$ soit suffisamment petite

$$\alpha_n = \frac{(r_n, r_n)}{(Az_n, z_n)}, \quad \nu_n := 1 + \alpha_n(Az_n, b),$$

$$x_{n+1} = \frac{1}{\nu_n}(x_n + \alpha_n z_n),$$

$$r_{n+1} = \frac{1}{\nu_n}(r_n - \alpha_n r(z_n)) = \frac{1}{\nu_n}(r_n - \alpha_n PAz_n) = \frac{1}{\nu_n}(r_n - \alpha_n[Az_n - (Az_n, b)b]),$$

$$\beta_n = \frac{(r_{n+1}, r_{n+1})}{(r_n, r_n)}, \quad z_{n+1} = r_{n+1} + \nu_n \beta_n z_n.$$

Remarquons que cette méthode, comme MinRes et CG, nécessite un produit matrice-vecteur par itération.

Nous concluons cette section en remarquant que la relation de récurrence pour r_{n+1} dans l'algorithme ACG peut être réécrite comme

$$r_{n+1} = \frac{1}{\nu_n} r_n - \frac{\alpha_n}{\nu_n}(Az_n - \frac{\nu_n - 1}{\alpha_n} b) = \frac{1}{\nu_n}(r_n - \alpha_n Az_n) + (1 - \frac{1}{\nu_n})b.$$

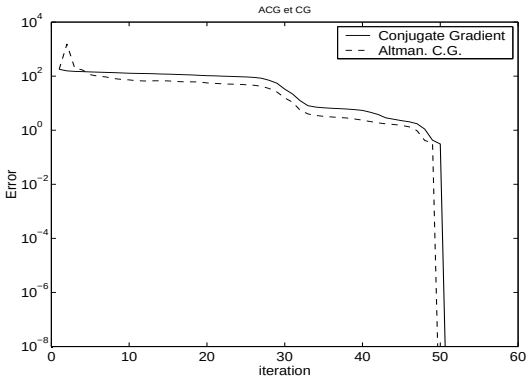


FIG. 1.1 – ACG (ligne pointillée) et CG (ligne pleine) pour $A = \text{tridiag}([-1, 2, -1])$ de dimension 50 résultant de la discrétisation du Laplacien en une dimension.

Pour $n = 0$, cela signifie que le résidu de la version à un pas (1.5) de ACG est obtenu en prenant une combinaison convexe du résidu de la méthode de la plus profonde descente (c'est à dire une itération de CG appliquée à (1.1)) et le résidu initial. Cette interprétation fut déjà énoncée par Altman dans [6].

1.5 Expériences Numériques

Donnons maintenant quelques résultats numériques pour illustrer la méthode ACG et la comparer à l'algorithme classique du gradient conjugué pour la résolution du système $Ax = b$ avec deux types différents de matrice A. Tous les calculs ont été réalisés en MATLAB et toutes les normes sont euclidiennes.

Nous présentons premièrement un exemple d'un système de type (1.1) avec une matrice A de dimension $p = 50$ résultant de la discrétisation du Laplacien en dimension une sur $[-1, 1]$, le second membre b et le vecteur initial y_0 sont choisis aléatoirement. Dans la Figure 1.1, nous remarquons que la convergence est essentiellement linéaire, jusqu'au moment où la convergence des itérations est brusque. En effet, la distribution des valeurs propres de la matrice A s'approche ici du cas le plus mauvais d'une distribution arcsin, c'est pourquoi la convergence super-linéaire se produit seulement pour des seconds membres b très spéciaux, voir [14]. Nous savons aussi, par le Lemme 2, que la quantité $\kappa(\widetilde{A})$ est comprise entre $\kappa(A) = \lambda_1(A)/\lambda_p(A)$ et $\lambda_2(A)/\lambda_{p-1}(A)$, ce qui pour notre exemple donne les valeurs numériques $1.05 \cdot 10^3$, et $0.26 \cdot 10^3$. Le conditionnement de $\widetilde{A}$ n'est donc pas très différent de celui de A, et par conséquent, le comportement de convergence de CG et ACG devrait être similaire. Ceci est clairement confirmé par le résultat présenté dans la Figure 1.1 : pour cet exemple nous obtenons que la méthode ACG, comparée à CG, permet de gagner seulement une itération.

Pour le second groupe d'exemples, nous considérons une matrice $A = QDQ^*$ de dimension $p = 1000$, où

$$Q = (I - 2w_3w_3^*)(I - 2w_2w_2^*)(I - 2w_1w_1^*),$$

numéro de l'exemple	ε	solution x	cond(A)	cond($\widetilde{A}$)	CG	ACG
I	10^{-6}	v_p	9.9e+08	9.9e+02	243	194
II		$v_p + 10^{-8}v_{p-1}$	9.9e+08	9.9e+02	237	188
III		$v_p + 10^{-3}v_{p-1}$	9.9e+08	4.9e+08	245	235
IV		aléatoire	9.9e+08	9.9e+08	274	274
V	10^{-3}	v_p	9.9e+05	9.9e+02	240	187
VI		aléatoire	9.9e+05	9.9e+05	238	235
VII	1	v_p	1e+03	5e+02	180	180

TAB. 1.1 – Nombre d'itérations pour CG et ACG.

w_1,w_2 et w_3 sont des vecteurs aléatoires unitaires, $D = diag(\lambda_1, \cdots, \lambda_p)$ est une matrice diagonale dont toutes les composantes sont $\lambda_i = \varepsilon + (i-1)$ pour $i = 1, \ldots, p$, et $\varepsilon > 0$ est un scalaire que nous préciserons. La solution x du système (1.1) (et par suite le second membre $b = Ax$) sera choisie soit aléatoirement soit en terme de vecteurs propres v_i correspondant aux valeurs propres $\lambda_i(A)$ de A.

Dans le tableau 1.1, pour sept choix différents de paramètres, nous reportons le choix de la variable ε et/ou la solution x, mais aussi les conditionnements des matrices A et $\widetilde{A}$, et le nombre d'itérations nécessaires pour obtenir une norme d'erreur $\| x_k - x \| \leq 10^{-8}$ par les méthodes CG et ACG.

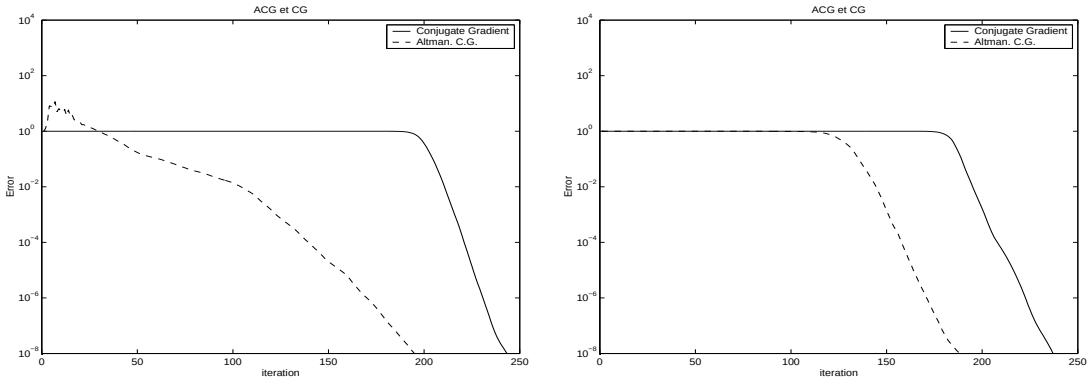


FIG. 1.2 – ACG (ligne pointillée) et CG (ligne pleine), exemples I (gauche) et II (droite).

Donnons quelques explications pour le comportement de convergence de la méthode CG pour ces sept exemples. Le comportement de convergence pour le cas de valeurs propres équidistantes ($\varepsilon = 1$) et des second membres généraux fut considéré dans [12, Corollaire 3.2] : pour cette valeur de ε, nous avons essentiellement une convergence super-linéaire et aucune phase de convergence linéaire, voir le graphe de l'Exemple VII dans la Figure 1.4. Pour ε approchant zéro, le conditionnement de la matrice A devient mauvais. Dans ce cas la convergence super-linéaire est retardée par des phases de convergence linéaire correspondant à une partie quasi-horizontale visible sur la gauche de la courbe de CG pour les exemples I, II, III (lesquels sont essentiellement les mêmes) et V et VI (encore

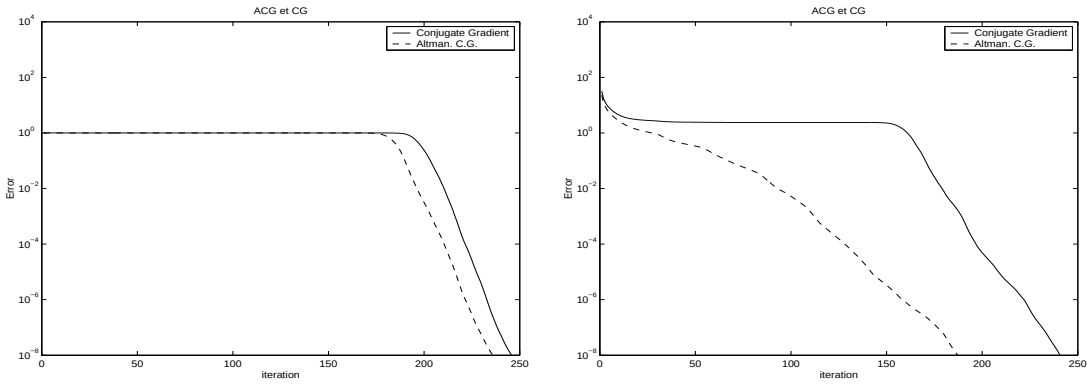


FIG. 1.3 – ACG (ligne pointillée) et CG (ligne pleine), exemples III (gauche) et V (droite).

essentiellement les mêmes).

Concernant le comportement de convergence de la méthode ACG, nous remarquons que, pour de petits ε, il y a un écart important entre $\kappa(A) = \lambda_1(A)/\lambda_p(A)$ et $\lambda_2(A)/\lambda_{p-1}(A)$, d'où une éventuelle amélioration de la méthode ACG sur CG dans la phase de convergence linéaire. En effet, pour les exemples I, II et V nous obtenons une importante amélioration pour le nombre d'itérations, voir le Tableau 1.1, la Figure 1.2 et la Figure 1.3. En tout cas, pour des seconds membres généraux comme dans les exemples III, IV, et VI, aucune amélioration n'est trouvée même dans la phase de convergence linéaire (comparer avec le Tableau 1.1, la Figure 1.3 et la Figure 1.4), confirmant nos remarques théoriques de la fin de la section 1.3. De plus, pour $\varepsilon = 1$ (donc quand A est bien conditionnée) comme dans l'exemple VII (voir la Figure 1.4), les deux méthodes CG et ACG se comportent de façon identique.

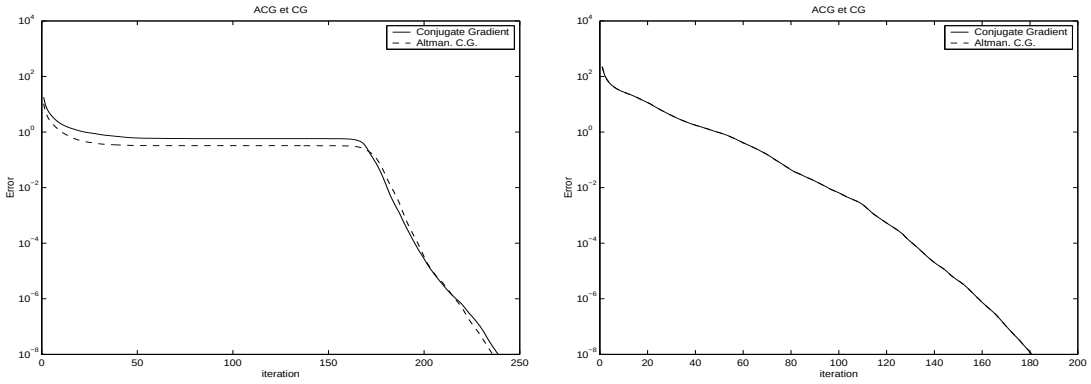


FIG. 1.4 – ACG (ligne pointillée) et CG (ligne pleine), exemples VI (gauche) et VII (droite).

1.6 Conclusion

Dans ce chapitre, nous avons montré que les deux approches différentes présentées par Altman pour résoudre des systèmes linéaires sont mathématiquement équivalentes aux approches classiques des méthodes de sous-espaces de Krylov. Cette équivalence nous a permis d'étudier le comportement de convergence des méthodes d'Altman, à la fois linéaire et super-linéaire. Des résultats tant sur le plan théorique (propriétés d'entrelacement de valeurs propres et taux de convergence) que sur le plan numérique, nous ont permis de conclure que les méthodes d'Altman se comportent toujours au moins aussi bien que les méthodes classiques correspondantes, avec une amélioration de la vitesse de convergence uniquement pour des seconds membres très particuliers.

1.7 Annexe A : Représentation de la forme quadratique

Notons H_b le complément orthogonal du sous-espace vectoriel engendré par le vecteur b. Dans [4], Altman utilise la forme quadratique

$$z \in H_b : \qquad F(z) := (r^{-1}(z), z). \tag{1.16}$$

Afin de justifier que cette formule a un sens, Altman montre que la restriction $r : H_b \mapsto H_b$ est hermitienne, auto-adjointe et définie positive, et de ce fait que l'inverse de r existe sur H_b. En fait, pour $u \in H_b$, nous avons $r(u) = PAu = PAPu = \widetilde{A}u$, et ainsi $r^{-1}(u) = \widetilde{A}^{\dagger}u$.

Cette observation nous permet de relier la forme quadratique F de (1.16) à la forme quadratique G de (1.4) : pour un vecteur y, nous avons selon (1.10) et (1.12)

$$\begin{aligned} F(r(y)) &= (\widetilde{A}^{\dagger}r(y), r(y)) = (\widetilde{A}^{\dagger}PAy, PAy) = ((PA)^*\widetilde{A}^{\dagger}PAy, y) \\ &= ((PAQ)^*\widetilde{A}^{\dagger}PAQy, y) = (Q^*\widetilde{A}\widetilde{A}^{\dagger}\widetilde{A}Qy, y) = (Q^*\widetilde{A}Qy, y) = (Q^*AQy, y), \end{aligned}$$

et par (1.13), nous concluons que $F(r(y)) = G(y)$.

1.8 Annexe B : Biographie de Mieczysław Altman

Mieczysław Altman (1916-1997) was born in Kutno, Poland, 50km north of £ódź. He studied mathematics at the Warsaw University from 1937 until the outbreak of World War II in 1939. In 1940 he enrolled at Lwów University and worked directly under the tutelage of Stefan Banach for two years. He was his last student. After the Nazi invasion of USSR in 1941, he was forced to flee again, eventually settling at the University of Sverdlovsk (now Yekaterinburg), 1700km east of Moscow, in 1941-1942, and then in Tashkent, USSR (now Uzbekistan). There, he first obtained his Master in Mathematics in 1944 and he finished his Ph.D. in Mathematics in 1948. After his return to Poland in 1949, he learned that his entire family, including seven brothers and sisters, perished in the £ódź Jewish ghetto during the war. He held a position at the Institute of Mathematics of the Polish Academy

of Sciences in Warsaw from 1949 to 1969, first as an Assistant Professor, 1949-1957, then as an Associate Professor, 1957-1958, and finally as a Full Professor and Director of the Numerical Analysis Department from 1958 to 1969.

In 1953, he married Wanda Kusal, M.D., and they had two children, Barbara (in 1956) and Tom (in 1958). For two years (1959-1960) Professor Altman took visiting positions at the California Institute of Technology in Pasadena and the Courant Institute in New York. He received the Poland's Banach Prize in Functional Analysis in 1958 and was Vice President of the Polish Mathematical Society in 1962-1963. Due to political pressures, in 1969 the Altman family left Poland and eventually settled in Baton Rouge, Louisiana, where M. Altman worked as a Professor of Mathematics, Louisiana State University, from 1970 until his retirement in 1987.

His 1977 book on *Contractors and Contractor Directions - Theory and Applications* (Marcel Dekker, New York, 1977) received international acclaim and recognition among mathematicians as the most encompassing theory for solving equations by analytical means. To honor the memory of his relatives, friends, and countrymen, he had dedicated his 1986 book on the *A Unified Theory of Nonlinear Operator and Evolution Equations with Applications - A New Approach to Nonlinear Partial Differential Equations* (Marcel Dekker, New York, 1986) to the victims of the Holocaust.

Even after his retirement, Professor Altman remained professionally active, publishing several journal papers and a book *A Theory of Optimization and Optimal Control for Nonlinear Evolution and Singular Equations with Applications to Nonlinear Partial Differential Equations* (World Scientific, Singapore, 1990).

Prof. Altman is the author of over 200 research papers in pure and applied mathematics, including functional and numerical analysis, mathematical programming, general optimization and optimal control theory, nonlinear differential and integral equations, and Banach algebras. A man of many talents, Professor Altman published his mathematical papers in a number of languages, including Russian, Polish, French, German, and English. He even coined the french word "contracteurs" (Contracteurs dans les algèbres de Banach, C.R. Acad. Sci. Paris, 274 (1972), 399-400).

He was a visiting professor at the Instituto per le Applicazioni del Calcolo, Consiglio Nazionale delle Ricerche, Rome, Italy, 1969-1970, and Newcastle University, Australia, in 1973.

Chapitre 2

Estimations du vecteur erreur pour les systèmes linéaires

2.1 Introduction

Considérons un système $n \times n$ d'équations linéaires

$$Ax = b$$

et notons $\tilde{x}$ une approximation du vecteur x obtenue soit par une méthode directe, soit par une méthode itérative. Auchmuty [10] puis Brezinski [27] ont obtenu des estimations de la norme de l'erreur $e = x - \tilde{x}$ qui sont indépendantes de la méthode utilisée et sont valables pour n'importe quelle norme vectorielle. Ces estimées peuvent être obtenues directement ou, comme montré dans [27], par une procédure d'extrapolation. Précisons qu'il existe d'autres approches pour obtenir des estimations de la norme de l'erreur, voir par exemple [67]. Dans ce travail, nous allons utiliser l'approche de [27] afin d'obtenir des estimations du vecteur erreur lui-même. Le but est d'utiliser cette nouvelle approche afin d'obtenir plusieurs méthodes connues de projection pour résoudre un système d'équations linéaires. Habituellement, ces méthodes sont obtenues par des approches différentes : projection sur différents plans, minimisation du résidu correspondant, ou quand la matrice est symétrique et définie positive, une minimisation d'une fonctionnelle quadratique. Nous allons donc montrer à partir d'une estimation du vecteur erreur grâce à la décomposition en valeurs singulières, en abrégé S.V.D., que ces méthodes (la méthode de la plus profonde descente, le gradient conjugué, les méthodes multiparamètres) habituellement présentées indépendamment peuvent découler de cette approche. Cela clarifie la connection entre les méthodes étudiées et simplifie leurs présentations. De cette idée découlent des procédures pour accélérer la convergence d'une méthode itérative.

2.2 S.V.D. et Estimations du vecteur erreur

Considérons la décomposition en valeurs singulières de la matrice

$$A = U\Sigma V^T$$

où $U = [u_1, \ldots, u_n]$ et $V = [v_1, \ldots, v_n]$ sont des matrices orthogonales et $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_n)$ avec $0 < \sigma_n \leq \ldots \leq \sigma_1$.

Etant donné un vecteur w, nous avons alors

$$Aw = \sum_{i=1}^{n} \sigma_i (v_i, w) u_i \tag{2.1}$$

$$A^T w = \sum_{i=1}^{n} \sigma_i (u_i, w) v_i \tag{2.2}$$

$$A^{-1} w = \sum_{i=1}^{n} {\sigma_i}^{-1} (u_i, w) v_i. \tag{2.3}$$

Si r est le vecteur résidu par rapport à l'itéré $\tilde{x}$, c'est à dire $r = b - A\tilde{x}$, alors $A^{-1}r = e$. Dans la suite, r sera toujours le résidu. Dans [27], Brezinski s'est intéressé à l'approximation de la norme de l'erreur e. En effet, par l'équation (2.3) avec $w = r$, nous avons

$$|| e ||^2 = (A^{-1}r, A^{-1}r) = \sum_{i=1}^{n} \sigma_i^{-2} (u_i, r)^2. \tag{2.4}$$

De ce fait, le calcul de la norme de l'erreur par cette équation nécessite la connaissance de tous les scalaires σ_i et des vecteurs u_i apparaissant dans la somme. L'idée donnée dans [27] est d'approcher cette norme en gardant seulement un terme dans la formule (2.4), c'est-à-dire que la norme de l'erreur e (au carré) est approchée par $\sigma^{-2} s^{-2}$. Le but est de déterminer ces deux paramètres au moyen de conditions d'interpolation à définir. Dans ce chapitre, nous utilisons la même approche mais pour estimer le vecteur erreur e lui-même. En effet, le vecteur erreur peut être calculé par (2.3) en prenant $w = r$. Mais, cette formule nécessite la connaissance de tous les scalaires σ_i et des vecteurs u_i et v_i. Une approximation du vecteur erreur e peut donc être obtenue en gardant seulement un terme dans la somme de la formule (2.3), ce qui signifie que le vecteur e est approché par

$$A^{-1}r = e \approx \sigma^{-1}(u, r) v.$$

Pour déterminer le scalaire σ et les deux vecteurs u et v, nous définissons les conditions suivantes d'interpolation

$$\begin{aligned} Aw &= \sigma(v, w) u \\ A^T w &= \sigma(u, w) v. \end{aligned} \tag{2.5}$$

Le système (2.5) a $2n$ équations à $2n + 1$ inconnues, et comme nous allons le voir, il a plusieurs solutions. Par (2.5),

$$v = \sigma^{-1} (u, w)^{-1} A^T w$$

et il s'en suit

$$e \approx \beta A^T w \quad \text{avec} \quad \beta = \sigma^{-2} \frac{(u, r)}{(u, w)}. \tag{2.6}$$

Reste à trouver une approximation du scalaire β.
Par les formules (2.1)-(2.2) et par orthogonalité de V, nous avons

$$\begin{aligned} AA^T w &= \sum_{i=1}^{n} \sigma_i (v_i, A^T w) u_i \\ &= \sum_{i=1}^{n} \sum_{j=1}^{n} \sigma_j \ \sigma_i (u_j, w)(v_i, v_j) u_i \\ &= \sum_{i=1}^{n} {\sigma_i}^2 (u_i, w) u_i . \end{aligned}$$

De même, mais par orthogonalité de U,

$$A^T A w = \sum_{i=1}^{n} {\sigma_i}^2 (v_i, w) v_i .$$

Par suite et par la SVD, nous avons donc

$$\begin{aligned} (r, w) &= \sum_{i=1}^{n} (u_i, r)(u_i, w) \\ &= \sum_{i=1}^{n} (v_i, r)(v_i, w) \\ (Aw, r) &= \sum_{i=1}^{n} \sigma_i (v_i, w)(u_i, r) \\ (A^T w, A^T w) &= \sum_{i=1}^{n} {\sigma_i}^2 (u_i, w)^2 \\ (A^T r, A^T w) &= \sum_{i=1}^{n} {\sigma_i}^2 (u_i, r)(u_i, w) \\ (A^T A w, A^T w) &= \sum_{i=1}^{n} {\sigma_i}^3 (v_i, w)(u_i, w) \\ (AA^T w, AA^T w) &= \sum_{i=1}^{n} {\sigma_i}^4 (u_i, w)^2 . \end{aligned}$$

Nous pouvons alors utiliser les approximations suivantes qui sont bien cohérentes avec les conditions d'interpolation (2.5) :

$$
\begin{aligned}
(r,w) &\approx (u,r)(u,w) && (2.7)\\
&\approx (v,r)(v,w)\\
(Aw,r) &\approx \sigma(v,w)(u,r) && (2.8)\\
(A^Tw,A^Tw) &\approx \sigma^2(u,w)^2 && (2.9)\\
(A^Tr,A^Tw) &\approx \sigma^2(u,r)(u,w) && (2.10)\\
(A^TAw,A^Tw) &\approx \sigma^3(v,w)(u,w) && (2.11)\\
(AA^Tw,AA^Tw) &\approx \sigma^4(u,w)^2. && (2.12)
\end{aligned}
$$

En fait, ces approximations vont permettre d'obtenir une approximation du scalaire β et donc d'ajouter aux conditions d'interpolation (2.5) une condition supplémentaire. Il s'en suit que

$$
\begin{aligned}
\beta &= \frac{(Aw,r)}{(A^Tw,AA^Tw)} && \text{par (2.8) et (2.11)}\\
\beta &= \frac{(A^Tr,A^Tw)}{(AA^Tw,AA^Tw)} && \text{par (2.10) et (2.12)}\\
\beta &= \frac{(r,w)}{(A^Tw,A^Tw)} && \text{par (2.7) et (2.9).}
\end{aligned}
$$

Ainsi, en utilisant ces trois expressions dans (2.6), cela mène à trois approximations du vecteur erreur e

$$
\begin{aligned}
e^{(1)} &= \frac{(Aw,r)}{(A^Tw,AA^Tw)}A^Tw && (2.13)\\
e^{(2)} &= \frac{(A^Tr,A^Tw)}{(AA^Tw,AA^Tw)}A^Tw && (2.14)\\
e^{(3)} &= \frac{(r,w)}{(A^Tw,A^Tw)}A^Tw. && (2.15)
\end{aligned}
$$

2.3 Méthodes itératives de projection

Montrons maintenant comment construire les méthodes itératives à partir des estimations précédentes de l'erreur. Dans ces expressions, nous remplaçons la solution approchée $\tilde{x}$ par un itéré x_k, r par $r_k = b - Ax_k$, w par w_k et $e^{(i)}$ devient $e_k^{(i)}$. Nous obtenons alors $x - x_k \approx e_k^{(i)}$, c'est à dire $x \approx x_k + e_k^{(i)}$, qui mène à trois méthodes itératives

$$
\begin{aligned}
x_{k+1} &= x_k + \frac{(Aw_k,r_k)}{(A^Tw_k,AA^Tw_k)}A^Tw_k \quad k=0,1,\ldots && (2.16)\\
x_{k+1} &= x_k + \frac{(A^Tr_k,A^Tw_k)}{(AA^Tw_k,AA^Tw_k)}A^Tw_k \quad k=0,1,\ldots && (2.17)\\
x_{k+1} &= x_k + \frac{(r_k,w_k)}{(A^Tw_k,A^Tw_k)}A^Tw_k \quad k=0,1,\ldots && (2.18)
\end{aligned}
$$

Premièrement, étudions la méthode itérative définie par (2.16) et supposons dans ce cas, la matrice A symétrique définie positive.
Par symétrie de A, $(Aw_k, r_k) = (A^T w_k, r_k)$ et en posant $z_k = A^T w_k$, cette méthode est de la forme

$$x_{k+1} = x_k + \lambda_k z_k \quad \text{avec} \quad \lambda_k = \frac{(z_k, r_k)}{(z_k, Az_k)}.$$

Comme expliqué dans [21, 22], ce choix de λ_k minimise la norme $(x_{k+1}-x, A(x_{k+1}-x))$ et nous avons

$$|| x_{k+1} - x ||_A{}^2 = || x_k - x ||_A{}^2 - \frac{(z_k, r_k)^2}{(z_k, Az_k)},$$

où $|| u ||_A{}^2 = (u, Au)$. Alors, $|| x_{k+1} - x ||_A \leq || x_k - x ||_A$. Les méthodes usuelles de projection pour les systèmes avec une matrice symétrique définie positive sont retrouvées (voir par exemple [66]). La méthode de la plus profonde descente proposée par Temple [121] est retrouvée pour le choix $z_k = r_k$; cette méthode converge toujours [46]. Le choix $(z_i, Az_k) = 0$ pour $i = 0, \ldots, n-1$, correspond à la méthode du gradient conjuguée [70] qui converge en n itérations au plus, où n est la dimension du système.

Supposons désormais la matrice A quelconque.
Étudions la méthode itérative définie par (2.17). En posant $z_k = A^T w_k$, la méthode est de la forme

$$x_{k+1} = x_k + \lambda_k z_k \quad \text{avec} \quad \lambda_k = \frac{(r_k, Az_k)}{(Az_k, Az_k)}.$$

Comme expliqué dans [21, 22], ce choix de λ_k minimise (r_{k+1}, r_{k+1}) et nous avons

$$|| r_{k+1} ||^2 = || r_k ||^2 - \frac{(r_k, Az_k)^2}{(Az_k, Az_k)}.$$

Si θ_k est l'angle entre les vecteurs r_k et Az_k, nous avons $|| r_{k+1} ||^2 = || r_k ||^2 \sin^2(\theta_k)$. D'où $|| r_{k+1} || \leq || r_k ||$, ce qui montre que la convergence est monotone. En définissant x'_k par $x'_k = x_k - z_k$, cette méthode de projection est donc équivalente à appliquer la *minimal smoothing procedure* [109, 108], notée MRS, à la suite (x'_k).

Si $z_k = r_k$, nous obtenons la méthode itérative de Richardson [102], aussi appelée *minimal residual iterations*. Si $z_k = C_k r_k$ où C_k est une approximation de A^{-1}, cette méthode peut être considérée comme une méthode de Richardson préconditionnée. Un tel choix est étudié en détail dans [30] où des résultats de convergence sont prouvés. Si nous prenons les vecteurs z_k définis par la relation de récurrence suivante

$$z_{k+1} = A^T r_{k+1} + \beta_{k+1} z_k \quad \text{avec} \quad \beta_{k+1} = \frac{(A^T r_{k+1}, A^T r_{k+1})}{(A^T r_k, A^T r_k)},$$

la méthode du CGNR introduite par Hestenes et Stiefel [71] est retrouvée.

Enfin, étudions la méthode itérative définie par (2.18). La méthode est de la forme

$$x_{k+1} = x_k + \lambda_k A^T w_k \quad \text{avec} \quad \lambda_k = \frac{(r_k, w_k)}{(A^T w_k, A^T w_k)}.$$

Comme expliqué dans [21, 22], ce choix de λ_k minimise $(x_{k+1} - x, x_{k+1} - x)$ et en définissant ϕ_k l'angle entre les vecteurs $A^T w_k$ et $x - x_k$, il est possible de montrer que

$$|| x_{k+1} - x ||^2 = || x_k - x ||^2 \sin^2(\phi_k).$$

Ainsi, $|| x_{k+1} - x || \leq || x_k - x ||$, ce qui montre que la convergence est monotone. Les méthodes usuelles de projection sont retrouvées (voir par exemple [66]). Pour le choix $w_k = r_k$, la méthode peut être considérée comme une extension de la méthode de la plus profonde descente à une matrice arbitraire. Pour cette raison, elle est appelée la *nonsymmetric steepest descent*.
Définissant x'_k par $x'_k = x_k - w_k$, la méthode est équivalente à appliquer la MRS à la suite (x'_k), mais avec ce choix de λ_k, elle minimise l'erreur $x_{k+1} - x$ à la place du résidu r_{k+1} comme proposé dans [108, 109]. Toutefois, il doit être remarqué qu'une telle méthode nécessite l'utilisation de la transposée de la matrice A.
En posant $z_k = A^T w_k$, la méthode est de la forme $x_{k+1} = x_k + \lambda_k z_k$ et si nous prenons les vecteurs z_k définis par la relation de récurrence suivante

$$z_{k+1} = A^T r_{k+1} + \beta_{k+1} z_k \quad \text{avec} \quad \beta_{k+1} = \frac{(r_{k+1}, r_{k+1})}{(r_k, r_k)},$$

la méthode du CGNE essentiellement due à Craig [49], est retrouvée. Elle peut être trouvée sous sa forme exacte dans [57].

2.4 Procédures d'accélération

Une procédure d'accélération consiste à transformer une suite donnée (x_k) en une nouvelle suite (y_k) avec une convergence plus rapide (sous certaines conditions à définir).

Considérons x_k les itérés obtenus par une méthode arbitraire. Comme $x_k + {e_k}^{(i)}$ est une approximation de x, nous définissons une nouvelle suite (y_k) par $y_k = x_k + {e_k}^{(i)}$. Ainsi, nous obtenons trois transformations de suites différentes définies par

$$y_k = x_k + \frac{(Aw_k, r_k)}{(A^T w_k, AA^T w_k)} A^T w_k \quad k = 0, 1, \ldots \tag{2.19}$$

$$y_k = x_k + \frac{(A^T r_k, A^T w_k)}{(AA^T w_k, AA^T w_k)} A^T w_k \quad k = 0, 1, \ldots \tag{2.20}$$

$$y_k = x_k + \frac{(r_k, w_k)}{(A^T w_k, A^T w_k)} A^T w_k \quad k = 0, 1, \ldots \tag{2.21}$$

Premièrement, étudions la procédure définie par (2.19) et supposons dans ce cas la matrice A symétrique définie positive.
Comme expliqué dans la section 2.3, en posant $z_k = A^T w_k$, cette procédure est de la forme

$$y_k = x_k + \lambda_k z_k \quad \text{avec} \quad \lambda_k = \frac{(z_k, r_k)}{(z_k, Az_k)},$$

et ce choix de λ_k minimise $(y_k - x, A(y_k - x))$ et nous avons

$$|| y_k - x ||_A{}^2 = || x_k - x ||_A{}^2 - \frac{(z_k, r_k)^2}{(z_k, Az_k)},$$

où $|| u ||_A{}^2 = (u, Au)$. De ce fait, $|| y_k - x ||_A \leq || x_k - x ||_A$. Quand $z_k = r_k$, cette procédure peut être considérée comme une *steepest descent acceleration*.

Supposons désormais la matrice A quelconque.
Etudions la procédure d'accélération définie par (2.20). Comme expliqué dans la section 2.3, en posant $z_k = A^T w_k$, la procédure est de la forme

$$y_k = x_k + \lambda_k z_k \quad \text{avec} \quad \lambda_k = \frac{(r_k, Az_k)}{(Az_k, Az_k)},$$

et ce choix de λ_k minimise (ρ_k, ρ_k) où $\rho_k = b - Ay_k$. Nous avons

$$|| \rho_k ||^2 = || r_k ||^2 - \frac{(r_k, Az_k)^2}{(Az_k, Az_k)} = || r_k ||^2 \sin^2(\theta_k),$$

où, θ_k est l'angle entre les vecteurs r_k et Az_k. Ainsi, $|| \rho_k || \leq || r_k ||$. En définissant x'_k par $x'_k = x_k - z_k$, cette procédure équivaut donc à appliquer la procédure hybride introduite dans [29] aux suites (x_k) et (x'_k). Des résultats d'accélération peuvent être trouvés dans [1].

Quand $z_k = r_k$, nous obtenons la procédure d'accélération de Richardson. Un choix optimal pour le vecteur z_k est donné dans [30] où des résultats d'accélération peuvent être trouvés.

Enfin, étudions la procédure définie par (2.21). La procédure est de la forme

$$y_k = x_k + \lambda_k A^T w_k \quad \text{avec} \quad \lambda_k = \frac{(r_k, w_k)}{(A^T w_k, A^T w_k)}.$$

Ce choix de λ_k minimise $(y_k - x, y_k - x)$ et en définissant ϕ_k l'angle entre les vecteurs $A^T w_k$ et $x - x_k$, il est possible de montrer que

$$|| y_k - x ||^2 = || x_k - x ||^2 \sin^2(\phi_k).$$

Alors, $|| y_k - x || \leq || x_k - x ||$. Pour le choix $w_k = r_k$, la procédure peut être considérée comme une extension de la *steepest descent acceleration* à une matrice arbitraire.

Définissant x'_k par $x'_k = x_k - w_k$, la procédure équivaut à appliquer la procédure hybride à la suite (x'_k) mais avec ce choix de λ_k, elle minimise l'erreur $y_k - x$ à la place du résidu ρ_k comme proposé dans [29]. Néanmoins, il doit être remarqué qu'une telle procédure nécessite la transposée de la matrice A.

2.5 Plus d'Estimées

L'idée d'extrapolation peut être utilisée pour obtenir d'autres estimées. En effet, il suffit de prendre plus de termes dans la somme des formules (2.1)-(2.3) et (2.7)-(2.12).

Intéressons-nous aux estimées à deux termes. Le vecteur erreur peut donc être approché par

$$A^{-1}r = e \approx {\sigma_1}^{-1}(u_1, r)v_1 + {\sigma_2}^{-1}(u_2, r)v_2. \tag{2.22}$$

Soient w_1, w_2, des vecteurs arbitraires distincts.
Pour simplifier les calculs, notons pour $i, j = 1$ ou 2,

$$\begin{aligned} \alpha_{i,j} &= (u_i, w_j) \\ \alpha_{i,0} &= (u_i, r), \end{aligned}$$

puis pour tout k entier positif,

$$\begin{aligned} {c_{i,j}}^{(2k)} &= ((AA^T)^k w_i, w_j) \\ {c_{i,0}}^{(2k)} &= ((AA^T)^k w_i, r) \\ {c_{i,j}}^{(2k+1)} &= (A(AA^T)^k w_i, w_j) \\ {c_{i,0}}^{(2k+1)} &= (A(AA^T)^k w_i, r), \end{aligned}$$

et

$$D = \alpha_{1,1}\ \alpha_{2,2} - \alpha_{1,2}\ \alpha_{2,1}.$$

En gardant deux termes dans (2.2), considérons les conditions d'interpolation suivantes

$$\begin{aligned} A^T w_1 &= \sigma_1(u_1, w_1)v_1 + \sigma_2(u_2, w_1)v_2 \\ &= \sigma_1\alpha_{1,1}v_1 + \sigma_2\alpha_{2,1}v_2 \\ A^T w_2 &= \sigma_1(u_1, w_2)v_1 + \sigma_2(u_2, w_2)v_2 \\ &= \sigma_1\alpha_{1,2}v_1 + \sigma_2\alpha_{2,2}v_2. \end{aligned} \tag{2.23}$$

Remarque 1.
Nous n'utilisons pas, contrairement à la section 2.2, la condition d'interpolation avec le produit Aw. Effectivement, en gardant deux termes dans ce produit, nous avons

$$Aw = \sigma_1(v_1, w)u_1 + \sigma_2(v_2, w)u_2,$$

et de ce fait, il apparaît deux nouvelles quantités (v_1, w) et (v_2, w) qui rendent les calculs plus difficiles.

Par (2.23), nous avons

$$\begin{aligned} v_1 &= \frac{1}{\sigma_1 D}[\alpha_{2,2}A^T w_1 - \alpha_{2,1}A^T w_2] \\ v_2 &= \frac{1}{\sigma_2 D}[\alpha_{1,1}A^T w_2 - \alpha_{1,2}A^T w_1]. \end{aligned}$$

Par suite dans (2.22),

$$\begin{aligned} e &\approx \frac{1}{D}[({\sigma_1}^{-2}\alpha_{1,0}\alpha_{2,2} - {\sigma_2}^{-2}\alpha_{2,0}\alpha_{1,2})A^T w_1 \\ &\quad +({\sigma_2}^{-2}\alpha_{2,0}\alpha_{1,1} - {\sigma_1}^{-2}\alpha_{1,0}\alpha_{2,1})A^T w_2] \\ &\approx \frac{1}{D}\left[A^T w_1, A^T w_2\right]\begin{bmatrix} {\sigma_1}^{-2}\alpha_{1,0}\alpha_{2,2} - {\sigma_2}^{-2}\alpha_{2,0}\alpha_{1,2} \\ {\sigma_2}^{-2}\alpha_{2,0}\alpha_{1,1} - {\sigma_1}^{-2}\alpha_{1,0}\alpha_{2,1} \end{bmatrix} \\ &\approx \frac{1}{D}\left[A^T w_1, A^T w_2\right]\Lambda \end{aligned} \tag{2.24}$$

avec $\Lambda = \begin{bmatrix} {\sigma_1}^{-2}\alpha_{1,0}\alpha_{2,2} - {\sigma_2}^{-2}\alpha_{2,0}\alpha_{1,2} \\ {\sigma_2}^{-2}\alpha_{2,0}\alpha_{1,1} - {\sigma_1}^{-2}\alpha_{1,0}\alpha_{2,1} \end{bmatrix}$.

Reste à trouver une approximation de D et des termes intervenant dans la matrice Λ. Précisons que malgré leur lourdeur, les calculs sont détaillés rigoureusement dans les trois cas traités.

Cas 1.
Considérons les approximations suivantes bien cohérentes avec les conditions d'interpolation (2.23)

$$\begin{array}{lcll} {c_{1,1}}^{(4)} = (AA^T w_1, AA^T w_1) & \approx & {\sigma_1}^4{\alpha_{1,1}}^2 + {\sigma_2}^4{\alpha_{2,1}}^2 & \text{(i)} \\ {c_{2,2}}^{(4)} = (AA^T w_2, AA^T w_2) & \approx & {\sigma_1}^4{\alpha_{1,2}}^2 + {\sigma_2}^4{\alpha_{2,2}}^2 & \text{(ii)} \\ {c_{1,2}}^{(4)} = (AA^T w_1, AA^T w_2) & \approx & {\sigma_1}^4\alpha_{1,1}\alpha_{1,2} + {\sigma_2}^4\alpha_{2,1}\alpha_{2,2} & \text{(iii)} \\ {c_{2,1}}^{(4)} = (AA^T w_2, AA^T w_1) & \approx & {\sigma_1}^4\alpha_{1,2}\alpha_{1,1} + {\sigma_2}^4\alpha_{2,2}\alpha_{2,1} & \text{(iv)} \\ {c_{1,0}}^{(2)} = (AA^T w_1, r) & \approx & {\sigma_1}^2\alpha_{1,1}\alpha_{1,0} + {\sigma_2}^2\alpha_{2,1}\alpha_{2,0} & \text{(v)} \\ {c_{2,0}}^{(2)} = (AA^T w_2, r) & \approx & {\sigma_1}^2\alpha_{1,2}\alpha_{1,0} + {\sigma_2}^2\alpha_{2,2}\alpha_{2,0} & \text{(vi)}. \end{array}$$

En fait, ces approximations vont permettre d'obtenir une approximation de D et de chaque composante de Λ, et donc d'ajouter aux conditions d'interpolation (2.23) les conditions supplémentaires nécessaires pour conclure.
D'après (i), (ii), (iii) et (iv), nous avons

$$D^2 = (\sigma_1\sigma_2)^{-4}[{c_{1,1}}^{(4)}{c_{2,2}}^{(4)} - {c_{2,1}}^{(4)}{c_{1,2}}^{(4)}].$$

De plus, d'après (v) et (vi),

$$\begin{aligned} \alpha_{1,0} &= \frac{1}{{\sigma_1}^2 D}[\alpha_{2,2}{c_{1,0}}^{(2)} - \alpha_{2,1}{c_{2,0}}^{(2)}] \\ \alpha_{2,0} &= \frac{1}{{\sigma_2}^2 D}[\alpha_{1,1}{c_{2,0}}^{(2)} - \alpha_{1,2}{c_{1,0}}^{(2)}]. \end{aligned}$$

Ainsi, en utilisant ces approximations dans (2.24),

$$
\begin{aligned}
e &\approx \left[A^T w_1, A^T w_2\right] \begin{bmatrix} \dfrac{{c_{1,0}}^{(2)}[{\sigma_2}^{-4}{\alpha_{1,2}}^2 + {\sigma_1}^{-4}{\alpha_{2,2}}^2] - {c_{2,0}}^{(2)}[{\sigma_2}^{-4}\alpha_{1,1}\alpha_{1,2} + {\sigma_1}^{-4}\alpha_{2,1}\alpha_{2,2}]}{D^2} \\ \dfrac{{c_{2,0}}^{(2)}[{\sigma_2}^{-4}{\alpha_{1,1}}^2 + {\sigma_1}^{-4}{\alpha_{2,1}}^2] - {c_{1,0}}^{(2)}[{\sigma_2}^{-4}\alpha_{1,2}\alpha_{1,1} + {\sigma_1}^{-4}\alpha_{2,2}\alpha_{2,1}]}{D^2} \end{bmatrix} \\
&\approx \left[A^T w_1, A^T w_2\right] \begin{bmatrix} \dfrac{{c_{1,0}}^{(2)}[{\sigma_1}^{4}{\alpha_{1,2}}^2 + {\sigma_2}^{4}{\alpha_{2,2}}^2] - {c_{2,0}}^{(2)}[{\sigma_1}^{4}\alpha_{1,1}\alpha_{1,2} + {\sigma_2}^{4}\alpha_{2,1}\alpha_{2,2}]}{{c_{1,1}}^{(4)}{c_{2,2}}^{(4)} - {c_{2,1}}^{(4)}{c_{1,2}}^{(4)}} \\ \dfrac{{c_{2,0}}^{(2)}[{\sigma_1}^{4}{\alpha_{1,1}}^2 + {\sigma_2}^{4}{\alpha_{2,1}}^2] - {c_{1,0}}^{(2)}[{\sigma_1}^{4}\alpha_{1,2}\alpha_{1,1} + {\sigma_2}^{4}\alpha_{2,2}\alpha_{2,1}]}{{c_{1,1}}^{(4)}{c_{2,2}}^{(4)} - {c_{2,1}}^{(4)}{c_{1,2}}^{(4)}} \end{bmatrix} \\
&\approx \left[A^T w_1, A^T w_2\right] \begin{bmatrix} \dfrac{{c_{1,0}}^{(2)}{c_{2,2}}^{(4)} - {c_{2,0}}^{(2)}{c_{1,2}}^{(4)}}{{c_{1,1}}^{(4)}{c_{2,2}}^{(4)} - {c_{2,1}}^{(4)}{c_{1,2}}^{(4)}} \\ \dfrac{{c_{2,0}}^{(2)}{c_{1,1}}^{(4)} - {c_{1,0}}^{(2)}{c_{2,1}}^{(4)}}{{c_{1,1}}^{(4)}{c_{2,2}}^{(4)} - {c_{2,1}}^{(4)}{c_{1,2}}^{(4)}} \end{bmatrix} \\
&\approx \frac{1}{{c_{1,1}}^{(4)}{c_{2,2}}^{(4)} - {c_{2,1}}^{(4)}{c_{1,2}}^{(4)}} \left[A^T w_1, A^T w_2\right] \begin{bmatrix} {c_{2,2}}^{(4)} & -{c_{1,2}}^{(4)} \\ -{c_{2,1}}^{(4)} & {c_{1,1}}^{(4)} \end{bmatrix} \begin{bmatrix} {c_{1,0}}^{(2)} \\ {c_{2,0}}^{(2)} \end{bmatrix} \\
&\approx \left[A^T w_1, A^T w_2\right] \begin{bmatrix} (AA^T w_1, AA^T w_1) & (AA^T w_1, AA^T w_2) \\ (AA^T w_2, AA^T w_1) & (AA^T w_2, AA^T w_2) \end{bmatrix}^{-1} \begin{bmatrix} (AA^T w_1, r) \\ (AA^T w_2, r) \end{bmatrix}.
\end{aligned}
$$

En posant $z_1 = A^T w_1$, $z_2 = A^T w_2$ et $Z = [z_1, z_2]$,

$$
\begin{aligned}
e &\approx Z \begin{bmatrix} (Az_1, Az_1) & (Az_1, Az_2) \\ (Az_2, Az_1) & (Az_2, Az_2) \end{bmatrix}^{-1} \begin{bmatrix} (Az_1, r) \\ (Az_2, r) \end{bmatrix} \\
&\approx Z[(AZ)^T AZ]^{-1}(AZ)^T r.
\end{aligned}
$$

Cas 2.

Nous procédons de la même manière que dans le cas 1, mais nous prenons les conditions d'interpolation suivantes

$$
\begin{array}{lcll}
{c_{1,1}}^{(2)} = (A^T w_1, A^T w_1) \approx {\sigma_1}^2{\alpha_{1,1}}^2 + {\sigma_2}^2{\alpha_{2,1}}^2 & \text{(i)} \\
{c_{2,2}}^{(2)} = (A^T w_2, A^T w_2) \approx {\sigma_1}^2{\alpha_{1,2}}^2 + {\sigma_2}^2{\alpha_{2,2}}^2 & \text{(ii)} \\
{c_{1,2}}^{(2)} = (A^T w_1, A^T w_2) \approx {\sigma_1}^2\alpha_{1,1}\alpha_{1,2} + {\sigma_2}^2\alpha_{2,1}\alpha_{2,2} & \text{(iii)} \\
{c_{2,1}}^{(2)} = (A^T w_2, A^T w_1) \approx {\sigma_1}^2\alpha_{1,2}\alpha_{1,1} + {\sigma_2}^2\alpha_{2,2}\alpha_{2,1} & \text{(iv)} \\
{c_{1,0}}^{(0)} = (w_1, r) \approx \alpha_{1,1}\alpha_{1,0} + \alpha_{2,1}\alpha_{2,0} & \text{(v)} \\
{c_{2,0}}^{(0)} = (w_2, r) \approx \alpha_{1,2}\alpha_{1,0} + \alpha_{2,2}\alpha_{2,0} & \text{(vi)}.
\end{array}
$$

D'après (i), (ii), (iii) et (iv), nous avons

$$D^2 = (\sigma_1\sigma_2)^{-2}[{c_{1,1}}^{(2)}{c_{2,2}}^{(2)} - {c_{2,1}}^{(2)}{c_{1,2}}^{(2)}].$$

De plus, d'après (v) et (vi),

$$\begin{aligned}\alpha_{1,0} &= \frac{1}{D}[\alpha_{2,2}{c_{1,0}}^{(0)} - \alpha_{2,1}{c_{2,0}}^{(0)}] \\ \alpha_{2,0} &= \frac{1}{D}[\alpha_{1,1}{c_{2,0}}^{(0)} - \alpha_{1,2}{c_{1,0}}^{(0)}].\end{aligned}$$

Ainsi, en utilisant ces approximations dans (2.24),

$$\begin{aligned}e &\approx \left[A^T w_1, A^T w_2\right] \begin{bmatrix} \dfrac{{c_{1,0}}^{(0)}[{\sigma_2}^{-2}{\alpha_{1,2}}^2 + {\sigma_1}^{-2}{\alpha_{2,2}}^2] - {c_{2,0}}^{(0)}[{\sigma_2}^{-2}\alpha_{1,1}\alpha_{1,2} + {\sigma_1}^{-2}\alpha_{2,1}\alpha_{2,2}]}{D^2} \\ \dfrac{{c_{2,0}}^{(0)}[{\sigma_2}^{-2}{\alpha_{1,1}}^2 + {\sigma_1}^{-2}{\alpha_{2,1}}^2] - {c_{1,0}}^{(0)}[{\sigma_2}^{-2}\alpha_{1,2}\alpha_{1,1} + {\sigma_1}^{-2}\alpha_{2,2}\alpha_{2,1}]}{D^2} \end{bmatrix} \\ &\approx \left[A^T w_1, A^T w_2\right] \begin{bmatrix} \dfrac{{c_{1,0}}^{(0)}[{\sigma_1}^{2}{\alpha_{1,2}}^2 + {\sigma_2}^{2}{\alpha_{2,2}}^2] - {c_{2,0}}^{(0)}[{\sigma_1}^{2}\alpha_{1,1}\alpha_{1,2} + {\sigma_2}^{2}\alpha_{2,1}\alpha_{2,2}]}{{c_{1,1}}^{(2)}{c_{2,2}}^{(2)} - {c_{2,1}}^{(2)}{c_{1,2}}^{(2)}} \\ \dfrac{{c_{2,0}}^{(0)}[{\sigma_1}^{2}{\alpha_{1,1}}^2 + {\sigma_2}^{2}{\alpha_{2,1}}^2] - {c_{1,0}}^{(0)}[{\sigma_1}^{2}\alpha_{1,2}\alpha_{1,1} + {\sigma_2}^{2}\alpha_{2,2}\alpha_{2,1}]}{{c_{1,1}}^{(2)}{c_{2,2}}^{(2)} - {c_{2,1}}^{(2)}{c_{1,2}}^{(2)}} \end{bmatrix} \\ &\approx \left[A^T w_1, A^T w_2\right] \begin{bmatrix} \dfrac{{c_{2,2}}^{(2)}{c_{1,0}}^{(0)} - {c_{1,2}}^{(2)}{c_{2,0}}^{(0)}}{{c_{1,1}}^{(2)}{c_{2,2}}^{(2)} - {c_{2,1}}^{(2)}{c_{1,2}}^{(2)}} \\ \dfrac{{c_{1,1}}^{(2)}{c_{2,0}}^{(0)} - {c_{2,1}}^{(2)}{c_{1,0}}^{(0)}}{{c_{1,1}}^{(2)}{c_{2,2}}^{(2)} - {c_{2,1}}^{(2)}{c_{1,2}}^{(2)}} \end{bmatrix} \\ &\approx \frac{1}{{c_{1,1}}^{(2)}{c_{2,2}}^{(2)} - {c_{2,1}}^{(2)}{c_{1,2}}^{(2)}} \left[A^T w_1, A^T w_2\right] \begin{bmatrix} {c_{2,2}}^{(2)} & -{c_{1,2}}^{(2)} \\ -{c_{2,1}}^{(2)} & {c_{1,1}}^{(2)} \end{bmatrix} \begin{bmatrix} {c_{1,0}}^{(0)} \\ {c_{2,0}}^{(0)} \end{bmatrix} \\ &\approx \left[A^T w_1, A^T w_2\right] \begin{bmatrix} (A^T w_1, A^T w_1) & (A^T w_1, A^T w_2) \\ (A^T w_2, A^T w_1) & (A^T w_2, A^T w_2) \end{bmatrix}^{-1} \begin{bmatrix} (w_1, r) \\ (w_2, r) \end{bmatrix}.\end{aligned}$$

En posant $W = [w_1, w_2]$,

$$\begin{aligned}e &\approx A^T W \begin{bmatrix} (A^T w_1, A^T w_1) & (A^T w_1, A^T w_2) \\ (A^T w_2, A^T w_1) & (A^T w_2, A^T w_2) \end{bmatrix}^{-1} \begin{bmatrix} (w_1, r) \\ (w_2, r) \end{bmatrix} \\ &\approx A^T W [(A^T W)^T A^T W]^{-1} W^T r.\end{aligned}$$

Cas 3.

Supposons dans ce cas la matrice symétrique, c'est à dire $A = A^T$. Dans la décomposition en valeurs singulières, les matrices U et V sont alors égales. Considérons les conditions d'interpolation suivantes

$$\begin{array}{lcccll} {c_{1,0}}^{(1)} & = & (Aw_1, r) & = & {\sigma_1}\alpha_{1,0}\alpha_{1,1} + {\sigma_2}\alpha_{2,0}\alpha_{2,1} & \text{(i)} \\ {c_{2,0}}^{(1)} & = & (Aw_2, r) & = & {\sigma_1}\alpha_{1,0}\alpha_{1,2} + {\sigma_2}\alpha_{2,0}\alpha_{2,2} & \text{(ii)} \\ {c_{1,1}}^{(3)} & = & (A^3 w_1, w_1) & = & {\sigma_1}^3{\alpha_{1,1}}^2 + {\sigma_2}^3{\alpha_{2,1}}^2 & \text{(iii)} \\ {c_{2,2}}^{(3)} & = & (A^3 w_2, w_2) & = & {\sigma_1}^3{\alpha_{1,2}}^2 + {\sigma_2}^3{\alpha_{2,2}}^2 & \text{(iv)} \\ {c_{1,2}}^{(3)} & = & (A^3 w_1, w_2) & = & {\sigma_1}^3\alpha_{1,1}\alpha_{1,2} + {\sigma_2}^3\alpha_{2,1}\alpha_{2,2} & \text{(v)} \\ {c_{2,1}}^{(3)} & = & (A^3 w_2, w_1) & = & {\sigma_1}^3\alpha_{1,2}\alpha_{1,1} + {\sigma_2}^3\alpha_{2,2}\alpha_{2,1} & \text{(vi)}. \end{array}$$

D'après (iii), (iv), (v) et (vi), nous avons

$$D^2 = (\sigma_1\sigma_2)^{-3}[{c_{1,1}}^{(3)}{c_{2,2}}^{(3)} - {c_{2,1}}^{(3)}{c_{1,2}}^{(3)}].$$

De plus, d'après (i) et (ii),

$$\begin{aligned}\alpha_{1,0} &= \frac{1}{\sigma_1 D}[\alpha_{2,2}{c_{1,0}}^{(1)} - \alpha_{2,1}{c_{2,0}}^{(1)}]\\ \alpha_{2,0} &= \frac{1}{\sigma_2 D}[\alpha_{1,1}{c_{2,0}}^{(1)} - \alpha_{1,2}{c_{1,0}}^{(1)}].\end{aligned}$$

Ainsi, en utilisant ces approximations dans (2.24),

$$\begin{aligned}
e &\approx [A^T w_1, A^T w_2]\begin{bmatrix} \dfrac{{c_{1,0}}^{(1)}[{\sigma_2}^{-3}{\alpha_{1,2}}^2 + {\sigma_1}^{-3}{\alpha_{2,2}}^2] - {c_{2,0}}^{(1)}[{\sigma_2}^{-3}\alpha_{1,1}\alpha_{1,2} + {\sigma_1}^{-3}\alpha_{2,1}\alpha_{2,2}]}{D^2} \\ \dfrac{{c_{2,0}}^{(1)}[{\sigma_2}^{-3}{\alpha_{1,1}}^2 + {\sigma_1}^{-3}{\alpha_{2,1}}^2] - {c_{1,0}}^{(1)}[{\sigma_2}^{-3}\alpha_{1,2}\alpha_{1,1} + {\sigma_1}^{-3}\alpha_{2,2}\alpha_{2,1}]}{D^2} \end{bmatrix}\\
&\approx [A^T w_1, A^T w_2]\begin{bmatrix} \dfrac{{c_{1,0}}^{(1)}[{\sigma_1}^3{\alpha_{1,2}}^2 + {\sigma_2}^3{\alpha_{2,2}}^2] - {c_{2,0}}^{(1)}[{\sigma_1}^3\alpha_{1,1}\alpha_{1,2} + {\sigma_2}^3\alpha_{2,1}\alpha_{2,2}]}{{c_{1,1}}^{(3)}{c_{2,2}}^{(3)} - {c_{2,1}}^{(3)}{c_{1,2}}^{(3)}} \\ \dfrac{{c_{2,0}}^{(1)}[{\sigma_1}^3{\alpha_{1,1}}^2 + {\sigma_2}^3{\alpha_{2,1}}^2] - {c_{1,0}}^{(1)}[{\sigma_1}^3\alpha_{1,2}\alpha_{1,1} + {\sigma_2}^3\alpha_{2,2}\alpha_{2,1}]}{{c_{1,1}}^{(3)}{c_{2,2}}^{(3)} - {c_{2,1}}^{(3)}{c_{1,2}}^{(3)}} \end{bmatrix}\\
&\approx [A^T w_1, A^T w_2]\begin{bmatrix} \dfrac{{c_{2,2}}^{(3)}{c_{1,0}}^{(1)} - {c_{1,2}}^{(3)}{c_{2,0}}^{(1)}}{{c_{1,1}}^{(3)}{c_{2,2}}^{(3)} - {c_{2,1}}^{(3)}{c_{1,2}}^{(3)}} \\ \dfrac{{c_{1,1}}^{(3)}{c_{2,0}}^{(1)} - {c_{2,1}}^{(3)}{c_{1,0}}^{(1)}}{{c_{1,1}}^{(3)}{c_{2,2}}^{(3)} - {c_{2,1}}^{(3)}{c_{1,2}}^{(3)}} \end{bmatrix}\\
&\approx \frac{1}{{c_{1,1}}^{(3)}{c_{2,2}}^{(3)} - {c_{2,1}}^{(3)}{c_{1,2}}^{(3)}}[A^T w_1, A^T w_2]\begin{bmatrix} {c_{2,2}}^{(3)} & -{c_{1,2}}^{(3)} \\ -{c_{2,1}}^{(3)} & {c_{1,1}}^{(3)} \end{bmatrix}\begin{bmatrix} {c_{1,0}}^{(1)} \\ {c_{2,0}}^{(1)} \end{bmatrix}\\
&\approx [Aw_1, Aw_2]\begin{bmatrix} (AAw_1, Aw_1) & (AAw_1, Aw_2) \\ (AAw_2, Aw_1) & (AAw_2, Aw_2) \end{bmatrix}^{-1}\begin{bmatrix} (Aw_1, r) \\ (Aw_2, r) \end{bmatrix}.
\end{aligned}$$

En posant $z_1 = Aw_1$, $z_2 = Aw_2$ et $Z = [z_1, z_2]$,

$$\begin{aligned}
e &\approx Z\begin{bmatrix} (Az_1, z_1) & (Az_1, z_2) \\ (Az_2, z_1) & (Az_2, z_2) \end{bmatrix}^{-1}\begin{bmatrix} (z_1, r) \\ (z_2, r) \end{bmatrix}\\
&\approx Z[Z^T AZ]^{-1}Z^T r.
\end{aligned}$$

Ainsi, en regroupant ces trois cas, cela mène à trois approximations du vecteur erreur e

$$e^{(4)} = Z[(AZ)^T AZ]^{-1}(AZ)^T r \tag{2.25}$$

$$e^{(5)} = A^T Z[(A^T Z)^T A^T Z]^{-1}Z^T r \tag{2.26}$$

$$e^{(6)} = Z[Z^T AZ]^{-1}Z^T r, \tag{2.27}$$

où $Z = [z_1, z_2]$ avec z_1 et z_2 deux vecteurs arbitraires distincts.

2.6 Multiparamètres

Les méthodes itératives de projection (respectivement les procédures d'accélération) retrouvées dans la section 2.3 (respectivement dans la section 2.4) sont de la forme

$$x_{k+1} = x_k + \lambda_k z_k, \quad k = 0, 1, \ldots,$$
$$(\text{resp.} \quad y_k = x_k + \lambda_k z_k, \quad k = 0, 1, \ldots,)$$

où z_k est un vecteur et λ_k est un paramètre. Dans quelques cas, les composantes de x_k peuvent avoir des comportements différents de convergence et donc utiliser le même paramètre λ_k pour toutes les composantes ne mènera pas à une convergence (resp. accélération) suffisante. C'est pour cette raison que les méthodes itératives (resp. procédures d'accélération) de la forme

$$x_{k+1} = x_k + Z_k \Lambda_k, \quad k = 0, 1, \ldots, \tag{2.28}$$
$$(\text{resp.} \quad y_k = x_k + Z_k \Lambda_k, \quad k = 0, 1, \ldots,) \tag{2.29}$$

où Z_k est une matrice de dimension $n \times n_k$ et $\Lambda_k \in I\!R^{n_k}$, ont été introduites. En particulier, dans la série d'articles [30, 31, 32, 33], Brezinski a étudié des choix du vecteur Λ_k qui ont mené à des méthodes itératives (resp. procédures d'accélération) de projection, appelées *les méthodes multiparamètres.* D'autres choix sont possibles : par exemple, le choix de Λ_k, introduit dans [25] et étudié dans [33], mène aux transformations appelées les *vector θ-type transformations.*

Dans les méthodes multiparamètres, n_k est fixé quand k tend vers l'infini. Si nous prenons $n_k = 1$, les méthodes itératives (resp. procédures d'accélération) de projection décrites dans la section 2.3 et 2.4 sont retrouvées. Montrons que les estimations du vecteur erreur e à n_k termes, permettent de retrouver les méthodes multiparamètres. Pour $n_k = 1$, cela est vérifié grâce aux sections 2.3 et 2.4. Vérifions qu'il en est de même pour $n_k = 2$.

Similairement à la section 2.3, dans les estimations d'erreur à deux termes, nous remplaçons la solution approchée $\tilde{x}$ par un itéré x_k, r par $r_k = b - Ax_k$, w par w_k et $e^{(i)}$ devient $e_k^{(i)}$. De ce fait, nous obtenons $x - x_k \approx e_k^{(i)}$, c'est à dire $x \approx x_k + e_k^{(i)}$, qui mène aux trois méthodes itératives suivantes

$$x_{k+1} = x_k + Z_k[(AZ_k)^T AZ_k]^{-1}(AZ_k)^T r_k \tag{2.30}$$
$$x_{k+1} = x_k + A^T Z_k[(A^T Z_k)^T A^T Z_k]^{-1} {Z_k}^T r_k \tag{2.31}$$
$$x_{k+1} = x_k + Z_k[Z_k^T AZ_k]^{-1} Z_k^T r_k. \tag{2.32}$$

Et similairement à la section 2.4, considérons x_k les itérés obtenus par une méthode arbitraire. Comme $x_k + {e_k}^{(i)}$ est une approximation de x, nous définissons une nouvelle suite (y_k) par $y_k = x_k + {e_k}^{(i)}$. Ainsi, nous obtenons trois transformations de suites différentes définies par

$$y_k = x_k + Z_k[(AZ_k)^T AZ_k]^{-1}(AZ_k)^T r_k \tag{2.33}$$
$$y_k = x_k + A^T Z_k[(A^T Z_k)^T A^T Z_k]^{-1} {Z_k}^T r_k \tag{2.34}$$
$$y_k = x_k + Z_k[Z_k^T AZ_k]^{-1} Z_k^T r_k. \tag{2.35}$$

Nous remarquons que les méthodes définies par (2.30), (2.31) et (2.32) (resp. les procédures définies par (2.33), (2.34) et (2.35)) sont de la même forme que (2.28) (resp. (2.29)).

Tout d'abord, étudions la méthode itérative définie par (2.30) (resp. la procédure d'accélération définie par (2.33)). Nous avons

$$\Lambda_k = \left[(AZ_k)^T AZ_k\right]^{-1} (AZ_k)^T r_k.$$

Ce choix de Λ_k, solution au sens des moindres carrées du système $r_k - AZ_k\Lambda_k = 0$, minimise $||\, r_{k+1}\,||$ (resp. $||\, \rho_k\,||$, où $\rho_k = b - Ay_k$). Pour le choix $Z_k = C_k r_k$ où C_k est une approximation de A^{-1}, nous retrouvons une méthode multiparamètre introduite dans [30] considérée comme une généralisation de la méthode itérative PR2 (resp. de la *PR2 acceleration* [32]).

Ensuite, pour la méthode itérative définie par (2.31) (resp. la procédure d'accélération définie par (2.34)), nous avons

$$\Lambda_k = \left[\left(A^T Z_k\right)^T A^T Z_k\right]^{-1} {Z_k}^T r_k.$$

Ce choix de Λ_k, solution au sens des moindres carrées du système $e_k - A^T Z_k \Lambda_k = 0$, minimise $||\, x - x_{k+1}\,||$ (resp. $||\, x - y_k\,||$). Dans ce cas, le prix à payer pour avoir une meilleure méthode (puisque si la matrice est mal conditionnée, la norme du résidu peut être petite tandis que la norme de l'erreur reste large), est l'utilisation de la transposée.

Enfin, étudions la méthode itérative définie par (2.32) (resp. la procédure d'accélération définie par (2.35)). Supposons dans ce cas la matrice symétrique définie positive. De ce fait, il existe une matrice H telle que $G = H^T H$. Nous avons

$$\Lambda_k = \left[Z_k^T AZ_k\right]^{-1} Z_k^T r_k,$$

solution au sens des moindres carrées de $He_k - HZ_k\Lambda_k = 0$. Ce choix minimise $||\, x - x_k\,||_A$ (resp. $||\, x - y_k\,||_A$) où $||\, u\,||_A = (u, Au)^{1/2}$.

Pour ces trois méthodes (resp. procédures) multiparamètres, plusieurs choix pour la matrice Z_k, appelée la matrice de descente, sont possibles.

1. Nous pouvons choisir la matrice Z_k telle qu'il existe $\alpha_k \in I\!R^{n_k}$ satisfaisant

$$C_k r_k = Z_k \alpha_k,$$

où C_k est une approximation de A^{-1}, c'est-à-dire un préconditionnement. Ce choix est étudié dans [33], où une stratégie de partionnement est appliquée.

2. De même, nous pouvons choisir Z_k telle qu'il existe $\alpha_k \in I\!R^{n_k}$ satisfaisant

$$C_k b - x_k = Z_k \alpha_k.$$

Un tel choix est considéré dans [21]. Il est à étudier plus en détails.

3. Les matrices Z_k peuvent être choisies conjuguées, c'est-à-dire $\forall\ i \neq k, {Z_i}^T A Z_k = 0$. Ce choix mène, pour les méthodes itératives correspondantes, à une généralisation multi-paramètre donnée dans [31] de l'algorithme du gradient conjugué et biconjugué.

Enfin, il doit être remarqué que si nous considérons plus de termes dans les estimations du vecteur erreur, en particulier n_k, nous retrouvons toutes les méthodes (resp. procédures d'accélération) multiparamètres.

2.7 Conclusion

Dans ce chapitre, nous avons proposé une nouvelle approche basée sur l'idée d'extrapolation afin d'obtenir une estimation du vecteur erreur dans le cas d'un système linéaire. Elle permet de simplifier la présentation de nombreuses méthodes habituellement présentées différemment et surtout indépendamment. De plus, il est possible d'obtenir de nouveaux schémas itératifs pour la résolution d'un système linéaire. En effet, il suffit de prendre des conditions d'interpolation différentes de celles présentées dans ce chapitre. Ces nouveaux schémas restent à étudier tant sur le plan pratique que sur le plan théorique.

Deuxième partie

Résolution de Problèmes de Point Fixe

Chapitre 3

Quelques Schémas pour des problèmes de point fixe

3.1 Introduction

Considérons le problème non linéaire suivant

$$\begin{cases} \text{Trouver } X \in I\!R^p \text{ tel que } X = T(X), \\ \text{où } T : I\!R^p \to I\!R^p \text{ est une fonction non linéaire.} \end{cases} \tag{3.1}$$

Une des méthodes les plus simples pour résoudre (3.1) est la méthode de Picard

$$X_{n+1} = T(X_n), \qquad X_0 \text{ donné.}$$

La simplicité de programmation recommande cette méthode, mais une critique qui a été faite sur la méthode de Picard est que sa convergence peut être lente. De plus, il est bien connu que nous ne pouvons pas utiliser cette méthode pour trouver les solutions instables des problèmes de réaction-diffusion avec bifurcations [88]. Pour ces raisons, quelques schémas basés sur des relaxations de plusieurs itérations successives de Picard ont été introduits : Lemaréchal [85] pour une méthode à deux pas, Marder et Weitzner [88] pour une méthode à trois pas, notée respectivement LM et MW dans la suite. Brezinski et Chehab ont introduit dans [28, 32] les méthodes Δ^k. Pour k=1, la méthode de Lemaréchal est retrouvée, tandis que, pour k=2, nous obtenons une méthode due à Marder et Weitzner avec le choix de paramètre proposé par Chehab [43]. Elles sont définies par

$$\begin{cases} \text{Soit } X_0 \text{ donné.} \\ \text{Pour } n = 0, 1, \ldots \\ X_{n+1} = X_n - (-1)^k \lambda_n \Delta^k X_n. \end{cases} \tag{3.2}$$

avec

$$\Delta^k X_n = \sum_{i=0}^{k} (-1)^{i-k} C_i^k T^i(X_n) \tag{3.3}$$

où $T^0(X_n) = X_n$, T^j désigne T répétée j fois et C_i^k les coefficients binomiaux. Le paramètre λ_n est choisi afin de minimiser la norme d'une approximation du résidu $R_n = T(X_n) - X_n$, à l'étape $n^{\text{ième}}$ du schéma Δ^k et il est donné par

$$\lambda_n = (-1)^k \frac{(\Delta X_n, \Delta^{k+1} X_n)}{(\Delta^{k+1} X_n, \Delta^{k+1} X_n)}. \tag{3.4}$$

Dans ce chapitre, nous proposons une nouvelle méthode qui peut être considérée comme une modification des méthodes Δ^k. A chaque itération du nouveau schéma, nous évaluons le pas de descente λ_n de la méthode Δ^k une fois et l'utilisons deux fois. Ce chapitre est organisé comme suit. Dans la section 3.2, nous rappelons une méthode intéressante présentée dans [100] pour résoudre des systèmes linéaires. En particulier, nous effectuons une remarque sur l'équation d'erreur. Cette remarque nous permet de définir le nouveau schéma appelé la méthode Δ^k-ajustée. Sa convergence est étudiée dans la section 3.3. Finalement, dans la dernière section, nous présentons des résultats numériques impliquant un problème non linéaire elliptique, lesquels illustrent l'intérêt du nouveau schéma. Un résultat numérique supplémentaire impliquant une distribution de Poisson est donné. Il suggére que le nouveau schéma peut être étendu avec succès à un problème important en statistique qui est l'accélération de l'algorithme E.M. (Espérance-Maximisation).

3.2 Les méthodes Δ^k ajustées

3.2.1 La méthode de Cauchy-Barzilai-Borwein

Considérons le problème linéaire suivant

$$\text{Trouver } x \in I\!R^p \text{ tel que } Qx = b \tag{3.5}$$

où Q est une matrice symétrique définie positive de dimension p. Une des méthodes les plus simples pour résoudre (3.5) est la méthode de Cauchy [40] définie par

$$x_{n+1} = x_n - \frac{(r_n, r_n)}{(Qr_n, r_n)}\, r_n \tag{3.6}$$

où $r_n = Qx_n - b$. Mais cette méthode converge lentement dans de nombreux cas. Pour éviter cet inconvénient, Barzilai et Borwein [11] présentent une nouvelle méthode définie par

$$x_{n+1} = x_n - t_{n-1} r_n$$

où $t_{n-1} = (r_{n-1}, r_{n-1})/(Qr_{n-1}, r_{n-1})$ est le choix optimal (Choix de Cauchy) à l'itération précédente et $t_{-1} = 1$. La convergence est étudiée dans [99], pour plus de détails voir aussi [98]. Récemment, Raydan et al [100] présentent un nouveau schéma appelé méthode de Cauchy-Barzilai-Borwein et notée CBB, qui améliore les méthodes de Barzilai-Borwein et de Cauchy. Précisons que cette méthode appartient à une famille générale de méthodes à gradient avec retard introduites dans [65]. Elle est décrite par l'algorithme suivant

Prendre x_0 dans $\mathbb{R}^p$, et à chaque itération n, calculer

$$t_n = (r_n, r_n)/(r_n, Qr_n)$$
$$x_{n+1} = x_n - 2t_n r_n + t_n^2 Q r_n. \tag{3.7}$$

Des résultats numériques encourageants furent obtenus dans [100] et montrent la performance de la méthode CBB. Précisons qu'à notre connaissance, aucun résultat théorique ne justifie la performance de cette méthode par rapport à la méthode de Cauchy ou/et la méthode de Barzilai et Borwein. Mais suite à ces résultats numériques, il semble naturel d'essayer d'adapter la même idée au cas non linéaire. Pour cela, nous remarquons la relation suivante entre l'équation d'erreur de la méthode de Cauchy (Eq. (3.6)) et celle de la méthode de Cauchy-Barzilai-Borwein (Eq. (3.7))

$$\begin{aligned} e_{n+1} &= (I - t_n Q)^2 e_n \text{ pour CBB} \quad \text{et} \\ e_{n+1} &= (I - t_n Q) e_n \text{ pour Cauchy,} \end{aligned} \tag{3.8}$$

où $e_n = x_n - x$. Précisons que ces deux équations découlent du fait que $r_n = Qe_n$ et $Qr_n = Q^2 e_n$. De ce fait, nous considérons la méthode CBB comme une méthode de Cauchy au carré. Grâce à cette remarque, nous proposons dans la section suivante un nouveau schéma pour le cas non linéaire.

3.2.2 Les nouveaux schémas

Nous supposons que nous sommes en dimension finie et que T est différentiable en chacun de ses points fixes. Nous notons ψ la matrice jacobienne de T au point fixe X. En posant $\varepsilon_n = X_n - X$ l'erreur à la $n^{\text{ième}}$ étape, nous avons

$$T^j(X_n) = X + \psi^j \varepsilon_n + o(\varepsilon_n).$$

Mais

$$\Delta^k X_n = \sum_{i=0}^{k} (-1)^{i-k} C_i^k T^i(X_n)$$

où $T^0(X_n) = X_n$ et les C_i^k désignent les coefficients binomiaux. Alors,

$$\begin{aligned} \Delta^k X_n &= \sum_{i=0}^{k} (-1)^{i-k} C_i^k X + \sum_{i=0}^{k} (-1)^{i-k} C_i^k \psi^i \varepsilon_n + o(\varepsilon_n) \\ &= \sum_{i=0}^{k} (-1)^{i-k} C_i^k \psi^i \varepsilon_n + o(\varepsilon_n) \\ &= (\psi - I)^k \varepsilon_n + o(\varepsilon_n), \end{aligned}$$

puisque $\sum_{i=0}^{k} (-1)^{i-k} C_i^k = (1-1)^k = 0$.
Ainsi, pour la méthode Δ^k donnée par (3.2), nous avons

$$\varepsilon_{n+1} = \left[I - (-1)^k \lambda_n (\psi - I)^k \right] \varepsilon_n + o(\varepsilon_n). \tag{3.9}$$

Par la remarque (3.8), nous voulons définir un nouveau schéma dont l'équation d'erreur est

$$\varepsilon_{n+1} = \left[I - (-1)^k \lambda_n (\psi - I)^k\right]^2 \varepsilon_n + o(\varepsilon_n). \tag{3.10}$$

La nouvelle méthode peut être alors décrite par l'algorithme suivant

$$\left\{ \begin{array}{l} \text{Soit } X_0 \text{ donné et pour } n = 0, 1, \ldots \\ \quad \text{Calculer le paramètre de relaxation} \\ \qquad \lambda_n = (-1)^k \dfrac{(\Delta X_n, \Delta^{k+1} X_n)}{(\Delta^{k+1} X_n, \Delta^{k+1} X_n)} \\ \quad \text{Calculer } X_{n+1} \text{ par} \\ \qquad X_{n+1} = X_n - 2\lambda_n (-1)^k \Delta^k X_n + \lambda_n^2 \Delta^{2k} X_n. \end{array} \right. \tag{3.11}$$

Nous l'appelons la méthode Δ^k-ajustée. Pour k=1, le pas de relaxation de la méthode Δ^1-ajustée correspond au paramètre de Lemaréchal [85] et, pour k=2, nous retrouvons le paramètre proposé par Chehab, c'est-à-dire le pas de relaxation de la méthode adaptative de Marder-Weitzner [43]. Remarquons que, pour k=1, notre méthode requiert seulement une somme vectorielle et un produit scalaire de plus par rapport à la méthode Δ^1 [28]. D'autre part, pour k=2, le nombre d'évaluations du terme non linéaire n'est pas le même. En effet, la méthode Δ^2-ajustée requiert une évaluation supplémentaire du terme non linéaire par rapport à la méthode Δ^2. Néanmoins, les expériences numériques montrent que pour k=2, notre schéma est efficace. Mais, dans les autres cas, c'est-à-dire pour $k > 2$, la méthode Δ^k-ajustée ne semble pas être convenable puisque le coût du calcul du vecteur $\Delta^{2k} X_n$ devient important. Dans la suite, nous étudions seulement les cas k=1 et k=2.

3.3 Convergence

Discutons de la convergence des méthodes Δ^1 et Δ^2 ajustées.

En accord avec [28, Théorème 3] et [85, Eqn (1)], nous supposons que T est monotone décroissante et satisfait une condition de Lipschitz, c'est-à-dire

$$\forall\ y, z, \quad (T(y) - T(z), y - z) \leq 0 \tag{3.12}$$

$$\exists\ L > 0 \text{ tel que } \forall\ y, z, \quad ||\, T(y) - T(z)\,|| \leq L ||\, y - z\,|| \tag{3.13}$$

où $||\, x\,||^2 = (x, x)$ pour tout x. Une propriété de monotonie, soit décroissante (≤ 0) soit croissante (≥ 0), est souvent considérée par les auteurs. Voir (en plus de [28, 85]), [79] où Korpelevich montre que son algorithme dit extragradient converge sous la condition de monotonie croissante et aussi [87] pour les applications à des problèmes d'équilibre.

Remarquons que la condition (3.12) implique l'unicité de la solution X. Par (3.3) et (3.4), nous avons

$$\lambda_n \quad = \quad -\frac{(T(X_n) - X_n, T^2(X_n) - 2T(X_n) + X_n)}{(T^2(X_n) - 2T(X_n) + X_n, T^2(X_n) - 2T(X_n) + X_n)}. \tag{3.14}$$

Dans la suite, l'itéré X_n est supposé différent de $T(X_n)$, sinon le calcul de X_{n+1} s'avére inutile. Soit $D_n > 0$ le dénominateur de λ_n. Nous avons les relations suivantes

$$\lambda_n D_n = || T(X_n) - X_n ||^2 - (T(X_n) - X_n, T^2(X_n) - T(X_n))$$

$$(1-\lambda_n) D_n = || T^2(X_n) - T(X_n) ||^2 - (T(X_n) - X_n, T^2(X_n) - T(X_n))$$

qui montrent, par (3.12), que $\lambda_n \in]0,1]$. Précisons qu'il peut-être montré de la même façon que D_n est *strictement* positif. De plus, par définition des termes $\Delta^k X_n$ et par (3.11) avec k=1, nous avons

$$X_{n+1} - X = a_n(X_n - X) + b_n(T(X_n) - X) + c_n(T^2(X_n) - X)$$

avec les variables positives a_n, b_n et c_n définies par

$$\begin{aligned} a_n &= (1-\lambda_n)^2 \\ b_n &= 2\lambda_n(1-\lambda_n) \\ c_n &= \lambda_n^2. \end{aligned}$$

Ainsi, nous avons

$$\begin{aligned} || X_{n+1} - X ||^2 &= a_n^2 || X_n - X ||^2 + b_n^2 || T(X_n) - X ||^2 + c_n^2 || T^2(X_n) - X ||^2 \\ &\quad + 2a_n b_n (X_n - X, T(X_n) - X) + 2a_n c_n (X_n - X, T^2(X_n) - X) \\ &\quad + 2b_n c_n (T(X_n) - X, T^2(X_n) - X). \end{aligned}$$

Finalement, nous en déduisons à partir des hypothèses (3.12), (3.13), de l'inégalité de Cauchy-Schwarz, et rappelant que $T(X) = X$, que

$$|| X_{n+1} - X ||^2 \leq M_n^2 \, || X_n - X ||^2,$$

avec $M_n^2 = (a_n + L^2 c_n)^2 + b_n^2 L^2 > 0$.
Mais, M_n^2 est un polynôme de degré 4 en la variable λ_n

$$M_n^2 = P(\lambda_n) = 1 - 4\lambda_n + (6 + 6L^2)\lambda_n^2 - (4 + 12L^2)\lambda_n^3 + (1 + L^4 + 6L^2)\lambda_n^4.$$

Ce polynôme décroît sur $[0, \xi]$ et croît sur $[\xi, 1]$, où ξ est l'unique zéro appartenant à [0,1] de la dérivée de P, ce qui peut-être prouvé en dérivant P trois fois et en utilisant le fait que $\lambda_n \in]0,1]$ (voir l'annexe C (page 46) pour le détail des calculs). D'autre part, $P(0) = 1$ et $P(1) = L^4$. Nous en déduisons donc le résultat suivant

Théorème 2.

- *Si $L < 1$, alors $M_n < 1$ et la méthode Δ^1-ajustée converge,*
- *Si $L \geq 1$ et $\lambda_n < \mu$, où μ est tel que $P(\mu) = 1$, alors $M_n < 1$ et le nouveau schéma converge.*

Remarque 2.
Dans le cas où k=2, nous supposons que le paramètre de relaxation λ_n=λ est constant. Par (3.10), nous en déduisons la condition suivante de convergence pour cette méthode

$$\rho\left(\left[I-\lambda(\psi-I)^2\right]^2\right)<1 \quad \textit{équivalente à} \quad 0<\lambda<\frac{2}{a^2}, \tag{3.15}$$

où ρ désigne le rayon spectral et a=sup $_{t\in sp(\psi)}|1-t|$. Si le paramètre de relaxation λ dépend de n et est donné par (3.11), rien ne peut-être dit sur la convergence du nouveau schéma.

3.4 Résultats numériques

3.4.1 Problème non linéaire elliptique

Nous considérons le problème non linéaire elliptique suivant

$$\left\{\begin{array}{rl} -\Delta u &= \gamma u-\nu|u|u \quad \text{dans } \Omega=]0,1[^2 \\ u &= 0 \quad \text{sur } \partial\Omega \end{array}\right. \tag{3.16}$$

avec $\gamma\geq\nu>0$. Le problème discrétisé correspondant peut être écrit de la façon suivante

$$AU=F(U)\in I\!R^p, \tag{3.17}$$

où A est la matrice de discrétisation de l'opérateur auto-adjoint $-\Delta$, obtenue par les différences finies sur une grille régulière, $F: I\!R^p\to I\!R^p$ est la fonction non linéaire tel que $F(U)=(\gamma U_i-\nu|U_i|U_i)_{1\leq i\leq p}$ et U est le vecteur contenant l'approximation de la solution du problème continu aux points de la grille. Remarquons qu'en posant $T=A^{-1}F(.)$, le problème (3.17) est équivalent à $U=T(U)$, c'est-à-dire le problème initial (3.1).

Comme expliqué dans [43, 44], la valeur du paramètre γ joue un rôle important dans ce problème. En effet, selon la valeur de ce paramètre, l'ensemble des solutions n'est pas le même. De plus, des solutions dîtes stables peuvent devenir instables. Nous rappelons qu'une solution est dîte stable (resp. instable) si la méthode de Picard converge (resp. ne converge pas) vers cette solution. Pour plus de détails voir [19, 20, 42, 43] et le rappel ci-après. Ici, nous calculons quelques solutions instables pour des valeurs choisies des paramètres γ et ν, avec différents vecteurs de départ $U^0_{i,j}=u_0(i.h,j.h)$, $i,j=1,\ldots,q$, $h=1/(q+1)$ avec $q^2=p$. Pour choisir les données initiales, nous rappelons plusieurs résultats [20] sur les solutions du problème (3.16).

Solutions instables et données initiales

Soient $\Lambda_{p,q}=\pi^2(p^2+q^2)$, $p.q\neq 0$, une valeur propre de l'opérateur $-\Delta$ et $\Phi_{p,q}=\sin(p\pi x)\sin(q\pi y)$ la fonction propre correspondante.

- Quand $\gamma<\Lambda_{1,1}$, la solution triviale $u\equiv 0$ est la seule solution et elle est stable.
- Quand $\Lambda_{1,1}<\gamma\leq\Lambda_{1,2}$, la solution triviale devient instable et il existe des solutions stables notées $K(1,1)$, lesquelles sont des déformations de la fonction propre $\Phi_{1,1}$, c'est-à-dire ce sont des fonctions qui ont leurs zéros et leurs extrema aux mêmes points.

- Quand $\Lambda_{p,q} < \gamma$, $p^2 + q^2 > 2$, toutes les solutions (incluant la solution triviale) sont instables sauf celles du type $K(1,1)$.
  Soient a et b tels que $a^2 + b^2 \leq p^2 + q^2$. Les solutions instables sont répertoriées en deux catégories
  - les solutions dîtes de type $K(a,b)$, lesquelles sont des déformations des fonctions $\Phi_{a,b}$, fonctions propres de $-\Delta$. Pour les calculer, nous prenons $U_0 = k\Phi_{a,b}$ comme vecteur initial,
  - les solutions dîtes de type $\Delta(a,b)$ qui sont des déformations des fonctions $\Theta_{a,b} = \sin(a\pi x)sin(b\pi y)Z(x,y)$, où $Z(x,y)$ s'annule aux points appartenant aux droites d'équations $y = x$ ou $y = -x$. Pour les calculer, nous prenons $U_0 = \Theta_{a,b}$ comme vecteur initial.

Critère d'arrêt

Les schémas considérés ici sont des méthodes itératives. Il est alors nécessaire de définir un critère d'arrêt qui indique si l'itéré calculé est assez proche de la solution. Si X_n est l'approximation de la solution X à l'étape n, nous définissons un résidu itératif relatif ω_n par

$$\omega_n = \frac{|| X_n - X_{n-1} ||}{|| X_{n-1} ||}.$$

Nous considérons que l'approximation X_n est suffisamment proche de la solution X si $\omega_n < 10^{-7}$.

Comparaison

Dans les tests suivants, nous distinguons les résultats fournis par les méthodes avec k=1 de ceux avec k=2. En effet, à chaque itération, les méthodes avec le choix k=2 nécessitent toujours plus d'évaluations de la fonction T que celles avec k=1 : il peut ainsi y avoir une différence importante en temps CPU entre les deux types de méthodes. Nous considérons les trois exemples suivants dont les résultats sont respectivement présentés dans les figures 3.1, 3.2 et 3.3 :

(a) Nous prenons $\gamma = 180 = \nu$. Nous calculons une solution $\Delta(1,1)$ et le vecteur initial est $u_0(x,y) = sin(\pi x)sin(\pi y)sin(\pi|x-y|)$. Nous choisissons $q = 100$ et alors la grille est composée de 100×100 points.

(b) Ici, nous considérons d'autres valeurs de γ(=150) et ν(=100).

(c) $\gamma = 120$, $\nu = 90$. Nous prenons une grille avec 150×150 points. La solution est de type $K(2,1)$ et la fonction initiale est $u_0(x,y) = sin(2\pi x)sin(\pi y)$.

Les résultats numériques montrent que les nouveaux schémas améliorent les méthodes Δ^k. Nous constatons toujours un gain important en itérations et donc en temps CPU. En général, dans le cas k=1, le nouveau schéma a un gain en itérations de 35%, et 30% pour

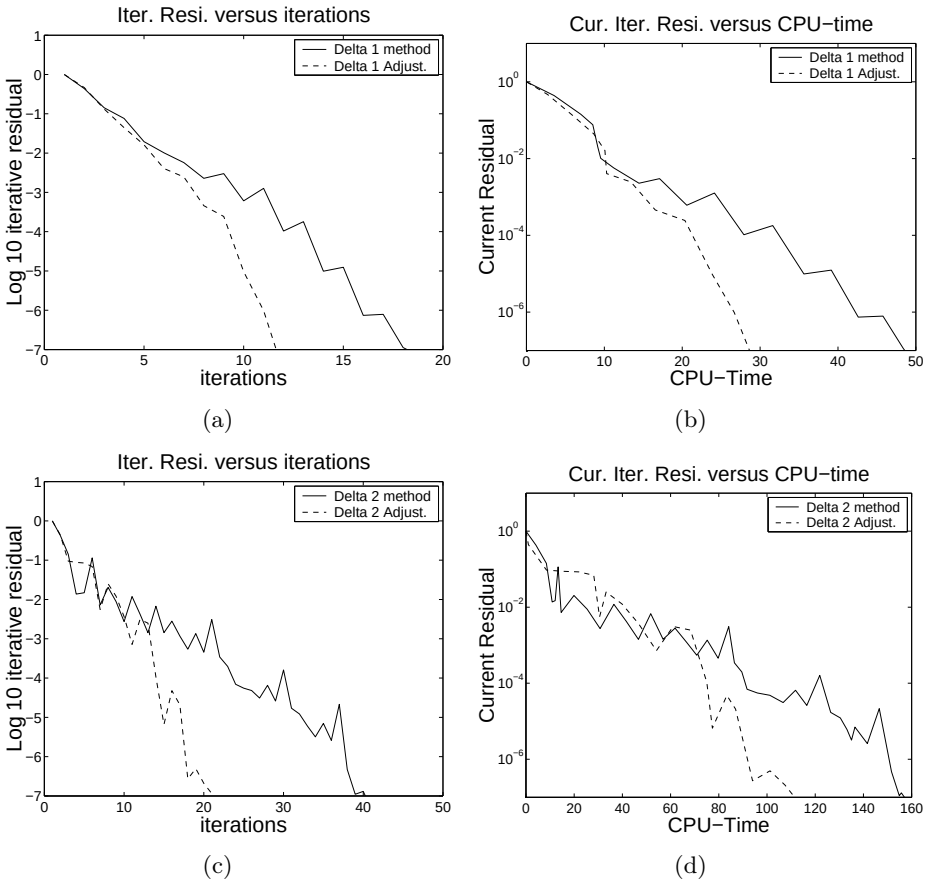


FIG. 3.1 – Exemple (a)

le temps CPU. En effet, il requiert seulement une somme vectorielle et un produit scalaire supplémentaires par rapport à la méthode Δ^1 à chaque itération. De ce fait, le gain en temps CPU est légérement inférieur au gain en itérations. Par contre, le nouveau schéma avec k=2 requiert une évaluation supplémentaire de la fonction T à chaque itération par rapport à la méthode classique. Cela explique que le pourcentage en gain de 25% en temps CPU est moins important que celui en itérations qui est de 45%.
Dans le tableau 3.1, nous reportons pour l'exemple (a) les valeurs du paramètre λ_n pour la méthode Δ^1 ajustée. Nous remarquons que le paramètre λ_n n'est pas toujours compris entre 0 et 1. Cela implique que la condition (3.12) n'est pas vérifiée dans nos applications. Des résultats de convergence sous des conditions plus faibles sont en préparation [105].

Donnons un autre exemple basé sur un problème de distribution de Poisson.

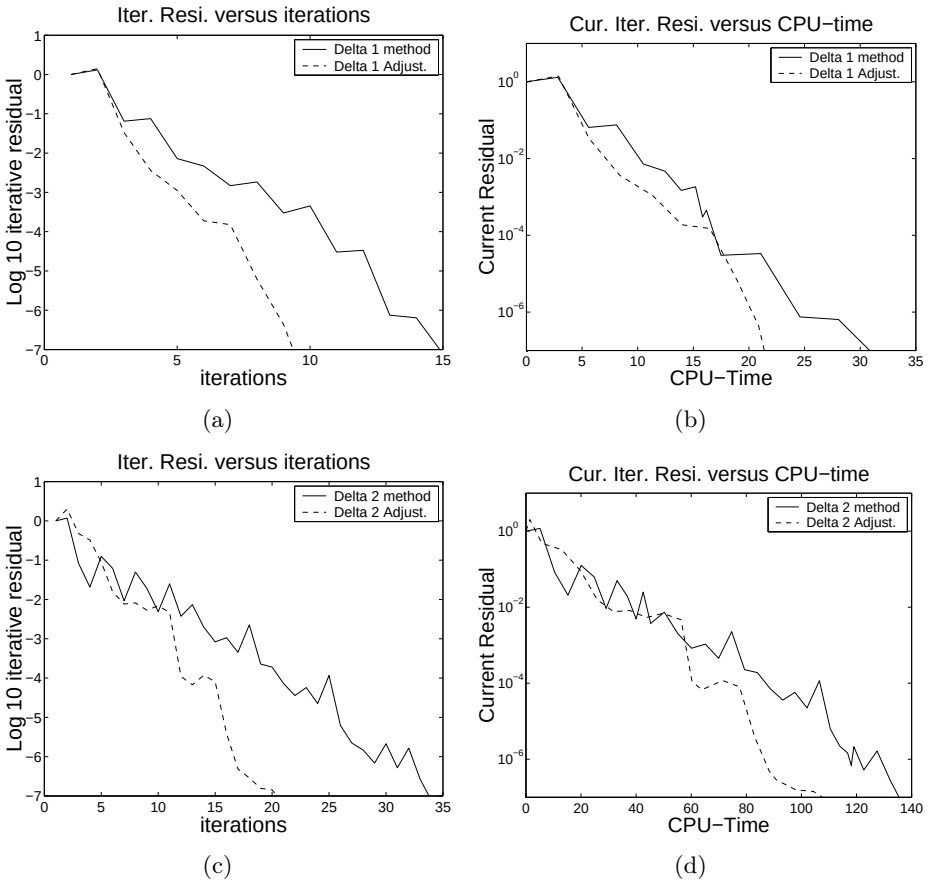


FIG. 3.2 – Exemple (b)

3.4.2 L'algorithme E.M. et Distribution de Poisson

L'algorithme E.M. de Dempster, Laird et Rubin [52] est une approche qui est très souvent appliquée dans des problèmes de données incomplètes dans le but de calculer itérativement les paramètres pour lesquels une fonction de densité de probabilité donnée est maximale. Ces paramètres sont appelés estimateurs du maximum de vraisemblance. Notre but dans ce chapitre n'est pas d'expliquer ce problème difficile. Pour un bref rappel, le lecteur est convié à lire le chapitre suivant et pour une étude complète il peut se référer à [52, 89, 123]. Comme expliqué dans [89], chaque cas de l'algorithme EM définit une fonction $\psi \to M(\psi)$, définie sur l'espace Ω, vers lui-même tel que

$$\psi_{m+1} = M(\psi_m). \tag{3.18}$$

Si ψ_m converge vers un point ψ^* et M est continue, alors ψ^* doit satisfaire $\psi^* = M(\psi^*)$ c'est-à-dire ψ^* est un point fixe de la fonction M. Nous pouvons ainsi considérer l'algorithme EM comme la méthode de Picard appliquée au problème de point fixe $M(\psi) = \psi$. Il est alors naturel d'adapter les méthodes Δ^k et notre amélioration à ce problème de sta-

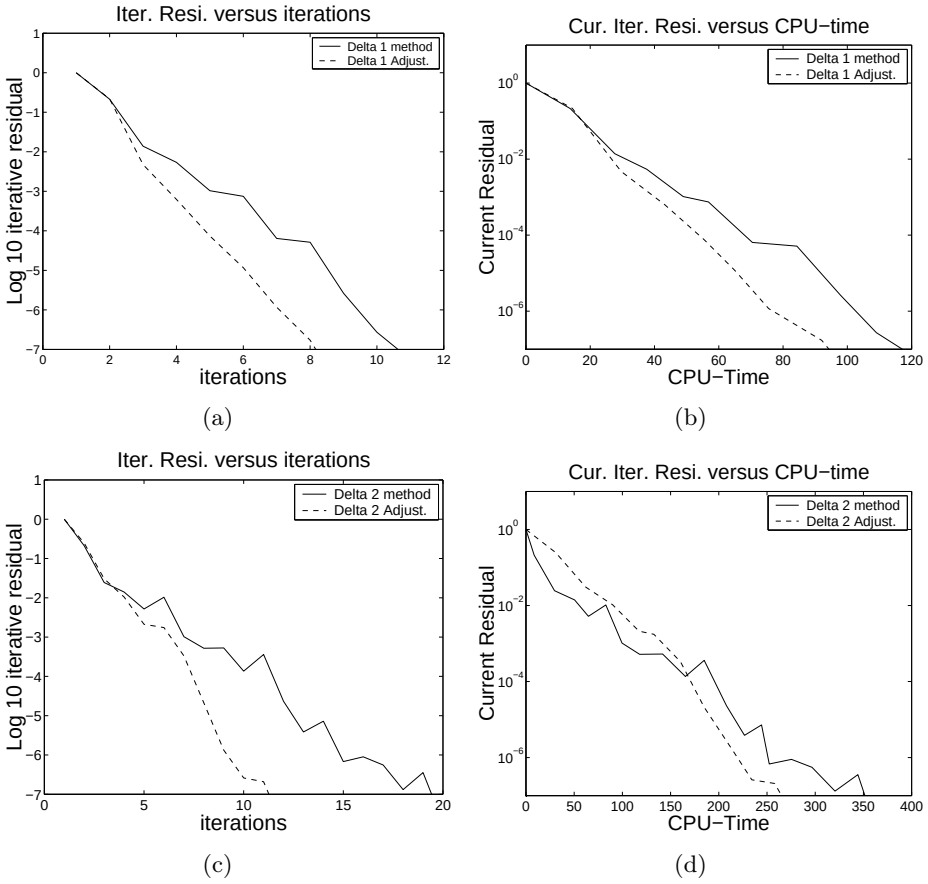


FIG. 3.3 – Exemple (c)

tistique. Un exemple classique est donné dans [123, pp.128] : il concerne une distribution de Poisson. Soit $\theta = (\alpha, \mu_1, \mu_2)^T$ et

$$z_i(\theta) = \frac{\alpha e^{-\mu_1}\mu_1^i}{\alpha e^{-\mu_1}\mu_1^i + (1-\alpha)e^{-\mu_2}\mu_2^i} \qquad \text{pour} \quad i = 0, \ldots, 9.$$

L'algorithme EM est donné par

$$\begin{aligned}
\alpha_{m+1} &= \frac{\sum_i n_i z_i(\theta_m)}{\sum_i n_i} \\
\mu_{m+1,1} &= \frac{\sum_i i n_i z_i(\theta_m)}{\sum_i n_i z_i(\theta_m)} \\
\mu_{m+1,2} &= \frac{\sum_i n_i i[1 - z_i(\theta_m)]}{\sum_i n_i [1 - z_i(\theta_m)]}
\end{aligned}$$

avec $\theta_m = (\alpha_m, \mu_{m,1}, \mu_{m,2})^T$ et les données observées (i, n_i) reportées dans le tableau 3.2. Les colonnes nommées "Décès i" désignent le nombre de décès de femmes de 80 ans ou

n	valeur de λ_n	n	valeur de λ_n	n	valeur de λ_n
0	0.467	4	0.441	8	0.345
1	0.379	5	0.895	9	1.12
2	0.452	6	0.3523	10	0.446
3	0.591	7	1.47		

TAB. 3.1 – Le paramètre λ_n pour la méthode Δ^1 ajustée

Décès i	Fréquence n_i	Décès i	Fréquence n_i
0	162	5	61
1	267	6	27
2	271	7	8
3	185	8	3
4	111	9	1

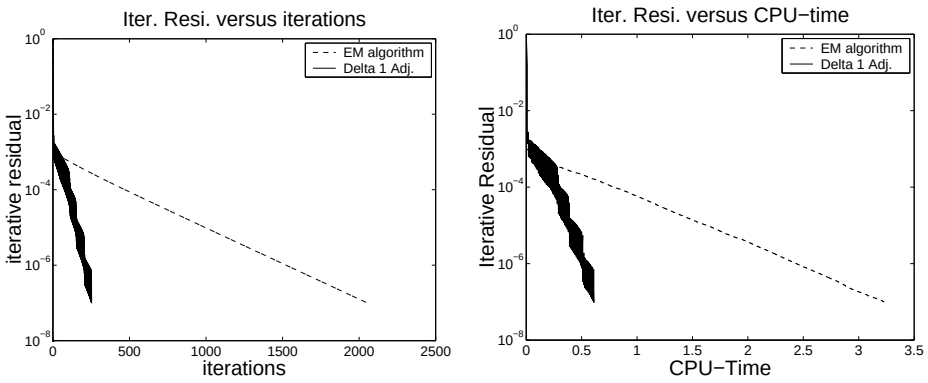
TAB. 3.2 – Les données observées

plus et les colonnes nommées "fréquence n_i" précisent le nombre de jours avec i décès de femmes de 80 ans ou plus. Précisons que le logarithme de la fonction de densité de probabilité (sous-entendu des données observées) est

$$L(\theta) = \sum_{i=0}^{9} n_i \log\left(\alpha e^{-\mu_1}\frac{\mu_1^i}{i!} + (1-\alpha)e^{-\mu_2}\frac{\mu_2^i}{i!}\right).$$

En prenant $\theta_0 = (0.3, 1, 2.5)^T$ comme vecteur initial, l'algorithme EM prend 534 itérations pour que le logarithme de la fonction de densité de probabilité atteigne son maximum de -1989.946. Il prend 2082 itérations pour obtenir les estimateurs du maximum de vraisemblance $\theta^* = (0.3599, 1.256, 2.663)^T$. Sur cet exemple, nous testons aussi les méthodes Δ^k et les méthodes Δ^k ajustées avec k=1 et k=2. Précisons que nous utilisons le même critère d'arrêt que celui décrit dans la sous-section 3.4.1. Les méthodes Δ^k n'atteignent pas le maximum du logarithme de la fonction de densité de probabilité. En effet, elles ne convergent pas vers les estimateurs θ^*, mais vers des estimateurs inappropriés. Dans [125], Roland et Varadhan montrent que ces schémas souffrent de problème de division par zéro ou/et de stagnation, et ils proposent un autre schéma plus stable. Par contre, l'algorithme EM et les méthodes Δ^k ajustées convergent vers la solution θ^*. La méthode Δ^2 ajustée converge très lentement et elle prend 25486 itérations pour atteindre la solution. Par contre, la méthode Δ^1 ajustée converge rapidement comme nous pouvons le voir dans la Figure 3.4 qui donne l'évolution du résidu itératif w_m selon les itérations dans la Figure 3.4(a) et selon le temps CPU dans la Figure 3.4(b) obtenue par les méthodes EM et Δ^1 ajustée. Le comportement précis de cette dernière ne peut être observé sur le graphe à cause d'importantes oscillations.

Nous observons sur cet exemple que notre méthode avec le choix k=1 est efficace. Nous obtenons encore une réduction significative du nombre d'itérations et du temps CPU. Dans [125], d'autres exemples ont été étudiés et confirment que le nouveau schéma avec k=1,



(a) Iterations (b) Cpu-Time

FIG. 3.4 – Exemple de distribution de Poisson

appelé méthode Δ^1 ajustée, est très prometteur.

3.5 Conclusion

Les nouveaux schémas introduits dans ce chapitre peuvent être considérés comme des améliorations efficaces des méthodes Δ^k. Leurs gains en itérations et en temps CPU montrent que ces méthodes sont intéressantes pour la résolution de problèmes non linéaires et peuvent être adaptées avec succès au problème de statistique présenté dans ce chapitre, à savoir l'accélération de l'algorithme EM. Un travail sur ce sujet peut être trouvé dans [125]. Plusieurs tests sont en étude afin de mieux comprendre ces méthodes et surtout de fournir des résultats de convergence sous des conditions plus faibles.

3.6 Annexe C

Montrons que le polynôme P de degré 4 en la variable λ défini par

$$P(\lambda) = 1 - 4\lambda + (6 + 6L^2)\lambda^2 - (4 + 12L^2)\lambda^3 + (1 + L^4 + 6L^2)\lambda^4$$

décroît sur $[0, \xi]$ et croît sur $[\xi, 1]$, où ξ est l'unique zéro appartenant à [0,1] de la dérivée de P. Dans ce but, calculons les dérivées successives de P. Elles sont données par

$$\begin{aligned} P^{(1)}(\lambda) &= -4 + 2(6 + 6L^2)\lambda - 3(4 + 12L^2)\lambda^2 + 4(1 + L^4 + 6L^2)\lambda^3 \\ P^{(2)}(\lambda) &= 2(6 + 6L^2) - 6(4 + 12L^2)\lambda + 12(1 + L^4 + 6L^2)\lambda^2 \\ P^{(3)}(\lambda) &= -6(4 + 12L^2) + 24(1 + L^4 + 6L^2)\lambda \end{aligned}$$

où $P^{(i)}$ désigne la $i^{\text{ème}}$ dérivée du polynôme P. Distinguons le cas $L \neq 1$ et $L = 1$.

Cas $L \neq 1$

Le polynôme $P^{(3)}$ s'annule en $z_0 = \dfrac{4 + 12L^2}{4 + 24L^2 + 4L^4}$ avec $0 < z_0 < 1$. Nous en déduisons le

tableau de variations 3.3 pour le polynôme $P^{(2)}$.

λ	0		z_0		1
$P^{(3)}(\lambda)$		$-$	0	$+$	
$P^{(2)}(\lambda)$	$2(6+6L^2)$	$\searrow$	$\dfrac{12L^2(1-L^2)^2}{1+L^4+6L^2} > 0$	$\nearrow$	$12L^2+12L^4$

TAB. 3.3 – Tableau de variations pour le polynôme $P^{(2)}$

Comme $P^{(2)}(z_0)$ est strictement positif, le polynôme $P^{(1)}$ est strictement croissant sur [0,1]. De plus, $P^{(1)}(0) = -4 < 0$ et $P^{(1)}(1) = 4L^4 > 0$. Par le théorème des valeurs intermédiaires, il existe ainsi un unique $\xi \in [0,1]$ tel que $P^{(1)}(\xi) = 0$. Il en suit le tableau de variations 3.4 pour le polynôme P.

λ	0		ξ		1
$P^{(1)}(\lambda)$		$-$	0	$+$	
$P(\lambda)$	1	$\searrow$	$P(\xi)$	$\nearrow$	L^4

TAB. 3.4 – Tableau de variations pour le polynôme P

Ainsi, dans le cas $L \neq 1$, le polynôme P décroît sur $[0,\xi]$ et croît sur $[\xi,1]$, où ξ est l'unique zéro du polynôme $P^{(1)}$ sur [0,1].

Cas $L = 1$

Dans ce cas, il peut-être montré que $P^{(1)}$ admet une racine unique de multiplicité trois égale à 0.5 et que P est décroissante sur $[0,\xi]$ et croissante sur $[\xi,1]$ où $\xi = 0.5$ est l'unique zéro de la dérivée de P sur $\mathbb{R}$ et donc sur [0,1].

Chapitre 4

Théorie de l'Algorithme E.M.

4.1 Introduction

L'algorithme E.M. Espérance-Maximisation est une approche qui est utilisée dans des problèmes de données incomplètes dans le but de calculer itérativement les paramètres pour lesquels une fonction de densité de probabilité donnée est maximale. Ces paramètres sont appelés estimateurs du maximum de vraisemblance. Conformément à la littérature, ils sont notés dans la suite en abrégé EMV. Dans les modèles statistiques, il est courant d'avoir des situations où des informations sont *incomplètes*. Ce fait peut s'expliquer par le manque direct d'informations sur des quantités mesurées ou mesurables, ou bien encore par une impossibilité d'observer quelques variables du modèle. Ces situations interviennent dans de nombreuses applications telles que l'évaluation de comportements sociaux, en médecine, en santé publique, en tomographie et en épidémiologie. L'algorithme E.M. est la méthode la plus utilisée pour résoudre ces problèmes de données incomplètes, en particulier quand le nombre de paramètres est relativement élévé, auquel cas des algorithmes tels que la méthode de Newton-Raphson peuvent s'avérer coûteux en stockage des données et surtout en temps de calcul. A chaque itération, l'algorithme E.M. est composé de deux étapes : l'étape de l'espérance (ou Étape E) et l'étape de maximisation (ou Étape M). C'est pour cette raison que dans un article fondamental [52], Dempster, Laird et Rubin lui ont donné le nom d'algorithme E.M.. L'idée de l'algorithme E.M. est d'associer au problème de données incomplètes un problème de données complètes pour lequel l'estimation du maximum de vraisemblance est plus facile. Il faut donc reformuler le problème initial en terme de problème de données complètes, établir une relation entre les fonctions de densité de probabilité de ces deux problèmes et exploiter la simplicité du calcul des E.M.V. pour le problème des données complètes dans l'étape M de l'algorithme. L'intérêt de ce chapitre est de présenter quelques éléments nécessaires à la compréhension de l'algorithme E.M.. Dans la première section, nous nous intéressons à sa formulation mathématique et sa convergence. Puis, dans la deuxième section, nous illustrons la théorie présentée à l'aide d'un exemple simple concernant une loi multinomiale.

4.2 Formulation et Convergence

Soient des observations y lesquelles sont supposées être générées par un modèle statistique ayant la fonction de densité de probabilité $g(y;\theta)$, où $y = (y_1, \ldots, y_n)$ est fixé et $\theta \in \Omega \subset \mathbb{R}^p$. Le but est de trouver le vecteur θ^* tel que

$$\theta^* = \text{argmax } \{L(\theta;y) \ ; \ \theta \in \Omega\} \tag{4.1}$$

où $L(\theta;y) = \log(g(y;\theta))$. L'algorithme E.M. permet de calculer itérativement ce vecteur θ^* dans les situations où par l'absence d'informations supplémentaires, la fonction $L(\theta;y)$ ne peut pas être maximisée analytiquement. Par ailleurs, même quand nous disposons de toutes les informations nécessaires, il peut s'avérer être plus facile de reformuler ce problème en un problème de données incomplètes pour pouvoir appliquer l'algorithme E.M.. Dans ce contexte, le vecteur des données observées y est dit incomplet et il est considéré comme une fonction observable d'un vecteur de données complètes noté x. Soit $g_c(x;\theta)$ la fonction de densité de probabilité du vecteur aléatoire X correspondant au vecteur des données complètes x. Notons $L_c(\theta;x)$, le logarithme de g_c, avec θ comme variable et x étant fixé. Formellement, nous avons ainsi deux espaces d'échantillons, $\mathbb{X}$ et $\mathbb{Y}$, et une fonction définie sur $\mathbb{X}$ vers $\mathbb{Y}$. Au lieu d'observer le vecteur des données complètes x dans $\mathbb{X}$, nous observons le vecteur des données incomplètes $y = y(x)$ dans $\mathbb{Y}$. Il s'en suit que

$$g(y;\theta) = \int_{\mathbb{X}(y)} g_c(x;\theta)\, dx \tag{4.2}$$

où $\mathbb{X}(y)$ est le sous-espace de $\mathbb{X}$ déterminé par l'équation $y = y(x)$.

L'algorithme E.M. permet de résoudre le problème (4.1) indirectement en procédant itérativement au moyen du logarithme de la fonction de densité de probabilité des données complètes $L_c(\theta;x)$. Puisque le vecteur des données complètes x n'est pas complètement observable, la fonction $L_c(\theta;x)$ est remplacée par son espérance conditionnelle sachant le vecteur des données observées y. Plus précisement, soit $\theta^{(0)}$ une valeur initiale pour θ. Alors pour la première itération, l'étape E requiert le calcul de

$$Q\left(\theta;\theta^{(0)}\right) = E\left[L_c(\theta;x);y;\theta^{(0)}\right]$$

Puis, l'étape M nécessite de trouver $\theta^{(1)}$ dans l'espace Ω, tel que

$$Q\left(\theta^{(1)};\theta^{(0)}\right) \geq Q\left(\theta;\theta^{(0)}\right)$$

pour tout $\theta \in \Omega$. Les étapes E et M sont répétées mais cette fois avec $\theta^{(0)}$ remplacé par $\theta^{(1)}$. Pour la $(k+1)^{\text{ième}}$ itération, les étapes E et M sont définies comme suit

Étape E. Calculer $Q\left(\theta;\theta^{(k)}\right)$ où

$$Q\left(\theta;\theta^{(k)}\right) = E\left[L_c(\theta;x);y;\theta^{(k)}\right]. \tag{4.3}$$

Étape M. Déterminer $\theta^{(k+1)}$ tel que

$$\theta^{(k+1)} = \text{argmax} \left\{ Q\left(\theta; \theta^{(k)}\right) \ ; \ \theta \in \Omega \right\}. \tag{4.4}$$

Les étapes E et M sont alternativement répétées jusqu'à ce que la différence $L(\theta^{(k+1)}; y) - L(\theta^{(k)}; y)$ soit plus petite que δ, où δ est une valeur à préciser. Remarquons qu'il n'est pas nécessaire de spécifier la relation entre les fonctions g et g_c donnée par l'équation (4.2) pour appliquer l'algorithme E.M.. Dans [52, Théorème 1], Dempster *et al* ont montré qu'à chaque itération k de l'algorithme E.M.

$$L(\theta^{(k+1)}; y) \geq L(\theta^{(k)}; y)$$

avec l'égalité si et seulement si

$$Q\left(\theta^{(k+1)}; \theta^{(k)}\right) = Q\left(\theta^{(k)}; \theta^{(k)}\right).$$

Par suite, si la fonction $L(\theta; y)$ est majorée sur Ω, la suite $L(\theta^{(k)}; y)$ converge de façon monotone vers une valeur L^*. Dans presque toutes les applications, L^* est une valeur stationnaire, c'est-à-dire $L^* = L(\theta^*; y)$ avec θ^* tel que $\partial L(\theta^*; y)/\partial\theta = 0$. Si $L(\theta; y)$ a plusieurs points stationnaires, la suite $(\theta^{(k)})$ de l'algorithme EM, peut converger vers un maximum local mais aussi un point selle de $L(\theta; y)$. La nature de la limite dépend de la valeur initiale $\theta^{(0)}$ (voir [16, p.34]). Dans le cas où $L(\theta; y)$ est unimodale dans Ω, c'est-à-dire $L(\theta; y)$ possède un unique point stationnaire et sous une certaine condition de différentiabilité sur la fonction Q (voir [128]), l'algorithme E.M. converge vers l'unique EMV θ^*. Pour un traitement rigoureux des théorèmes de convergence pour l'algorithme EM, le lecteur intéressé est convié à se référer à [128]. Le principal inconvénient de l'algorithme E.M. est que son taux de convergence r défini par

$$r = \lim_{k \to \infty} || \, \theta^{(k+1)} - \theta^* \, || / || \, \theta^{(k)} - \theta^* \, ||$$

est souvent voisin de 1 et donc sa convergence peut être extrêmement lente. Comme expliqué dans [52, Eq. 3.4] et aussi [89, Section 3], chaque cas de l'algorithme E.M. définit une fonction $\theta \to F(\theta)$ sur l'espace $\Omega \subset \mathbb{R}^p$ vers lui même tel que

$$\theta^{(k+1)} = F(\theta^{(k)}). \tag{4.5}$$

Si la suite $(\theta^{(k)})$ converge vers un point θ^* et si F est continue, alors θ^* doit satisfaire $\theta^* = F(\theta^*)$ c'est-à-dire θ^* est un point fixe de F. Par un développement de Taylor de $F(\theta^{(k)})$, nous avons dans un voisinage de θ^*

$$\begin{aligned} \theta^{(k+1)} - \theta^* &= F(\theta^{(k)}) - F(\theta^*) \\ &= J(\theta^*)(\theta^{(k)} - \theta^*) + o(\|\theta^{(k)} - \theta^*\|^2) \end{aligned} \tag{4.6}$$

où $J(\theta)$ est la matrice jacobienne de $F(\theta) = (F_1(\theta), \ldots, F_p(\theta))$ au point θ et ses éléments $J_{ij}(\theta)$ sont donnés par

$$J_{ij}(\theta) = \frac{\partial F_i(\theta)}{\partial \theta_j}.$$

Il fut montré dans [52, Eq. 3.26] que la matrice jacobienne au point fixe θ^* peut être écrite comme

$$J(\theta^*) = I_p - I_{obs}(\theta^*; y) I_{comp}^{-1}(\theta^*; y), \tag{4.7}$$

où I_p est la matrice identité de dimension p, $I_{obs}(\theta; y)$ est la matrice hessienne (au signe près) de l'information des données observées définie par

$$I_{obs}(\theta; y) = -\frac{\partial^2 L(\theta; y)}{\partial \theta^2} \tag{4.8}$$

et

$$I_{comp}(\theta; y) = E\left[-\frac{\partial^2 L_c(\theta; x)}{\partial \theta^2}; y; \theta \right]. \tag{4.9}$$

De ce fait, par l'équation (4.6), l'algorithme E.M. est essentiellement une itération linéaire dans un voisinage du point θ^* avec une matrice d'itération $J(\theta^*)$. Le taux de convergence de l'algorithme E.M. r est donné par le rayon spectral de la matrice $J(\theta^*)$. Si la matrice jacobienne de la fonction E.M. n'est pas symétrique, ses valeurs propres peuvent être complexes. Mais, en général, la matrice $I_{obs}(\theta^*; y)$ est définie positive auquel cas (voir [52, Page 10]) les valeurs propres de la matrice $J(\theta^*)$ sont réelles et surtout dans [0,1). Sous cette condition, les itérations $\theta^{(k)}$ de l'algorithme E.M. convergent vers θ^* et de plus, si le rayon spectral de la matrice $J(\theta^*)$ est proche de 1, la convergence de l'algorithme E.M. peut être très lente.

Pour illustrer cette théorie, donnons un exemple simple concernant une loi multinomiale.

4.3 Exemple : la Loi Multinomiale

Nous considérons un exemple multinomial que Dempster, Laird et Rubin [52, Page 2] ont utilisé pour introduire l'algorithme E.M. et qui a été utilisé de nombreuses fois dans la littérature pour illustrer des modifications et extensions de l'algorithme EM. Pour une meilleure compréhension de cet algorithme, nous tenterons de détailler et justifier tous les calculs.

Soit le vecteur des données observées

$$y = (y_1, y_2, y_3, y_4)^T$$

supposé être généré par une distribution multinomiale avec les probabilités respectives

$$\frac{1}{2} + \frac{1}{4}\,\theta, \quad \frac{1}{4}(1-\theta), \quad \frac{1}{4}(1-\theta) \quad \text{et} \quad \frac{1}{4}\,\theta \tag{4.10}$$

avec $0 \leq \theta \leq 1$. Dans [52], le vecteur y est donné par

$$y = (125, 18, 20, 34)^T \tag{4.11}$$

et l'échantillon est donc de taille $n = \sum_{i=1}^{4} y_i = 197$. Par contre, Thisted [122, Section 4.2.6] a considéré le même exemple mais avec un échantillon de taille $n = 3839$ en définissant

$$y = (1997, 906, 904, 32)^T. \tag{4.12}$$

Par définition d'une loi multinomiale, la fonction de densité de probabilité, généralement notée en abrégé f.d.p, pour le vecteur des données observées y, est donnée par

$$g(y;\theta) = \frac{n!}{y_1!y_2!y_3!y_4!}\left(\frac{1}{2}+\frac{1}{4}\theta\right)^{y_1}\left(\frac{1}{4}(1-\theta)\right)^{y_2}\left(\frac{1}{4}(1-\theta)\right)^{y_3}\left(\frac{1}{4}\theta\right)^{y_4}.$$

A une constante additive près n'impliquant pas la variable θ, le logarithme de la fonction de densité de probabilité est donc

$$L(\theta;y) = \log(g(y;\theta)) = y_1\log(2+\theta) + (y_2+y_3)\log(1-\theta) + y_4\log(\theta). \tag{4.13}$$

Par différenciation de (4.13) par rapport à la variable θ, nous avons

$$\frac{\partial L(\theta;y)}{\partial\theta} = \frac{y_1}{2+\theta} - \frac{y_2+y_3}{1-\theta} + \frac{y_4}{\theta}. \tag{4.14}$$

La solution θ^* du problème (4.1), que nous cherchons, est la solution de l'équation $\partial L(\theta;y)/\partial\theta$=0 avec la contrainte $\theta \in [0,1]$. Or, l'équation (4.14) peut être réécrite comme une fonction rationnelle dont le numérateur est quadratique en la variable θ. En résolvant cette équation du second degré, avec le vecteur y donné par (4.11), nous trouvons $\theta^* \approx 0.62682$. Par contre, avec le vecteur y de l'équation (4.12), nous obtenons $\theta^* \approx 0.03571$. En pratique, il n'est pas alors nécessaire pour cet exemple d'utiliser l'algorithme E.M., puisque l'estimateur θ^* peut être obtenu analytiquement. Reformulons maintenant l'exemple en un problème de données incomplètes pour pouvoir appliquer et comprendre la théorie de l'algorithme EM. Posons $y_1 = y_{11} + y_{12}$ et définissons le vecteur

$$x = (y_{11}, y_{12}, y_2, y_3, y_4)^T \tag{4.15}$$

supposé être généré par une distribution multinomiale avec les probabilités respectives

$$\frac{1}{2},\quad \frac{1}{4}\,\theta,\quad \frac{1}{4}(1-\theta),\quad \frac{1}{4}(1-\theta)\ \text{ et }\ \frac{1}{4}\,\theta. \tag{4.16}$$

Le vecteur x est considéré comme le vecteur des données complètes. Remarquons que le vecteur des données observées y est une fonction du vecteur des données complètes x puisque nous avons $y = (y_{11}+y_{12}, y_2, y_3, y_4)^T$. Par cette reformulation, les vecteurs y_{11} et y_{12} sont considérés comme des données inobservables ou manquantes, puisque seule leur somme y_1 est observée et les données observées sont considérées incomplètes. Nous avons ainsi la relation suivante entre le modèle des données complètes et le modèle des données observables

$$g(y;\theta) = \sum_{x\ \in\ \mathbb{X}(y)} g_c(x;\theta) \tag{4.17}$$

où

$$g_c(x;\theta) = \frac{n!}{y_{11}!y_{12}!y_2!y_3!y_4!}\left(\frac{1}{2}\right)^{y_{11}}\left(\frac{\theta}{4}\right)^{y_{12}}\left(\frac{1-\theta}{4}\right)^{y_2}\left(\frac{1-\theta}{4}\right)^{y_3}\left(\frac{\theta}{4}\right)^{y_4} \tag{4.18}$$

et

$$\mathbb{X}(y) = \left\{x = (x_1,x_2,x_3,x_4,x_5)^T :\ \text{tel que } y = (x_1+x_2,x_3,x_4,x_5)^T\right\}.$$

Pour montrer l'équation (4.17), il suffit d'utiliser les définitions de g et g_c et de montrer que

$$\frac{1}{y_1!}\left(\frac{1}{2}+\frac{1}{4}\theta\right)^{y_1} = \sum_{x \in \mathbb{X}(y)} \frac{1}{x_1!}\frac{1}{x_2!}\left(\frac{1}{2}\right)^{x_1}\left(\frac{\theta}{4}\right)^{x_2}.$$

A une constante additive près n'impliquant pas la variable θ, le logarithme de la fonction de densité de probabilité des données complètes est donc

$$L_c(\theta;x) = \log(g_c(x;\theta)) = (y_{12}+y_4)\log(\theta) + (y_2+y_3)\log(1-\theta). \tag{4.19}$$

La solution de l'équation $\partial L_c(\theta;x)/\partial\theta = 0$, que nous notons θ_c^* correspondant au EMV pour les données complètes, est

$$(y_{12}+y_4)/(y_{12}+y_2+y_3+y_4). \tag{4.20}$$

Puisque la donnée y_{12} n'est pas observable, nous ne sommes pas capables d'estimer le vecteur θ_c^* par l'équation (4.20). C'est dans de telles situations que l'algorithme EM est efficace puisqu'il permet de surmonter cet obstacle par l'étape E (Eq. 4.3), c'est-à-dire en calculant l'espérance conditionnelle du logarithme de la fonction de densité de probabilité des données complètes sachant le vecteur des données observées y. Soit $\theta^{(0)}$ une valeur initiale pour le paramètre θ. Alors pour la première itération de l'algorithme EM, l'étape E requiert le calcul de la quantité

$$Q\left(\theta;\theta^{(0)}\right) = E\left[L_c(\theta;x);y;\theta^{(0)}\right].$$

Par l'équation (4.19) et la propriété de linéarité de l'espérance conditionnelle, nous avons

$$\begin{aligned} Q\left(\theta;\theta^{(0)}\right) &= \left(E\left[y_{12};y;\theta^{(0)}\right] + E\left[y_4;y;\theta^{(0)}\right]\right)\ \log(\theta) \\ &\quad + \left(E\left[y_2;y;\theta^{(0)}\right] + E\left[y_3;y;\theta^{(0)}\right]\right)\ \log(1-\theta). \end{aligned} \tag{4.21}$$

Or les données y_2, y_3 et y_4 sont observées. De ce fait, par définition de l'espérance conditionnelle et la formule de Bayes, nous avons

$$E\left[y_2;y;\theta^{(0)}\right] = E\left[y_2;y_2;\theta^{(0)}\right] = y_2.$$

Similairement, nous avons $E\left[y_3;y;\theta^{(0)}\right] = y_3$ et $E\left[y_4;y;\theta^{(0)}\right] = y_4$. Reste à calculer la quantité

$$E\left[y_{12};y;\theta^{(0)}\right] = E\left[y_{12};y_1;\theta^{(0)}\right].$$

En considérant la variable aléatoire Y_{12} (resp. Y_1), correspondant à y_{12} (resp. y_1), et par définition du conditionnement d'une variable aléatoire par une autre, nous en déduisons que la variable aléatoire Y_{12} conditionnée par Y_1 suit une loi binomiale $B(y_1;p)$ avec

$$p = \left(\frac{1}{4}\theta^{(0)}\right) / \left(\frac{1}{2}+\frac{1}{4}\theta^{(0)}\right).$$

Par cette remarque et la formule de Bayes, l'espérance conditionnelle de Y_{12} sachant y_1 est donc

$$E\left[y_{12}; y; \theta^{(0)}\right] = y_{12}^{(0)}$$

avec

$$y_{12}^{(0)} = \left(\frac{1}{4} y_1 \theta^{(0)}\right) / \left(\frac{1}{2} + \frac{1}{4}\theta^{(0)}\right).$$

De ce fait, nous obtenons pour l'étape E

$$Q\left(\theta; \theta^{(0)}\right) = (y_{12}^{(0)} + y_4)\log(\theta) + (y_2 + y_3)\log(1-\theta).$$

L'étape M (Eq. 4.4) consiste à choisir à la première itération pour $\theta^{(1)}$ la valeur de θ qui maximise $Q(\theta; \theta^{(0)})$. Par (4.20), (4.21) et le calcul des espérances conditionnelles ci-dessus, nous obtenons

$$\theta^{(1)} = \frac{y_{12}^{(0)} + y_4}{y_{12}^{(0)} + y_2 + y_3 + y_4}. \tag{4.22}$$

Les étapes E et M sont répétées mais cette fois-ci avec $\theta^{(0)}$ remplacé par $\theta^{(1)}$. Pour la $(k+1)^{\text{ième}}$ itération, l'algorithme EM est donné par

- $y_{12}^{(k)} = \left(\frac{1}{4} y_1 \theta^{(k)}\right) / \left(\frac{1}{2} + \frac{1}{4}\theta^{(k)}\right).$
- $\theta^{(k+1)} = \dfrac{y_{12}^{(k)} + y_4}{y_{12}^{(k)} + y_2 + y_3 + y_4}.$

Il est ainsi vérifié que l'algorithme EM définit bien une fonction F telle que $\theta^{(k+1)} = F(\theta^{(k)})$ (Eq. (4.5)). Dans les tableaux 4.1 et 4.2, nous reportons les résultats de l'algorithme E.M. appliqué aux données y considérées par Dempster *et al* (Eq. 4.11) et par Thisted (Eq. 4.12). Précisons que les résultats présentés ici diffèrent légèrement de ceux obtenus dans [52, 122]. Cela pourrait s'expliquer par la précision choisie pour le vecteur θ^*. Dans le tableau 4.1, nous remarquons qu'à partir d'une valeur initiale $\theta^{(0)} = 0.5$, l'algorithme E.M. donne la solution du problème en 8 itérations. Il peut être observé qu'à partir de $k \geq 4$, la quantité $(||\,\theta^{(k)} - \theta^*\,||)/(||\,\theta^{(k-1)} - \theta^*\,||)$ est constante. L'algorithme EM converge donc linéairement avec un taux de convergence égal à 0.1328, d'où une convergence rapide. Par contre, dans le tableau 4.2, nous observons qu'à partir d'une valeur initiale $\theta^{(0)} = 0.057$, l'algorithme EM nécessite 21 itérations pour obtenir la solution exacte. En effet, le taux de convergence égal à 0.49511 est plus proche de 1, donc la convergence linéaire est moins rapide. Il peut être vérifié que les deux taux de convergence obtenus coïncident avec le rayon spectral de la matrice $J(\theta^*)$ définie par l'équation (4.6).

Itération k	$\theta^{(k)}$	$\theta^{(k)} - \theta^*$	$\frac{\|\| \theta^{(k)} - \theta^* \|\|}{\|\| \theta^{(k-1)} - \theta^* \|\|}$	$\text{Log}(L(\theta^{(k)})$
0	0.500000000	0.126821497	-	64.62974
1	0.608247422	0.018574074	0.1484	67.32017
2	0.624321050	0.002500446	0.1348	67.38292
3	0.626488879	0.000332617	0.1330	67.38408
4	0.626777322	0.000044174	0.1328	67.38410
5	0.626815632	0.000005864	0.1328	67.38410
6	0.626820719	0.000000777	0.1328	67.38410
7	0.626821394	0.000000102	0.1328	67.38410
8	0.626821484	0.000000012	-	67.38410

TAB. 4.1 – Résultats de l'algorithme EM pour $y = (125, 18, 20, 34)^T$

Itération k	$\theta^{(k)}$	$\theta^{(k)} - \theta^*$	$\frac{\|\| \theta^{(k)} - \theta^* \|\|}{\|\| \theta^{(k-1)} - \theta^* \|\|}$	$\text{Log}(L(\theta^{(k)})$
0	0.057000000	0.021287698	-	1242.4358
1	0.046031552	0.010319250	0.47977	1245.8511
2	0.040769136	0.005056834	0.48756	1246.7804
3	0.038203383	0.002491081	0.49138	1247.0230
4	0.036942605	0.001230303	0.49327	1247.0845
5	0.036320697	0.000608395	0.49420	1247.0999
6	0.036013346	0.000301044	0.49466	1247.1037
7	0.035861310	0.000149008	0.49489	1247.1046
8	0.035786068	0.000073766	0.49500	1247.1049
9	0.035748822	0.000036520	0.49506	1247.1050
10	0.035730383	0.000018081	0.49509	1247.1050
11	0.035721254	0.000008952	0.49509	1247.1050
12	0.035716735	0.000004433	0.49511	1247.1050
13	0.035714497	0.000002195	0.49511	1247.1050
14	0.035713389	0.000001087	0.49511	1247.1050
15	0.035712840	0.000000538	0.49511	1247.1050
16	0.035712569	0.000000267	0.49511	1247.1050
17	0.035712434	0.000000132	0.49511	1247.1050
18	0.035712368	0.000000066	0.49511	1247.1050
19	0.035712335	0.000000033	0.49511	1247.1050
20	0.035712318	0.000000016	0.49511	1247.1050
21	0.035712310	0.000000008	-	1247.1050

TAB. 4.2 – Résultats de l'algorithme EM pour $y = (1997, 906, 904, 32)^T$

4.4 Conclusion

Dans ce chapitre, nous avons donc présenté quelques outils nécessaires à la compréhension de l'algorithme E.M., approche très utilisée pour calculer itérativement des estimateurs du maximum de vraisemblance pour des problèmes avec des données incomplètes. Malgré quelques propriétés intéressantes (croissance de la fonction de densité de probabilité des données observées, stabilité numérique), l'algorithme E.M. a un inconvénient majeur : sa convergence linéaire peut être extrêmement lente. Il est par conséquent nécessaire de proposer de nouveaux schémas numériques afin d'accélérer la convergence de cet algorithme.

Nous concluons ce chapitre par une citation de Lange [82, Page 16], laquelle nous a grandement motivés à proposer des nouveaux schémas pour accélérer la convergence de l'algorithme E.M. :

> The missing data paradigm is ubiquitous in statistical applications, and the EM algorithm enjoys ever wider use. Any measure that improves the EM algorithm can only benefit consumers of statistics.

Chapitre 5

Accélération de la convergence de l'algorithme E.M.

5.1 Introduction

Considérons une fonction non linéaire $F : \Omega \subset I\!R^p \rightarrow \Omega$. Le but est de trouver, s'il existe, le point fixe x^*, de la fonction F, c'est-à-dire la solution de l'équation suivante

$$x = F(x). \tag{5.1}$$

Une des méthodes les plus simples pour résoudre (5.1) est la méthode de Picard définie par

$$x_{n+1} = F(x_n) \quad (n = 0, 1, 2, \ldots). \tag{5.2}$$

Si la suite (x_n) converge vers un vecteur x^* et si la fonction F est continue, alors $x^* = F(x^*)$. Ainsi, x^* est un point fixe de la fonction F. Dans les discussions qui suivent, nous supposons que la fonction F est Lipschitz-continue, avec une constante de Lipschitz inférieure à 1, c'est-à-dire

$$\forall\ x, y \in \Omega : || F(x) - F(y) || \leq L || x - y || \tag{5.3}$$

où $L < 1$ est la constante de Lipschitz. Nous supposons aussi que la fonction F admet des dérivées partielles continues et bornées. Sous ces conditions, la méthode de Picard (Eq. (5.2)) est linéairement convergente. Le taux de convergence de cette méthode est $\rho(J(x^*))$, où $\rho(M)$ désigne le rayon spectral de la matrice M et $J(x^*)$ est la matrice jacobienne de la fonction F évaluée au point fixe x^*. Par suite, si la plus grande valeur propre (en module) de la matrice $J(x^*)$ est proche de 1, alors la méthode de Picard converge lentement.

Dans ce chapitre, nous nous intéressons à l'accélération de la convergence de la méthode de Picard. Nous exploitons la connection entre les méthodes d'extrapolation et les méthodes de point fixe pour résoudre l'équation (5.1), en utilisant une stratégie de cyclage. L'exemple le plus connu de cette connexion est le lien entre le processus Δ^2 d'Aitken et la méthode de Steffensen, dans le cas scalaire ($p = 1$). La stratégie de cyclage consiste à définir des cycles : à l'intérieur de chaque cycle, à partir d'un vecteur initial, des itérations de Picard, Eq. (5.2), sont calculées, puis une méthode d'extrapolation est appliquée à ces itérations de Picard

pour obtenir un vecteur dit extrapolé qui sera le vecteur initial au prochain cycle. Puis, nous définissons une nouvelle stratégie appelée *squaring* que nous appliquons à des méthodes itératives de point fixe obtenues en cyclant des méthodes d'extrapolation. Nous obtenons alors une nouvelle classe de méthodes itératives de point fixe, convergeant linéairement mais plus rapidement que les itérations de Picard. Elles sont appelées les méthodes squarem. Dans les méthodes squarem, le vecteur x_{n+1}, itéré de la nouvelle méthode à l'itération n, est obtenu en deux étapes. La première étape consiste à appliquer à l'itéré x_n une méthode d'extrapolation pour déterminer un vecteur intermédiaire z_n auquel nous appliquons une deuxième fois la même méthode d'extrapolation pour obtenir le vecteur x_{n+1}.

Les méthodes itératives basées sur la stratégie de cyclage, sont depuis longtemps utilisées pour la résolution de problèmes non linéaires et linéaires de point fixe (voir par exemple [118]). Par contre, la stratégie *squaring* est relativement nouvelle. Elle fut premièrement employée par Raydan et Svaiter [100] pour accélérer la convergence de la méthode classique de Cauchy [40], aussi appelée méthode de la plus profonde descente, utilisée afin de résoudre un problème de minimisation d'une fonctionnelle quadratique. Roland et Varadhan [104] ont élargi la technique *squaring* pour résoudre des problèmes non linéaires de point fixe. Ils ont proposé une version squarem du schéma itératif de Lemaréchal [85] lequel peut être considéré comme une extension de la méthode de Steffensen au cas vectoriel. Ici, nous proposons une classe plus large de méthodes itératives. En effet, la stratégie *squaring* est ici appliquée à des méthodes itératives issues de la stratégie de cyclage. En particulier, nous nous focalisons sur deux types de méthodes d'extrapolation : la *minimal polynomial extrapolation method*, M.P.E. [39] et *la reduced rank extrapolation method*, R.R.E. [93].

Notre but principal est d'utiliser les méthodes itératives RRE et MPE, et leurs versions squarem afin d'accélérer la convergence de l'algorithme EM [52]. Cet algorithme peut être considéré comme une méthode de Picard (5.2) utilisée afin de résoudre des problèmes de maximisation d'une fonction de densité de probabilité en présence de données incomplètes. L'algorithme EM (voir chapitre 4) possède des propriétés intéressantes : il est (a) simple à appliquer pour des problèmes avec des données manquantes importantes, (b) globalement convergeant, (c) exhibe une croissance monotone de la fonction de densité de probabilité, et (d) satisfait naturellement des contraintes de paramètres. Mais, l'inconvénient majeur est que sa convergence peut être très lente, en particulier quand la part des données manquantes est importante. Il est par conséquent justifié d'accélérer la convergence de cet algorithme. Précisons que les nouveaux schémas peuvent aussi être employés pour accélérer la convergence linéaire des variantes de l'algorithme EM telles que les algorithmes G.E.M. [83], et E.C.M. [91]. Ces variantes qui possèdent des propriétés de stabilité et de convergence similaires à l'algorithme EM, sont utilisées quand l'étape M de l'algorithme EM est analytiquement intraitable. De nombreux schémas numériques ont été proposés pour accélérer l'algorithme EM. Ils incluent les méthodes de Quasi-Newton ([74],[82]) et les méthodes du gradient conjugué [73]. Bien que ces schémas améliorent généralement la convergence de l'algorithme EM, ils tendent à perdre une, voire plusieurs, des propriétés intéressantes de l'algorithme EM. Par exemple, les méthodes de type Quasi-Newton perdent la propriété de monotonie de la fonction de densité de probabilité. De plus, la superlinéarité de ces méthodes n'est réalisée que pour des valeurs initiales proches de la solution x^*. Par suite, la convergence est généralement linéaire. Et enfin, elles nécessitent le

calcul de quantités auxiliaires. Par exemple, les méthodes du gradient conjugué impliquent le calcul du logarithme de la fonction de densité de probabilité des données complètes et son gradient.

Comme expliqué dans [73], nous pouvons considérer deux types de coûts associés au développement d'un schéma : les coûts associés à son implémentation, *coût de l'analyse* et les coûts associés à l'utilisation d'un ordinateur et le temps qu'elle prend pour donner le résultat, *le coût machine*. Dans de nombreux problèmes de recherche scientifique, c'est typiquement les coûts de l'analyse qui sont limités. Mais, dans des modèles statistiques complexes, il est parfois très coûteux d'évaluer le gradient et la hessienne de la fonction de densité de probabilité des données complètes à chaque étape EM. Par conséquent, la capacité de développer de bons modèles est sévérement réduit à cause du temps de calcul requis pour lancer une simulation. Dans quelques applications technologiques telles que la reconstruction d'images en tomographie, les modèles et les étapes de l'algorithme EM peuvent être simples, mais elles impliquent l'estimation de dizaines de millions de paramètres ce qui rend le coût machine critique. Les méthodes squarem donnent un juste milieu entre les coûts de calcul et d'analyse puisqu'elles possédent les avantages suivants : (a) elles sont simples et requièrent seulement quelques itérations de base de l'algorithme EM, (b) elles ne requièrent pas la programmation de quantités auxiliaires telles que les fonctions de densité de probabilité des données incomplètes ou complètes ou leurs gradients, (c) par (a) et (b) elles peuvent être intégrées facilement dans les sous-programmes existants de l'algorithme EM, (d) elles n'impliquent pas de stockage supplémentaire de vecteurs ou de matrices et (e) elles convergent linéairement, comme l'algorithme EM, mais avec un taux de convergence plus rapide, où les gains peuvent être importants, en particulier dans les problèmes où l'algorithme EM est très lent.

5.2 Quelques Rappels

5.2.1 Transformation Scalaire et Extrapolation

La nouvelle classe de schémas proposée dans ce chapitre, pour l'accélération des itérations de point fixe et en particulier l'algorithme EM, exploite la connection entre les méthodes d'extrapolation, les transformations de suites, et les itérations de point fixe. Dans cette sous-section et les suivantes, nous présentons les éléments essentiels sur les transformations de suites et les méthodes d'extrapolation pour permettre par la suite une meilleure compréhension des nouveaux schémas. Pour une étude complète sur ce sujet, le lecteur est convié à lire [36].

Le but de cette section est de présenter pour le cas scalaire ($p = 1$) des méthodes d'accélération de convergence habituellement obtenues par une procédure d'extrapolation.

Considérons une suite scalaire (x_n) qui converge vers le scalaire x, mais dont la convergence est lente et doit par conséquent être accélérée. Nous transformons alors la suite (x_n) en une nouvelle suite (t_n) et nous notons T une telle transformation. Afin de présenter un intérêt, la suite (t_n) doit vérifier les propriétés suivantes

1. (t_n) converge

2. (t_n) converge aussi vers x
3. (t_n) converge vers x plus vite que (x_n), c'est-à-dire

$$\lim_{n\to\infty}(t_n - x)/(x_n - x) = 0.$$

Si ces trois conditions sont satisfaites alors nous disons que la suite (t_n) converge plus vite que la suite (x_n) ou bien encore, la transformation T accélére la convergence de la suite (x_n). Par exemple, nous pouvons avoir

$$t_n = (x_n + x_{n+1})/2$$

ou

$$t_n = x_n - \frac{(x_{n+1} - x_n)^2}{x_{n+2} - 2x_{n+1} + x_n}.$$

La première transformation est un processus de sommation et la seconde est le processus Δ^2 d'Aitken [2]. Étudions quelle classe de suites ces deux transformations permettent d'accélérer. Pour la première transformation, nous écrivons

$$\frac{t_n - x}{x_n - x} = \frac{1}{2}\left(1 + \frac{x_{n+1} - x}{x_n - x}\right).$$

De ce fait,

$$\lim_{n\to\infty}\frac{t_n - x}{x_n - x} = 0 \text{ si et seulement si } \lim_{n\to\infty}\frac{x_{n+1} - x}{x_n - x} = -1$$

ce qui montre que cette transformation est seulement capable d'accélérer la convergence d'une classe très restrictive de suites, ce qui est essentiellement le cas pour tous les processus de sommation. Pour le processus d'Aitken, il peut être prouvé qu'il accélére la convergence de toutes les suites pour lesquelles il existe un $\rho \in [-1, 1[$ tel que

$$\lim_{n\to\infty}\frac{x_{n+1} - x}{x_n - x} = \rho$$

ce qui constitue clairement une classe plus large de suites par rapport à la première transformation. Des exemples de suites convergentes (x_n) pour lesquelles la suite (t_n) obtenue par le processus d'Aitken a deux points d'accumulation sont connus (voir [36, pp.84])). Mais il peut être prouvé que si (t_n) converge, alors sa limite est la même que celle de (x_n) [124].

Dans l'étude des transformations de suites, la première question à se poser (avant celles sur la convergence et l'accélération) est algébrique : elle concerne le *noyau* de la transformation, c'est-à-dire l'ensemble des suites pour lesquelles il existe x tel que pour tout $n > N$, $t_n = x$. Pour la première transformation, le noyau est l'ensemble des suites de la forme

$$x_n = x + c(-1)^n$$

où c est un scalaire. Pour le processus d'Aitken, le noyau est donné par

$$x_n = x + c\lambda^n$$

où c et λ sont des scalaires avec $c \neq 0$ et $\lambda \neq 1$. Nous constatons que le noyau du processus d'Aitken contient bien celui du processus de sommation. Comme nous pouvons le voir, dans les deux exemples, le noyau dépend de quelques paramètres, x et c dans le cas du processus de sommation et x, c et λ dans le cas du processus Δ^2 d'Aitken.

Si la suite (x_n) à accélérer appartient au noyau de la transformation utilisée, alors par construction, nous avons $t_n = x$ pour tout $n > N$. Bien entendu, habituellement, x est la limite de la suite (x_n), mais ce n'est pas toujours le cas ; la question doit donc être étudiée. Par exemple, dans le processus d'Aitken, x est la limite de la suite (x_n) si $|\lambda| < 1$. Si $|\lambda| > 1$, la suite (x_n) diverge et x est appelée son anti-limite.

Dans les deux exemples, nous sommes capables d'obtenir une forme explicite du noyau. Mais le noyau peut aussi être donné dans une forme implicite au moyen d'une relation qui prend en compte plusieurs termes consécutifs de la suite. Pour le processus de sommation, il est équivalent d'écrire que

$$x_{n+1} - x = -(x_n - x), \quad \forall\ n > N$$

tandis que pour le processus d'Aitken, nous avons

$$x_{n+1} - x = \lambda(x_n - x), \quad \forall\ n > N.$$

De telles équations sont appelées forme implicite du noyau car elles ne donnent pas explicitement la forme des suites appartenant au noyau mais seulement implicitement comme solution de ces équations. Résoudre ces équations, ce qui est facile dans nos deux exemples, mène à la forme explicite du noyau. Bien entendu, les deux formes sont équivalentes.

La forme implicite du noyau d'une transformation est généralement décrite de la façon suivante

$$K(x_n, \ldots, x_{n+q}; x, c_1, \ldots, c_p) = 0$$

ce qui doit être satisfait si et seulement si la suite (x_n) appartient au noyau $\mathcal{K}_T$ de la transformation T. Une transformation de suites $T : x_n \mapsto t_n$ est dite méthode d'extrapolation si elle est telle que $\forall\ n, t_n = x$ si et seulement si $(x_n) \in \mathcal{K}_T$. Expliquons comment une méthode d'extrapolation est construite à partir de son noyau, dont la forme implicite est K. Soient $x_n, x_{n+1}, \ldots, x_{n+p+q}$, et $(u_n) \in \mathcal{K}_T$ une suite satisfaisant les conditions suivantes d'interpolation

$$u_i = x_i, \quad i = n, n+1, \ldots, n+p+q.$$

Comme $(u_n) \in \mathcal{K}_T$, elle satisfait la relation implicite, et nous avons

$$K(u_i, \ldots, u_{i+q}; x, c_1, \ldots, c_p) = 0, \quad i = n, \ldots, n+p.$$

C'est un système de $(p+1)$ équations à $(p+1)$ inconnues, $x, c_1, \ldots, c_p$ dont la solution (si elle existe) dépend de l'indice n. Afin de déterminer la solution de ce système, nous supposons que $\frac{\partial K}{\partial x} \neq 0$. Ceci garantit par le théorème des fonctions implicites, l'existence d'une fonction G (dépendant des paramètres inconnus $c_1, \ldots, c_p$) telle que

$$x = G(x_i, \ldots, x_{i+q}),\ i = n, \ldots, n+p.$$

La solution $t_n = x$ de ce système dépend seulement des termes de la suite initiale, $x_n, \ldots, x_{n+p+q}$. De ce fait, nous obtenons la méthode d'extrapolation suivante qui dépend de n et est notée t_n

$$t_n = F(x_n, \ldots, x_{n+k}).$$

Elle est aussi parfois notée $t_n^{(k)}$ pour signifier qu'elle dépend également de $k = p + q$. Illustrons le développement d'une méthode d'extrapolation sur un exemple. Supposons que le noyau implicite soit de la forme suivante

$$K(u_i, u_{i+1}; x, c_1, c_2) = c_1(u_i - x) + c_2(u_{i+1} - x) = 0$$

où $c_1 + c_2 \neq 0$. Nous pouvons supposer sans perte de généralité que $c_1 + c_2 = 1$. Nous devons alors résoudre le système

$$\begin{aligned} c_1(x_i - x) + c_2(x_{i+1} - x) &= 0 \\ c_1(x_{i+1} - x) + c_2(x_{i+2} - x) &= 0. \end{aligned}$$

Ce système a une solution unique x, puisque la dérivée $\frac{\partial K}{\partial x} = -(c_1 + c_2) = -1$. La fonction G est donnée par

$$G(u_i, u_{i+1}) = c_1 u_i + c_2 u_{i+1}$$

et le système à résoudre devient alors

$$\begin{aligned} t_n &= x = c_1 x_n + (1 - c_1) x_{n+1} \\ t_n &= x = c_1 x_{n+1} + (1 - c_1) x_{n+2}. \end{aligned}$$

En ajoutant et en soustrayant x_n dans la première équation et x_{n+1} dans la seconde équation, il en résulte le système équivalent suivant

$$\begin{aligned} x_n &= t_n + (c_1 - 1)\Delta x_n \\ x_{n+1} &= t_n + (c_1 - 1)\Delta x_{n+1}, \end{aligned}$$

où Δ est l'opérateur différence défini par $\Delta x_i = x_{i+1} - x_i$. La solution pour t_n peut être écrite en utilisant les règles de Cramer comme un rapport de deux déterminants

$$t_n = \frac{\begin{vmatrix} x_n & x_{n+1} \\ \Delta x_n & \Delta x_{n+1} \end{vmatrix}}{\begin{vmatrix} 1 & 1 \\ \Delta x_n & \Delta x_{n+1} \end{vmatrix}}. \tag{5.4}$$

En développant les déterminants et par définition de l'opérateur Δ, nous obtenons

$$\begin{aligned} t_n &= \frac{x_n \Delta x_{n+1} - x_{n+1} \Delta x_n}{\Delta x_{n+1} - \Delta x_n} \\ &= \frac{x_n x_{n+2} - x_{n+1}^2}{x_{n+2} - 2x_{n+1} + x_n}, \end{aligned} \tag{5.5}$$

lequel est le processus Δ^2 d'Aitken.

Considérons maintenant un autre problème plus compliqué. Nous supposons que le noyau implicite K est de la forme

$$K(u_i, \ldots, u_{i+q}; x, c_1, \ldots, c_p) = c_1(u_i - x) + c_2(u_{i+1} - x) + \cdots + c_{p+1}(u_{i+q} - x) = 0$$

où $c_1 c_{p+1} \neq 0$, $\sum_{i=1}^{p+1} c_i = 1$ et $p = q = k$. En utilisant les techniques utilisées auparavant pour la méthode d'Aitken, nous obtenons le système $(k+1) \times (k+1)$ suivant

$$\begin{array}{ccccccccc} t_n & + & b_1\Delta x_n & + & \cdots & + & b_k\Delta x_{n+k-1} & = & x_n \\ t_n & + & b_1\Delta x_{n+1} & + & \cdots & + & b_k\Delta x_{n+k} & = & x_{n+1} \\ & & & & & & & \vdots & \\ t_n & + & b_1\Delta x_{n+k} & + & \cdots & + & b_k\Delta x_{n+2k-1} & = & x_{n+k} \end{array} \tag{5.6}$$

où les variables b_i dépendent de $c_j, j = 1, \ldots, p$. Résoudre ce système en utilisant de nouveau les règles classiques de Cramer, mène à

$$t_n^{(k)} = \frac{\begin{vmatrix} x_n & x_{n+1} & \ldots & x_{n+k} \\ \Delta x_n & \Delta x_{n+1} & \ldots & \Delta x_{n+k} \\ \vdots & \vdots & \ldots & \vdots \\ \Delta x_{n+k-1} & \Delta x_{n+k} & \ldots & \Delta x_{n+2k-1} \end{vmatrix}}{\begin{vmatrix} 1 & 1 & \ldots & 1 \\ \Delta x_n & \Delta x_{n+1} & \ldots & \Delta x_{n+k} \\ \vdots & \vdots & \ldots & \vdots \\ \Delta x_{n+k-1} & \Delta x_{n+k} & \ldots & \Delta x_{n+2k-1} \end{vmatrix}} \tag{5.7}$$

Cela correspond à une transformation bien connue de suites appelée *la transformation de Shanks* [113]. Elle implique le calcul de deux déterminants de dimension $k+1$, et donc elle requiert $2(k+1)(k+1)!$ multiplications. Cela est ainsi prohibitif pour des grandes valeurs de k. De plus, les résultats peuvent être affectés par des erreurs d'arrondis dues à l'utilisation d'un ordinateur. Calculer directement ces déterminants pour obtenir l'itéré $t_n^{(k)}$ n'est donc pas la meilleure solution. Pour résoudre ce problème, des algorithmes récursifs (voir [61, 76]) furent développés afin de calculer le rapport de deux déterminants avec de telles structures. Voir aussi l'ε algorithme de Wynn [129] pour implémenter récursivement la transformation de Shanks.

5.2.2 Transformation Vectorielle et Extrapolation

Examinons le déterminant du numérateur de la transformation de Shanks, Eq. (5.7). Si le déterminant est développé suivant sa première ligne, nous obtenons

$$t_n^{(k)} = \alpha_0 x_n + \ldots + \alpha_k x_{n+k} \tag{5.8}$$

où les α_i sont les solutions du système suivant

$$\begin{array}{ccccccccc} \alpha_0 & + & \alpha_1 & + & \cdots & + & \alpha_k & = & 1 \\ \alpha_0\Delta x_n & + & \alpha_1\Delta x_{n+1} & + & \cdots & + & \alpha_k\Delta x_{n+k} & = & 0 \\ & & & & & & & \vdots & \\ \alpha_0\Delta x_{n+k-1} & + & \alpha_1\Delta x_{n+k} & + & \cdots & + & \alpha_k\Delta x_{n+2k-1} & = & 0. \end{array} \tag{5.9}$$

Exprimer la transformation de Shanks de cette manière facilite son adaptation aux suites vectorielles, c'est-à-dire le cas où $x_n \in I\!R^p$. Dans ce cas, les éléments $\Delta x_n, \ldots, \Delta x_{n+2k-1}$ dans les équations du système (5.9) sont tous des vecteurs, c'est-à-dire pour tout i, $\Delta x_{n+i} \in I\!R^p$. De ce fait, chaque équation (exceptée la première) dans le système (5.9) est une équation vectorielle décrite par un système de p équations. En effectuant le produit scalaire de chaque équation du système (exceptée la première) avec un vecteur $y_i^{(n)}$ ($i \in [1, \ldots, k]$) à préciser, nous obtenons le système suivant

$$\begin{array}{ccccccccc} \alpha_0 & + & \alpha_1 & + & \ldots & + & \alpha_k & = & 1 \\ \alpha_0(y_1^{(n)}, \Delta x_n) & + & \alpha_1(y_1^{(n)}, \Delta x_{n+1}) & + & \ldots & + & \alpha_k(y_1^{(n)}, \Delta x_{n+k}) & = & 0 \\ & & & & & & & \vdots & \\ \alpha_0(y_k^{(n)}, \Delta x_{n+k-1}) & + & \alpha_1(y_k^{(n)}, \Delta x_{n+k}) & + & \ldots & + & \alpha_k(y_k^{(n)}, \Delta x_{n+2k-1}) & = & 0. \end{array} \tag{5.10}$$

Ce système a k+1 équations pour k+1 inconnues. Si le déterminant de ce système est non singulier, sa solution existe et correspond aux paramètres $\alpha_0, \ldots, \alpha_k$ utilisés pour définir la transformation vectorielle via l'équation (5.8). Par analogie à l'équation (5.7), la transformation vectorielle de Shanks peut maintenant être définie comme suit

$$t_n^{(k)} = \frac{\begin{vmatrix} x_n & x_{n+1} & \ldots & x_{n+k} \\ (y_1^{(n)}, \Delta x_n) & (y_1^{(n)}, \Delta x_{n+1}) & \ldots & (y_1^{(n)}, \Delta x_{n+k}) \\ \vdots & \vdots & \ldots & \vdots \\ (y_k^{(n)}, \Delta x_{n+k-1}) & (y_k^{(n)}, \Delta x_{n+k}) & \ldots & (y_k^{(n)}, \Delta x_{n+2k-1}) \end{vmatrix}}{\begin{vmatrix} 1 & 1 & \ldots & 1 \\ (y_1^{(n)}, \Delta x_n) & (y_1^{(n)}, \Delta x_{n+1}) & \ldots & (y_1^{(n)}, \Delta x_{n+k}) \\ \vdots & \vdots & \ldots & \vdots \\ (y_k^{(n)}, \Delta x_{n+k-1}) & (y_k^{(n)}, \Delta x_{n+k}) & \ldots & (y_k^{(n)}, \Delta x_{n+2k-1}) \end{vmatrix}} \tag{5.11}$$

A partir de (5.11), plusieurs méthodes classiques d'extrapolation vectorielles sont retrouvées par le biais du choix des vecteurs $y_i^{(n)}$. En effet, nous avons

- Minimal Polynomial Extrapolation (MPE) de Cabay et Jackson [39] : $y_i^{(n)} = \Delta x_{n+i}$.
- Reduced Rank Extrapolation (RRE) de Mesina [93] et Eddy [55] : $y_i^{(n)} = \Delta^2 x_{n+i}$.
- Topological Epsilon Algorithm (TEA) de Brezinski [34] : $y_i^{(n)} = y$.
- Modified Minimal Polynomial Extrapolation (MMPE) de Sidi [114] : $y_i^{(n)} = y_{i+1}$.

– La Transformation d'Henrici [69] : k=p et $y_i^{(n)} = e_i$, où e_i sont les vecteurs de la base canonique de $\mathbb{R}^p$.

Dans le TEA, $y \in \mathbb{R}^p$ est un vecteur arbitraire fixé, et dans l'algorithme MMPE, $(y_1, \ldots, y_k)$ est une suite de vecteurs linéairement indépendants. Notons Y_k et $\Delta^j X_{k,n} (j = 1, 2)$ les matrices dont les colonnes sont respectivement, $y_1^{(n)}, \ldots, y_k^{(n)}$ et $\Delta^j x_n, \ldots, \Delta^j x_{n+k-1}$. Dans le dénominateur et le numérateur de l'équation (5.11), nous remplaçons chaque colonne, en commençant par la deuxième, par sa différence avec la colonne précédente, et nous obtenons

$$t_n^{(k)} = \frac{\begin{vmatrix} x_n & \Delta X_{k,n} \\ Y_{k,n}^T \Delta x_n & Y_{k,n}^T \Delta^2 X_{k,n} \end{vmatrix}}{\begin{vmatrix} Y_{k,n}^T \Delta^2 X_{k,n} \end{vmatrix}}. \tag{5.12}$$

En utilisant la formule de Schur pour les déterminants [35], nous pouvons exprimer $t_n^{(k)}$ sous la forme matricielle suivante

$$t_n^{(k)} = x_n - \Delta X_{k,n} (Y_{k,n}^T \Delta^2 X_{k,n})^{-1} Y_{k,n}^T \Delta x_n \tag{5.13}$$

où $Y_{k,n} = \Delta X_{k,n}$ pour MPE, et $Y_{k,n} = \Delta^2 X_{k,n}$ pour RRE. La matrice $(Y_{k,n}^T \Delta^2 X_{k,n})$ doit bien entendu être non singulière pour que l'itéré $t_n^{(k)}$ soit défini. Dans [114], Sidi donne des conditions pour lesquelles cela est vérifié. Pour $k \in [1, 2]$, l'itéré d'une méthode d'extrapolation, $t_n^{(k)}$, peut directement être calculé à partir de l'équation (5.13). Pour des valeurs plus élevées de k, le calcul de $t_n^{(k)}$ peut se faire en utilisant des algorithmes récursifs, voir par exemple [61] et [76].

5.2.3 Résolution des systèmes d'équations par extrapolation

Considérons le problème linéaire de point fixe suivant

$$x = Ax + b \tag{5.14}$$

où $x \in \mathbb{R}^p$ et A est une matrice supposée diagonalisable de dimension p. Soient $\lambda_1, \ldots, \lambda_p$ les valeurs propres et $v_1, \ldots, v_p$ les vecteurs propres correspondants de la matrice A. Supposons que $\lambda_i \neq 1$, pour tout i. Ainsi, le problème (5.14) a une solution unique x^*. Pour un vecteur x_0 donné, nous définissons la suite (x_n) de la façon suivante

$$x_{n+1} = Ax_n + b \qquad n = 0, 1, \ldots \tag{5.15}$$

Comme A est supposée diagonalisable, tout vecteur de $\mathbb{R}^p$ s'exprime dans la base de vecteurs propres v_i. Par suite, posons $x_0 - x^* = \sum_{i=1}^p \rho_i v_i$, où ρ_i sont des scalaires. Nous avons alors

$$x_n = x^* + \sum_{i=1}^{p} \rho_i v_i \lambda_i^n \quad n = 0, 1, \ldots \tag{5.16}$$

Si le module de la plus grande valeur propre de la matrice A est inférieure à l'unité, c'est-à-dire $|\lambda_1| < 1$, alors la limite de la suite (x_n) existe et est simplement x^*. Sinon (si

$|\lambda_1| \geq 1$) la limite n'existe pas, et le vecteur x^* est appelé l'anti-limite de la suite (x_n). Sidi [115] a prouvé le lemme suivant pour établir que les méthodes d'extrapolation données par l'équation (5.13), en particulier les méthodes RRE et MPE, sont des procédures d'accélération de la méthode itérative définie par l'équation (5.15) pour résoudre le problème (5.14).

Lemme 3.
Si A est diagonalisable et si ses valeurs propres distinctes de zéro notées λ_j, $j = 1, 2, \ldots, p$, sont ordonnées de la façon suivante

$$|\lambda_1| \geq |\lambda_2| \geq |\lambda_3|, \ldots$$

et si

$$|\lambda_k| > |\lambda_{k+1}|,$$

alors les méthodes MPE et RRE accélérent la convergence de la méthode itérative donnée par l'équation (5.15), c'est-à-dire

$$\frac{||\, t_n^{(k)} - x^* \,||}{||\, x_{n+k+1} - x^* \,||} = O\left[\left(\frac{\lambda_{k+1}}{\lambda_1}\right)^n\right] \quad \textit{quand} \quad n \to \infty. \tag{5.17}$$

Ainsi, si $x_n \to x^*$, c'est-à-dire $|\lambda_1| < 1$, alors $t_n^{(k)} \to x^*$ plus vite. De plus, si $\lim_{n\to\infty} x_n$ n'existe pas, c'est-à-dire $|\lambda_1| > 1$, alors $t_n^{(k)} \to x^*$, à condition que $|\lambda_{k+1}| < 1$. La raison pour laquelle le terme x_{n+k+1} intervient dans l'équation (5.17) est que l'itéré $t_n^{(k)}$ des méthodes d'extrapolation MPE et RRE est calculé à partir des termes $x_n, x_{n+1}, \ldots, x_{n+k+1}$. Le lemme implique aussi que les méthodes RRE et MPE sont particulièrement efficaces pour accélérer une suite générée par l'équation (5.15) quand la matrice d'itération A a un spectre tel que les valeurs propres dominantes soient en nombre restreint (k au plus quand $t_n^{(k)}$ est utilisé) et les autres valeurs propres soient nettement plus petites. Précisons qu'un résultat analogue est vrai avec l'ϵ-algorithme, voir [38].

Si la suite (x_n) est générée par l'algorithme EM, $x_{n+1} = F(x_n)$ (voir Chap. 4, Eq. 4.5) avec une fonction F non linéaire, le lemme (5.17) semble aussi asymptotiquement vrai. En effet, pour n suffisamment large, les itérés x_n, $x_{n+1}, \ldots$ de l'algorithme EM sont dans un voisinage du point fixe x^* de la fonction F, et par un développement de Taylor, nous avons

$$x_{n+1} - x^* = J(x^*)(x_n - x^*) + O(||\, x_n - x^* \,||^2) \tag{5.18}$$

où $J(x^*)$ est la matrice jacobienne de la fonction non linéaire F évaluée au point fixe x^*. Cela implique que la suite (x_n) se comporte de façon linéaire à l'infini dans le sens où

$$x_{n+1} \approx J(x^*)x_n + (I_p - J(x^*))x^* \tag{5.19}$$

pour n suffisamment large. Par identification, nous avons $A = J(x^*)$ la matrice de linéarisation et $b = (I_p - J(x^*))x^*$ le vecteur second membre. En pratique, nous ne connaissons pas A et b, mais seulement les vecteurs x_n. Cela n'est pas un problème, puisque les méthodes itératives proposées ne nécessitent pas la connaissance explicite de A et de b.

5.2.4 Comment cycler des Méthodes d'Extrapolation ?

Discutons maintenant de la stratégie dîte de cyclage qui utilise les avantages des méthodes d'extrapolation données par l'équation (5.13) et que nous utiliserons pour résoudre le problème de l'accélération de convergence de l'algorithme EM. Précisons que les méthodes itératives proposées ici, basées sur cette stratégie peuvent être utilisées plus généralement pour accélérer les itérations de Picard pour déterminer le ou les points fixes d'une fonction F. Pour la suite du chapitre, il est important de distinguer les termes cycles et itérations. Une itération de base signifie une simple itération du schéma de base à accélérer, par exemple la méthode de Picard ou l'algorithme EM, tandis qu'un cycle désigne l'application d'une méthode d'extrapolation utilisant plusieurs itérations de base et repartant du vecteur donné par l'extrapolation. La stratégie de cyclage consiste au début de chaque cycle, à partir d'un vecteur initial, à calculer quelques itérations de base et ensuite à appliquer un schéma d'extrapolation de la forme (5.13) à ces itérations pour obtenir un vecteur qui sera le vecteur initial au prochain cycle, et ainsi de suite. Précisons que la méthode d'extrapolation utilisée est la même pour tous les cycles. Nous obtenons alors le schéma itératif suivant en cyclant une méthode d'extrapolation de la forme (5.13)

1. Soit x_n le vecteur initial au début du $(n+1)^{\text{ième}}$ cycle, et soit $u_0 = x_n$.
2. Appliquons le schéma de base k fois pour obtenir les itérations de base $u_1, \ldots, u_k$, où
$$u_{i+1} = F(u_i) \qquad i = 0, \ldots, k-1.$$
3. Appliquons le schéma d'extrapolation donné par l'équation (5.13) à la suite $u_0, \ldots, u_k$ pour obtenir $t_n^{(k)}$.
4. Posons $x_{n+1} = t_n^{(k)}$, et vérifions le critère de convergence.
5. Si le critère est satisfait, nous nous arrêtons, sinon nous revenons à l'étape 1 pour un autre cycle.

L'idée de cycler est la voie la plus naturelle et efficace qui utilise les méthodes d'extrapolation. Donnons quelques raisons de cycler des méthodes d'extrapolation [117, section 6]

1. Cycler une méthode d'extrapolation d'ordre k tel que la MPE ou RRE avec m cycles est comparable, mais pas équivalent, à l'utilisation d'une méthode d'extrapolation sans cycle qui calcule un itéré $t_0^{(mk)}$, obtenu à partir de $x_0, x_1, \ldots, x_{mk}$.
2. Cycler requiert m fois moins de stockage qu'une pure extrapolation, puisqu'il utilise seulement $k+1$ vecteurs à chaque cycle tandis qu'une pure extrapolation nécessite le stockage des $m(k+1)$ vecteurs.
3. En cyclant nous pouvons contrôler l'erreur d'extrapolation à chaque cycle et modifier le nombre k d'itérations de base tandis que dans une extrapolation pure, le nombre de termes est fixé à priori.

Un inconvénient mineur avec les cycles est que les itérations de base que nous calculons ne peuvent être utilisées qu'une seule fois, en particulier pour calculer l'itéré x_{n+1} du $(n+1)^{\text{ième}}$ cycle. Par contre, dans une extrapolation pure (sans cycle), les itérations de base sont calculées une fois puis stockées et enfin elles peuvent être utilisées avec n'importe

quel schéma d'extrapolation. Bien entendu, le stockage des vecteurs est prohibitif si la dimension des vecteurs est élévée. Rappelons quelques définitions utiles pour la suite [81].

Définition 1.
Le polynôme caractéristique d'une matrice carrée T est $det(T - \lambda I)$. Son degré est p, la dimension de T. Nous avons $P(T) = 0$ (Théorème d'Hamilton-Cayley). Le polynôme minimal d'une matrice T est le polynôme P de plus petit degré tel que $P(T) = 0$. Il divise le polynôme caractéristique de T. Le polynôme minimal d'une matrice T pour un vecteur u est le polynôme Q de plus petit degré tel que $Q(T)u = 0$. Q divise le polynôme minimal.

Sous les conditions suivantes

1. La dérivée au sens de Fréchet dans Ω de la fonction F est continue
2. La matrice jacobienne $J(x^*)$ évaluée au point fixe x^* n'admet pas l'unité comme valeur propre
3. Le paramètre k est choisi pour le $(n+1)^{\text{ième}}$ cycle comme étant le degré du polynôme minimal de $J(x^*)$ pour le vecteur $x_n - x^*$
4. x_0 est suffisamment proche du point fixe x^*,

Smith et al [118] ont montré que la méthode itérative décrite à la page 69 et obtenue en cyclant une méthode d'extrapolation, est quadratiquement convergente dans le sens où

$$|| x_{n+1} - x^* || = O(|| x_n - x^* ||^2).$$

Voir aussi [84] pour un résultat sur la convergence quadratique du Topological Epsilon algorithm (TEA). Un exemple classique qui illustre cette stratégie de cyclage est la connection entre la méthode Δ^2 d'Aitken et le schéma itératif de Steffensen pour la résolution de problème de point fixe dans le cas scalaire. Rappelons que la méthode Δ^2 d'Aitken, étudiée dans la section 5.2.1, est définie par

$$t_n = x_n - \frac{(x_{n+1} - x_n)^2}{x_{n+2} - 2x_{n+1} + x_n}.$$

D'autre part, la méthode de Steffensen est un schéma itératif qui peut être explicitement défini par

$$x_{n+1} = x_n - \frac{(F(x_n) - x_n)^2}{F(F(x_n)) - 2F(x_n) + x_n}. \tag{5.20}$$

Ce schéma peut être obtenu à partir du processus Δ^2 d'Aitken, en utilisant la stratégie de cyclage, de la façon suivante

1. Soit x_n le vecteur initial au début du $(n+1)^{\text{ième}}$. Posons $u_0 = x_n$.
2. Appliquons le schéma de Picard deux fois pour obtenir
   $u_1 = F(u_0)$ et $u_2 = F(u_1)$.
3. Appliquons le Δ^2 d'Aitken, Equation (5.5), à u_0, u_1 et u_2 pour obtenir x_{n+1}.
4. Incrémenter le nombre de cycles de 1, et répéter les étapes (1)-(3) jusqu'à convergence.

Bien que les itérations de base, précisement les itérations de Picard (Eq. (5.1)), soient linéairement convergentes, la méthode de Steffensen produit une suite qui converge quadratiquement vers le point fixe x^* de la fonction F, à condition que $F'(x) \neq 1$ et que le vecteur initial x_0 est suffisamment proche du point fixe x^*. Ce résultat ne devrait pas être surprenant puisque la méthode de Steffensen est très proche de la méthode de Newton-Raphson pour résoudre l'équation non linéaire, $f(x) = F(x) - x = 0$. En effet, la méthode de Newton-Raphson utilise la dérivée de $f(x)$, $f'(x) = F'(x) - 1$ tandis que la méthode de Steffensen utilise une approximation de la dérivée de F par la méthode de la sécante, c'est-à-dire

$$\begin{aligned} F'(x_n) &= (F(x_{n+1}) - F(x_n))/(x_{n+1} - x_n) \\ &= (F(x_{n+1}) - F(x_n))/(F(x_n) - x_n). \end{aligned}$$

Nous remarquons que la fonction F doit être contractive afin que les itérations de base convergent, mais cette condition n'est pas nécessaire pour avoir la convergence de la méthode de Steffensen. Ce fut aussi observé par Henrici [69] pour un schéma qu'il a proposé comme une extension de la méthode de Steffensen au cas vectoriel. En effet, Henrici a donné un exemple (dans $\mathbb{R}^2$) où cette extension de la méthode de Steffensen converge tandis que la fonction F est telle que non seulement $|| J(x^*) || > 1$, mais aussi $| \lambda_{min}(J(x^*)) | > 1$. Précisons qu'il fut prouvé plus tard par Nievergelt [95] que des conditions sur les dérivées partielles secondes de F et la coïncidence du polynôme caractéristique et du polynôme minimal de la jacobienne de F au point x^* (ce qui est trivialement vrai dans le cas scalaire) sont suffisantes pour garantir la stabilité et la convergence du schéma proposé par Henrici et appelé la *multivariate Steffensen's method*.

La méthode d'Henrici peut être aussi retrouvée, via le cyclage, à partir des méthodes d'extrapolation, Eq. (5.13), en posant $k = p$ et $y_i = e_i$, où les vecteurs e_i sont les vecteurs de la base canonique de $\mathbb{R}^p$. Cette méthode a aussi été considérée par Louis [86] sous le nom de *the multivariate Aitken's method* et elle fut implémentée dans [80]. Un algorithme récursif pour la mettre en oeuvre, le H-algorithme, a été donné par Sadok [107], voir aussi [26, p. 238]. Comme Smith et al [118] l'ont démontré, une extension plus naturelle de la méthode Δ^2 d'Aitken pour le cas vectoriel, est obtenue en cyclant la méthode d'extrapolation RRE (p. 66) avec le paramètre $k = k^* \leq p$, où k^* est le degré du polynôme minimal de $J(x^*)$ pour le vecteur $x_n - x^*$, où $J(x^*)$ désigne la matrice jacobienne de F évaluée au point fixe x^*. Rappelons que la méthode d'extrapolation RRE (Eq. (5.13)) avec le choix $y_i^{(n)} = \Delta^2 x_{n+i}$, est définie par

$$t_n^{(k)} = x_n - \Delta X_{n,k} (\Delta^2 X_{k,n})^\dagger \Delta x_{k,n} \tag{5.21}$$

où $A^\dagger = (A^T A)^{-1} A^T$ est l'inverse généralisé de Moore-Penrose ou le pseudo-inverse de la matrice A. Quand A est une matrice carrée non singulière, alors $A^\dagger = A^{-1}$, dans quel cas la méthode RRE avec $k = p$ coïncide avec la méthode d'Henrici.

Le degré du polynôme minimal est plus petit que p la dimension des vecteurs. De ce fait, compte tenu du résultat de convergence de Smith et al [118] énoncé précédemment, la méthode d'Henrici pourrait s'avérer moins efficace que la méthode RRE. De plus, pour des problèmes avec un très grand nombre d'inconnues, il peut être prohibitif d'utiliser la

méthode d'Henrici. Néanmoins, les méthodes RRE et MPE ont quelques inconvénients : (1) nous ne connaissons pas le degré du polynôme minimal puisque nous ne connaissons pas $J(x^*)$ et (2) le degré dépend du vecteur $x_n - x^*$ et de ce fait, il peut varier à chaque cycle. Précisons qu'une solution basée sur l'évolution de la norme du résidu $|| x - F(x) ||$ pour un certain $x \in \mathbb{R}^p$, a été proposée pour ce problème dans [118]. De plus, même si des valeurs pour le paramètre k sont choisies correctement, en particulier $k \leq k^*$, ces méthodes sont assujetties à des problèmes numériques tels que la stagnation, dans le cas de la méthode RRE, et la division par zéro dans le cas de la méthode MPE. Une discussion sur ces problèmes est présentée dans la section 5.4.2. Les méthodes RRE et MPE peuvent aussi être utilisées avec des valeurs k plus petites que k^*. Même une valeur comme $k = 1$ peut donner des résultats satisfaisants. Dans ce chapitre, nous nous intéressons aux méthodes RRE et MPE avec un ordre $k = 1$ pour deux raisons : (1) leur simplicité et (2) leur coût de calcul relativement faible. Une nouvelle technique, appelée *squaring* appliquée à ces méthodes d'ordre 1, est donnée afin d'obtenir une nouvelle classe de méthodes appelées squarem. Nous démontrerons que les méthodes squarem de premier ordre ($k = 1$) peuvent être considérées comme une amélioration importante des méthodes RRE et MPE d'ordre 1, puisqu'elles nécessitent des efforts supplémentaires négligeables de calcul et convergent plus rapidement. Bien que nous n'espérons pas la convergence quadratique de ce type de méthodes, il sera démontré, pour divers problèmes, que la convergence des méthodes squarem est plus rapide que les méthodes RRE et MPE d'ordre 1 et que l'algorithme EM. Par contre, les méthodes squarem d'ordre plus élevé impliquent des efforts de calcul plus importants et pourraient s'avérer moins efficaces. L'évaluation numérique de la performance de ces méthodes d'ordre supérieur pour accélérer l'algorithme EM fera l'objet d'études futures.

5.3 Méthodes itératives à un pas

Pour résoudre le problème de point fixe

$$f(x) = x - F(x) = 0,$$

considérons les schémas de la forme suivante

$$x_{n+1} = x_n - A_n(F(x_n) - x_n) \tag{5.22}$$

où A_n est une matrice de dimension $p \times p$. A partir de cette représentation très générale, nous pouvons à la fois retrouver des schémas numériques existants mais aussi en obtenir de nouveaux, par le biais du choix de la matrice A_n. Remarquons que les schémas d'extrapolation donnés par l'équation (5.13), via la stratégie de cyclage, sont de la forme (5.22) avec le choix

$$A_n = \Delta X_{k,n}(Y_{k,n}^T \Delta^2 X_{k,n})^{-1} Y_{k,n}^T.$$

Brezinski [37] a donné une classification des différentes méthodes de Quasi-Newton basée sur différentes stratégies pour le choix de A_n. Les trois stratégies principales sont (1) A_n est une matrice pleine, (2) A_n est une matrice diagonale, et (3) A_n est une matrice

scalaire. Le cas où A_n est considérée comme une matrice pleine approchant $M(x^*) = I - F'(x^*)$, la jacobienne de $f(x)$ au point fixe x^*, mène aux méthodes de Quasi-Newton pour l'accélération de l'algorithme EM. Par exemple, le schéma donné par Louis [86] est retrouvé quand

$$A_n = I_{obs}(x_n; y)^{-1} I_{comp}(x_n; y)$$

où I_{obs} et I_{comp} sont données respectivement par l'équation (4.8) et (4.9) du chapitre 4. Le cas où A_n est une matrice diagonale correspond à utiliser un paramètre différent, α_n^j, $j = 1, \ldots, p$, pour chaque composante du vecteur $f(x_n)$. Voir par exemple [32] où Brezinski et Chehab ont développé des schémas itératifs multiparamètres pour trouver les solutions de systèmes linéaires et non linéaires.

Quand A_n est une matrice scalaire de la forme $\alpha_n I_p$, où I_p est la matrice identité de taille p, nous retrouvons les algorithmes classiques dits de gradient, qui incluent la méthode de la plus profonde descente

$$x_{n+1} = x_n - \alpha_n (F(x_n) - x_n). \tag{5.23}$$

Quand $\alpha_n = -1$, pour tout n, nous obtenons l'algorithme EM, et plus généralement, quand $\alpha_n = \alpha$, nous obtenons les méthodes de relaxation, où le vecteur $F(x_n) - x_n$ est relaxé par le facteur $-\alpha$. Ces méthodes peuvent être aussi considérées comme des schémas obtenus en cyclant une méthode d'extrapolation, avec une transformation de suite définie par

$$t_n = (1 - \alpha) x_n + \alpha x_{n+1}.$$

Cependant, ces méthodes avec α_n constant sont différentes des méthodes obtenues en cyclant des méthodes d'extrapolation données par l'équation (5.13) (avec $k = 1$), puisque ces dernières sont non linéaires. En effet, le coefficient α_n dépend non linéairement de plusieurs paramètres. Ainsi, les schémas avec un pas de relaxation constant (par conséquent linéaire) sont invariants par une transformation de paramètres, tandis que ce n'est pas le cas pour les schémas d'extrapolation. Ceci est un point important puisque dans de nombreux problèmes où nous devons déterminer des estimateurs du maximum de vraisemblance, il est possible d'accélérer les schémas itératifs non linéaires par une transformation appropriée de paramètres. Nous démontrerons ceci dans les exemples numériques.

Dans l'équation (5.23), la quantité $F(x_n) - x_n$ est la direction de recherche ou la direction du gradient et le scalaire α_n est le pas de descente. Dans le contexte de l'algorithme EM, $F(x_n) - x_n$ peut être interprété comme un gradient généralisé, puisqu'il peut-être montré que

$$F(x_n) - x_n \approx -\left(\partial^2 Q(x; x_n)/\partial x \partial x^T\right)^{-1}_{x=x_n} (\partial Q(x; x_n)/\partial x)_{x=x_n}$$

où la quantité $Q(x; x_n)$ est définie par l'équation (4.3) du chapitre 4. En effet, ce résultat constitue la base de l'algorithme du gradient EM discuté par Lange [83, Eq. 6]

$$\begin{aligned} x_{n+1} &= F(x_n) \\ &= x_n + (F(x_n) - x_n) \\ &\approx x_n - \left(\partial^2 Q(x; x_n)/\partial x \partial x^T\right)^{-1}_{x=x_n} (\partial Q(x; x_n)/\partial x)_{x=x_n}. \end{aligned}$$

Cet algorithme du gradient est potentiellement utile dans les situations où le maximum de la fonction Q a été calculé itérativement, puisqu'il évite l'étape M de l'algorithme EM (Eq. (4.4)) simplement en calculant une itération de Newton. De plus, il préserve les propriétés de stabilité et de convergence de l'algorithme EM. Toutefois, à la différence de l'algorithme EM, la propriété de croissance de la fonction de densité de probabilité à chaque itération n n'est pas toujours vérifiée, puisque la matrice $\left(\partial^2 Q(x;x_n)/\partial x \partial x^T\right)_{x=x_n}$ n'est pas nécessairement définie négative, excepté dans les problèmes où les données complètes proviennent d'une distribution appartenant à une famille exponentielle linéaire.

Les méthodes d'extrapolation les plus simples sont obtenues à partir de l'équation (5.13) en posant k=1. Dans ce cas, les matrices $\Delta^i X_{n,k} (i = 1, 2)$ ont seulement une colonne $\Delta^i x_n$. En appliquant la stratégie de cyclage à ces méthodes d'extrapolation d'ordre 1, nous obtenons des méthodes itératives de la forme (5.22) où A_n est une matrice scalaire. En particulier pour les méthodes d'extrapolation RRE et MPE d'ordre 1, nous obtenons les procédures itératives suivantes, que nous appelons MPE1 et RRE1 :
MPE1 :

$$\begin{aligned} x_{n+1} &= x_n - \frac{||\, r_n \,||^2}{(r_n, v_n)} r_n \\ &:= x_n - \alpha_n^{MPE1} r_n \end{aligned} \tag{5.24}$$

RRE1 :

$$\begin{aligned} x_{n+1} &= x_n - \frac{(r_n, v_n)}{||\, v_n \,||^2} r_n \\ &:= x_n - \alpha_n^{RRE1} r_n \end{aligned} \tag{5.25}$$

où $r_n = F(x_n) - x_n$ et $v_n = F(F(x_n)) - 2F(x_n) + x_n$.

Le schéma RRE1 (Eq. (5.25)) est aussi connu sous le nom de la méthode de Lemaréchal [85]. Dans [37], elle est présentée de la manière suivante :
Soit une méthode itérative de la forme (5.23). Posons $u_0 = x_n$ et $u_{i+1} = F(u_i)$, $i = 0, 1, \ldots$. Définissons $z_0 = x_{n+1}$ et $z_i = u_i - \alpha_n (u_{i+1} - u_i)$. Nous considérons les vecteurs

$$\Delta z_i = \Delta u_i - \alpha_n \Delta^2 u_i,$$

où Δ^k est l'opérateur différence défini par $\Delta^k u_i = \Delta^{k-1} u_{i+1} - \Delta^{k-1} u_i$ avec $\Delta^0 u_i = u_i$. Pour un vecteur arbitraire $y \in I\!R^p$, nous choisissons α_n tel que $(y, \Delta z_0) = 0$ et nous obtenons

$$\alpha_n = \frac{(y, \Delta u_0)}{(y, \Delta^2 u_0)}.$$

Il peut être montré que le choix $y = \Delta u_0$ minimise $||\, \Delta z_0 \,||^2$ et mène à la méthode de Lemaréchal ou la RRE1. Similairement, le choix $y = \Delta u_0$, qui mène au pas de descente du

schéma MPE1, minimise $|| \Delta z_0 ||^2/\alpha_n^2$. Selon Brezinski [37], le schéma MPE1 de l'équation (5.24) définit une nouvelle méthode itérative.

Il doit être remarqué que les procédures itératives présentées ci-dessus avec un pas de descente nécessitent des efforts de calcul négligeables par rapport à ceux de l'algorithme EM. Elles requièrent seulement le calcul de deux vecteurs (r_n et v_n), deux produits scalaires, une multiplication scalaire-vecteur ainsi qu'une somme de vecteurs, très faciles à programmer.

5.4 Les Méthodes SQUAREM

Ici, nous proposons une nouvelle classe de méthodes basées sur l'idée d'appliquer des schémas d'extrapolation décrits dans la section 5.3, en deux étapes. Dans la première étape, le schéma d'extrapolation avec un paramètre de descente α_n est appliqué une fois et il en résulte un vecteur dit intermédiaire. Puis le schéma d'extrapolation est appliqué une seconde fois avec le même paramètre α_n, mais cette fois-ci à partir du vecteur intermédiaire. Cette idée fut d'abord proposée dans [100] pour améliorer la performance de la méthode classique de Cauchy [40], généralement appelée la méthode de la plus profonde descente et utilisée afin de minimiser une fonctionnelle quadratique. Elle fut ensuite utilisée par Roland et Varadhan [104] pour accélérer les itérations de Picard afin de résoudre un problème non linéaire de point fixe. En particulier, Roland et Varadhan ont développé les versions dites squarem des schémas de Lemaréchal [85] et de Marder-Weitzner [88]. Ici, nous développons davantage cette idée pour donner une classe plus large de schémas itératifs afin de résoudre le problème non linéaire de point fixe généré par l'algorithme EM. En particulier, nous nous focalisons sur les schémas de premier ordre ($k = 1$). Nous proposons aussi un schéma itératif hybride, pour les méthodes d'extrapolation de premier ordre, lequel combine les propriétés agréables des schémas MPE et RRE, en évitant leurs problèmes de stagnation et de division par zéro.

5.4.1 La méthode de Cauchy-Barzilai-Borwein

Considérons le problème suivant

$$\text{Minimiser } f(x) = \frac{1}{2}x^T Q x - b^T x \qquad \text{pour } x \in \mathbb{R}^p \tag{5.26}$$

où Q est une matrice symétrique définie positive de dimension p. Ce problème est équivalent à résoudre le système linéaire $Qx = b$. Une des méthodes possibles pour résoudre ce système est la méthode de la plus profonde descente [40] définie par

$$x_{n+1} = x_n - \lambda_n g_n$$

où $g_n = \nabla f(x_n) = Qx_n - b$, et le choix dit optimal du pas de descente est donné par

$$\lambda_n = \frac{g_n^T g_n}{g_n^T Q g_n}.$$

Pour ce choix optimal, la méthode de la plus profonde descente converge linéairement. En effet, il peut-être montré que

$$\|x_{n+1} - x^*\|_Q \leq \frac{\lambda_{max} - \lambda_{min}}{\lambda_{max} + \lambda_{min}} \|x_n - x^*\|_Q$$

où pour tout $z \in \mathbb{R}^p$, la Q-norme est définie par $\|z\|_Q^2 = z^T Q z$, λ_{max} et λ_{min} sont respectivement la plus grande et la plus petite valeur propre de la matrice Q et x^* est la solution du problème (5.26).

Il est alors clair que la méthode de la plus profonde descente peut être inédaquate si la valeur propre λ_{min} est très petite par rapport à la valeur propre λ_{max}, c'est-à-dire le problème est mal conditionné. Raydan et Svaiter [100] ont montré que la convergence lente de la méthode de la plus profonde descente, même dans des problèmes moyennement mal conditionnés, est due au choix optimal du pas de descente, et non au choix de la direction de descente g_n. Considérons un schéma relaxé de Cauchy de la forme suivante

$$x_{n+1} = x_n - \theta_n \lambda_n g_n$$

où $0 \leq \theta_n \leq 2$ est un paramètre de relaxation. Remarquons que $\theta_n = 1$ (correspondant à aucune relaxation du pas de descente) est la méthode de Cauchy elle-même. Raydan et Svaiter ont démontré que relaxer le pas de descente de cette façon améliore la performance de la méthode de Cauchy, sauf si la direction de descente g_n est un vecteur propre de la matrice Q, auquel cas le pas de descente de Cauchy donne la solution en une itération. Notons que cette situation est rare en pratique. En choisissant, à chaque itération, le paramètre de relaxation aléatoirement dans l'intervalle $(0, 2)$, Raydan et Svaiter ont montré que la méthode de Cauchy, relaxée aléatoirement, améliore significativement la méthode classique de Cauchy.

Pour résoudre le problème (5.26), Barzilai et Borwein [11] ont proposé la méthode définie de la façon suivante

$$x_{n+1} = x_n - \lambda_{n-1} g_n$$

où λ_{n-1} est le choix optimal du pas de descente (pas de descente de Cauchy) à l'itération précédente et $\lambda_{-1} = 1$. La convergence de cette méthode est étudiée dans [99], pour plus de détails voir aussi [98]. Récemment, Raydan et Svaiter [100] ont combiné la méthode de la plus profonde descente et la méthode de Barzilai-Borwein pour obtenir une nouvelle méthode plus performante. Elle s'appelle la méthode de Cauchy-Barzilai-Borwein notée CBB. Précisons que cette méthode appartient à une famille générale de méthodes à gradient avec retard introduites dans [65]. Elle est définie par l'algorithme suivant
Soit $x_0 \in \mathbb{R}^p$. A chaque itération n, nous posons

$$\begin{aligned} h_n &= Q g_n, \\ t_n &= \frac{g_n^T g_n}{g_n^T h_n}, \\ z_n &= x_n - t_n g(x_n), \\ x_{n+1} = z_n - t_n g(z_n) &= z_n - t_n (Q z_n - b). \end{aligned}$$

Puisque

$$Qz_n - b = Q(x_n - t_n g_n) - b = g_n - t_n h_n,$$

nous obtenons

$$\begin{aligned} x_{n+1} &= z_n - t_n(g_n - t_n h_n) \\ &= x_n - 2\,t_n g_n + t_n^2 h_n. \end{aligned} \tag{5.27}$$

5.4.2 Description des Nouveaux Schémas

Il est facile d'obtenir les équations d'erreur pour les méthodes de Cauchy et de CBB

$$\begin{aligned} e_{n+1} &= (I - t_n Q)^2 e_n \text{ pour CBB} \quad \text{et} \\ e_{n+1} &= (I - t_n Q) e_n \text{ pour Cauchy,} \end{aligned} \tag{5.28}$$

où $e_n = x_n - x^*$. De ce fait, nous pouvons considérer la méthode CBB comme une méthode de Cauchy au carré. Grâce à cette remarque, nous allons maintenant élargir l'idée de *squaring* aux problèmes non linéaires de point fixe. Ceci, bien entendu, signifie que l'idée de *squaring* est aussi applicable à l'algorithme EM [52] et à ses extensions [89] telles que GEM, ECM, et ECME, puisqu'elles sont toutes des itérations non linéaires de point fixe. Pour se faire, nous appliquons d'abord un schéma d'extrapolation d'ordre 1 ($k = 1$) de la forme (5.13) à l'itéré x_n pour obtenir un vecteur intermédiaire z_n. Puis nous appliquons encore une fois la méthode d'extrapolation, mais cette fois-ci au vecteur intermédiaire z_n, pour obtenir un vecteur x_{n+1} qui sera le vecteur initial au prochain cycle. Remarquons que nous pouvons bien entendu utiliser une méthode différente d'extrapolation à la seconde étape, ce qui donnera des schémas mixés. Des études sur ce sujet sont en cours avec des applications en génétique [78]. Le schéma résultant peut être représenté par

$$\begin{aligned} z_n &= x_n - \alpha_n \Delta x_n \\ x_{n+1} &= z_n - \alpha_n \Delta z_n \\ &= x_n - 2\alpha_n \Delta x_n + {\alpha_n}^2 \Delta^2 x_n \\ &= x_n - 2\alpha_n r_n + {\alpha_n}^2 v_n \end{aligned} \tag{5.29}$$

où $r_n = \Delta x_n = F(x_n) - x_n$ et $v_n = \Delta^2 x_n = F(F(x_n)) - 2F(x_n) + x_n$. Ainsi, pour les nouveaux schémas, nous avons l'équation suivante pour la propagation de l'erreur (voir Chapitre 3, Section 3.2.2)

$$\varepsilon_{n+1} = [I_p - \alpha_n(\psi - I_p)]^2 \varepsilon_n + o(\varepsilon_n) \tag{5.30}$$

où $\varepsilon_n = x_n - x^*$ est l'erreur du schéma à la $n^{\text{ième}}$ itération et ψ est la jacobienne de F au point fixe x^*. Par contre, pour les méthodes définies par (5.23) sans *squaring*, nous avons

$$\varepsilon_{n+1} = [I_p - \alpha_n(\psi - I_p)]\varepsilon_n + o(\varepsilon_n). \tag{5.31}$$

Les nouvelles méthodes peuvent ainsi, via la remarque (5.28), être considérées comme les méthodes de type (5.23) élevées au carré, d'où la désignation méthodes squarem.

Si la méthode d'extrapolation d'ordre 1 utilisée est la MPE (voir p. 66), le paramètre α_n est donné par

$$\alpha_n^{MPE1} = \frac{(r_n, r_n)}{(v_n, r_n)}. \tag{5.32}$$

Nous appelons la nouvelle méthode définie par (5.29) avec le choix de α_n donné par (5.32), la méthode SqMPE1. Elle correspond à la méthode MPE1 (Eq. (5.24)) au carré.

Similairement, si la méthode d'extrapolation d'ordre 1 utilisée est la RRE (voir p. 66), le paramètre α_n est donné par

$$\alpha_n^{RRE1} = \frac{(v_n, r_n)}{(v_n, v_n)}. \tag{5.33}$$

Nous appelons la nouvelle méthode définie par (5.29) avec le choix de α_n donné par (5.33), la méthode SqRRE1. Elle correspond à la méthode RRE1 (Eq. (5.25)) au carré.

Pour les nouveaux schémas, le pas de descente α_n est calculé une fois, mais utilisé deux fois. Deux évaluations d'itérations de base sont calculées à chaque cycle, comme pour les schémas MPE1 et RRE1. Les méthodes squarem requièrent seulement un produit scalaire-vecteur et une addition de vecteurs supplémentaires par rapport aux schémas d'ordre 1, MPE1 et RRE1. Le coût supplémentaire de calculs est ainsi négligeable. De plus, aucun stockage supplémentaire de vecteurs n'est nécessaire par rapport aux méthodes d'ordre 1. Si, pour un certain n, les vecteurs r_n et v_n sont pratiquement orthogonaux, c'est-à-dire $|(v_n, r_n)| \leq \varepsilon$, où la valeur de $\varepsilon > 0$ est très petite mais ni la norme de r_n, ni la norme de v_n ne le sont, alors les schémas sont sujets à des problèmes numériques de stabilité. Les schémas MPE1 et SqMPE1 deviennent instables à cause d'une valeur très grande pour le paramètre de relaxation α_n^{MPE1}. Cette situation porte le nom de *near breakdown* et dans la suite, nous parlerons plutôt de division par zéro. Par contre, les méthodes RRE1 et SqRRE1 subissent une *stagnation*. En effet, le paramètre α_n^{RRE1} devient presque nul, et ainsi $x_{n+1} \approx x_n$. Une voie naturelle pour résoudre, dans un premier temps, ce problème lié soit à la stagnation soit à la division par zéro est de ne pas effectuer l'étape de la méthode squarem, mais d'utiliser à la place les deux itérés, $F(x_n)$ et $F(F(x_n))$, de l'algorithme EM calculés dans chaque cycle. Ainsi, si à l'itération n, nous avons $|\langle v_n, r_n \rangle| \leq \varepsilon$, nous posons $x_{n+1} = F(F(x_n))$. Nous parlons de *redémarrages*. Une valeur de 0.01 semble être un bon choix pour la variable ε. Des valeurs plus grandes pour ε pourraient ralentir la convergence des méthodes squarem en effectuant fréquemment des étapes de l'algorithme EM, tandis que des valeurs trop petites pourraient être inefficaces pour éviter les stagnations et les divisions par zéro. Des études à la fois théorique et numérique pour fournir une solution à ce problème d'instabilité sont en cours [105].

Nous avons aussi développé un nouveau schéma hybride, SqHyb1, qui combine les schémas MPE1 et RRE1. En remarquant que

$$-1 \leq \cos\theta_n = \frac{(v_n, r_n)}{\|r_n\|\|v_n\|} \leq 1$$

où θ_n est l'angle entre r_n et v_n, nous définissons $w_n = |\cos\theta_n|$. De plus, nous remarquons

aussi que

$$w_n = \left| \frac{\alpha_n^{RRE1}}{\alpha_n^{MPE1}} \right|^{1/2} \tag{5.34}$$

ce qui signifie que le paramètre d'extrapolation du schéma RRE n'est jamais plus grand, en amplitude, que celui du schéma MPE1 (ils ont toujours le même signe). Cela pourrait expliquer la stabilité plus élevée des schémas RRE1 et SqRRE1 comparée à celle de MPE1 et SqMPE1. Le schéma hybride, SqHyb1, est donné par

$$x_{n+1} = x_n - 2\alpha_n^{Hyb1} r_n + \left(\alpha_n^{Hyb1}\right)^2 v_n \tag{5.35}$$

où

$$\alpha_n^{Hyb1} = w_n \alpha_n^{MPE1} + (1 - w_n)\alpha_n^{RRE1}.$$

Les divisions par zéro et les stagnations ont lieu quand r_n est presque orthogonal à v_n, c'est-à-dire quand pour un certain ε, $(v_n, r_n) < \varepsilon$, mais ni la norme de r_n, ni la norme de v_n ne le sont. Dans cette condition et en utilisant (5.34), nous pouvons approcher le pas de descente du schéma hybride par

$$\alpha_n^{Hyb1} \approx sgn((v_n, r_n)) \frac{\|r_n\|}{\|v_n\|} + \alpha_n^{RRE1} \tag{5.36}$$

où $sgn(x) = x/|x|, x \neq 0$. Ainsi, le schéma hybride semblerait éviter les divisions par zéro et la stagnation. En tout cas, la possibilité que la norme de v_n devienne très petite, sans que la norme de r_n le soit, est encore envisageable. Si un tel phénomène a lieu à l'itération n, nous effectuons un redémarrage, c'est-à-dire nous posons $x_{n+1} = F(F(x_n))$.

Comme énoncé à la page 77, les schémas d'extrapolation MPE et RRE peuvent aussi être combinés, c'est-à-dire nous pouvons appliquer le schéma d'extrapolation MPE (resp. RRE) pour obtenir z_n, suivi par une application du schéma d'extrapolation RRE (resp. MPE) pour obtenir x_{n+1}. Ainsi, il en résulte le schéma suivant

$$x_{n+1} = x_n - \left(\alpha_n^{MPE1} + \alpha_n^{RRE1}\right) r_n + \left(\alpha_n^{MPE1}\alpha_n^{RRE1}\right) v_n. \tag{5.37}$$

Une étude tant sur le plan théorique que sur le plan numérique est en cours [78].

5.4.3 Convergence des méthodes SQUAREM

Nous allons dans un premier temps discuter de la convergence des méthodes squarem quand le paramètre de relaxation est constant c'est à dire $\alpha_n = \alpha$. En accord avec [28, Section 5], nous parlons alors de stabilité du schéma au lieu de convergence.

Stabilité

Soit x^* un point fixe de la fonction F et x_0 le vecteur initial choisi dans un voisinage du point fixe x^*. Par (5.30), nous avons l'équation suivante pour la propagation de l'erreur des méthodes squarem

$$\varepsilon_{n+1} = [I_p - \alpha(\psi - I_p)]^2 \varepsilon_n + o(\varepsilon_n)$$

où $\varepsilon_n = x_n - x^*$ est l'erreur du schéma à la $n^{\text{ième}}$ itération et ψ est la jacobienne de F au point fixe x^*.

Une condition suffisante de stabilité pour les méthodes squarem en découle

$$\rho\left([I_p - \alpha(\psi - I_p)]^2\right) < 1 \quad \text{équivalent à} \quad -\frac{2}{a} < \alpha < 0 \tag{5.38}$$

où $a{=}sup_{\ t\in sp(\psi)}|1-t|$ et ρ désigne le rayon spectral. Alors, par (5.38) et similairement à [28], nous en déduisons le résultat suivant

Lemme 4.
Supposons que $I_p - \psi$ soit non singulière. Alors il existe un voisinage V de x^* tel que, pour $x_0 \in V$ et pour $-\frac{2}{a} < \alpha < 0$, la méthode squarem soit convergente.

De plus, nous pouvons obtenir la valeur optimale α_{opt}, c'est-à-dire la valeur de α pour laquelle la convergence linéaire est la plus rapide. En effet pour avoir la stabilité du schéma, nous devons avoir

$$\rho\left([I_p - \alpha(\psi - I_p)]^2\right) < 1$$

c'est-à-dire que le rayon spectral (module de la plus grande valeur propre) doit être strictement inférieur à l'unité. Si la matrice jacobienne de la fonction E.M. n'est pas symétrique, ses valeurs propres peuvent être complexes. Mais, en général (voir [52, Page 10]), les valeurs propres de la matrice ψ sont réelles et surtout dans $[0, 1)$. Donc, pour la stabilité du schéma, il est suffisant d'avoir

$$\sup_i \left[1 - \alpha\left(\lambda_i - 1\right)\right]^2 \ < \ 1$$

où les λ_i sont les valeurs propres de la matrice ψ et vérifient $0 \leq \lambda_i < 1$. Cette condition peut être réécrite comme

$$\frac{-2}{1-\lambda_i} \ < \ \alpha \ < \ 0 \quad \text{pour tout } i. \tag{5.39}$$

Il peut être montré (voir [96, pp. 310]) que la valeur optimale α_{opt} est donnée par

$$\alpha_{opt} = -\frac{2}{a+b} \tag{5.40}$$

avec $a = \min_i |1 - \lambda_i|$ et $b = \max_i |1 - \lambda_i|$. Cependant, cette analyse de la convergence n'est pas très pratique car le paramètre optimal de relaxation requiert la connaissance de la solution x^*, qui est, bien entendu, inconnue. Maintenant intéressons nous à la convergence pour le cas général où le paramètre de relaxation α_n n'est pas constant.

Convergence

En accord avec [28, 85], nous supposons que F est monotone décroissante et satisfait une condition de Lipschitz, c'est-à-dire

$$\forall\ y, z, \quad (F(y) - F(z), y - z) \leq 0 \tag{5.41}$$

$$\exists\ L > 0 \text{ tel que } \forall\ y, z, \quad || F(y) - F(z) || \leq L || y - z || \tag{5.42}$$

où $\forall\ x, || x ||^2 = (x, x)$. Remarquons que la condition (5.41) implique l'unicité de la solution x^*. Une étape importante pour la preuve du théorème de convergence est de montrer que le paramètre α_n prend ses valeurs dans $[-1, 0[$. Nous établissons alors ce résultat pour les trois méthodes squarem, SqRRE1, SqMPE1, et SqHyb1.

Pour le schéma SqRRE1, nous avons

$$\alpha_n^{RRE} = \frac{(r_n, v_n)}{(v_n, v_n)}. \tag{5.43}$$

En posant $D_n > 0$ le dénominateur de α_n^{RRE}, nous avons les relations suivantes

$$\begin{aligned} \alpha_n D_n &= -|| F(x_n) - x_n ||^2 + (F(x_n) - x_n, F^2(x_n) - F(x_n)) \\ (1 + \alpha_n) D_n &= || F^2(x_n) - F(x_n) ||^2 - (F(x_n) - x_n, F^2(x_n) - F(x_n)). \end{aligned}$$

Par suite, nous avons en vertu de (5.41) et comme $|| r_n || > 0$, $\alpha_n^{RRE} \in [-1, 0[$.

Pour le schéma SqMPE1, nous avons

$$\alpha_n^{MPE} = \frac{(r_n, r_n)}{(r_n, v_n)}. \tag{5.44}$$

Mais,

$$\begin{aligned} (r_n, v_n) &= (F(F(x_n)) - 2F(x_n) + x_n, F(x_n) - x_n) \\ &= (F(F(x_n)) - F(x_n), F(x_n) - x_n) - \|r_n\|^2 \\ &< 0 \quad \text{(grâce à la monotonie, Eq. (5.41) et } || r_n || > 0). \end{aligned}$$

D'où $\alpha_n^{MPE} < 0$. De plus, nous avons

$$\begin{aligned} 1 + \alpha_n^{MPE} &= \frac{(v_n + r_n, r_n)}{(v_n, r_n)} \\ (v_n + r_n, r_n) &= (F(F(x_n)) - F(x_n), F(x_n) - x_n) \\ &\leq 0 \quad \text{(par la monotonie, Eq. (5.41)).} \end{aligned}$$

Mais nous avons aussi $(v_n, r_n) < 0$, et donc $1 + \alpha_n^{MPE} \geq 0$. Par suite nous avons $-1 \leq \alpha_n^{MPE} < 0$.

Comme les deux paramètres α_n^{MPE} et α_n^{RRE} sont dans $[-1, 0[$, n'importe quelle combinaison convexe de ces deux paramètres est dans cet intervalle. Ainsi, $\alpha_n^{Hyb} \in [-1, 0[$.

Par définition des termes r_n et v_n dans les méthodes squarem (Eq. (5.29)), nous avons

$$x_{n+1} - x^* = a_n(x_n - x^*) + b_n(F(x_n) - x^*) + c_n(F^2(x_n) - x^*)$$

avec les variables positives a_n, b_n et c_n définies par

$$\begin{aligned} a_n &= (1 + \alpha_n)^2 \\ b_n &= -2\alpha_n(1 + \alpha_n) \\ c_n &= \alpha_n^2. \end{aligned}$$

Alors, nous avons

$$\begin{aligned} || x_{n+1} - x^* ||^2 &= a_n^2 || x_n - x^* ||^2 + b_n^2 || F(x_n) - x^* ||^2 + c_n^2 || F^2(x_n) - x^* ||^2 \\ &\quad + 2a_n b_n (x_n - x^*, F(x_n) - x^*) + 2a_n c_n (x_n - x^*, F^2(x_n) - x^*) \\ &\quad + 2b_n c_n (F(x_n) - x^*, F^2(x_n) - x^*). \end{aligned}$$

Par suite et par les conditions (5.41) et (5.42), l'inégalité de Cauchy-Schwartz, et en rappelant que $F(x^*) = x^*$, nous en déduisons que

$$|| x_{n+1} - x^* ||^2 \leq M_n^2 \, || x_n - x^* ||^2,$$

avec $M_n^2 = (a_n + L^2 c_n)^2 + b_n^2 L^2 > 0$. Mais, M_n^2 est un polynôme de degré 4 en la variable α_n

$$M_n^2 = P(\alpha_n) = 1 + 4\alpha_n + (6 + 6L^2)\alpha_n^2 + (4 + 12L^2)\alpha_n^3 + (1 + L^4 + 6L^2)\alpha_n^4.$$

Ce polynôme décroît dans $[-1, \alpha^*]$ et croît dans $[\alpha^*, 0]$, où α^* est l'unique racine appartenant à $[-1, 0]$ de la dérivée de P, ce qui peut être prouvé en dérivant P trois fois et en utilisant le fait que $\alpha_n \in [-1, 0[$ (Pour le détail des calculs, voir Annexe C (page 46)). D'autre part, $P(0) = 1$ et $P(-1) = L^4$. Nous en déduisons donc le résultat suivant

Théorème 3.

- *Si $L < 1$, alors $M_n < 1$ et donc la méthode squarem converge.*
- *Si $L \geq 1$ et $\mu < \alpha_n < 0$, où μ est tel que $P(\mu) = 1$, alors $M_n < 1$ et la nouvelle méthode converge.*

La figure 5.1 représente le polynôme de convergence, $P(\alpha_n)$, pour différentes valeurs de la constante L. Quand $L < 1$, nous avons la convergence pour tout $-1 \leq \alpha_n < 0$. Sinon ($L \geq 1$), la valeur minimale, μ de α_n pour laquelle la convergence est garantie devient de plus en plus petite. Cette valeur μ peut être obtenue à partir des courbes en désignant l'abscisse du point d'intersection de la courbe de $P(\alpha_n)$ avec la droite en pointillée d'équation $y = 1$.

5.5 Résultats Numériques

Nous testons la performance des méthodes squarem sur trois exemples statistiques différents où l'algorithme EM est choisi afin de déterminer des estimateurs du maximum de vraisemblance, notés en abrégé EMV. Dans ces problèmes, les méthodes squarem sont évaluées sur trois critères. Le premier critère est d'évaluer les méthodes squarem sur leurs efficacités à accélérer l'algorithme EM. Puis, elles sont comparées à leurs homologues à un pas. Et enfin, elles sont évaluées entre elles pour déterminer la méthode squarem la plus efficace pour chaque problème. Avant de présenter les résultats numériques, nous effectuons une remarque intéressante concernant des transformations de paramètres.

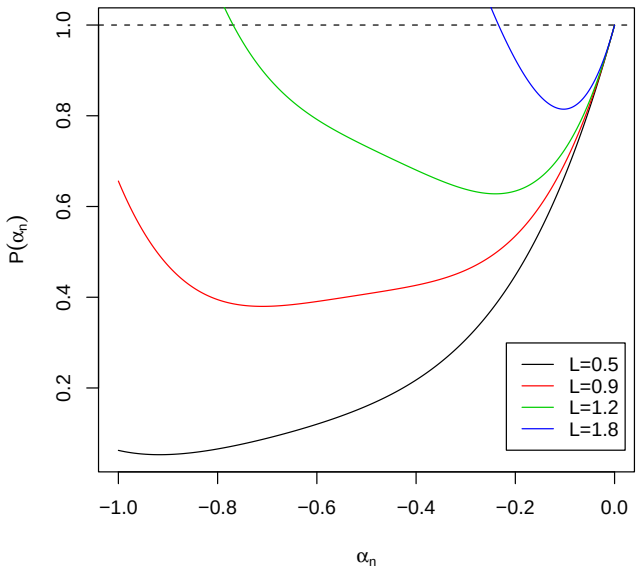


FIG. 5.1 – Polynôme de convergence $P(\alpha_n)$ pour différentes valeurs de la constante L

5.5.1 Transformations de Paramètres

Soit (x_n) la suite générée par l'algorithme EM. Nous rappelons que l'algorithme EM peut être considéré comme la méthode de Picard appliquée à des problèmes de maximisation de fonction de densité de probabilité. En effet, chaque cas de l'algorithme EM (Chap. 4, Eq. (4.5)) définit une fonction $x \to F(x)$ sur l'espace $\Omega \subset \mathbb{R}^p$ vers lui même telle que

$$x_{n+1} = F(x_n).$$

Dans ce contexte, il est intéressant de remarquer que l'algorithme EM n'est pas affecté par des transformations de paramètres lesquelles sont *homéomorphes*. En effet, la convergence de la suite transformée est identique à celle de la suite initiale. Bien que cette remarque est relativement facile à démontrer, elle n'a pas été faite à notre connaissance dans la littérature de l'algorithme EM. Nous rappelons qu'une transformation $T : D \subset \mathbb{R}^p \mapsto \Omega$ est un homéomorphisme de D dans Ω si T est bijective et T et T^{-1} sont continues sur D et Ω, respectivement. Nous avons donc le théorème suivant

Théorème 4. *Etant donné F la fonction de l'algorithme EM et un homéomorphisme T tel que $T^{-1}FT$ est une contraction sur D, alors F a précisement le même nombre de point fixe dans Ω qu'en a $T^{-1}FT$ dans D. Pour n'importe quel $x_0 \in \Omega$, les itérés (x_n) de l'algorithme EM restent dans Ω et convergent si et seulement si les itérés $z_{n+1} = T^{-1}FTz_n$ avec $z_0 = T^{-1}x_0$, restent dans D et convergent.*

Décès i	Fréquence n_i	Décès i	Fréquence n_i
0	162	5	61
1	267	6	27
2	271	7	8
3	185	8	3
4	111	9	1

TAB. 5.1 – Les données observées

Démonstration. Si $x^* \in \Omega$ est un point fixe de F, alors $z^* = T^{-1}x^*$ est un point fixe de $T^{-1}FT$, et inversement. De ce fait, puisque les suites (x_n) et (z_n) sont reliées par $x_n = T\,z_n$, elles doivent avoir le même comportement de convergence. □

Nous remarquons par analogie à la théorie des matrices que la fonction F est semblable à la contraction $T^{-1}FT$. De ce fait, elles ont les mêmes valeurs propres et par suite cette transformation n'affecte pas le taux de convergence de l'algorithme EM. Par contre, comme nous le verrons dans les exemples numériques, nous pouvons améliorer significativement le taux de convergence des méthodes squarem et leurs homologues à un pas en utilisant une transformation appropriée des paramètres.

5.5.2 Distribution de Poisson

Considérons un exemple classique donné dans [123, p.128] : il concerne une distribution de Poisson. Les données observées (i, n_i) sont reportées dans le tableau 5.1. Les colonnes nommées "Décès i" désignent le nombre de décès de femmes de 80 ans ou plus et les colonnes nommées "fréquence n_i" précisent le nombre de jours avec i décès de femmes de 80 ans ou plus. Pour ce problème, la fonction de densité de probabilité des données observées est

$$f(p, \mu_1, \mu_2) = \prod_{i=0}^{9} \left[p e^{-\mu_1} \frac{\mu_1^i}{i!} + (1-p) e^{-\mu_2} \frac{\mu_2^i}{i!} \right]^{n_i} .$$

Le but est de trouver le vecteur $\theta^* = (p^*, \mu_1^*, \mu_2^*)$ tel que

$$\theta^* = \operatorname{argmax} \{ \ \log\left(f(p, \mu_1, \mu_2)\right) \ ; \ p \in [0,1], \quad \mu_1, \mu_2 \in \mathbb{R} \} \, . \tag{5.45}$$

Pour calculer itérativement ce vecteur θ^*, nous utilisons l'algorithme EM qui est donné par

$$\begin{aligned}
p^{(m+1)} &= \frac{\sum_i n_i \, \pi_{i1}^{(m)}}{\sum_i n_i} \\
\mu_1^{(m+1)} &= \frac{\sum_i i \, n_i \, \pi_{i1}^{(m)}}{\sum_i n_i \, \pi_{i1}^{(m)}} \\
\mu_2^{(m+1)} &= \frac{\sum_i i \, n_i \left(1 - \pi_{i1}^{(m)}\right)}{\sum_i n_i \left(1 - \pi_{i1}^{(m)}\right)}
\end{aligned}$$

	EM	MPE1	RRE1	SqMPE1	SqRRE1	SqHyb1
Nb. Eval. de F	2045	1986	1242	308	584	462
Redémarrage	0	0	308	0	1	0
log-Prob.	−1989.95	−1989.95	−1994.05	−1989.95	−1989.95	−1989.95

TAB. 5.2 – Vecteur initial 1 : $\theta^{(0)} = (0.2870, 1.101, 2.582)$

	EM	MPE1	RRE1	SqMPE1	SqRRE1	SqHyb1
Nb. Eval. de F	2056	1800	2342	244	572	268
Redémarrage	0	0	363	0	0	0
log-Prob.	−1989.95	−1989.95	−1994.05	−1989.95	−1989.95	−1989.95

TAB. 5.3 – Vecteur initial 2 : $\theta^{(0)} = (0.3, 1.0, 2.5)$

où

$$\pi_{i1}^{(m)} = \frac{p^{(m)} \left(\mu_1^{(m)}\right)^i e^{-\mu_1^{(m)}}}{p^{(m)} \left(\mu_1^{(m)}\right)^i e^{-\mu_1^{(m)}} + (1-p^{(m)}) \left(\mu_2^{(m)}\right)^i e^{-\mu_2^{(m)}}}.$$

Remarquons que l'algorithme EM définit bien une fonction non linéaire F telle que $\theta^{(m+1)} = F(\theta^{(m)})$ avec $\theta^{(m)} = (p^{(m)}, u_1^{(m)}, u_2^{(m)})$. Pour ce problème, le vecteur solution θ^* est $(0.3599, 1.256, 2.663)$. L'algorithme EM converge très lentement vers θ^*. En effet, les valeurs propres de la matrice jacobienne de la fonction F au point fixe θ^*, $J(\theta^*)$, sont 0.9957, 0.7204 et 0. Par suite, comme la plus grande valeur propre en valeur absolue de la matrice $J(\theta^*)$, c'est-à-dire le rayon spectral, est très proche de 1, l'algorithme EM converge lentement (voir Chap. 4, Eq. (4.6)). Les résultats pour les différentes méthodes d'extrapolation et pour l'algorithme EM sont présentés dans les tableaux 5.2 et 5.3 pour deux vecteurs initiaux différents. Le premier vecteur initial utilisé par Lange dans [82] est $\theta^{(0)}$ = (0.2870,1.101,2.582) et l'autre vecteur est $\theta^{(0)}$ =(0.3,1.0,2.5). Nous pouvons remarquer que la méthode squarem SqMPE1 donne les meilleurs résultats. Elle accélére l'algorithme EM par un facteur entre 6 et 7. Les autres méthodes squarem donnent une accélération par un facteur 4. Pour les méthodes à un pas, la méthode RRE1 n'accélére pas l'algorithme EM tandis que la méthode MPE1 montre un gain très modeste. Il doit être remarqué que le nombre de redémarrages est élévé pour la méthode RRE1. Cela s'explique par le fait que cette méthode est sujette à des problèmes numériques, en particulier des stagnations.

L'algorithme EM satisfait naturellement la contrainte sur le paramètre p, c'est-à-dire $p \in [0, 1]$. Ce n'est pas toujours le cas pour les méthodes d'extrapolation, et par suite, le paramètre p à une itération m peut ne pas vérifier cette contrainte, même si cela n'a aucune incidence sur la convergence. Les transformations de paramètres sont une issue possible pour contraindre la variable p à être dans l'intervalle [0,1]. Dans les tableaux 5.4 et 5.5, nous reportons les résultats des différentes méthodes utilisant une transformation de paramètres

	EM	MPE1	RRE1	SqMPE1	SqRRE1	SqHyb1
Nb. Eval. de G	2211	1482	212	46	72	94
Redémarrage	0	0	0	0	0	0
log-Prob.	−1989.95	−1989.95	−1989.95	−1989.95	−1989.95	−1989.95

TAB. 5.4 – Vecteur initial 1, avec transformation de paramètres

simple qui consiste à transformer le paramètre p en p' tel que $p' = \log\frac{p}{1-p}$, et qui laisse les autres paramètres invariants c'est-à-dire $\mu_1' = \mu_1$ et $\mu_2' = \mu_2$. Nous observons quelques résultats très intéressants concernant cette transformation de paramètres. Elle permet d'améliorer significativement le taux de convergence des méthodes d'extrapolation squarem par un facteur de 4 à 10. La meilleure amélioration est observée pour les méthodes de type RRE. En effet, grâce à cette transformation de paramètres, la méthode RRE1 converge nettement plus vite et surtout les problèmes numériques liés à la stagnation sont éliminés. Par contre, aucune amélioration importante n'est observée pour la méthode MPE1. Bien entendu, cette transformation n'a aucun impact sur la convergence de l'algorithme EM puisqu'il est invariant par transformation homéomorphe (voir la section précédente). La légère différence dans le nombre d'évaluations de la fonction non linéaire pour l'algorithme EM, quand nous comparons le tableau 5.2 avec le tableau 5.4 et le tableau 5.3 avec le tableau 5.5, est due au critère d'arrêt. Précisons que nous considérons qu'une itération $\theta^{(m)}$ d'une méthode est suffisamment proche de la solution θ^*, si $|| \theta^{(m)} - F(\theta^{(m)}) || < 10^{-7}$. Soit la transformation de paramètres $T : z = (z_1, z_2, z_3) \mapsto \theta = (p, \mu_1, \mu_2)$ définie par

$$p = (1 + \exp(-z_1))^{-1} \quad , \quad \mu_1 = z_2 \quad , \quad \mu_2 = z_3.$$

Les itérés $z^{(m)}$ de l'algorithme EM transformé sont donnés par $z^{(m+1)} = G(z^{(m)})$ où la fonction G est définie par $T^{-1}FT$, et z^* est le point fixe de G. Puisque les itérés $\theta^{(m)}$ et $z^{(m)}$ sont reliés par $\theta^{(m)} = T(z^{(m)})$ et $\|\theta^{(m)} - F(\theta^{(m)})\| = \|T(z^{(m)}) - T(G(z^{(m)})\|$, le critère d'arrêt pour les itérations de l'algorithme EM transformé, $z^{(m)}$, est approximativement égal au critère d'arrêt pour les itérés $\theta^{(m)}$ multiplié par un facteur égal à $\det\left[\partial T(z^*)\right]$, où $\partial T(z^*)$ est la jacobienne de la transformation T évaluée au point fixe z^*, donné par $T(z^*) = x^*$. La valeur pour z^* est $(-0.575, 1.256, 2.663)$ et d'où

$$\det\left[\partial T(z^*)\right] = (\exp(z_1^*/2) + \exp(-z_1^*/2))^{-2} = 0.230.$$

Le critère d'arrêt sur les $z^{(m)}$ est donc plus pointu que celui sur les $\theta^{(m)}$, expliquant pourquoi le nombre d'évaluations dans les tableaux 5.4 et 5.5 est plus élévé que celui dans les tableaux correspondants 5.2 et 5.3.

5.5.3 Distribution de von Mises

La distribution de von Mises est souvent utilisée pour décrire des données sur la circonférence d'un cercle. Des données circulaires interviennent dans de nombreuses applications telles que : (i) l'évolution journalière d'événements défavorables (infarctus du myocarde,

	EM	MPE1	RRE1	SqMPE1	SqRRE1	SqHyb1
Nb. Eval. de G	2223	1736	212	40	46	86
Redémarrage	0	0	0	0	0	0
log-Prob.	−1989.95	−1989.95	−1989.95	−1989.95	−1989.95	−1989.95

TAB. 5.5 – Vecteur initial 2, avec transformation de paramètres

morts), (ii) variations saisonnières d'événements défavorables (certains types de cancer, suicides) et (iii) l'étude de processus hydrologiques, par exemple la quantité de précipitations par mois. La distribution de von Mises est donnée par

$$g(y;\mu,\kappa) = \frac{1}{2\pi I_0(\kappa)}\exp(\kappa(y-\mu)), \tag{5.46}$$

où y est une variable distribuée sur le cercle, c'est-à-dire $y \in (0, 2\pi)$, les autres paramètres de la distribution $\mu \in (0, 2\pi)$ et $\kappa > 0$ sont respectivement les paramètres de localisation et de concentration et $I_0(.)$ désigne la fonction modifiée de Bessel d'ordre zéro. Remarquons que pour la valeur $\kappa = 0$, nous obtenons la densité uniforme $1/2\pi$. Il n'est pas inhabituel pour des données circulaires d'être bimodales, indiquant soit la présence de deux processus différents soit le même processus opérant à deux moments différents. Par exemple, l'écoulement mensuel de l'eau au niveau d'un barrage pourrait avoir deux sources dominantes, une provenant des précipitations en automne et l'autre de la fonte de neige au printemps. Dans de telles situations, une combinaison à deux composantes d'une distribution de von Mises, pourrait fournir une description plus juste des données. L'équation d'une telle combinaison est donnée par

$$g(y;p_1,p_2,\mu_1,\kappa_1,\mu_2,\kappa_2) = p_1\ f(y;\kappa_1,\mu_1)\ +\ p_2\ f(y;\kappa_2,\mu_2) \tag{5.47}$$

où $p_1 + p_2 = 1$ et pour $i \in [1, 2]$,

$$f(y;\kappa_i,\mu_i) = \frac{\exp(\kappa_i(y-\mu_i))}{2\pi I_0(\kappa_i)}.$$

Soit $y = (y_1, \ldots, y_n)$ les directions observées en radians. Nous pouvons écrire la fonction de densité de probabilité des données observées de la façon suivante

$$\prod_{i=1}^{n} g(y_i;p_1,p_2,\mu_1,\kappa_1,\mu_2,\kappa_2). \tag{5.48}$$

Le but est donc de trouver les paramètres p_1, μ_1, μ_2, κ_1 et κ_2 tels que le logarithme de cette fonction de densité de probabilité soit maximale. Le paramètre p_2 se déduit directement de p_1 grâce à la contrainte $p_1 + p_2 = 1$. Pour calculer itérativement ces paramètres appelés estimateurs du maximum de vraisemblance, nous utilisons l'algorithme EM qui est donné par [53]

$$
\begin{aligned}
p_1^{(m+1)} &= \frac{1}{n}\sum_{i=1}^{n}\pi_{i1}^{(m)} \\
p_2^{(m+1)} &= 1 - p_1^{(m+1)} \\
\mu_j^{(m+1)} &= \arctan\left(\frac{S_j^{(m)}}{C_j^{(m)}}\right) \qquad \text{pour } j \in [1,2] \\
\kappa_j^{(m+1)} &= \left(1.28 - 0.53\left(A_j^{(m)}\right)^2\right)\tan\left(\frac{\pi A_j^{(m)}}{2}\right) \qquad \text{pour } j \in [1,2]
\end{aligned}
$$

où

$$
C_j^{(m)} = \frac{1}{n}\sum_{i=1}^{n}\pi_{ij}^{(m)}\cos y_i, \qquad S_j^{(m)} = \frac{1}{n}\sum_{i=1}^{n}\pi_{ij}^{(m)}\sin y_i,
$$

et

$$
A_j^{(m)} = \sqrt{\left(C_j^{(m)}\right)^2 + \left(S_j^{(m)}\right)^2} \quad \text{et} \quad \pi_{ij}^{(m)} = \frac{p_j^{(m)}\, f(y_i;\kappa_j^{(m)},\mu_j^{(m)})}{\sum_{j=1}^{2} p_j^{(m)}\, f(y_i;\kappa_j^{(m)},\mu_j^{(m)})}.
$$

Avec les paramètres suivants de simulation $(p_1,\mu_1,\kappa_1,\mu_2,\kappa_2) = (0.75,\pi/2,0.8,3\pi/2,1.6)$, nous générons 200 échantillons de taille $n = 1000$. Nous choisissons aléatoirement le vecteur initial qui est le même pour toutes les méthodes. Comme précisé par Biernacki [16, p.35], l'étape d'initialisation est importante : il pourrait s'avérer plus judicieux d'utiliser les stratégies définies dans [17]. Les résultats donnés par l'algorithme EM et les différentes méthodes d'extrapolation sur les 200 exemples sont rassemblés dans les tableaux 5.6 et 5.7 afin de comparer leur efficacité à trouver les estimateurs du maximum de vraisemblance, sans et avec une transformation des paramètres. Précisons que la transformation de paramètres est de type logarithmique. Nous transformons les paramètres de concentration et de proportion de la façon suivante

$$
\kappa_i' = \log\frac{\kappa_i}{1-\kappa_i} \quad \text{et} \quad p_i' = \log\frac{p_i}{1-p_i}.
$$

Les autres paramètres sont laissés invariants, c'est-à-dire $\mu_i' = \mu_i$. Les transformations de paramètres améliorent non seulement le taux de convergence des méthodes squarem significativement (par un facteur de 2 ou plus), comme dans le cas de la distribution de Poisson, mais elles permettent aussi de garantir les contraintes pour chaque paramètre. En particulier, les paramètres de concentration doivent être positifs. Une fois encore, la transformation de paramètres n'influe pas sur l'algorithme EM, puisque les valeurs propres de la matrice jacobienne sont invariantes par une tranformation bijective des paramètres. La méthode MPE1 se comporte de la même façon que l'algorithme EM et son taux de convergence n'est guère meilleur. Si nous regardons le premier quartile et la médiane, la méthode RRE1 a un taux de convergence plus rapide que l'algorithme EM, mais en général

	1^{er} quartile	médiane	moyenne	$3^{ème}$ quartile	# Echecs
EM	656	1030	1343	1403	1
MPE1	508	840	1393	1330	11
RRE1	138	340	3822	10000	43
SqMPE1	132	171	184	235	20
SqRRE1	206	388	889	878	2
SqHyb1	148	209	247	289	2

TAB. 5.6 – Résultats pour le problème VM - sans transformation. Le nombre dans les quatre premières colonnes désigne le nombre d'évaluations de la fonction non linéaire F.

	1^{er} quartile	médiane	moyenne	$3^{ème}$ quartile	# Echecs
EM	697	958	1500	1442	0
MPE1	577	800	1235	1148	0
RRE1	138	212	2721	5656	23
SqMPE1	44	56	75	83	35
SqRRE1	52	86	1894	1994	15
SqHyb1	56	80	284	184	2

TAB. 5.7 – Résultats pour le problème VM - avec transformation. Le nombre dans les quatre premières colonnes désigne le nombre d'évaluations de la fonction non linéaire F.

elle est assujettie à un problème numérique, la stagnation. La méthode SqMPE1 semble la plus rapide. Si nous regardons la médiane, elle accélére l'algorithme EM par un facteur de 6 à 18. Toutefois, cette méthode souffre de problèmes de division par zéro dans quelques simulations. La méthode SqHyb1 est clairement la plus efficace pour ce problème à la fois en terme de vitesse de convergence et de capacité à éviter les problèmes numériques, à savoir la division par zéro et la stagnation. Précisons que nous désignons par le terme échec dans les tableaux 5.6 et 5.7, les trois situations suivantes : (1) dépassement du seuil maximum du nombre d'évaluations de la fonction F que nous avons fixé à 10000, (2) dépassement du nombre de redémarrages qui est fixé à 100 et (3) valeurs inadmissibles ou impossibles pour les paramètres. Seuls l'algorithme EM et les méthodes de premier ordre RRE1 et MPE1 souffrent du premier type de problème. Dans ce cas rare, la convergence est presque sublinéaire. Les méthodes RRE1, à la fois à un pas et squarem, sont sujettes au second type de problème, dû à des stagnations. Quand les méthodes stagnent, la technique de redémarrage utilisant les itérations de base de l'algorithme EM est utilisée. Cependant, si après 100 redémarrages, nous n'observons aucune amélioration en termes de réduction de la norme du résidu, nous considérons cette situation comme un échec. Le troisième type de problème intervient dans les méthodes MPE1, dû à une instabilité numérique appelée *nearbreakdown*, dans quel cas les erreurs s'amplifient à chaque cycle jusqu'au moment où la méthode ne donne plus de résultats mathématiquement possibles.

5.5.4 Analyse des classes latentes (A.C.L.)

Soit un vecteur $\mathbf{y} = (y_1, \ldots, y_d) \in \mathbb{R}^d$ supposé être généré par un modèle de type classe latente (voir [56, Chap. 2]). Par définition de ce modèle, la fonction de densité de probabilité pour ce vecteur $\mathbf{y}$ est donnée par

$$g(\mathbf{y}; p, \Theta) = \sum_{j=1}^{c} p_j \, g_j(\mathbf{y}; \theta_j) \tag{5.49}$$

où $\Theta = (\theta_1, \ldots, \theta_c)$ est une matrice de dimension $d \times c$, $\theta_j = (\theta_{j_1}, \ldots, \theta_{j_d})$ représente le vecteur colonne j de la matrice Θ, $p = (p_1, \ldots, p_c)$ est un vecteur de $\mathbb{R}^c$ qui vérifie la contrainte $\sum_i p_i = 1$ et

$$g_j(\mathbf{y}; \theta_j) = \prod_{k=1}^{d} \theta_{j_k}^{y_k} \, (1 - \theta_{j_k})^{1 - y_k}. \tag{5.50}$$

Soient $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$, n vecteurs observés de dimension d. Nous pouvons écrire la fonction de densité de probabilité des données observées de la façon suivante

$$f(\mathbf{Y}; p, \Theta) = \prod_{i=1}^{n} g(\mathbf{y}_i; p, \Theta). \tag{5.51}$$

Le but est de trouver le vecteur p et les éléments de la matrice Θ pour lesquels le logarithme de la fonction de densité de probabilité des données observées $f(\mathbf{Y}; p, \Theta)$ donnée par l'équation (5.51) atteint son maximum. Pour calculer itérativement tous ces paramètres, nous utilisons l'algorithme EM donné par les équations suivantes [56, Chap. 2]

$$p_j^{(m+1)} = (1/n) \sum_{i=1}^{n} \hat{\pi}_{ij}^{(m)} \tag{5.52}$$

$$\theta_j^{(m+1)} = \frac{1}{n \, p_j^{(m)}} \sum_{i=1}^{n} \mathbf{y}_i \hat{\pi}_{ij}^{(m)} \tag{5.53}$$

où

$$\hat{\pi}_{ij}^{(m)} = \frac{p_j^{(m)} \, g_j(\mathbf{y}_i; \theta_j^{(m)})}{\sum_{j=1}^{c} p_j^{(m)} \, g_j(\mathbf{y}_i; \theta_j^{(m)})}. \tag{5.54}$$

Pour trouver ces équations, il suffit d'appliquer la théorie de l'algorithme EM donnée et illustrée dans le chapitre 4. Dans la suite, la fonction non linéaire définie par l'algorithme EM (Eqs. (5.52) et (5.53)) est notée F. Avec le choix $c = 3$ et $d = 5$ dans les équations (5.49) et (5.50) et en définissant le jeu suivant de paramètres de simulation

$$\begin{aligned} p &= (1/3, 1/3, 1/3) \\ \theta_1 &= (0.5, 0.5, 0.2, 0.3, 0.1) \\ \theta_2 &= (0.3, 0.2, 0.7, 0.6, 0.4) \\ \theta_3 &= (0.9, 0.7, 0.5, 0.1, 0.7), \end{aligned}$$

	1^{er} quartile	médiane	moyenne	$3^{ème}$ quartile	# Echecs
EM	586	1147	1652	2504	6
MPE1	613	1106	1749	2150	6
RRE1	203	415	652	966	40
SqMPE1	30	42	959	134	9
SqRRE1	32	41	121	72	0
SqHyb1	30	40	155	59	2

TAB. 5.8 – Résultats pour le problème ACL : 200 simulations

nous générons 200 échantillons de taille $n = 200$, c'est-à-dire 200 données observées $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$ avec $n = 200$. Les résultats des différentes méthodes sur ces 200 simulations sont rassemblés dans le tableau 5.8. Précisons que nous initialisons les méthodes aléatoirement. L'intervalle interquartile pour le nombre d'évaluations de la fonction F pour l'algorithme EM est (586,2504) avec une moyenne de 1652. Pour les méthodes à un pas, RRE1 est clairement supérieur. Son intervalle interquartile est (203,966) avec une moyenne de 652. Les résultats obtenus par la méthode MPE1 ne sont pas très différents de l'algorithme EM. La performance des méthodes squarem pour ce problème est spectaculaire. Elles sont 20 à 25 fois plus rapides que l'algorithme EM. La méthode SqMPE1 échoue 9 fois, c'est-à-dire elle dépasse les 5000 évaluations de la fonction F en ne donnant pas le résultat correct, tandis que la méthode SqHyb1 échoue 2 fois. Par contre, la méthode SqRRE1 n'a aucun problème numérique.

5.6 Discussion et Conclusion

Dans ce chapitre, nous avons proposé une nouvelle classe de méthodes itératives dites squarem pour déterminer des estimateurs du maximum de vraisemblance, via l'algorithme EM, en considérant cet algorithme comme une méthode itérative de point fixe, $x_{n+1} = F(x_n)$. Ces nouvelles méthodes sont basées sur une nouvelle stratégie appelée *squaring*, laquelle est une application en deux étapes à l'intérieur de chaque cycle, de schémas d'extrapolation d'ordre 1, ($k = 1$ dans Eq. (5.13)), de la forme $y_n = x_n + \alpha_n(F(x_n) - x_n)$. Les méthodes se différencient par le choix du pas de descente α_n dans les schémas d'extrapolation. Les méthodes avec un pas de descente constant sont dites *stationnaires* et les autres *non stationnaires*. Les propriétés de convergence des méthodes stationnaires sont faciles à caractériser, mais elles apportent, en général, un gain modeste par rapport aux itérations de point fixe, c'est-à-dire l'algorithme EM. Ce constat est identique pour leurs versions squarem, même si ces dernières convergent plus rapidement. De ce fait, nous nous sommes uniquement intéressés aux méthodes itératives non stationnaires à un pas, et leurs versions squarem, pour accélérer la convergence de l'algorithme EM.

Les méthodes itératives non stationnaires à un pas peuvent être développées à l'aide des schémas d'extrapolation, Eq. (5.13). En effet, en cyclant ces schémas d'extrapolation, nous obtenons des méthodes itératives à la fois efficace et simple pour résoudre des pro-

blèmes non linéaires de point fixe. Précisons que de tels schémas d'extrapolation ont déjà été considérés pour résoudre des systèmes linéaires ([23], [76], [116]). Dans ce chapitre, les schémas d'extrapolation considérés sont d'ordre 1, c'est-à-dire $k = 1$ dans l'équation (5.13). En particulier, nous utilisons la *minimal polynomial extrapolation* (MPE) et *reduced rank extrapolation* (RRE). La stratégie *squaring* permet d'obtenir des méthodes dites squarem, qui convergent plus rapidement avec un coût de calcul négligeable. En effet, elles requièrent seulement un produit scalaire-vecteur et une addition vectorielle supplémentaires par rapport à leurs homologues classiques d'ordre 1, RRE1 et MPE1. Les résultats de convergence sur les méthodes itératives classiques ou sur les schémas d'extrapolation, ne sont pas très nombreux. Jbilou et Sadok [76] ont montré que les méthodes itératives classiques obtenues en cyclant un schéma d'extrapolation donné par l'équation (5.13), sont quadratiquement convergentes si pour tout n, l'ordre k au cycle $n + 1$ du schéma d'extrapolation utilisé est égal au degré du polynôme minimal de la matrice jacobienne $J(x^*)$ pour le vecteur $x_n - x_0$. Pour les schémas d'extrapolation, nous disposons d'un résultat de Sidi [115] : il donne des résultats asymptotiques (c'est-à-dire quand le nombre de cycles n devient large) concernant le taux de convergence des schémas d'extrapolation, RRE et MPE. Bien que ce résultat soit donné pour des problèmes linéaires de point fixe, il semble être possible de l'adapter à des problèmes non linéaires. Des résultats analogues sur les méthodes squarem n'existent pas et ceci fera l'objet d'études futures. En tout cas, nos expériences numériques démontrent clairement que les méthodes squarem améliorent significativement leurs homologues à un pas, par un facteur au moins égal à deux qui reste à démontrer par la théorie.

Nous pouvons aussi obtenir des méthodes squarem d'ordre plus élevé, en appliquant la nouvelle stratégie *squaring* avec des schémas d'extrapolation d'ordre $k > 1$. Par exemple, en cyclant un schéma d'extrapolation d'ordre $k = 2$, les méthodes itératives classiques sont de la forme

$$x_{n+1} = x_n + \alpha_n(F(x_n) - x_n) + \beta_n(F^2(x_n) - 2F(x_n) + x_n) \tag{5.55}$$

où α_n et β_n dépendent de $F^j(x_n), j = 0, 1, 2, 3$. Appliquer la stratégie *squaring* à ces méthodes itératives, implique ainsi le calcul de $F^4(x_n)$, et donc, une évaluation supplémentaire de la fonction F est nécessaire. Plus généralement, les méthodes itératives classiques, RRE et MPE d'ordre k requièrent $k+1$ évaluations de la fonction F et leurs homologues squarem $2k$ évaluations : la stratégie *squaring* nécessite donc $k-1$ évaluations supplémentaires. Par ailleurs, le nombre relatif d'évaluations de la fonction F est $(k-1)/(k+1)$, lequel est zéro pour les schémas d'ordre 1, et approche 1 quand l'ordre croît. Ainsi, la stratégie *squaring* pourrait ne pas être nécessaire pour des schémas d'ordre plus élevés. Ce point fera l'objet d'une étude approfondie.

Les méthodes itératives et leurs versions squarem peuvent être sujettes soit à des stagnations soit à des divisions par zéro. Dans le cas de la stagnation, la méthode itérative ne progresse plus, c'est-à-dire le pas de descente est presque nul, et les itérés calculés ne sont guère différents. Dans le cas de la division par zéro, les erreurs numériques s'amplifient, le pas de descente de la méthode prend des valeurs disproportionnellement élevées et la mise à jour des itérés n'est plus possible. Les méthodes RRE, en particulier RRE1 et SqRRE1,

sont sujettes aux stagnations. Par contre, les méthodes MPE, SqMPE1 et MPE1 souffrent de problèmes de division par zéro. Pour éviter ces problèmes, nous avons conseillé une stratégie simple de redémarrage en utilisant les itérations de base de l'algorithme EM qui sont calculées à chaque cycle. Les expériences numériques montrent clairement les limites de cette stratégie. Pour réduire ce problème, nous avons aussi développé une méthode hybride qui est, avec succès, moins sujette aux problèmes de division par zéro que les méthodes RRE et aux problèmes de stagnation que les méthodes MPE. Un développement de stratégies plus efficaces, pour éviter la stagnation et les divisions par zéro, est en cours.

Ainsi, les méthodes squarem présentées dans ce chapitre, semblent être efficaces pour accélérer la convergence de l'algorithme EM puisqu'elles possédent les avantages suivants : (a) elles sont simples et requièrent seulement quelques itérations de base de l'algorithme EM, (b) elles ne requièrent pas la programmation de quantités auxiliaires telles que les fonctions de densité de probabilité des données incomplètes ou complètes ou leurs gradients, (c) par (a) et (b) elles peuvent être intégrées facilement dans les sous-programmes existants de l'algorithme EM, (d) elles n'impliquent pas de stockage supplémentaire de vecteurs ou de matrices et (e) elles convergent linéairement, comme l'algorithme EM, mais avec un taux de convergence plus rapide, en particulier dans les problèmes où l'algorithme EM est très lent.

Chapitre 6

Application en Tomographie

6.1 Introduction

La tomographie par émission de positons notée T.E.P. est une technologie médicale très utilisée pour visualiser l'activité d'un organe dans le but de détecter les régions dont l'activité est anormale, par exemple une tumeur. Une dose d'une substance biochimique marquée par des composants radioactifs émettant des positons est administrée au patient et les émissions radioactives sont comptées à l'aide d'un scanner, machine composée de plusieurs détecteurs formant un ou plusieurs anneaux et placée autour de l'organe. Le choix de la substance dépend bien entendu de l'organe à étudier. Par exemple, pour le cerveau, du glucose marqué par un isotope radioactif sera utilisé puisque le glucose constitue la principale source d'énergie de cet organe. Si nous pouvions enregistrer la localisation (à l'intérieur du cerveau) de chaque émission de positons, nous pourrions alors produire un portrait de la consommation en glucose du cerveau et donc de son activité. Il est bien entendu impossible de détecter la localisation exacte de l'émission, par contre il est possible de déterminer un volume cylindrique dans lequel l'émission s'effectue. En effet, chaque positon émis à partir de la substance s'annule avec un électron libre, produisant deux photons se déplaçant à la vitesse de la lumière dont les directions sont (pratiquement) opposées. La paire de photons émis est au même instant détectée par une paire de détecteurs, qui définit un volume cylindrique appelée *le tube détecteur* ou simplement *le tube* (voir figure 6.1). Précisons que seulement une petite fraction des photons émis est détectée par les tubes. Pour les autres photons, soit ils ne sont pas interceptés par les tubes détecteurs, soit ils sont atténués par le corps. Les données collectées sont donc le nombre d'émissions de photons observées dans chaque tube formé par deux détecteurs du scanner. A partir de ces données, le but est d'obtenir la distribution spatiale des intensités des émissions de photons $\lambda(\mathbf{x})$, où $\mathbf{x}$ est un point dans la région $B \subset \mathbb{R}^3$, qui définit le cerveau. Précisons qu'une intensité $\lambda(\mathbf{x})$ correspond au nombre d'émissions de photons au point $\mathbf{x}$ par unité de volume et par unité de temps. Vardi, Shepp et Kaufman [127] ont développé un modèle statistique (VSK) basé sur le comportement physique des positons et sur la géométrie du scanner pour décrire ce problème de reconstruction d'image en T.E.P. comme un problème standard de données incomplètes (voir Chapitre 4). Suite à cette reformulation, l'algo-

rithme E.M. [52] est naturellement utilisé pour résoudre ce problème difficile, c'est-à-dire trouver les intensités $\lambda(\mathbf{x})$. De ce fait, il nous semble intéressant de comparer les schémas présentés dans le chapitre précédent à l'algorithme EM pour ce problème en tomographie. Le chapitre est organisé comme suit : dans la première section, nous décrivons une version simplifiée du modèle VSK et donnons la formulation de l'algorithme EM. Dans la deuxième section, nous rappelons brièvement les différentes méthodes testées. Finalement, dans la dernière section, nous présentons les résultats numériques et comparons les méthodes à la fois sur des critères numériques et visuels.

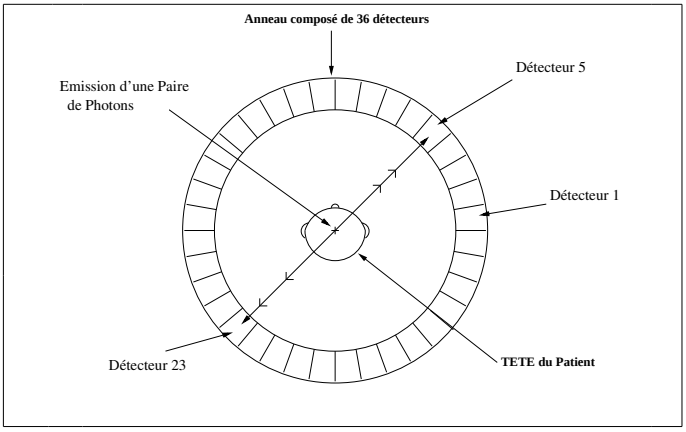


FIG. 6.1 – Émission d'une paire de photons observée dans le tube formé par les détecteurs 5 et 23 (Vue du dessus)

6.2 Modèle V.S.K. et l'algorithme E.M.

Soit $y = (y_1, \ldots, y_d)$, où y_i est le nombre d'émissions d'une paire de photons détectés dans le tube détecteur i, et d est le nombre total de tubes. Le vecteur y correspond ainsi au vecteur des données mesurées ou observées. La région $B \subset \mathbb{R}^3$, qui représente le cerveau, est discrétisée en s mailles de forme carrée appelées *pixels* et les intensités des émissions sont estimées au centre de chaque pixel. Nous notons alors λ_j l'intensité des émissions des photons au pixel j et $\lambda = (\lambda_1, \ldots, \lambda_s)$ le vecteur inconnu des intensités. Vardi, Shepp et Kaufman [127, Eq. (2.1)] ont supposé que les variables y_i sont des variables indépendantes suivant une loi de Poisson avec une moyenne $\sum_{j=1}^{s} c_{ij}\lambda_j$, c'est-à-dire

$$y_i \sim \text{Poisson}\left(\sum_{j=1}^{s} c_{ij}\lambda_j\right), \quad \text{pour } i = 1, \ldots, d \tag{6.1}$$

où j désigne un pixel, s est le nombre total de pixels, c_{ij} est la probabilité d'une émission à partir du pixel j détectée par le tube i. La matrice de transition, C, dont les éléments sont les c_{ij} ($i = 1, \ldots, d$ et $j = 1, \ldots, s$), est connue et peut être déterminée sous certaines

conditions basées sur la géométrie du cerveau et la configuration du scanner. Sans perte de généralité, nous pouvons supposer que

$$\sum_{i=1}^{d} c_{ij} = 1, \quad \text{pour tout } j \in [1, \ldots, s]. \tag{6.2}$$

Suite à ces définitions, la fonction de densité de probabilité des données observées est

$$g(y;\lambda) = \prod_{i=1}^{d} e^{-\left(\sum_{j=1}^{s} c_{ij}\lambda_j\right)} \frac{\left(\sum_{j=1}^{s} c_{ij}\lambda_j\right)^{y_i}}{y_i!}. \tag{6.3}$$

Reformulons maintenant ce problème en un problème de données incomplètes pour pouvoir appliquer la théorie de l'algorithme E.M. (Chap. 4). Notons Z la matrice de dimension $d \times s$, dont les éléments z_{ij} correspondent au nombre d'émissions au pixel j détectées par le tube i. Il suit que la somme des éléments de la ligne i, z_{i+}, est le nombre d'émissions détectées dans le tube i, qui est exactement y_i. Il est alors naturel de voir y_i comme des données observées ou incomplètes, et les éléments z_{ij} comme les données complètes. La fonction de densité de probabilité pour les données complètes peut être écrite de la façon suivante ([127, p. 21]) :

$$g_c(Z;\lambda) = \prod_{j=1}^{s} \left(\frac{e^{-\lambda_j}\, \lambda_j^{z_{+j}}}{z_{+j}!} \prod_{i=1}^{d} c_{ij}^{z_{ij}} \right) \tag{6.4}$$

où pour tout $j \in [1,\ldots,s]$, $z_{+j} = \sum_{k=1}^{d} z_{kj}$, est la somme des éléments de la colonne j de la matrice Z. L'équation (6.4) exprime le fait que la probabilité d'une détection par un tube i d'un photon émis au pixel j est simplement le produit de la probabilité d'une émission au pixel j et de la probabilité d'une détection par le tube i sachant que l'émission se fait au pixel j. Par définition des variables z_{+j} et par (6.2), l'équation (6.4) peut être réécrite de la façon suivante

$$g_c(Z;\lambda) = \prod_{j=1}^{s} \prod_{i=1}^{d} \frac{(\lambda_j\, c_{ij})^{z_{ij}}\, e^{-\lambda_j\, c_{ij}}}{z_{+j}!}. \tag{6.5}$$

Le logarithme de la fonction de densité de probabilité des données complètes est donc donné par

$$L_c(\lambda;Z) = \log g_c(Z;\lambda) = \sum_{j=1}^{s} \sum_{i=1}^{d} \left(z_{ij} \log(\lambda_j\, c_{ij}) \; - \; \lambda_j\, c_{ij}\right) + \text{constante}. \tag{6.6}$$

Pour obtenir l'algorithme EM, nous devons dans un premier temps calculer l'espérance de $L_c(\lambda;Z)$, sachant les données observées y_i. Soit $\lambda^{(0)} = (\lambda_1^{(0)}, \ldots, \lambda_s^{(0)})$ un vecteur initial pour le vecteur intensité λ. Alors, pour la première itération de l'algorithme EM, l'étape E requiert le calcul de la quantité

$$Q(\lambda;\lambda^{(0)}) = E\left[L_c(\lambda;Z); y; \lambda^{(0)}\right].$$

Par l'équation (6.6) et par les règles de calcul de l'espérance conditionnelle, nous avons

$$Q(\lambda;\lambda^{(0)}) = \sum_{j=1}^{s}\sum_{i=1}^{d}\left(z_{ij}^{(0)}\log(\lambda_j\, c_{ij}) \;-\; \lambda_j\, c_{ij}\right) + \text{constante}$$

avec

$$z_{ij}^{(0)} := E\left[z_{ij}; y; \lambda_j^{(0)}\right] = \frac{c_{ij}\lambda_j^{(0)}}{\sum_{k=1}^{s} c_{ik}\lambda_k^{(0)}}\;\; y_i. \tag{6.7}$$

L'espérance conditionnelle des données complètes z_{ij}, donnée par l'équation (6.7) peut être comprise comme suit. La probabilité qu'un photon soit détecté dans le tube i sachant qu'il a été émis au pixel j, est donnée par la fraction

$$\frac{c_{ij}\lambda_j^{(0)}}{\sum_k c_{ik}\lambda_k^{(0)}}.$$

De ce fait, l'espérance du nombre de photons émis au pixel j et détectés dans le tube i, est simplement le produit de cette fraction avec le nombre de détections dans le tube i. Nous devons maintenant calculer le vecteur $\lambda^{(1)}$ qui maximise la quantité $Q(\lambda;\lambda^{(0)})$, ce qui correspond à l'étape M de l'algorithme EM. En écrivant cette quantité et en posant sa dérivée par rapport à la variable λ_j égale à zéro, nous obtenons

$$\lambda_j^{(1)} = \sum_{i=1}^{d} z_{ij}^{(0)} \qquad \text{pour tout } j = [1,\ldots,s].$$

Les étapes E et M sont répétées mais cette fois-ci avec le vecteur $\lambda^{(0)}$ remplacé par le vecteur $\lambda^{(1)}$. Pour la $(m+1)^{\text{ième}}$ itération, l'algorithme E.M. est donnée par

- $$z_{ij}^{(m)} = \frac{c_{ij}\lambda_j^{(m)}}{\sum_{k=1}^{s} c_{ik}\lambda_k^{(m)}}\;\; y_i.$$
- $$\begin{aligned}\lambda_j^{(m+1)} &= \sum_{i=1}^{d} z_{ij}^{(m)} \\ &= \lambda_j^{(m)}\sum_{i=1}^{d}\left(c_{ij}y_i \Big/ \sum_{k=1}^{s} c_{ik}\lambda_k^{(m)}\right),\ \text{pour } j\in[1,\ldots,s].\end{aligned} \tag{6.8}$$

Par suite, l'algorithme EM définit bien une fonction $F:\mathbb{R}^s\to\mathbb{R}^s$ telle que

$$\lambda^{(m+1)} = F(\lambda^{(m)}), \qquad \lambda^{(0)} \text{ donné.} \tag{6.9}$$

Trouver le vecteur intensité λ^* pour ce problème de tomographie est par conséquent équivalent à trouver la solution du problème de point fixe $F(\lambda^*)=\lambda^*$ (voir aussi le chapitre 4 et l'équation (4.5)). Les méthodes d'accélération présentées dans le chapitre précédent peuvent donc être appliquées sur ce problème et comparées à l'algorithme EM. Avant de présenter les résultats numériques, récapitulons brièvement les méthodes testées.

6.3 Synthèse des méthodes squarem

Le but de cette section est juste de remémorer au lecteur les méthodes présentées dans le chapitre précédent et testées pour ce problème en tomographie. Pour plus de précisions, le lecteur est convié à se référer au chapitre précédent. Soit F la fonction définie par les équations (6.8) et (6.9). Nous posons $\lambda^{(n)} \in \mathbb{R}^s$ l'itéré d'une méthode à l'itération n, $r^{(n)} = F(\lambda^{(n)}) - \lambda^{(n)}$ le résidu et $v^{(n)} = F(F(\lambda^{(n)})) - 2F(\lambda^{(n)}) + \lambda^{(n)}$. Les méthodes notées RRE1 et MPE1 sont de la forme suivante

$$\lambda^{(n+1)} = \lambda^{(n)} - \alpha_n r^{(n)}$$

avec

$$\alpha_n = \frac{|| r^{(n)} ||^2}{(r^{(n)}, v^{(n)})} := \alpha_n^{MPE1} \quad \text{pour la méthode MPE1}$$

et

$$\alpha_n = \frac{(r^{(n)}, v^{(n)})}{|| v^{(n)} ||^2} := \alpha_n^{RRE1} \quad \text{pour la méthode RRE1.}$$

Les autres méthodes notées SqMPE1, SqRRE1 et SqHyb1 sont de la forme

$$\lambda^{(n+1)} = \lambda^{(n)} - 2\alpha_n r^{(n)} + \alpha_n^2 v^{(n)}$$

avec

$\alpha_n = \alpha_n^{MPE1}$ pour la méthode SqMPE1,

$\alpha_n = \alpha_n^{RRE1}$ pour la méthode SqRRE1,

$\alpha_n = w_n \alpha_n^{MPE1} + (1 - w_n)\alpha_n^{RRE1}$ où $w_n = \left| \frac{\alpha_n^{RRE1}}{\alpha_n^{MPE1}} \right|^{1/2}$ pour la méthode SqHyb1.

6.4 Résultats

Ici nous considérons un problème de reconstruction d'image en T.E.P. avec une géométrie simple pour la région B définissant le cerveau. Précisons que dans la littérature de la tomographie, le modèle mathématique qui simule une partie du corps est appelé un *fantôme mathématique* ou tout simplement un *fantôme*. Dans notre cas, le fantôme du cerveau sera de forme carrée plutôt qu'un fantôme de forme ovale qui est plus réaliste. Cette différence dans la géométrie et la dimension (deux au lieu de trois) n'a aucune conséquence sur l'évaluation des schémas numériques et la reconstruction d'image. Rappelons que notre principal but est de comparer les taux de convergence des schémas numériques pour trouver le vecteur intensité λ^* qui n'est autre que le point fixe de la fonction F définie par les itérations de l'algorithme E.M. (Equation (6.9)). Nous définissons alors B par

$$B = \left\{ (x, y) \in \mathbb{R}^2 : |\, x \,| < 1 \;\; et \;\; |\, y \,| < 1 \right\}$$

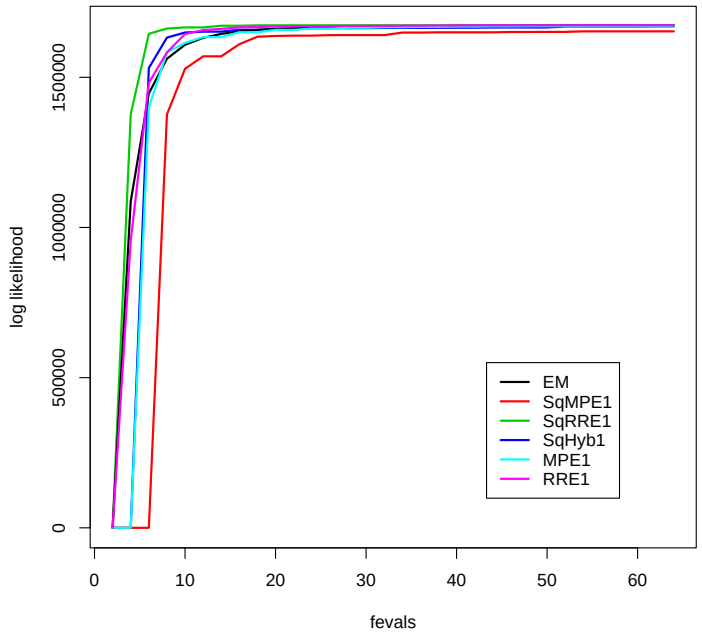


FIG. 6.2 – Évolution du logarithme de la fonction de densité de probabilité des données observées $\mathcal{L}$ pour les différents schémas numériques

qui sera discrétisé par 32 mailles de même taille dans chacune des directions. Le nombre de pixels s est par suite donné par $s = 32 \times 32 = 1024$. Nous identifions le scanner à un simple anneau avec 32 segments espacés de façon uniforme représentant les détecteurs et nous choisissons le nombre de tubes d dans lesquels le nombre d'émission de photons détectée est non nul, à 496. Par suite, la matrice de transition C sera de dimension 496×1024. Les éléments de cette matrice peuvent être calculés en utilisant des arguments géométriques, en particulier basés sur les représentations choisies pour le cerveau et le scanner. Mais ici, nous les simulons. Pour chaque pixel j, nous générons aléatoirement une direction entre $[0, 2\pi]$, et déterminons le tube i qui contient ce rayon ou direction. Pour déterminer ce tube, il suffit de calculer les deux points d'intersection de la droite contenant le rayon avec l'un des cercles formant l'anneau des détecteurs. Nous incrémentons alors c_{ij} de 1. Nous répétons cette opération 10000 fois. Enfin, chaque élément de la $j^{\text{ème}}$ colonne de la matrice C est divisé par 10000 afin de satisfaire l'hypothèse de l'équation (6.2). Cette procédure est répétée pour chaque pixel j avec $j \in [1, \ldots, 1024]$. Pour les données observées $y = (y_1, \ldots, y_d)$, nous avons choisi un échantillon de taille 10 millions, c'est-à-dire le nombre total d'émissions de photons correspondant à la somme des y_i, $\sum_{i=1}^{d} y_i$ est fixé à 10 millions. Pour le vecteur initial $\lambda^{(0)}$ de chaque schéma, nous choisissons le vecteur dont toutes les composantes sont égales à $\left(\sum_{i=1}^{d} y_i\right)/s$. La comparaison des différents schémas est basée

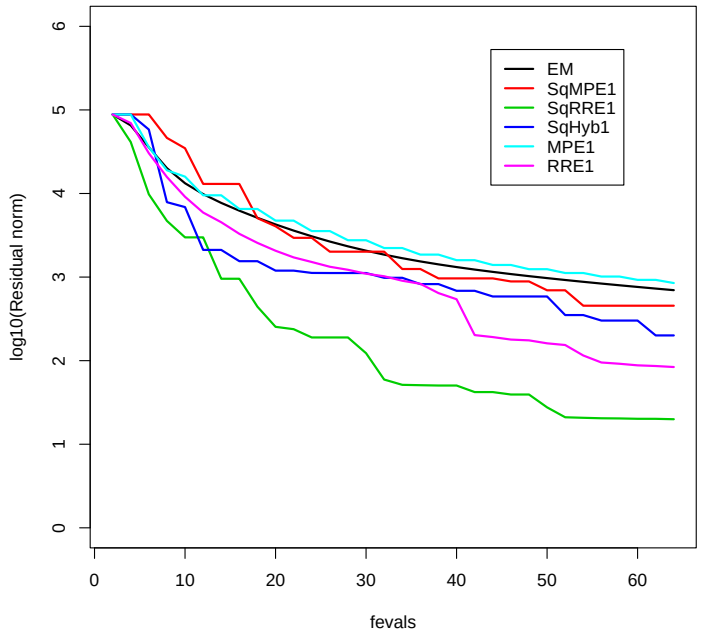


FIG. 6.3 – Évolution de la norme du résidu R pour les différents schémas numériques

sur les trois critères suivants

1. l'évolution de la norme du résidu R définie à l'itération n par

$$R = || F(\lambda^{(n)}) - \lambda^{(n)} ||^2 \tag{6.10}$$

   où F est la fonction définie par les itérations EM, Eq. (6.8), $\lambda^{(n)}$ est l'itéré d'un schéma à l'itération n et $||\,.\,||$ désigne la norme euclidienne,

2. l'évolution du logarithme de la fonction de densité de probabilité des données observées $\mathcal{L}$ (Eq. (6.3))

$$\mathcal{L} = \log\, g(y;\lambda^{(n)}) = \sum_{i=1}^{d} \left(y_i \log \hat{y}_i^{(n)} + \hat{y}_i^{(n)} \right) + \text{constante} \tag{6.11}$$

   où $\hat{y}_i^{(n)} = \sum_{j=1}^{s} c_{ij}\lambda_j^{(n)}$ et $\lambda_j^{(n)}$ est la composante j de l'itéré $\lambda^{(n)}$ du schéma,

3. la différence $\mathcal{D}$, entre les nombres d'émission de photons dans les tubes observés et calculés

$$\mathcal{D} = \sum_{i=1}^{d} (y_i - \hat{y}_i^{(n)})^2. \tag{6.12}$$

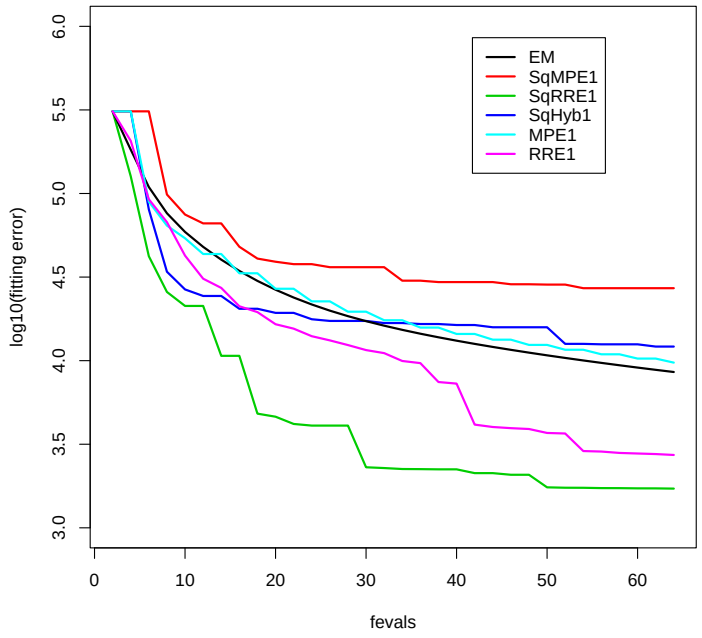


FIG. 6.4 – Évolution de la norme de l'erreur $\mathcal{D}$ pour les différents schémas numériques

Les figures 6.2, 6.3 et 6.4 présentent les résultats obtenus par l'algorithme E.M. et les différents schémas d'extrapolation. La figure 6.2 montre que tous les schémas convergent bien vers le même maximum pour le logarithme de la fonction de densité de probabilité des données observées et par équivalence, vers le point fixe de la fonction F définie par l'équation (6.8). Ceci est d'ailleurs confirmé par les figures 6.3 et 6.4 qui montrent à la fois que la norme du résidu $|| F(\lambda^{(n)}) - \lambda^{(n)} ||$ et la différence entre le nombre d'émissions observées et calculées, diminuent à chaque itération. Dans ces trois figures, les différents schémas sont stoppés au bout de 64 itérations. Par contre, dans le tableau 6.1, nous donnons le nombre d'évaluations de la fonction F nécessaires à chaque schéma pour vérifier la condition suivante $R = || \lambda^{(n)} - F(\lambda^{(n)}) || \ / \ || \lambda^{(0)} - F(\lambda^{(0)}) || < 10^{-4}$. Les méthodes squarem, SqRRE1 et SqMPE1 convergent plus vite que leurs homologues à un pas, RRE1 et MPE1 et l'algorithme EM. Ceci montre une fois de plus l'intérêt des méthodes squarem et l'efficacité de la technique *squaring* développée dans le chapitre précédent. Un fait intéressant est d'observer la supériorité des schémas de type RRE par rapport à ceux de type MPE. Ce phénomène a déjà été observé dans les expériences numériques du chapitre précédent. Le schéma squarem hybride Sqhyb, issu des expériences numériques est, quant à lui, légérement meilleur que l'algorithme EM et les schémas de type MPE, mais pas aussi efficace que les schémas de type RRE. Précisons qu'aucune des méthodes ne souffre

Schémas	Nombre d'évaluations de la fonction F
EM	1336
RRE1	392
MPE1	1512
SqRRE1	96
SqMPE1	690
SqHyb1	920

TAB. 6.1 – Nombre d'évaluations nécessaire à chaque schéma pour satisfaire $|| \lambda^{(n)} - F(\lambda^{(n)}) || \; / \; || \lambda^{(0)} - F(\lambda^{(0)}) || < 10^{-4}$

de divisions par zéro ou/et de stagnation. Ainsi, les méthodes de type RRE, en particulier, la version squarem, semblent donc être les plus efficaces et peuvent être considérées comme une alternative intéressante à l'algorithme EM. Les résultats présentés dans les figures 6.2, 6.3 et 6.4 évaluent les schémas sur leurs capacités à satisfaire correctement certains critères numériques. Évaluons désormais les schémas en comparant les images qu'ils produisent (figures 6.5 à 6.10). Nous rappelons la différence entre le problème mathématique de maximisation de la fonction de densité de probabilité et le problème pratique de la reconstruction d'images, qui devront être interprétées par des experts en diagnostics médicaux. Le problème mathématique est bien résolu et notre objectif est atteint. Ces reconstructions d'image sont ainsi données pour comprendre la difficulté du problème et pour montrer les limites du modèle statistique. Chaque figure est divisée en quatre parties. Le champ d'intensité réel λ^*, c'est-à-dire la solution de notre problème, est donné dans le premier cadre (quadrant en haut et à gauche). Précisons que ce champ d'intensité λ^* a été utilisé pour générer les données observées y_i. Les 3 autres cadres correspondent aux images produites par les schémas à un nombre d'évaluations donné de la fonction F. Nous utilisons les couleurs "chaudes" (blanc à rouge) pour traduire l'activité plus ou moins élevée du cerveau : la couleur blanche traduit des zones de faible activité (autour de 2000 émissions de photons) et la couleur rouge des zones de haute activité (autour de 50000 photons). Par exemple, le premier cadre de la figure 6.5 montre trois zones différentes d'activité. En principe, les schémas doivent être évalués sur la qualité des images qu'ils produisent, c'est-à-dire sur la reconstruction du champ d'intensité. Ce n'est pas aussi évident en pratique, puisque nous ne connaissons pas le champ d'intensité réel. Rappelons que ce dernier est solution du problème de point fixe $F(\lambda^*) = \lambda^*$. La difficulté principale réside dans le choix des critères d'arrêt pour les méthodes utilisées. Si nous effectuons peu d'itérations des schémas, le champ d'intensité obtenu peut être très différent du champ d'intensité réel. Par contre, si trop d'itérations sont effectuées, le champ d'intensité devrait être proche du champ réel, mais nous observons du bruit dans les images. En effet, l'algorithme EM (figure 6.5), mais aussi les schémas d'extrapolation (figures 6.6 à 6.10), capte bien le signal principal, c'est-à-dire les traits dominants de l'image, plutôt rapidement en 10 à 20 itérations. Si l'algorithme EM est utilisé au delà de ce seuil d'itérations, les itérés calculés se rapprochent de la solution du problème et la qualité de l'image commence à se détériorer : l'image devient floue et du bruit apparaît. De ce fait, il est commun en pratique

dans la littérature de la reconstruction d'image de stopper prématurement les itérations de l'algorithme EM pour obtenir une image lisse. Un tel critère d'arrêt est complètement arbitraire. Des approches basées sur des essais numériques ont été développées pour établir des critères d'arrêt objectifs, mais ils ne donnent aucune solution générale pour ce problème de bruits. Ce n'est pas une déficience de l'algorithme EM. Cet algorithme fait en effet ce qu'il est supposé faire, c'est-à-dire maximiser le logarithme de la fonction de densité de probabilité des données observées. Pour éviter ces effets indésirables et pour obtenir des images "lisses", il faudrait régulariser ce problème en imposant des conditions de lissage. En d'autres termes, nous serions amenés à résoudre un autre problème de point fixe mais ce n'est pas le but de notre chapitre. Pour les méthodes d'extrapolation, puisqu'elles sont plus rapides pour résoudre le problème de point fixe ou par équivalence le problème de maximisation de la fonction de densité de probabilité, elles seront stoppées plus prématurement que l'algorithme EM. En effet, puisque la convergence vers la solution est plus rapide, les méthodes tendent à produire des images qui exhibent du bruit plus tôt que l'algorithme EM. Ce phénomène est confirmé par chaque figure (6.6 à 6.10) en comparaison à la figure 6.5. Par exemple, pour l'algorithme EM (figure 6.5) et la SqRRE (figure 6.9), nous remarquons que du bruit apparaît pour l'algorithme EM au bout de 32 évaluations de la fonction F, tandis qu'il apparaît à la huitième évaluation de la fonction F pour le schéma SqRRE1.

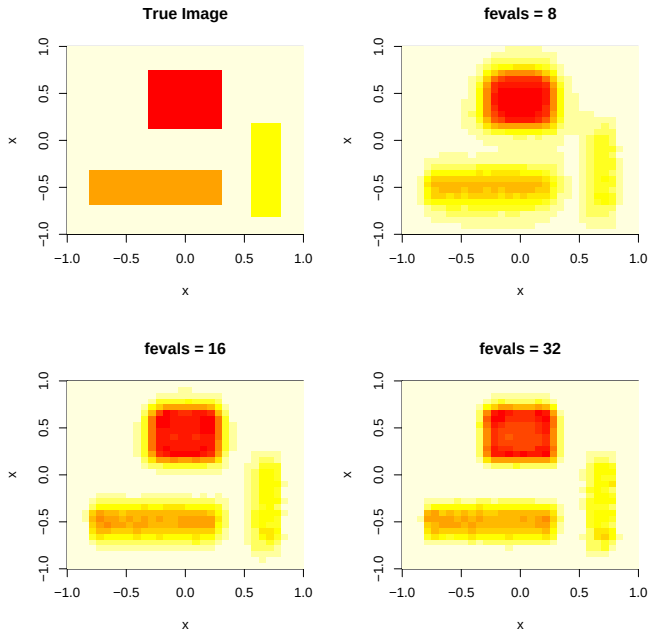


FIG. 6.5 – Comparaison entre l'image exacte et la suite d'images produites par l'algorithme EM à 8, 16 et 32 évaluations de la fonction F

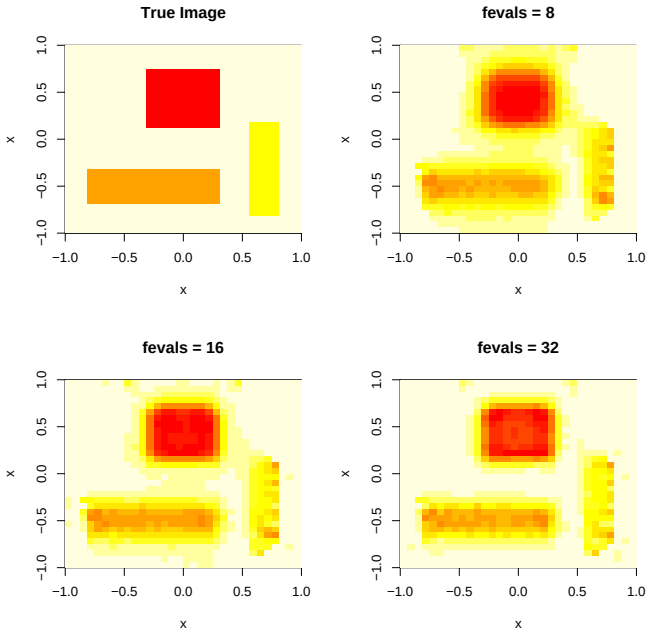


FIG. 6.6 – Comparaison entre l'image exacte et la suite d'images produites par le schéma MPE1 à 8, 16 et 32 évaluations de la fonction F

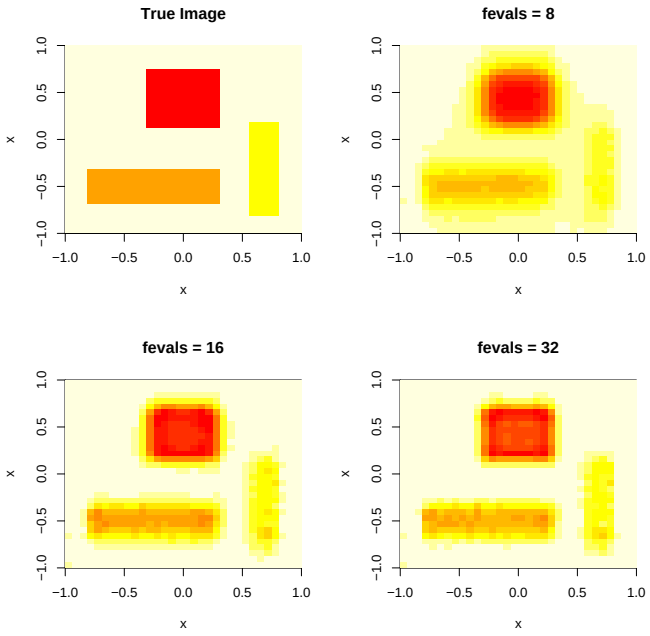


FIG. 6.7 – Comparaison entre l'image exacte et la suite d'images produites par le schéma RRE1 à 8, 16 et 32 évaluations de la fonction F

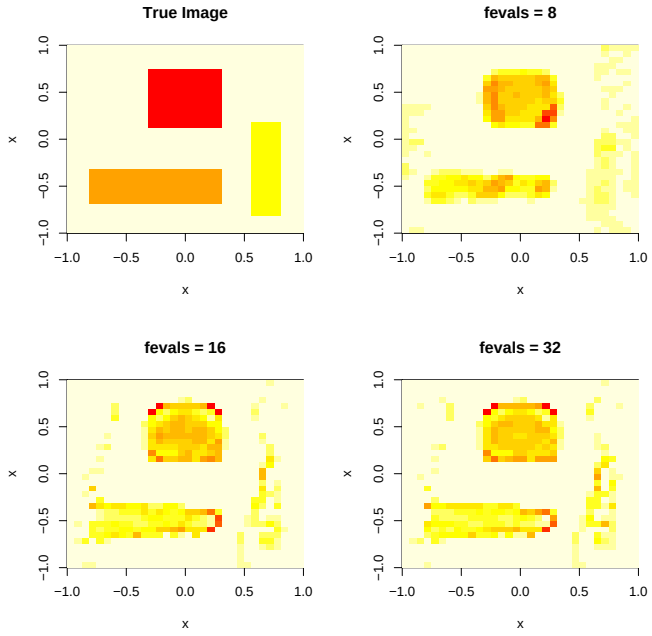


FIG. 6.8 – Comparaison entre l'image exacte et la suite d'images produites par le schéma SqMPE1 à 8, 16 et 32 évaluations de la fonction F

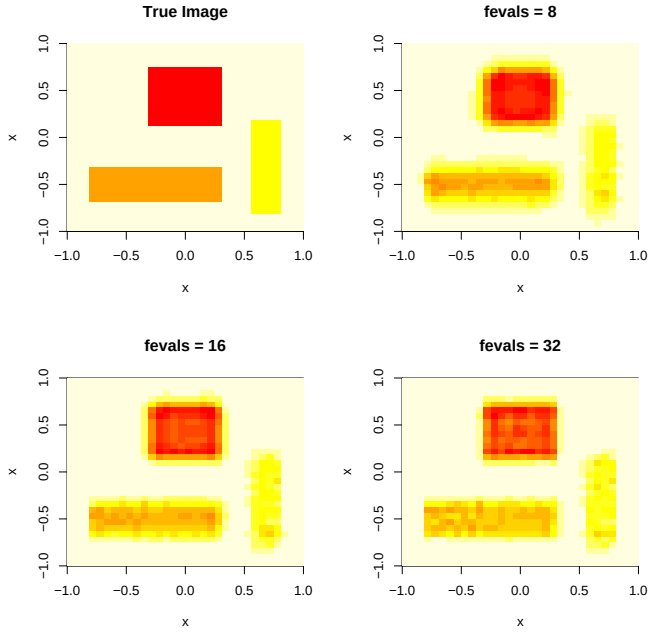


FIG. 6.9 – Comparaison entre l'image exacte et la suite d'images produites par le schéma SqRRE1 à 8, 16 et 32 évaluations de la fonction F

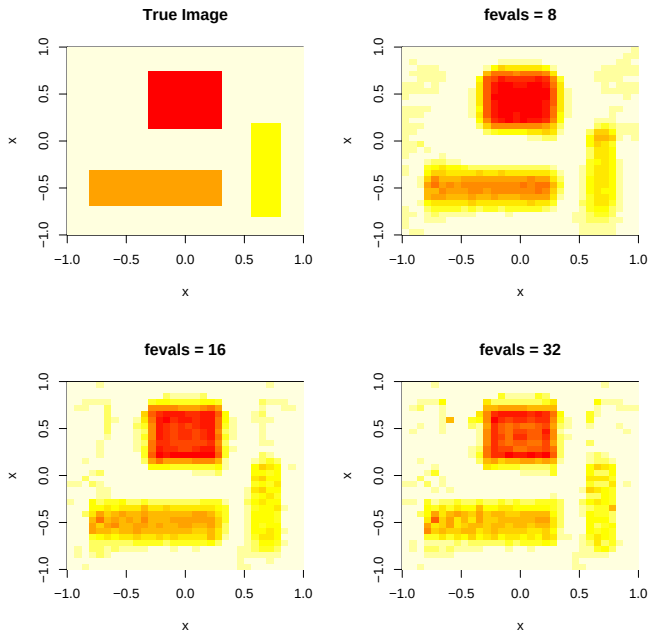


FIG. 6.10 – Comparaison entre l'image exacte et la suite d'images produites par le schéma SqHyb1 à 8, 16 et 32 évaluations de la fonction F

6.5 Conclusion

Dans ce chapitre, nous nous sommes intéressés à un problème en tomographie, qui est une technique très utilisée pour visualiser l'activité d'un organe, en particulier ici le cerveau. Ce problème en T.E.P. a été formulé par Vardi, Shepp et Kaufman [127] sous forme d'un problème de données incomplètes pour pouvoir utiliser la théorie de l'algorithme EM. Malgré quelques propriétés intéressantes (stabilité numérique et croissance monotone de la fonction de densité de probabilité des données observées), l'algorithme EM converge lentement. Pour cette raison, les méthodes squarem ont été comparées à l'algorithme EM et, une fois de plus, leur supériorité a été montrée numériquement. Suite à ces résultats prometteurs, plusieurs perspectives sont envisagées. En particulier, il semblerait intéressant de comparer les méthodes squarem à une alternative de l'algorithme EM, appelée *Order Subset E.M.* (OSEM) [72] qui est autant utilisée en pratique que l'algorithme EM pour résoudre ces problèmes en tomographie. Il faudrait aussi construire la matrice des probabilités C non plus par des simulations, mais en prenant en compte la géométrie du scanner. Enfin, il faudrait régulariser ce problème, c'est-à-dire trouver des conditions initiales à imposer afin d'obtenir par la suite des images moins bruitées. Plusieurs tests sont en cours afin de répondre à ces différentes perspectives.

Troisième partie

Une Étude en Physique des Plasmas

Chapitre 7

An Adaptive Particle-In-Cell method using multi-resolution analysis [1]

In this paper, we introduce a new PIC method based on an adaptive multi-resolution scheme for solving one dimensional Vlasov-Poisson equation. Our approach is based on a description of the solution by particles of unit weight and on a reconstruction of the density at each time step of the numerical scheme by an adaptive wavelet technique : the density is firstly estimated in a proper wavelet basis as a distribution function from the current empirical data and then "de-noised" by a thresholding procedure. The so-called Landau damping problem is considered for validating our method. The numerical results agree with those obtained by the classical PIC scheme, suggesting that this multi-resolution procedure could be extended with success to plasma dynamics in higher dimensions.

7.1 Introduction

The kinetic motion of a physic plasma of charged particles in which the collisions between particles are neglected is usually modeled by the Vlasov equation [41],

$$\frac{\partial f}{\partial t} + v \cdot \nabla_x f - (E \cdot \nabla_v) f = 0, \qquad (x, v) \in \mathbb{R}^d \times \mathbb{R}^d, \tag{7.1}$$

$f(x, v, t)$ being the distribution function and E the electrostatic field.. The self-consistent field produced by the charge of the particles is

$$E_{self}(x,t) = -\nabla_x \phi_{self}(x,t) \quad \text{where} \quad -\Delta_x \phi_{self} = \rho(x,t), \quad \rho(x,t) = \int f(x,v,t)\, dv. \tag{7.2}$$

The system is closed with an initial data $f(x, v, 0) = f_0(x, v)$ and some decay conditions for the Poisson equation.

1. Ce chapitre est le fruit du projet APICIB du Cemracs 2003 proposé par E. Sonnendrücker et A. Cohen. Il a donné lieu à une publication [45] dont les auteurs sont J.P. Chehab, A. Cohen, D. Jennequin, J.J. Nieto, J. Roche et Ch. Roland. Par respect pour le travail de chaque auteur, ce chapitre est laissé sous sa forme d'article.

If $E = E_{self}$, this model is clearly dispersive due to the repulsive forces and then, in order to confine the particles in a bounded domain, as usual, we consider an additional given external potential $\phi_{ext}(x)$ and rewrite E as :

$$E(x,t) = E_{self} + E_{ext} := -(\nabla_x \phi_{self} + \nabla_x \phi_{ext}). \tag{7.3}$$

In this project we are interested in the numerical resolution of the repulsive VP system (7.1)-(7.2)-(7.3) endowed with an appropriate initial data by means of the *particle in cell* (PIC) method. In the classical PIC method (see [48, 97]) the initial data is approximated by set of particles, and the method aims to follow the trajectories (the characteristic curves) of these particles. In order to reflect the distribution function f_0, the initial set of particles can either be uniformly distributed and weighted with the value of f_0 at the corresponding point, or distributed randomly according to the distribution function f_0 and identically weighted. In this paper we follow the second approach.

The main difficulty in the PIC method lies in the construction of the characteristic curves

$$\begin{cases} \dfrac{dX(t)}{dt} = V(t), \\ \\ \dfrac{dV(t)}{dt} = E(X(t),t), \end{cases}$$

because of the nonlinearity due to the self-consistent potential. This requires to rebuild the charge density ρ at each time step in order to solve the Poisson equation and to obtain the electric field. Generally, this method gives good results with a relatively small number of particles but produces some numerical noise which prevent from describing precisely the tail of the distribution function. To overcome this drawback, it has been proposed to solve the problem by Eulerian methods [9] or semi Lagrangian methods [15]. Here, we propose to combine the PIC method with density estimation techniques based on wavelet thresholding [54] in order to reduce the noise level : the density $\rho(x,t)$ is estimated at time $t > 0$ by an expansion of the type

$$\sum_k \hat{c}_{J_1,k}\ \varphi_{J_1,k}(x) + \sum_{j=J_1}^{J_0} \Big(\sum_{k=0}^{2^j-1} T_\eta(\hat{d}_{j,k})\ \psi_{j,k}(x) \Big). \tag{7.4}$$

Here $\varphi_{J_1,k}$ and $\psi_{j,k}$ are the scaling functions and the wavelets, respectively. The scaling and detail coefficients $\hat{c}_{J_1,k}$ and $\hat{d}_{j,k}$ are estimated from the particle distribution at time t, and T_η is a thresholding operator at level η. Both the threshold level η and the finest resolution level J_0 are chosen depending on the number of particles.

This paper is organized as follows : in section 7.2 we describe how the density is estimated by wavelets, in particular we present several thresholding strategies. Then, in section 7.3 we present the new numerical scheme, making a comparison with the classical PIC method. In section 7.4, we give a numerical illustration with the simulation of the so-called Landau damping.

7.2 Density estimation by wavelet thresholding

Wavelet decompositions have been widely studied since the last two decades both from the theoretical and practical point of view. In a nutshell, these decompositions are based on a hierarchy of nested approximation spaces $(V_j)_{j\geq 0}$ which should be thought as finite element spaces of mesh size $h \sim 2^{-j}$, endowed with a nodal basis of the form $\varphi_{j,k} := 2^{j/2}\varphi(2^j \cdot -k)$. The functions $\varphi_{j,k}$ are often referred to as primal scaling functions. A projector onto V_j is of the form

$$P_j f := \sum_k c_{j,k}\varphi_{j,k} \text{ with } c_{j,k} := \langle f, \tilde{\varphi}_{j,k}\rangle, \tag{7.5}$$

where $\tilde{\varphi}_{j,k}$ are dual scaling functions. The primal and dual wavelets $\psi_{j,k}$ and $\tilde{\psi}_{j,k}$ characterize the update between two successive level of approximation in the sense that

$$P_{j+1}f - P_j f := \sum_k d_{j,k}\psi_{j,k} \text{ with } d_{j,k} := \langle f, \tilde{\psi}_{j,k}\rangle, \tag{7.6}$$

We refer to [50] for a classical introduction on wavelets, [47] for more information on their application to numerical simulation of PDE's.

In the particular context of PIC methods, we are interested in the reconstruction of the density $\rho(x,t)$ from the locations $(x_i)_{i=1,\cdots,N}$ of the particles at time t. As explained in the introduction, this reconstruction has the form (7.4) where $\hat{c}_{J_1,k}$ and $\hat{d}_{j,k}$ are estimators of the exact coefficients $c_{J_1,k} := \int \rho(x,t)\tilde{\varphi}_{J_1,k}(x)dx$ and $d_{J_1,k} := \int \rho(x,t)\tilde{\psi}_{j,k}(x)dx$ from the empirical distribution according to

$$\hat{c}_{J_1,k} := \frac{1}{N}\sum_{i=1}^{N} \varphi_{J_1,k}(x_i), \tag{7.7}$$

and

$$\hat{d}_{j,k} := \frac{1}{N}\sum_{i=1}^{N} \psi_{j,k}(x_i). \tag{7.8}$$

The interest of the thresholding procedure, in contrast to a simple projection or regularization at a fixed scale j which would compute $\sum_k \hat{c}_{j,k}\varphi_{j,k}$ is twofold : (i) the regularization level j is allowed to vary locally in the sense that the procedure might retain coefficients $d_{j,k}$ at scale j only for some k which typically corresponds to the regions where the density has sharp transitions and requires more resolution, and (ii) the local regularization level automatically adapts to the unknown amount of smoothness of the density through the thresholding procedure which only depends on the number N of samples.

Following Donoho *et al* [54], the maximal scale level J_0 and the threshold η depend on the number of samples according to

$$2^{J_0} \sim N^{1/2} \tag{7.9}$$

and

$$\eta \sim \sqrt{\log(N)/N}. \tag{7.10}$$

Another choice proposed in [54] is a threshold parameter which also depends on the scale level j according to $\eta = \eta_j = K\sqrt{j/N}$. Two techniques are generally used to threshold the details : "hard" thresholding defined by $T_\eta(y) = y\chi_{\{|y|\geq\eta\}}$ and "soft" thresholding defined by $T_\eta(y) = \text{Sign}(y)\max\{0, |y| - \eta\}$. We shall precise thresholding strategies that we choose for our applications in section 7.4.

The density reconstruction method varies with the choice of the wavelet basis. This choice is dictated by two constraints :

1. Numerical simplicity : according to (7.7) and (7.8), the coefficients are estimated through the evaluation of dual scaling functions $\tilde{\varphi}_{J_1,k}$ and dual wavelets $\tilde{\psi}_{j,k}$ at the points x_i. It is therefore useful that these functions have a simple analytical form. In particular, high order compactly supported orthonormal wavelets cannot be used since they do not have an explicit analytical expression.
2. High order accuracy and smoothness : the primal wavelet system should have high order accuracy and smoothness in order to ensure the quality of the approximation of $\rho(x,t)$ by the expansion (7.4).

The choice of the Haar system is good with respect to the first constraint, since in this case the scaling function $\tilde{\varphi} = \varphi$ is simply the box function $\chi_{[0,1]}$, so that the estimation of a scaling coefficient $c_{j,k} = \langle \rho, \tilde{\varphi}_{j,k} \rangle$ simply amounts in counting the points falling in the interval $I_{j,k} = [2^{-j}k, 2^{-j}(k+1)[$:

$$\hat{c}_{j,k} := 2^{j/2}\frac{1}{N}\#\{i\ ;\ x_i \in I_{j,k}\}. \tag{7.11}$$

In particular, we can compute the $\hat{c}_{J_0,k}$ at the finest scale level and use the Haar transform algorithm to compute the $\hat{d}_{j,k}$ according to the classical relations :

$$\hat{c}_{j,k} = \frac{\hat{c}_{j+1,2k} + \hat{c}_{j+1,2k+1}}{\sqrt{2}} \quad \text{and} \quad \hat{d}_{j,k} = \frac{\hat{c}_{j+1,2k} - \hat{c}_{j+1,2k+1}}{\sqrt{2}}. \tag{7.12}$$

However, this choice is not good with respect to the second constraint since piecewise constant functions are low order accurate. In order to fix this defect, while preserving the numerical simplicity of the method, we propose to use a higher order (third order) reconstruction still based on the box function $\chi_{[0,1]}$ as $\tilde{\varphi}$, as proposed by Ami Harten in [68]. This means that the coefficients $\hat{c}_{j,k}$ are still defined by (7.11), but the relation between the approximation and detail coefficients $\hat{c}_{j,k}$ and $\hat{d}_{j,k}$ is modified according to

$$\hat{d}_{j,k} = \frac{1}{\sqrt{2}}\hat{c}_{j+1,2k} - \hat{c}_{j,k} - \frac{1}{8}(\hat{c}_{j,k-1} - \hat{c}_{j,k+1}). \tag{7.13}$$

This is an instance of the so-called *lifting scheme* introduced in [120]. Using this relation, we estimate all the coefficients $c_{j,k}$ and $d_{j,k}$ for $J = J_1, \cdot, J_0 - 1$ and we apply the thresholding operator T_η to the estimated coefficients $\hat{d}_{j,k}$.

It should be remarked that the primal scaling functions $\varphi_{J_1,k}$ and wavelets $\psi_{j,k}$ do not have an explicit analytical expression, in contrast to the dual scaling functions and

wavelets. However, we can reconstruct the estimator (7.4) at arbitrarily fine resolution by applying the reconstruction formulae

$$\hat{c}_{j+1,2k} = \sqrt{2}[\hat{c}_{j,k} - \frac{1}{8}(\hat{c}_{j,k-1} - \hat{c}_{j,k+1}) + T_\eta(\hat{d}_{j,k})], \tag{7.14}$$

and

$$\hat{c}_{j+1,2k+1} = \sqrt{2}\hat{c}_{j,k} - \hat{c}_{j+1,2k}. \tag{7.15}$$

It is also possible to construct wavelet-like multiscale decompositions where both the dual and primal functions have a simple analytical expression, based on the quasi-interpolation operator

$$P_j f := \sum_k c_{j,k}\varphi_{j,k}, \quad c_{j,k} = \langle f, \varphi_{j,k}\rangle, \tag{7.16}$$

where $\varphi = (1-|x|)_+$ is the classical hat function. We therefore estimate the coefficients by

$$\hat{c}_{j,k} := \frac{1}{N}\sum_{i=1}^{N} \varphi_{j,k}(X_i), \tag{7.17}$$

at the finest scale $j = J_0$ and derive them recursively at coarser levels by the formula

$$\hat{c}_{j,k} = \frac{1}{\sqrt{2}}\hat{c}_{j+1,2k} + \frac{1}{2\sqrt{2}}(\hat{c}_{j+1,2k+1} + \hat{c}_{j+1,2k-1}).$$

In this case, the detail components at level j reads

$$(P_{j+1} - P_j)f = \sum_k [\hat{d}^0_{j,k}\varphi_{j+1,2k} + \hat{d}^+_{j,k}\varphi_{j+1,2k+1} + \hat{d}^-_{j,k}\varphi_{j+1,2k-1}], \tag{7.18}$$

with

$$\begin{aligned}
\hat{d}^0_{j,k} &:= \tfrac{\sqrt{2}-1}{\sqrt{2}}\hat{c}_{j+1,2k} - \tfrac{1}{2\sqrt{2}}\hat{c}_{j+1,2k+1} - \tfrac{1}{2\sqrt{2}}\hat{c}_{j+1,2k-1},\\
\hat{d}^+_{j,k} &:= \tfrac{2\sqrt{2}-1}{4\sqrt{2}}\hat{c}_{j+1,2k+1} - \tfrac{1}{2\sqrt{2}}\hat{c}_{j+1,2k} - \tfrac{1}{4\sqrt{2}}\hat{c}_{j+1,2k-1},\\
\hat{d}^-_{j,k} &:= \tfrac{2\sqrt{2}-1}{4\sqrt{2}}\hat{c}_{j+1,2k-1} - \tfrac{1}{2\sqrt{2}}\hat{c}_{j+1,2k} - \tfrac{1}{4\sqrt{2}}\hat{c}_{j+1,2k}.
\end{aligned}$$

The triplet $(d^0_{j,k}, d^+_{j,k}, d^-_{j,k})$ plays the role of the wavelet coefficient and it is jointly thresholded in order to preserve the density mass.

In the sequel of the paper, we shall denote W0 for the first algorithm based on the lifting scheme we have described and W1 for the second algorithm based on the quasi-interpolation operator. In the numerical scheme, we apply these algorithms and reconstruct the denoised density at the finest level J_0 on which we apply the Poisson solver to derive the electric field.

7.3 Numerical schemes

We present here the new scheme we introduce in this paper (PICONU [2]) as a modification of the classical PIC method which will be used to compare the numerical results. Of course, the considered Vlasov-Poisson equation is one dimensional in space and in velocity, so we can write a formal expression using the fundamental solution of the Poisson equation. Indeed, we have

$$-\triangle\Phi_{self} = \rho \Leftrightarrow \Phi_{self} = \frac{1}{2}|x| * \rho \Leftrightarrow E_{self} = \frac{1}{2}\frac{x}{|x|} * \rho.$$

On the other hand, if we denote $X_i(t)$ the position of the i^{th} particle at time t for $i = 1 \dots N$, the density is

$$\rho(x,t) = \frac{1}{N}\Big(\sum_{i=1}^{N} \delta_{X_i(t)}\Big),$$

where δ_ξ stands for the Dirac measure at point ξ. Hence, the self-consistent field E_{self} can be computed by the following formula (written in general dimension d)

$$E_{self}(x,t) = \frac{1}{2N}\Big(\sum_{i=1}^{N} \frac{x - X_i}{|x - X_i|^d}\Big).$$

However, we underline that, in the practical point of view, we can not proceed in such a way in higher dimensions ($d > 1$) due to the singularity of the Green kernel, and that the adaptive method proposed in this paper is aimed at being extended, e.g., to 2-D Vlasov-Poisson problem.

7.3.1 The PIC method

The PIC method consists in following the track to particles with position X_i and velocity V_i along the characteristic curves

$$\begin{aligned} &\frac{dX_i(t)}{dt} = V_i(t), \\ &\frac{dV_i(t)}{dt} = E(X_i(t), t), \\ &X_i(0) = x_i,\ V_i(0) = v_i. \end{aligned}$$

Let f_0 be the initial distribution, the distribution at time $t = T$ is computed as follows :

- Initialization : build (x_i, v_i), N pair of random variables drawn of the initial distribution f_0.
- Time marching scheme : the (nonlinear) characteristic equation is split and integrated as follows : set $\delta t = \frac{T}{N_{max}}$ where N_{max} is the number of time steps, then, for

2. which stands in French for PIC Ondelettes NUmérique

$n = 0, \cdots$ c

$$V^{n+1/2} = V^n + \frac{\delta t}{2} E^n(X^n) \tag{7.19a}$$

$$X^{n+1} = X^n + \delta t\, V^{n+1/2} \tag{7.19b}$$

$$\texttt{Build } \rho^{n+1} \tag{7.19c}$$

$$\texttt{Solve } -\Delta_h \phi^{n+1} = \rho^{n+1} \tag{7.19d}$$

$$E^{n+1}(X^{n+1}) = \nabla_h \phi^{n+1}(X^{n+1}) \tag{7.19e}$$

$$V^{n+1} = V^{n+1/2} + \frac{\delta t}{2}\, E^{n+1}(X^{n+1}) \tag{7.19f}$$

– Build f^{Nmax} by interpolation.

In step 7.19c, we must compute the charge density ρ on the discrete grid points. Two classical methods are Nearest Grid Point (NGP) and Cloud In Cell (CIC) : they consist on $P0$ and $P1$ interpolations respectively. According to Birdsall and Langdon in [18, p.19-23], CIC reduces the noise relative to the NGP. Higher order techniques could be used too, some of them consist on quadratic and cubic spline interpolations. In our numerical results, we will compare our schemes to a PIC method with CIC and NGP density reconstruction.

7.3.2 The adaptive scheme (PICONU)

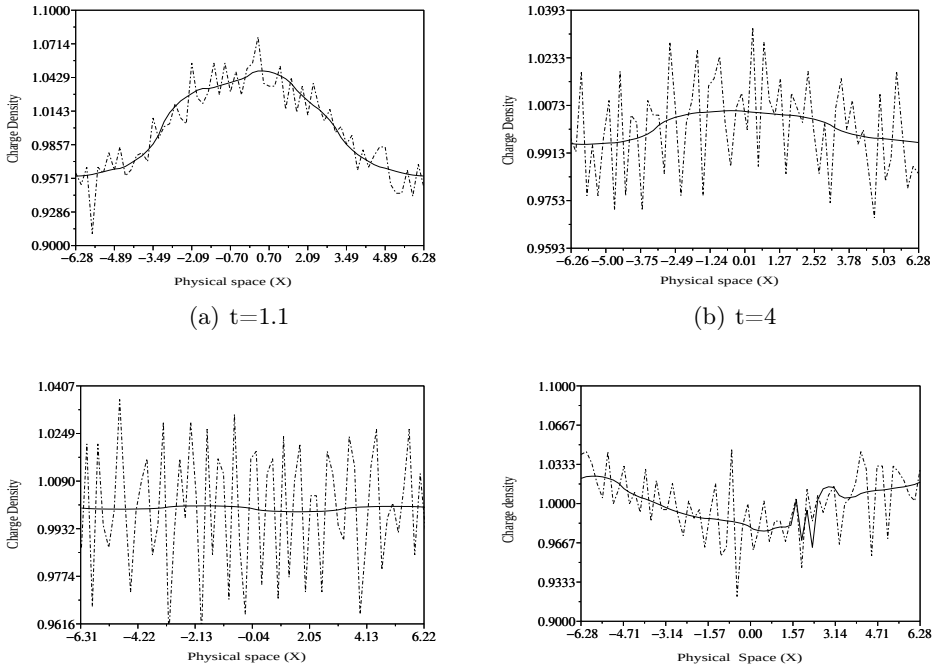
The PICONU scheme differs from the classical PIC method in the step (7.19c). The density is computed by Donoho's technique described in section 7.2. For this method, we have to select the finest and the coarsest resolution level (J_0 and J_1), the threshold and the mesh size for the Poisson equation (7.19d). In the classical density estimation, the noise appears when the mesh size is locally too small. We expect to find the "good threshold" which refines the density mesh only in the region where there are a lot of particles.

7.4 Numerical results

The numerical results presented hereafter were obtained with SCILAB, the (free) numerical software of the INRIA [111]. As a validation of our scheme, we consider the simulation of the so-called Landau damping. This is indeed a significant numerical test, due to its difficulty in simulating, and it has been considered by several authors for validating a code, see, e.g., [18, 58] and references therein. This test consists in the observation of the decay rate of the electrostatic energy obtained when the initial distribution is the perturbed Maxwellian distribution defined by

$$\forall (x,v) \in [-\frac{L}{2}, \frac{L}{2}] \times \mathbb{R}, \quad f_0(x,v) = \frac{1}{\sqrt{2\pi}} \exp\left(-v^2/2{v_{th}}^2\right)(1 + \alpha \cos(kx))$$

where v_{th} is the thermal mean velocity, $L = \dfrac{2\pi}{k}$ and k, α are positive constants with $\alpha \ll 1$. Let us recall that if E denotes the electric field, the electrostatic energy is defined



(a) t=1.1 (b) t=4

(c) t=6.1 (d) t=7.4

FIG. 7.1 – Landau damping with $\alpha = 0.1$. Charge density computed with 26000 particles, in solid line : the PICONU method (W0), in dashed line : the classical PIC (NGP).

by $\int_{-L/2}^{L/2} E^2\,dx$. We consider that the tail of the distribution does not contribute to the problem for $|v| > v_{max}$ for some v_{max} large enough. We will choose $v_{th} = 1$ and $v_{max} = 6$. More precisely, one must observe numerically that

– The decay rate of electrostatic energy defines a line of director coefficient

$$\gamma_L = \sqrt{\frac{\pi}{8}} k^{-3} \exp(-\frac{1}{2k^2} - \frac{3}{2}),$$

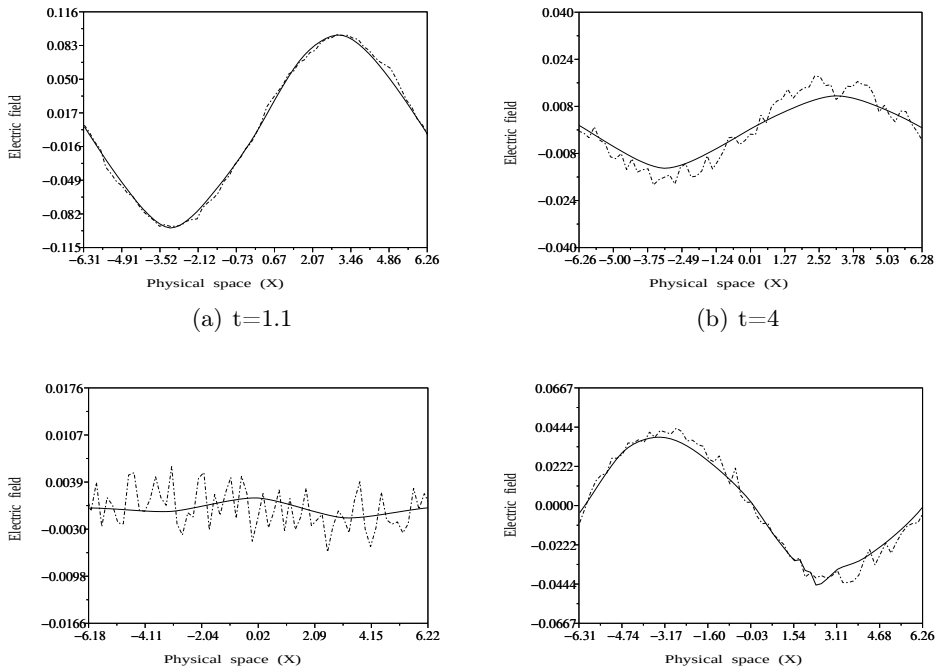
– Oscillations frequency of the electrostatic energy must be

$$\omega^2 = 1 + 3k^2.$$

The Landau damping is very sensitive to the initial distribution and we follow [18] using for that purpose the so-called "quiet start" initialization for α small enough. For all the tests, we do vary only α taking as fixed value $k = 0.5$.

7.4.1 Comparison between NGP and W0.

As a reference for validating our new scheme, we shall compare the results obtained with the classical PIC method where the charge density is computed using the NGP technique described in [18, pp 21,22] to the wavelets build from Haar system. In the finest



(a) t=1.1 (b) t=4

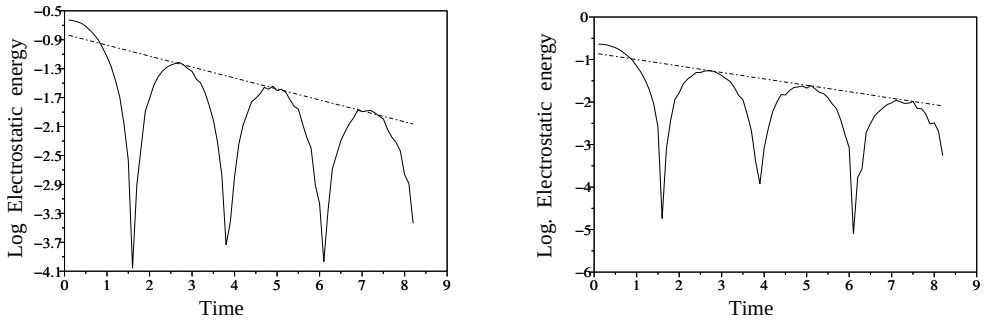
(c) t=6.1 (d) t=7.4

FIG. 7.2 – Landau damping with $\alpha = 0.1$. Electrostatic field computed with 26000 particles, in solid line : the PICONU method(W0), in dashed line :the classical PIC (NGP).

resolution level J_0 is equal to the coarsest one J_1, these two methods are equivalent. In the figures 7.1,7.2 and 7.3 we plotted the charge density, the electrostatic field and the discrete electrostatic energy of the plasma. The graph of the electrostatic energy is in a log-scale and the line corresponds to the theoretical decay of director coefficient γ_L.

We used the following parameters : the time step δt is chosen equal to 0.1. The threshold is the one given in (7.10) and more precisely, we choose $\eta = 0.5 \times \sqrt{j/N}$. The finest and coarsest resolution levels are 6 and 2, this corresponds to about 800 particles per cell. Then we use NGP on a grid of 2^6 intervals.

The NGP method requires a high number of particles per cell. The only way to reduce it is to increase the accuracy of the interpolation. The wavelets, based on the same interpolation, inherit the same problem. However, we observe that wavelets allow to reduce efficiently the noise of the method. This is obvious in figure 7.3 and 7.4 looking at the minima of electrostatic energy : the local minima are obtained theoretically when the electrostatic field vanishes and numerically, these minima have the same magnitude than the discrete L^2-norm of the noise. Furthermore, we observe that the charge density and the electrostatic field are smoother than in the simple $P0$ interpolation case. Whatever noisy is the charge density, the electrostatic field is nearly smooth. It is due to the particular case of the one dimensional integration which has a smoothing effect. In higher dimension, the smoothness will take a greater importance and the use of wavelets should



(a) Classical PIC (NGP) (b) PICONU method (W0)

FIG. 7.3 – Landau damping with $\alpha = 0.1, k = 0.5$. Electrostatic energy

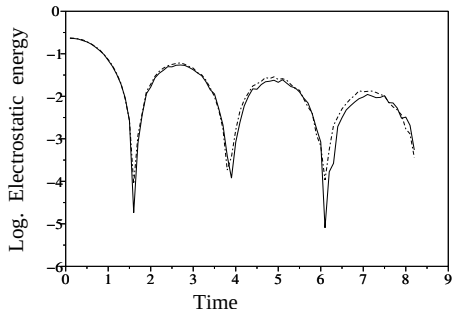
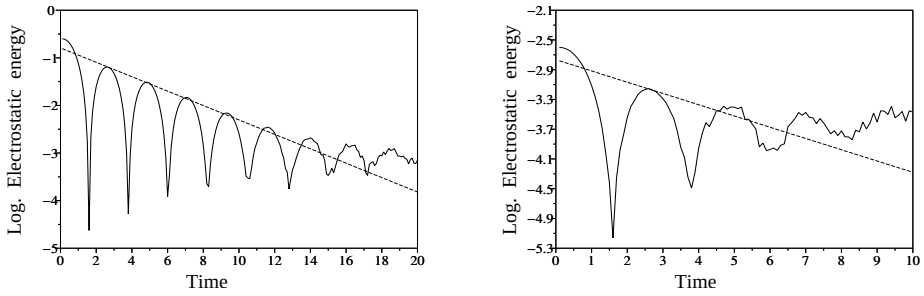


FIG. 7.4 – Superposition of electrostatic energy with 26000 particles. In solid line : damping obtain with PICONU method (W0), In dashed line : classical PIC method (NGP).

be more pertinent.

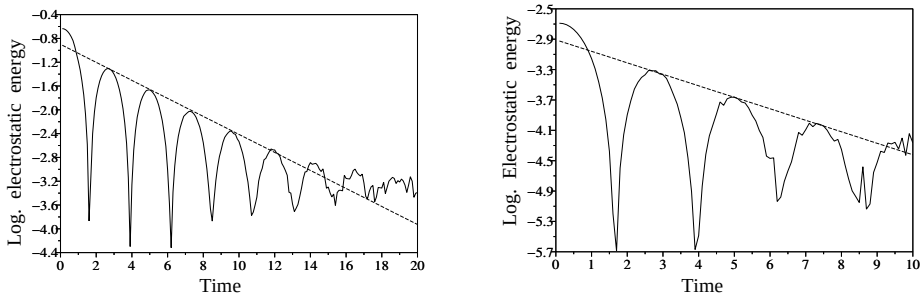
7.4.2 Comparison between CIC and W1.

The simulation parameters are chosen as follows : the highest level of resolution equals 7 and the number of particles equals 10 000. It implies that there is about 80 particles per cell. The time step δt equals 0.1. For the density estimation using wavelets, the coarsest resolution level is equal to 3 and the threshold is this one prescribed by Donoho, that is $K \times \sqrt{j/N}$. The coefficient R_T gives the rate of thresholded coefficients that is the ratio of the mean value of thresholded coefficients at each time step. The figure 7.5 gives the electrostatic energy computed with the classical PIC (CIC). We observe that the classical PIC fails when α becomes small. On the contrary, the adaptive method gives some better results (see figure 7.6). A finer analysis of the threshold shows that all the coefficients are thresholded on the two finest grid. This is a natural consequence of the landau test : the distribution of particles corresponds to a small perturbation of the uniform distribution. Since we use only a first order reconstruction, the charge density is less smooth than in



(a) $\alpha = 0.1$ (b) $\alpha = 0.01$

FIG. 7.5 – linear Landau damping with classical PIC (CIC). 10000 particles, 128 cells.



(a) $\alpha = 0.1, R_T = 13\%, K = 0.5$ (b) $\alpha = 0.01, R_T = 96\%, K = 1.5$

FIG. 7.6 – Linear landau damping, PICONU (W1), 10000 particles

the case of $W0$.

7.5 Concluding remarks and perspectives

The results presented in this paper show that the adaptive wavelet reconstruction of the density for the Vlasov-Poisson equation is a promising approach to solve such plasma dynamics in a Lagrangian framework even though the choice of appropriated wavelets must still be discussed. The W0 wavelets are not completely satisfying because they require a high number of particles. Moreover, these methods are unable to verify the landau damping test for small perturbation magnitude α. We are interested in the numerical simulation where there are less than 100 particles per cell. The W1 wavelets satisfy this condition but do not smooth the density. The results proved that threshold helps to find the appropriate (adaptive) mesh and it should become crucial in tests where particles are very dispersed. Moreover, highest order reconstruction is particularly efficient to reduce the noise. A compromise has to be found between the reconstruction and accuracy order which minimize the computational time.

We have considered here one dimensional Vlasov-Poisson but our approach will be ex-

tended in a near future to higher dimensional problems for which the Eulerian framework becomes more costly in terms of CPU time since large numbers of grid points must be used in that case.

Bibliographie

[1] A. Abkowicz, C. Brezinski, Acceleration properties of the hybrid procedure for solving linear systems, Appl. Math., 23 (1996) 417-432.

[2] A. C. Aitken, On Bernoulli's numerical solution of algebraic equations, Proc. R. Soc. Edinb., 46 (1926) 289-305.

[3] M. Altman, A linear iterative method with a vector parameter, Bull. Pol. Acad. Sci. Math., 9 (1961) 169-174.

[4] M. Altman, A method of steepest ortho-descent, Bull. Pol. Acad. Sci. Math., 9 (1961) 575-580.

[5] M. Altman, A general method of steespest ortho-descent, Bull. Pol. Acad. Sci. Math., 9 (1961) 645-651.

[6] M. Altman, A gradient relaxation method for linear algebraic equations, Bull. Pol. Acad. Sci. Math., 9 (1961) 653-657.

[7] M. Altman, A generalized method of steepest descent, Bull. Pol. Acad. Sci. Math., 9 (1961) 733-737.

[8] M. Altman, A general method with minimum ortho-residual, Bull. Pol. Acad. Sci. Math., 9 (1961) 739-744.

[9] T.D. Arber, R.G.L. Vann, A critical comparison of Eulerian-Grid-Based Vlasov solvers, J. Compt. Phys., 180 (2002) 339-357.

[10] G. Auchmuty, A posteriori error estimates for linear equations, Numer. Math., 61 (1992) 1-6.

[11] J. Barzilai, J.M. Borwein, Two-point step size gradient methods, IMA J. Numer. Anal., 8 (1988) 141-148.

[12] B. Beckermann, A.B.J. Kuijlaars, Superlinear convergence of conjugate gradients, SIAM J. Num. Anal, 39 (2001) 300-329.

[13] B. Beckermann, A.B.J. Kuijlaars, On the sharpness of an asymptotic error estimate for Conjugate Gradients, BIT, 41 (2001) 856–867.

[14] B. Beckermann, A.B.J. Kuijlaars, Superlinear CG convergence for special right-hand sides, Electr. Trans. Num. Anal., 14 (2002) 1-19.

[15] N. Besse, E. Sonnendrücker, Semi-Lagrangian schemes for the Vlasov equation on an unstructured mesh of phase space, J. Comput. Phys., 191 (2003) 341-376.

[16] C. Biernacki, *Choix de modèles en classification*, Thèse, Université de Technologie de Compiègne, 1997.

[17] C. Biernacki, G. Celeux, G. Govaert, Choosing starting values for the EM algorithm for getting the highest likelihood in multivariate gaussian mixture models, Comput. Statist. Data Anal., 41 (2003) 561-575.

[18] C. K. Birdsall, A. B. Langdon, *Plasma Physics via Computer Simulation*, Institute of Physics Publishing Bristol and Philadelphia, 1991.

[19] C. Bolley, Solutions numériques de problèmes de bifurcation, RAIRO Modél. Math. Anal. Numér., 14 (1980) 127-147.

[20] C. Bolley, Multiple solutions of a bifurcation problem, *in Bifurcation and Nonlinear Eigenvalue Problems*, C. Bardos ed., Lecture Notes in Math. 782, Springer-Verlag, Berlin, (1978) 42-53.

[21] C. Brezinski, *Projection methods for systems of equations*, North-Holland, Amsterdam, 1997.

[22] C. Brezinski, Projection methods for linear systems, J. Comput. Appl. Math., 77 (1997) 35-51.

[23] C. Brezinski, Vector sequence transformations : methodology and applications to linear systems, J. Comput. Appl. Math., 98 (1998) 149-175.

[24] C. Brezinski, Error estimates and convergence acceleration, dans Error Control and adaptivity in scientific computing, H. Bulgak and C. Zenger eds, Kluwer, Dordrecht, (1999) 87-94.

[25] C. Brezinski, M. Redivo Zaglia, Vector and matrix sequence transformations based on biorthogonality, Appl. Numer. Math., 21 (1996) 353-373.

[26] C. Brezinski, M. Redivo Zaglia, a Schur complement approach to a general algorithm, Linear Algebra Appl., 368 (2003) 279-301.

[27] C. Brezinski, Error estimates for the solution of linear systems, SIAM J. Sci. Comput., 21 (1999) 764-781.

[28] C. Brezinski, J.P. Chehab, Nonlinear hybrid procedures and fixed point iterations, Numer. Funct. Anal. Optim., 19 (1998) 465-487.

[29] C. Brezinski, M. Redivo Zaglia, Hybrid procedures for solving systems of linear equations, Numer. Math., 67 (1994) 1-19.

[30] C. Brezinski, Variations on Richardson's method and acceleration, Numerical Analysis. A Numerical Analysis Conf. in Honor of Jean Meinguet, Bull. Soc. Math. Belg., suppl., (1996) 33-44.

[31] C. Brezinski, Multiparameter descent methods, Linear Algebra Appl., 296 (1999) 113-142.

[32] C. Brezinski, J.P. Chehab, Multiparameter iterative schemes for the solution of systems of linear and nonlinear equations, SIAM J. Sci. Comput., 20 (1999) 2140-2159.

[33] C. Brezinski, Acceleration procedures for matrix iterative methods, Numer. Algorithms, 25 (2000) 63-73.

[34] C. Brezinski, Généralisation de la transformation de Shanks, de la table de Padé et de l'epsilon-algorithm, Calcolo, 12 (1975) 317-360.

[35] C. Brezinski, Other manifestations of the Schur complement, Linear Algebra Appl., 24 (1988) 231-247.

[36] C. Brezinski, M. Redivo Zaglia, *Extrapolation Methods Theory and Practice*, North-Holland, Amsterdam, 1991.

[37] C. Brezinski, A classification of quasi-Newton methods, Numer. Algorithms, 33 (2003) 123-135.

[38] C. Brezinski, Some results in the theory of the vector ϵ-algorithm, Linear Algebra Appl., 8 (1974) 77-86.

[39] S. Cabay, L.W. Jackson, A polynomial extrapolation method for finding limits and antilimits of vector sequences, SIAM J. Numer. Anal., 13 (1976) 734-751.

[40] C. Cauchy, Méthodes générales pour la résolution des systèmes d'équations simultanées, C.R. Acad. Sci. Paris, 25 (1847) 536-538.

[41] S. Chapman, T. G. Cowling, *The mathematical theory of non-uniform gases*, Cambridge University Press, 1970.

[42] J.P. Chehab, *Méthode des Inconnues Incrémentales. Applications au calcul des Bifurcations,* Thèse, Université de Paris XI, 1993.

[43] J.P. Chehab, A nonlinear adaptative multiresolution method in finite differences with incremental unknows, M2AN, 29 (1995) 451-475.

[44] J.P. Chehab, R. Temam, Incremental unknowns for solving nonlinear eigenvalues problems : new multiresolution methods, Numerical methods for PDE's, 11 (1995) 199-228.

[45] J.P. Chehab, A. Cohen, D. Jennequin, JJ. Nieto, Ch. Roland, J. Roche, An adaptative particle in cell method for the simulation of intense beams using multiresolution analysis, Numerical Methods for hyperbolic and kinetic problems, Cemracs 2003, IRMA Lectures in Mathematics and theoretical physics, (2005) 29-42.

[46] P.G. Ciarlet, *Introduction à l'Analyse Numérique Matricielle et à l'Optimisation*, Masson, Paris, 1982.

[47] A. Cohen, *Numerical Analysis of Wavelet Methods*, Elsevier, North-Holland, 2003.

[48] G.-H. Cottet, P.-A. Raviart, Particle methods for the one-dimensional Vlasov-Poisson equations, SIAM J. Numer. Anal., 21 (1984).

[49] E.J. Craig, The N-step iterations procedures, J. Math. Phys., 34 (1955) 64-73.

[50] I. Daubechies, *Ten Lectures on Wavelets*, SIAM, 1992.

[51] J. P. Delahaye, B. Germain-Bonne, Résultats négatifs en accélération de la convergence, Numer. Math., 35 (1980) 443-457.

[52] A.P. Dempster, N.M. Laird, D.B. Rubin, Maximum likelihood from incomplete data via the EM algorithm (with discussion), J. Roy. Statist. Soc., Ser. B., 39 (1977) 1-38.

[53] A.J. Dobson, Simple approximations for the von Mises concentration statistic, Appl. Statist., 27 (1978) 345-347.

[54] D.L. Donoho, I.M. Johnstone, G. Kerkyacharian, D. Picard, Density estimation by wavelet thresholding, Ann. Statist., 24 (1996) 508–539.

[55] R.P. Eddy, Extrapolation to the limit of a vector sequence, In Information Linkage Between Applied Mathematics and Industry, P.C.C. Wang (ed.), Academic Press, New York, (1979) 387-396.

[56] B.S. Everitt, D.J. Hand, *Finite Mixture Distributions*, Chapman and Hall, New York, 1981.

[57] D.K. Faddeev, V.N. Faddeeva, *Computational Methods of linear Algebra,* W.H. Freeman and Co., San Franscisco, 1963.

[58] F. Filbet, *Contribution à l'analyse et la simulation numérique de l'équation de Vlasov*, Thèse Université Henri Poincaré, Nancy, France, 2001.

[59] F. Filbet, E. Sonnendrücker, P. Bertrand, Conservative numerical schemes for the Vlasov equation, J. Comput. Phys., 172 (2001) 166-187.

[60] B. Fischer, M. Hanke, M. Hochbruck, A note on conjugate-gradient type methods for indefinite and/or inconsistent linear systems, Numer. Algo., 11 (1996) 181-189.

[61] W.F. Ford, A. Sidi, Recursive algorithms for vector extrapolation methods, Appl. Numer. Math., 4 (1988) 477-489.

[62] C.E. Frangakis, D.B. Rubin, Principal stratification in causal inference, Biometrics, 58 (2002) 21-29.

[63] C.E. Frangakis, R.S. Brookmeyer, R. Varadhan, et al., Methodology for evaluating a partially controlled longitudinal treatment using principal stratification, with application to a needle exchange program, J. Amer. Statist. Assoc., 99 (2004) 239-249.

[64] C.E. Frangakis, R. Varadhan, Systematizing the evaluation of partially controlled studies using principal stratification : from theory to practice, Statistica Sinica, 14 (2004) 945-947.

[65] A. Friedlander, J.M. Martinez, B. Molina, M. Raydan, Gradient Method with Retards and Generalizations, SIAM J. Numer. Anal., 36 (1999) 275-289.

[66] N. Gastinel, *Analyse Numérique Linéaire*, Hermann, Paris, 1966.

[67] G.H. GOLUB, G. MEURANT, Matrices, moments and quadrature II : how to compute the norm of the error in iterative methods, BIT, 37-3 (1997) 687-705.

[68] A. HARTEN, Discrete multiresolution and generalized wavelets, J. Appl. Num. Math., 12 (1993) 153-193.

[69] P. HENRICI, *Elements of Numerical Analysis*, John Wiley, New York, 1964.

[70] M.R. HESTENES, The conjugate-gradient method for linear systems, in : J. Curtiss, Ed., *Proc. 6th Symp. in Applied Mathematics*(American Mathematical Society, Providence, RI), (1956) 83-102.

[71] M.R. HESTENES, E. STIEFEL, Methods of conjugate gradients for solving linear systems, J. Res. Natl. Bur. Stand., 49 (1952) 409-436.

[72] H.M. HUDSON, R.S. LARKIN, Accelerated image reconstruction using ordered subsets of projection data, IEEE Trans Med Imaging, 13 (1994) 601-609.

[73] M. JAMSHIDIAN, R.I. JENNRICH, Conjugate gradient acceleration of the EM algorithm, J. Amer. Statist. Assoc., 88 (1993) 221-228.

[74] M. JAMSHIDIAN, R.I. JENNRICH, Acceleration of the EM algorithm by using quasi-Newton methods, J. Roy. Statist. Soc. Ser. B, 59 (1997) 569-587.

[75] K. JBILOU, H. SADOK, Some results about vector extrapolation methods and related fixed-point iterations, J. Comput. Appl. Math., 36 (1991) 385-398.

[76] K. JBILOU, H. SADOK, Analysis of some vector extrapolation methods for solving systems of linear equations, Numer. Math. (Electronic Edition), 70 (1995) 73-89.

[77] K. JBILOU, H. SADOK, LU implementation of the modified minimal polynomial extrapolation method for solving linear and nonlinear systems, IMA J. Numer. Anal., 19 (1999) 549-561.

[78] H.A. KATKI, R. VARADHAN, CH. ROLAND, B.E. CHEN, P.S. ROSENBERG, Accelerating the EM-algorithm for reconstructing haplotypes from population genotype data, en préparation.

[79] G.M. KORPELEVICH, The extragradient method for finding saddle points and other problems, Economics and Mathematical Methods, 12 (1976) 747-756.

[80] N. LAIRD, N. LANGE, D. STRAM, Maximum likelihood computation with repeated measures : application of the EM algorithm, J. Amer. Statist. Assoc., 82 (1987) 97-105.

[81] P. LANCASTER, *Theory of Matrices*, Academic Press, New York, 1969.

[82] K. LANGE, A quasi-Newton acceleration of the EM algorithm, Statisticia Sinica, 5 (1995b) 1-18.

[83] K. LANGE, A gradient algorithm locally equivalent to the EM algorithm, J. Roy. Statist. Soc. Ser. B, 57 (1995a) 425-437.

[84] H. LE FERRAND, The quadratic convergence of the topological epsilon algorithm for systems of nonlinear equations, Numer. Algorithms, 3 (1992) 273-284.

[85] C. LEMARÉCHAL, Une méthode de résolution de certains systèmes non linéaires bien posés, C.R. Acad. Sci. Paris, sér. A, 272 (1971) 605-607.

[86] T.A. LOUIS, Finding the observed information matrix when using the EM algorithm, J. Roy. Statist. Soc. Ser. B, 44 (1982) 226- 233.

[87] P. MARCOTTE, Application of Khobotov's algorithm to variational inequalities and network equilibrium problems, INFOR, 29 (1991) 1051-1065.

[88] H. MARDER, B. WEITZNER, A bifurcation problem in E-layer equilibria, Plasma Physics, 12 (1970) 435-445.

[89] G. J. MCLACHLAN, On Aitken's method and other approaches for accelerating convergence of the EM algorithm, Proceedings of the A.C. Aitken Centenary Conference, University of Otago, Dunedin, University of Otago Press, (1995) 201-209.

[90] I. MEILIJSON, A fast improvement to the EM algorithm on its own terms, J. Roy. Statist. Soc. Ser. B, 51 (1989) 127-138.

[91] X.L. MENG, D.B. RUBIN, A maximum likelihood estimation via the EM algorithm : a general framework, Biometrika, 80 (1993) 267-278.

[92] X.L. MENG, D.B. RUBIN, On the global and component-wise rates of convergence of the EM algorithm, Linear Algebra Appl., 199 (1994) 413-425.

[93] M. MESINA, Convergence acceleration for the iterative solution of x = Ax + f; Computational Methods in Applied Mechanics and Engineering, 10 (1977) 165-173.

[94] O. NEVANLINNA, *Convergence of Iterations for Linear Equations*, Birkhäuser, Basel, 1993.

[95] Y. NIEVERGELT, Aitken's and Steffensen's accelerations in several variables, Numer. Math., 59 (1991) 295-310.

[96] J.M. ORTEGA, W.C. RHEINBOLDT, *Iterative Solutions of Nonlinear Equations in Several Variables*, Academic Press, New York, 1970.

[97] P.A. RAVIART, An analysis of particle method, Lect. Notes Math. 1127, (1985) 243-324.

[98] M. RAYDAN, *Convergence Properties of the Barzilai and Borwein Gradient Method*, Ph. D. Thesis, Dept. of Mathematical Sciences, Rice University, Houston, TX, 1991.

[99] M. RAYDAN, On the Barzilai and Borwein choice of steplength for the gradient method, IMA J. Numer. Anal., 13 (1993) 321-326.

[100] M. RAYDAN, B. F. SVAITER, Relaxed Steepest Descent and Cauchy-Barzilai-Borwein Method, Comput. Optim. and Appl., 21 (2002) 155-167.

[101] R.A. REDNER, H.F. WALKER, Mixture densities, maximum likelihood and the EM algorithm, SIAM Review, 26 (1984) 195-239.

[102] L.F. RICHARDSON, The approximate arithmetical solution by finite differences of physical problems involving differential equations, with application to the stress in a masonry dam, Philos. Roy. Soc. London, ser. A, 226 (1910) 307-357.

[103] CH. ROLAND, B. BECKERMANN, C. BREZINSKI, Altman's methods revisited, Appl. Math., 31-3 (2004) 353-368.

[104] CH. ROLAND, R. VARADHAN, New iteratives schemes for nonlinear fixed point problems, with applications to problems with bifurcations and incomplete-data problems, Appl. Numer. Math., 55-2 (2005) 215-226.

[105] CH. ROLAND, R. VARADHAN, Convergence results for the squared extrapolation methods, en préparation.

[106] Y. SAAD, *Iterative methods for sparse linear systems*, PWS Publishing, Boston, MA, 1996.

[107] H. SADOK, About Henrici's transformation for accelerating vector sequences, J. Comput. Appl. Math., 29 (1990) 101-110.

[108] W. SCHÖNAUER, H. MÜLLER, E. SCHNEPF, Numerical tests with biconjugate gradient type methods, Z. Angew. Math. Mech., 65 (1985) T400-T402.

[109] W. SCHÖNAUER, *Scientific Computing on Vector Computers,* North-Holland, Amsterdam, 1987.

[110] G. SCHOU, Estimation of the concentration parameter in von Mises-Fisher distributions, Biometrika, 65 (1978) 369-377.

[111] SCILAB, http ://scilabsoft.inria.fr.

[112] M. SERMANGE, Une méthode numérique en bifurcation-application à un problème à frontière libre de la physique des plasmas, Appl. Math. Optim., (1979) 127-151.

[113] D. SHANKS, Non linear transformations of divergent and slowly convergent sequences, J. Math. Phys., 34 (1955) 1-42.

[114] A. SIDI, Acceleration of convergence of vector sequences, SIAM J. Numer. Anal., 23 (1986a) 178-196.

[115] A. SIDI, Convergence and stability properties of minimal polynomial and reduce rank extrapolation algorithms, SIAM J. Numer. Anal., 23 (1986b) 197-209.

[116] A. SIDI, Extrapolation vs. projection methods for linear systems of equations, J. Comp. Appl. Math., 22 (1988) 71-88.

[117] A. SIDI, Efficient implementation of minimal polynomial and reduced rank extrapolation methods, J. Comp. Appl. Math., 36 (1991) 305-337.

[118] D.A. SMITH, W.F. FORD, A. SIDI, Extrapolation methods for vector sequences, SIAM Review, 29 (1987) 199-233.

[119] R.V. SOUTHWELL, Stress-calculation in frameworks by the method of "systematic relaxation of contraints ", Proc. Roy. Soc. London, A 151 (1935) 56-95 ; A 153 (1935) 41-76.

[120] W. Sweldens, The lifting scheme : a custom design construction of biorthogonal wavelets, Appl. Comp. Harm. Anal., 3 (1996) 186-200.

[121] G. Temple, The general theory of relaxation methods applied to linear systems, Proc. Roy. Soc. London, A 169 (1939) 476-500.

[122] R. A. Thisted, *Elements of Statistical Computing : Numerical Computation*, Chapman and Hall, New York, 1988.

[123] D.M. Titterington, A.F.M. Smith, U.E. Makov, *Statistical Analysis of Finite Mixture Distributions*, Wiley, New York, 1985.

[124] R. Tucker, The δ^2-process and related topics, Pac. J. Math., 22 (1967) 349-359.

[125] R. Varadhan, Ch. Roland, Squared extrapolation methods (SQUAREM) : A new class of simple and efficient numerical schemes for accelerating the convergence of the EM algorithm, Department of Biostatistics Working Paper, Johns Hopkins University, 63 (2004) 1-70.

[126] R. Varadhan, Ch. Roland, Squared extrapolation methods : A new class of numerical schemes for accelerating the convergence of the EM algorithm, soumis.

[127] Y. Vardi, L.A. Shepp, L. Kaufman, A statistical model for positron emission tomography, J. Amer. Statist. Assoc., 80 (1985) 8-37.

[128] C.F.J. Wu, On the convergence properties of the EM algorithm, The Annals of Statistics, 11 (1983) 95-103.

[129] P. Wynn, On a device for computing the $e_m(S_n)$ transformation, MTAC, 10 (1956) 91-96.

