Contribution à la résolution numérique de certains systèmes d'équations

Carlos Espinoza

THESE

présentée à

Université Scientifique et Médicale de Grenoble

Institut National Polytechnique de Grenoble

pour obtenir le grade de

DOCTEUR DE 3ème CYCLE
Spécialité : Analyse numérique

par

Carlos ESPINOZA

CONTRIBUTION A LA RESOLUTION NUMERIQUE DE CERTAINS SYSTEMES D'EQUATIONS

Thèse soutenue le 27 mai 1977 devant la Commission d'Examen

Président : N. GASTINEL

Examinateurs : P. BERNA
C. BREZINSKI
F. ROBERT

UNIVERSITE SCIENTIFIQUE
ET MEDICALE DE GRENOBLE

Monsieur Gabriel CAU : Président
Monsieur Pierre JULLIEN : Vice Président

MEMBRES DU CORPS ENSEIGNANT DE L'U.S.M.G.

PROFESSEURS TITULAIRES

MM.	AMBLARD Pierre	Clinique de dermatologie
	ARNAUD Paul	Chimie
	ARVIEU Robert	I.S.N.
	AUBERT Guy	Physique
	AYANT Yves	Physique approfondie
Mme.	BARBIER Marie-Jeanne	Electrochimie
MM.	BARBIER Jean-Claude	Physique expérimentale
	BARBIER Reynold	Géologie appliquée
	BARJON Robert	Physique nucléaire
	BARNOU Fernand	Biosynthèse de la cellulose
	BARRA Jean-René	Statistiques
	BARRIE Joseph	Clinique chirurgicale
	BEAUDOING André	Clinique de pédiatrie et puériculture
	BELORIZKY Elie	Physique
	BERNARD Alain	Mathématiques pures
Mme.	BERTRANDIAS Françoise	Mathématiques pures
MM.	BERTRANDIAS Jean-Paul	Mathématiques pures
	BEZEZ Henri	Pathologie chirurgicale
	BLAMBERT Maurice	Mathématiques pures
	BOLLIET Louis	Informatique (IUT B)
	BONNET Jean-Louis	Clinique ophtalmologique
	BONNET-EYMARD Joseph	Clinique gastro-entérologique
Mme.	BONNIER Marie-Jeanne	Chimie générale
MM.	BOUCHERLE André	Chimie et toxicologie
	BOUCHEZ Robert	Physique nucléaire
	BOUSSARD Jean-Claude	Mathématiques appliquées
	BOUTET DE MONVEL Louis	Mathématiques pures
	BRAVARD Yves	Géographie
	CABANEL Guy	Clinique rhumatologique et hydrologique
	CALAS François	Anatomie
	CARLIER Georges	Biologie végétale
	CARRAZ Gilbert	Biologie animale et pharmocodynamie
	CAU Gabriel	Médecine légale et toxicologie
	CAUQUIS Georges	Chimie organique
	CHABAUTY Claude	Mathématiques pures
	CHARACHON Robert	Clinique oto-rhino-laryngologique
	CHATEAU Robert	Clinique de neurologie
	CHIBON Pierre	Biologie animale
	COEUR André	Pharmacie chimique et chimie analytique
	CONTAMIN Robert	Clinique gynécologique
	COUDERC Pierre	Anatomie pathologique

Mme.	DEBELMAS Anne-Marie	Matière médicale
MM.	DEBELMAS Jacques	Géologie générale
	DEGRANGE Charles	Zoologie
	DELORMAS Pierre	Pneumophtisiologie
	DEPORTES Charles	Chimie minérale
	DESRE Pierre	Métallurgie
	DESSAUX Georges	Physiologie animale
	DODU Jacques	Mécanique appliquée (IUT I)
	DOLIQUE Jean-Michel	Physique des plasmas
	DREYFUS Bernard	Thermodynamique
	DUCROS Pierre	Cristallographie
	GAGNAIRE Didier	Chimie physique
	GALVANI Octave	Mathématiques pures
	GASTINEL Noël	Analyse numérique
	GAVEND Michel	Pharmacologie
	GEINDRE Michel	Electroradiologie
	GERBER Robert	Mathématiques pures
	GERMAIN Jean-Pierre	Mécanique
	GIRAUD Pierre	Géologie
	JANIN Bernard	Géographie
	KAHANE André	Physique générale
	KOSZUL Jean-Louis	Mathématiques pures
	KLEIN Joseph	Mathématiques pures
	KRAVTCHENKO Julien	Mécanique
	KUNTZMANN Jean	Mathématiques appliquées
	LACAZE Albert	Thermodynamique
	LACHARME Jean	Biologie végétale
Mme.	LAJZEROWICZ Janine	Physique
MM.	LAJZEROWICZ Joseph	Physique
	LATREILLE René	Chirurgie générale
	LATURAZE Jean	Biochimie pharmaceutique
	LAURENT Pierre-Jean	Mathématiques Appliquées
	LEDRU Jean	Clinique médicale B
	LE ROY Philippe	Mécanique (IUT I)
	LLIBOUTRY Louis	Géophysique
	LOISEAUX Pierre	Sciences nucléaires
	LONGEQUEUE Jean-Pierre	Physique nucléaire
	LOUP Jean	Géographie
Melle	LUTZ Elisabeth	Mathématiques pures
MM.	MALINAS Yves	Clinique obstétricale
	MARTIN-NOEL Pierre	Clinique cardiologique
	MAZARE Yves	Clinique médicale A
	MICHEL Robert	Minéralogie et pétrographie
	MICOUD Max	Clinique maladies infectieuses
	MOURIQUAND Claude	Histologie
	MOUSSA André	Chimie nucléaire
	NOZIERES Philippe	Spectrométrie physique
	OZENDA Paul	Botanique
	PAYAN Jean-Jacques	Mathématiques pures
	PEBAY-PEYROULA Jean-Claude	Physique
	PERRET Jean	Semeiologie médicale (Neurologie)
	RASSAT André	Chimie systématique
	RENARD Michel	Thermodynamique
	REVOL Michel	Urologie
	RINALDI Renaud	Physique
	DE ROUGEMONT Jacques	Neuro-chirurgie
	SEIGNEURIN Raymond	Microbiologie et Hygiène
	SENGEL Philippe	Zoologie
	SIBILLE Robert	Construction mécanique (IUT I)

MM.	SOUTIF Michel	Physique générale
	TANCHE Maurice	Physiologie
	TRAYNARD Philippe	Chimie générale
	VAILLANT François	Zoologie
	VALENTIN Jacques	Physique nucléaire
	VAUQUOIS Bernard	Calcul électronique
Mme.	VERAIN Alice	Pharmacie galénique
MM.	VERAIN André	Physique
	VEYRET Paul	Géographie
	VIGNAIS Pierre	Biochimie médicale

PROFESSEURS ASSOCIES

MM.	CRABBE Pierre	CERMO
	DEMBICKI Eugéniuz	Mécanique
	JOHNSON Thomas	Mathématiques appliquées
	PENNEY Thomas	Physique

PROFESSEURS SANS CHAIRE

Melle	AGNIUS-DELORD Claudine	Physique pharmaceutique
	ALARY Josette	Chimie analytique
MM.	AMBROISE-THOMAS Pierre	Parasitologie
	ARMAND Gilbert	Géographie
	BENZAKEN Claude	Mathématiques appliquées
	BIAREZ Jean-Pierre	Mécanique
	BILLET Jean	Géographie
	BOUCHET Yves	Anatomie
	BRUGEL Lucien	Energétique (IUT I)
	BUISSON René	Physique (IUT I)
	BUTEL Jean	Orthopédie
	COHEN ADDAD Pierre	Spectrométrie physique
	COLOMB Maurice	Biochimie
	CONTE René	Physique (IUT I)
	DELOBEL Claude	M.I.A.G.
	DEPASSEL Roger	Mécanique des fluides
	FONTAINE Jean-Marc	Mathématiques pures
	GAUTRON René	Chimie
	GIDON Paul	Géologie et minéralogie
	GLENAT René	Chimie organique
	GROULADE Joseph	Biologie médicale
	HACQUES Gérard	Calcul numérique
	HOLLARD Daniel	Hématologie
	HUGONOT Robert	Hygiène et médecine préventive
	IDELMAN Simon	Physiologie animale
	JOLY Jean-René	Mathématiques pures
	JULLIEN Pierre	Mathématiques appliquées
Mme.	KAHANE Josette	Physique
MM.	KRAKOWIACK Sacha	Mathématiques appliquées
	KUHN Gérard	Physique (IUT I)
	LUU DUC Cuong	Chimie organique
	MAYNARD Roger	Physique du solide
Mme.	MINIER Colette	Physique (IUT I)
MM.	PELMONT Jean	Biochimie
	PERRIAUX Jean-Jacques	Géologie et minéralogie
	PFISTER Jean-Claude	Physique du solide
Melle	PIERY Yvette	Physiologie animale

MM.	RAYNAUD Hervé	M.I.A.G.
	REBECQ Jacques	Biologie (CUS)
	REYMOND Jean-Charles	Chirurgie générale
	RICHARD Lucien	Biologie végétale
Mme.	RINAUDO Marguerite	Chimie macromoléculaire
MM.	ROBERT André	Chimie papetière
	SARRAZIN Roger	Anatomie et chirurgie
	SARROT-REYNAULD Jean	Géologie
	SIROT Louis	Chirurgie générale
Mme.	SOUTIF Jeanne	Physique générale
MM.	STIEGLITZ Paul	Anesthésiologie
	VIALON Pierre	Géologie
	VAN CUTSEM Bernard	Mathématiques appliquées

MAITRES DE CONFERENCES ET MAITRES DE CONFERENCES AGREGES

MM.	ARMAND Yves	Chimie (IUT I)
	BACHELOT Yvan	Endocrinologie
	BARGE Michel	Neuro-chirurgie
	BEGUIN Claude	Chimie organique
Mme	BERIEL Hélène	Pharmacodynamie
MM.	BOST Michel	Pédiatrie
	BOUCHARLAT Jacques	Psychiatrie adultes
Mme.	BOUCHE Liane	Mathématiques (CUS)
MM.	BRODEAU François	Mathématiques (IUT B) (Personne étrangère habilitée à être directeur de thèse)
	CHAMBAZ Edmond	Biochimie médicale
	CHAMPETIER Jean	Anatomie et organogénèse
	CHARDON Michel	Géographie
	CHERADAME Hervé	Chimie papetière
	CHIAVERINA Jean	Biologie appliquée (EFP)
	CONTAMIN Charles	Chirurgie thoracique et cardio-vasculaire
	CORDONNIER Daniel	Néphrologie
	COULOMB Max	Radiologie
	CROUZET Guy	Radiologie
	CYROT Michel	Physique du solide
	DENIS Bernard	Cardiologie
	DOUCE Roland	Physiologie végétale
	DUSSAUD René	Mathématiques (CUS)
Mme.	ETERRADOSSI Jacqueline	Physiologie
MM.	FAURE Jacques	Médecine légale
	FAURE Gilbert	Urologie
	GAUTIER Robert	Chirurgie générale
	GIDON Maurice	Géologie
	GROS Yves	Physique (IUT I)
	GUIGNIER Michel	Thérapeutique
	GUITTON Jacques	Chimie
	HICTER Pierre	Chimie
	JALBERT Pierre	Histologie
	JULIEN-LAVILLAVROY Claude	O.R.L.
	KOLODIE Lucien	Hématologie
	LE NOC Pierre	Bactériologie-virologie
	MACHE Régis	Physiologie végétale
	MAGNIN Robert	Hygiène et médecine préventive
	MALLION Jean-Michel	Médecine du travail
	MARECHAL Jean	Mécanique (IUT I)
	MARTIN-BOUYER Michel	Chimie (CUS)
	MICHOULIER Jean	Physique (IUT I)

MM.	NEGRE Robert	Mécanique (IUT I)
	NEMOZ Alain	Thermodynamique
	NOUGARET Marcel	Automatique (IUT I)
	PARAMELLE Bernard	Pneumologie
	PECCOUD François	Analyse (IUT B) (Personnalité étrangère habilité à être directeur de thèse)
	PEFFEN René	Métallurgie (IUT I)
	PERRIER Guy	Géophysique-Glaciologie
	PHELIP Xavier	Rhumatologie
	RACHAIL Michel	Médecine interne
	RACINET Claude	Gynécologie et obstétrique
	RAMBAUD André	Hygiène et hydrologie (Pharmacie)
	RAMBAUD Pierre	Pédiatrie
	RAPHAEL Bernard	Stomatologie
Mme.	RENAUDET Jacqueline	Bactériologie (Pharmacie)
MM.	ROBERT Jean-Bernard	Chimie physique
	Romier Guy	Mathématiques (IUT B) (Personnalité étrangère habilité à être directeur de thèse)
	SCHAERER René	Cancérologie
	SHOM Jean-Claude	Chimie générale
	STOEBNER Pierre	Anatomie pathologie
	VROUSOS Constantin	Radiologie

MAITRES DE CONFERENCES ASSOCIES

MM.	DEVINE Roderick	Spectro physique
	HODGES Christopher	Transition de phases

Fait à SAINT MARTIN D'HERES, NOVEMBRE 1976.

INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

Monsieur Philippe TRAYNARD : Président

Monsieur Pierre-Jean LAURENT : Vice Président

PROFESSEURS TITULAIRES

MM.	BENOIT Jean	Radioélectricité
	BESSON Jean	Electrochimie
	BLOCH Daniel	Physique du solide
	BONNETAIN Lucien	Chimie minérale
	BONNIER Etienne	Electrochimie et électrométallurgie
	BOUDOURIS Georges	Radioélectricité
	BRISSONNEAU Pierre	Physique du solide
	BUYLE-BODIN Maurice	Electronique
	COUMES André	Radioélectricité
	DURAND Francis	Métallurgie
	FELICI Noël	Electrostatique
	FOULARD Claude	Automatique
	LESPINARD Georges	Mécanique
	MOREAU René	Mécanique
	PARIAUD Jean-Charles	Chimie-Physique
	PAUTHENET René	Physique du solide
	PERRET René	Servomécanismes
	POLOUJADOFF Michel	Electrotechnique
	SILBER Robert	Mécanique des fluides

PROFESSEUR ASSOCIE

M.	ROUXEL Roland	Automatique

PROFESSEURS SANS CHAIRE

MM.	BLIMAN Samuel	Electronique
	BOUVARD Maurice	Génie mécanique
	COHEN Joseph	Electrotechnique
	LACOUME Jean-Louis	Géophysique
	LANCIA Roland	Electronique
	ROBERT François	Analyse Numérique
	VEILLON Gérard	Informatique fondamentale et appliquée
	ZADWORNY François	Electronique

MAITRES DE CONFERENCES

MM.	ANCEAU François	Mathématiques appliquées
	CHARTIER Germain	Electronique
	GUYOT Pierre	Chimie minérale
	IVANES Marcel	Electrotechnique
	JOUBERT Jean-Claude	Physique du solide
	MORET Roger	Electrotechnique nucléaire
	PIERRARD Jean-Marie	Mécanique
	SABONNADIERE Jean-Claude	Informatique fondamentale et appliquée
Mme.	SAUCIER Gabrièle	Informatique fondamentale et appliquée

MAITRE DE CONFERENCES ASSOCIE

M.	LANDAU Ioan	Automatique

CHERCHEURS DU C.N.R.S. (Directeur et Maîtres de Recherche)

MM.	FRUCHART Robert	Directeur de Recherche
	ANSARA Ibrahim	Maître de Recherche
	CARRE René	Maître de Recherche
	DRIOLE Jean	Maître de Recherche
	MATHIEU Jean-Claude	Maître de Recherche
	MUNIER Jacques	Maître de Recherche

A mes Parents,

ma Femme et ma Fille.

Monsieur N. GASTINEL a bien voulu m'accueillir dans son Equipe. Il m'est agréable de témoigner ici de tout ce que le présent travail doit à ses conseils précieux et son soutien incessant et de lui exprimer ma reconnaissance pour l'honneur qu'il me fait en présidant ce jury.

Monsieur C. BREZINSKI m'a proposé ce sujet dont il a constamment suivi la progression. Qu'il trouve ici l'expression de mes vifs remerciements pour ses conseils judicieux et son aide bienveillante.

Je remercie vivement Monsieur F. ROBERT pour l'intérêt qu'il a bien voulu porter à cette étude et pour avoir accepté de faire partie du jury.

J'exprime tous mes remerciements à Monsieur P. BERNA pour sa collaboration dans l'élaboration de la troisième partie de ce travail et pour avoir eu l'amabilité de participer au jury.

Je tiens aussi à remercier :

Mes collègues des équipes d'Analyse Numérique des Universités de Lille, de Grenoble, du C.E.N.-Cadarache, et de l'équipe APL de l'I.F.C. de Grenoble pour l'accueil amical qu'ils m'ont réservé. Notamment Messieurs GALLI, ZEDDE, COSNARD, avec qui j'ai eu de fructueuses conversations,

Tous ceux qui d'une façon ou d'une autre ont facilité ma venue et la poursuite de mes études en France. En particulier mes amis DESCARPENTRIES et COMYN,

Mes camarades latino-américains pour leur amitié et leur constante collaboration,

Et ceux qui ont contribué à la réalisation matérielle de ce travail, en particulier Madame Meyrieux dont j'ai pu apprécier la gentillesse et la compétence.

P L A N

PREMIERE PARTIE : METHODES SIMULTANEES D'ACCELERATION DE LA CONVERGENCE.

CHAPITRE I : Etude d'un procédé adapté aux suites obtenues par des itérations linéaires.

CHAPITRE II : Généralisations de l'ε-algorithme à des suites de vecteurs.

CHAPITRE III : Algorithmes d'accélération de la convergence des suites de nombres.

CHAPITRE IV : Algorithmes de losange pour les suites de vecteurs.

CHAPITRE V : Etude de l'efficacité des méthodes itératives.

DEUXIEME PARTIE : ACCELERATION DE LA CONVERGENCE DES METHODES DE PROJECTION.

CHAPITRE I : Méthodes de projection généralisée.

CHAPITRE II : Applications de la méthode.

TROISIEME PARTIE : CONVERGENCE D'UNE METHODE DE RESOLUTION DES EQUATIONS D'EQUILIBRE DANS LES RESEAUX THERMIQUES.

CHAPITRE I : Présentation du problème.

CHAPITRE II : Etude variationnelle et convergence des méthodes de minimisation.

TABLE DES MATIERES

page

PREMIERE PARTIE : METHODES SIMULTANEES D'ACCELERATION DE LA CONVERGENCE

INTRODUCTION..... 1

NOTATIONS ET DEFINITIONS..... 3

CHAPITRE I : ETUDE D'UN PROCEDE ADAPTE AUX SUITES OBTENUES PAR DES ITERATIONS LINEAIRES

I-1 Définition et étude du procédé..... 5

I-2 Itération de la méthode précédente..... 10

I-3 Application au calcul des éléments propres d'une matrice..... 13

I-4 Résultats numériques..... 16

CHAPITRE II : GENERALISATIONS DE L'ε-ALGORITHME A DES SUITES DE VECTEURS

II-1 L'ε-algorithme vectoriel..... 26

II-2 L'ε-algorithme topologique..... 33

CHAPITRE III : ALGORITHMES D'ACCELERATION DE LA CONVERGENCE DES SUITES DE NOMBRES

III-1 Les algorithmes de losange..... 35

III-2 Le θ-algorithme..... 38

III-3 Résultats numériques..... 42

CHAPITRE IV : ALGORITHMES DE LOSANGE POUR LES SUITES DE VECTEURS

IV-1 Généralisation des algorithmes de losange.... 45

IV-2 Introduction d'un paramètre d'accélération... 50

CHAPITRE V : ETUDE DE L'EFFICACITE DES METHODES ITERATIVES

V-1 Les critères d'efficacité et ses limitations. 54

V-2 Un indice général d'efficacité..... 57

V-3 Efficacité des algorithmes étudiés..... 61

REFERENCES..... 68

DEUXIEME PARTIE : ACCELERATION DE LA CONVERGENCE DES METHODES DE PROJECTION

INTRODUCTION..... 73

NOTATIONS ET DEFINITIONS..... 74

CHAPITRE I : METHODES DE PROJECTION GENERALISEES

I-1 Définition et propriétés..... 76

I-2 Une méthode d'accélération..... 81

I-3 Convergence de la méthode..... 84

I-4 Transformée de la méthode associée à la décomposition de φ_2 90

I-5 Minimisation d'une fonctionnelle non quadratique..... 91

CHAPITRE II : APPLICATIONS DE LA METHODE

II-1 Systèmes symétriques définis positifs..... 95

II-2 Systèmes quelconques..... 99

REFERENCES..... 107

TROISIEME PARTIE : CONVERGENCE D'UNE METHODE DE RESOLUTION DES EQUATIONS D'EQUILIBRE DANS LES RESEAUX THERMIQUES

INTRODUCTION..... 110

CHAPITRE I : PRESENTATION DU PROBLEME

I-1 Définition et propriétés des réseaux thermiques 112

I-2 Méthode de conservation de débit anticipée (MCAC)..... 116

CHAPITRE II : ETUDE VARIATIONNELLE ET CONVERGENCE DES METHODES DE MINIMISATION

II-1 Caractérisation et propriétés de la fonctionnelle a minimiser..... 121

II-2 Généralisation de la MCAC..... 127

ANNEXE A ALGORITHME POUR LA MISE EN NIVEAUX D'UN GRAPHE CONNEXE..... 130

REFERENCES..... 132

PREMIERE PARTIE

METHODES SIMULTANEES D'ACCELERATION DE LA CONVERGENCE

INTRODUCTION

Dans cette partie nous étudions les différentes méthodes simultanées d'accélération de suites de vecteurs. Nous développons en détail les propriétés concernant l'accélération de la convergence des suites obtenues par la résolution des systèmes d'équations par des itérations linéaires.

Dans le premier chapitre nous présentons une méthode d'accélération très simple, qui consiste à transformer la suite $\{s_n\}$ que l'on veut accélérer en une autre suite $\{u_n\}$ par la formule

$$u_n = s_n + \omega \cdot (s_n - s_{n-1}) \quad , \quad n=1,\ldots$$

L'idée est analogue à celle qui est à l'origine de la méthode de sur-relaxation, mais on ne réintroduit pas le u_n calculé dans la suite $\{s_n\}$, comme ça se fait dans la sur-relaxation et dans d'autres méthodes de résolution de systèmes d'équations (voir [11], [16], [21]). On caractérise la valeur optimale de ω et on envisage la possibilité de remplacer ω par une suite $\{\omega_n\}$ qui converge vers ω. On étudie ensuite une généralisation de cette méthode qui consiste à accélérer $\{u_n\}$ par le même procédé et ainsi de suite.

Le second chapitre est consacré à l'étude d'algorithmes plus connus d'accélération de la convergence des suites de vecteurs ; l'ε-algorithme vectoriel et l'ε-algorithme topologique. Les principaux résultats et applications de l'ε-algorithme vectoriel sont rappelés. Dans le cas des suites obtenues par des itérations linéaires, on donne des conditions suffisantes pour qu'il y ait accélération de la convergence, en passant d'une colonne paire à la suivante.

Dans le troisième chapitre, après un rappel des propriétés des algorithmes de losange et du θ-algorithme, pour accélérer les suites de nombres, on démontre que, quand on applique le θ-algorithme aux suites de la forme $s_n = s - a\,\lambda^n$, $\forall\, n \geq N$ ou $s_n = s - a/n$, $\forall\, n \geq N$, on obtient tout de suite la solution exacte s , à partir de s_N , s_{N+1} et s_{N+2} (en un nombre fini de pas).

Dans le quatrième chapitre, nous généralisons les algorithmes de losange aux suites de vecteurs. Dans son premier paragraphe on étudie les propriétés d'un tel algorithme puis dans le second, son comportement lors de l'introduction d'un paramètre d'accélération w_k .

Dans le cinquième chapitre nous abordons le problème du calcul de l'efficacité. On fait remarquer que pour les méthodes à convergence linéaire (ordre de convergence égal à 1), ni l'index d'efficacité d'OSTROWSKI ni celui de BRENT sont applicables.

D'autre part, dans la résolution d'un système de point fixe $X = F(X)$, avec F dérivable au sens de Fréchet, alors dans un voisinage de la solution les itérations $x_{n+1} = F(X_n)$ approchent une itération linéaire du type
$x_{n+1}-x = F'(x^*)(x_n-x^*)$ et on peut comparer les différentes méthodes, en comparant les rayons spectraux des matrices $F'(x^*)$. Nous ne pouvons pas nous servir de cette approche, parce que les suites obtenues par des méthodes d'accélération de la convergence, ne peuvent pas en général être mises sous la forme $x_{n+1} = F(x_n)$. Nous avons donc été conduit à proposer un indice général d'efficacité. On montre comment il est basé sur le même principe que l'index d'efficacité d'Ostrowski et cohérent avec celui-ci. Quand il s'agit de comparer deux itérations linéaires il est aussi cohérent avec la notion de taux asymptotique de convergence qui conduit, dans ce cas, à la comparaison des rayons spectraux. Nous finissons le chapitre et cette partie, en mesurant l'efficacité des méthodes précédemment étudiées, au moyen de l'indice qu'on vient de définir.

NOTATIONS ET DEFINITIONS

Nous donnons brièvement quelques notations et définitions qui seront utilisées dans la suite.

i) Dans tout ce qui suit on considère R^p comme un espace vectoriel (sur R), muni du produit scalaire $(x,y) = \sum_{i=1}^{p} x_i\, y_i$ et de la norme (euclidienne) qui en découle.

ii) Soit $\{s_n\}$ une suite d'éléments de R^p . On désigne par

$$\Delta^0 s_n = s_n \quad , \quad n=0,1,\ldots$$

$$\Delta^{k+1} s_n = \Delta^k s_{n+1} - \Delta^k s_n \quad , \quad n,k=0,1,\ldots$$

iii) Soient $\{s_n\}$ et $\{u_n\}$ deux suites de nombres (éléments de R), alors le symbole $s_n \sim u_n$ signifie que $\lim_{n\to\infty} s_n/u_n$ existe et est égal à 1.

iv) Soient $\{s_n\}$ et $\{u_n\}$ deux suites de vecteurs de R^p , alors le symbole $s_n \sim u_n$ signifie que pour tout i , $1 \leq i \leq p$, les suites de composantes $\{s_n^{(i)}\}$ et $\{u_n^{(i)}\}$ satisfont $s_n^{(i)} \sim u_n^{(i)}$.

v) Soit A une matrice carrée, réelle $(p \times p)$ et soient $\lambda_1,\lambda_2,\ldots,\lambda_p$ ses valeurs propres, supposées réelles et telles que $|\lambda_1| \geq |\lambda_2| \geq \ldots \geq |\lambda_p|$. Nous appelons respectivement (H_0) et (H_1) les hypothèses précédentes, avec, en plus :

$$(H_0) \quad \begin{cases} \text{Les vecteurs propres de A forment une base de } R^p \\ \lambda_1 \neq 1 \quad \text{et} \quad |\lambda_1| > |\lambda_2| \end{cases}$$

$$(H_1) \quad \begin{cases} \lambda_i \neq 1 \quad , \quad i=1,\ldots,p \\ |\lambda_1| > |\lambda_2| > \ldots |\lambda_p| \end{cases}$$

(vi) On utilise différents critères de comparaison de deux suites, qui d'ailleurs le plus souvent sont rappelés dans le texte.
(A) , pour $\{s_n\}$ et $\{u_n\}$ deux suites de vecteurs de R^p , on dit que $\{u_n\}$ converge plus vite que $\{s_n\}$ si

$$(A) \qquad \lim_{n\to\infty} \frac{(z,\ \Delta u_n)}{(z,\ \Delta s_n)} = 0$$

pour tout vecteur z non nul et tel que cette limite existe.
(B) , si les suites $\{s_n\}$ et $\{u_n\}$ convergent vers s , on dit aussi que $\{u_n\}$ converge plus vite que $\{s_n\}$ si :

$$(B) \qquad \lim_{n\to\infty} \frac{(z,u_n-s)}{(z,s_n-s)} = 0$$

pour tout vecteur z non nul et tel que cette limite existe.

(vii) On manipule des quantités doublement indexées du type $t_k^{(n)}$. Ces quantités sont considérées comme des éléments d'un tableau à double entrée T . Pour k fixé, on appelle colonne k (k-ième colonne) la suite indexée par n $\{t_k^{(n)}\}$ (k fixé). De la même façon, pour n fixé, appelle diagonale n (n-ième diagonale) la suite indexée par k $\{t_k^{(n)}\}$ (n fixé).

CHAPITRE I

ETUDE D'UN PROCEDE ADAPTE AUX SUITES OBTENUES PAR DES ITERATIONS LINEAIRES

I-1 DEFINITION ET ETUDE DU PROCEDE

Soit $\{s_n\}$ une suite de vecteurs de R^p, qui converge vers s. Nous voulons accélérer la convergence de cette suite, par le procédé suivant.

$$(1.1) \qquad u_n = s_{n+1} + \omega \Delta s_n \quad ; \quad n=0,1,\ldots$$

Nous nous proposons donc de déterminer ω de telle sorte que la suite $\{u_n\}$ converge vers s et cela plus vite que la suite initiale $\{s_n\}$, dans ce sens que $\lim_{n\to\infty} \dfrac{(z, \Delta u_n)}{(z, \Delta s_{n+1})} = 0$, pour z différent de zéro quelconque. En ce qui concerne la convergence, puisque $\{\Delta s_n\}$ converge vers zéro, on a :

$$(1.2) \qquad \lim_{n\to\infty} u_n = s \quad , \quad \forall\, \omega \in R$$

Mais pour qu'il ait accélération de la convergence, il faut bien choisir la valeur de ω. Sa valeur optimale est caractérisée par le théorème suivant.

THEOREME 1

Supposons l'existence de $\lim_{n\to\infty} \frac{(z, \Delta s_{n+1})}{(z, \Delta^2 s_n)}$, pour un z quelconque, différent de zéro. Alors, une condition nécessaire et suffisante pour que $\lim_{n\to\infty} \frac{(z, \Delta u_n)}{(z, \Delta s_{n+1})} = 0$ est de prendre

$$\omega = - \lim_{n\to\infty} \frac{(z, \Delta s_{n+1})}{(z, \Delta^2 s_n)} \tag{1.3}$$

DEMONSTRATION

$$u_n = s_{n+1} + \omega \Delta^2 s_n$$

$$(z, \Delta u_n) = (z, \Delta s_{n+1}) + (z, \Delta^2 s_n)$$

$$\frac{(z, \Delta u_n)}{(z, \Delta s_{n+1})} = 1 + \omega \frac{(z, \Delta^2 s_n)}{(z, \Delta s_{n+1})} \tag{1.4}$$

$$\lim_{n\to\infty} \frac{(z, \Delta u_n)}{(z, \Delta s_{n+1})} = 1 + \omega \lim_{n\to\infty} \frac{(z, \Delta^2 s_n)}{(z, \Delta s_{n+1})}$$

Ce qui montre que, pour que la limite du terme de gauche soit nulle, il faut et il suffit que (1.3) soit satisfait.

Pour certaines suites produites par des itérations linéaires on peut calculer cette limite et l'on a :

THEOREME 2

Pour la suite $\{s_n\}$ obtenue par l'itération $s_{n+1} = A\, s_n + b$, où A satisfait les hypothèses (H_0) on a :

$$\omega = \frac{\lambda_1}{1-\lambda_1} \tag{1.5}$$

DEMONSTRATION

Puisque $s_o - s$ peut être exprimé dans la base de R^p formée par les vecteurs propres de A et $s_n - s = A^n(s_o - s)$, on a :

$$s_n - s = \lambda^n (y + e_n) \quad ,$$

avec $\lim_{n\to\infty} e_n = 0$

étant donné que $|\lambda_1| > |\lambda_i|$, $i=2,\ldots,p$

$$\left\{\begin{array}{l} \Delta\, s_{n+1} = \lambda^{n+1}[(\lambda_1-1)y + \lambda_1\, e_{n+2} - e_{n+1}] \\ \\ \Delta^2\, s_n = \lambda^{n+1}[\dfrac{(\lambda_1-1)^2}{\lambda_1}\, y_1 + \lambda_1\, e_{n+2} - 2\,\lambda_1\, e_{n+1} + e_n] \\ \\ \dfrac{(z,\, \Delta s_{n+1})}{(z, \Delta^2 s_n)} = \dfrac{(\lambda_1-1)(z,y_1) + (z,\, \lambda_1\, e_{n+2} - e_{n+1})}{\dfrac{(\lambda_1-1)^2}{\lambda_1}(z,y_1) + (z,\, \lambda_1\, e_{n+2} - (\lambda_1+1)e_{n+1} + e_n)} \end{array}\right. \tag{1.6}$$

d'où, en supposant $(z,y_1) \neq 0$

$$\omega = -\lim_{n\to\infty} \frac{(z,\, \Delta s_{n+1})}{(z,\, \Delta^2 s_n)} = \frac{-\lambda_1}{\lambda_1 - 1} \tag{1.7}$$

Ce résultat n'est pas nouveau, l'utilisation de ω calculé par (1.5) , dans (1.1) à été déjà proposée pour améliorer la convergence des itérations linéaires (voir [12]). Mais en général, on ne connait pas la valeur de λ_1 et le calcul de ω par (1.5) devient difficile à effectuer. C'est aussi le cas pour d'autres suites, pour lesquelles, on n'a pas accès à la limite définie par l'équation (1.4). On peut en conséquence envisager plutôt une suite $\{\omega_n\}$ convergente vers ω , de façon à vérifier toujours $\lim_{n\to\infty} \frac{(z,\ \Delta u_n)}{(z,\Delta s_{n+1})} = 0$. Le choix le plus naturel est de prendre

$$(1.8) \qquad \omega_n = - \frac{(z,\ \Delta s_{n+1})}{(z,\ \Delta^2 s_n)}$$

ce qui revient, dans le cas particulier des itérations linéaires, à approcher λ_1 dans (1.5) par (méthode de la puissance)

$$(1.9) \qquad \lambda_1 \simeq \frac{(z,\ \Delta\ s_{n+1})}{(z,\ \Delta\ s_n)}$$

De cette façon, on retrouve le procédé Δ^2 d'Aitken, tel qu'il a été généralisé par Brezinski (voir [10])

$$(1.10) \qquad u_n = s_{n+1} - \frac{(z,\ \Delta s_{n+1})}{(z,\ \Delta^2 s_{n+1})}\ \Delta\ s_n$$

La suite $\{u_n\}$ converge plus vite que la suite $\{s_{n+1}\}$, pour toute suite $\{s_n\}$ convergente vers s , parce que par construction de $\{u_n\}$, $\lim_{n\to\infty} \frac{(z,\ \Delta u_n)}{(z,\ \Delta s_n)} = 0$. Pour les suites produites par des itérations linéaires, on a en plus :

THEOREME 3

Si on applique l'algorithme (1.10) à la suite $\{s_n\}$ obtenue par l'itération $s_{n+1} = A\, s_{n+1} + b$, où la matrice A satisfait les hypothèses (H_0) , alors la suite $\{u_n\}$ converge plus vite que la suite $\{s_{n+1}\}$ dans ce sens que :

$$(1.11) \quad \lim_{n\to\infty} \frac{(y,u_n-s)}{(y,s_{n+1}-s)} = 0 \quad , \quad \forall\, y \text{ tel que } (y,v_1) \neq 0$$

où v_1 est le vecteur propre de A associé à λ_1 .

DEMONSTRATION

Soit $s_0-s = a_1v_1 + a_2v_2 + \ldots + a_pv_p$

et supposons en plus $a_1 \neq 0$. Prenons $t_1 = a_1v_1$:

Du fait que $s_n-s = A^n(s_0-s)$ on a :

$$(1.12) \quad \begin{cases} s_{n+1}-s = \lambda_1^{n+1}(t_1+e_{n+1}) \\ \Delta\, s_n = \lambda_1^n[(\lambda_1-s)t_1 + \lambda_1 \quad e_{n+1}-e_n] \end{cases}$$

avec $\lim_{n\to\infty} e_n = 0$, étant donné que $|\lambda_1| > |\lambda_i|$; $i=2,\ldots,n$.

$$(1.13) \quad \frac{(y,\ \Delta s_n)}{(y,s_{n+1}-s)} = \frac{\lambda_1^n[(\lambda_1-1)(y,t_1) + (y,\lambda_1 e_{n+1}-e_n)]}{\lambda_1^{n+1}[(y,t_1) + (y,e_{n+1})]}$$

$$
\text{Donc,}\quad
\begin{cases}
\displaystyle\lim \frac{(y,\Delta s_n)}{(y,s_{n+1}-s)} = \frac{\lambda_1-1}{\lambda_1} \\[2ex]
\displaystyle\frac{(y,u_n-s)}{(y,s_{n+1}-s)} = 1 - \frac{(z,\Delta s_{n+1})}{(z,\ \Delta^2 s_n)} \cdot \frac{(y,\Delta s_n)}{(y,s_{n+1}-s)} \\[2ex]
\displaystyle\lim_{n\to\infty} \frac{(y,u_n-s)}{(y,s_{n+1}-s)} = 1 - \frac{\lambda_1}{(\lambda_1-1)} \cdot \frac{(\lambda_1-1)}{\lambda_1} \\[2ex]
\displaystyle\lim_{n\to\infty} \frac{(y,u_n-s)}{(y,s_{n+1}-s)} = 0
\end{cases}
\qquad (1.14)
$$

Le but de cette méthode, que l'on retrouvera par la suite, en étudiant l'ε-algorithme topologique, est de faire disparaître l'erreur liée à la valeur propre de plus grand module λ_1 . Dans le cas ou les valeurs propres de A sont suffisemment séparées, on peut envisager d'itérer la même méthode, afin de diminuer de façon simultanée les erreurs liées aux différentes valeurs propres de A ; c'est ce que nous allons étudier dans le paragraphe suivant .

I-2 ITERATION DE LA METHODE PRECEDENTE

Nous présentons maintenant une généralisation de la méthode précédente, qui consiste à accélérer la suite $\{u_n\}$ de la même façon qu'on a accéléré $\{s_n\}$ et ainsi de suite. Pour une suite $\{s_n\}$ donnée, on peut construire le tableau a double entrée, formé par les vecteurs $u_k^{(n)}$ définis par :

$$
\begin{cases}
u^{(n)} = s_n \quad , \quad n=0,1,\ldots \\[2ex]
u_{k+1}^{(n)} = u_k^{(n+1)} - \dfrac{(z,\Delta u_k^{(n+1)})}{(z,\Delta^2 u_k^{(n)})}\ \Delta u_k^{(n)} \qquad \begin{matrix} k=0,1,\ldots \\ n=0,1,\ldots \end{matrix}
\end{cases}
\qquad (1.15)
$$

avec $y \in R^p$ non nul choisi convenablement pour que la méthode soit bien définie. Dans le cas qui nous intéresse des suites obtenues par itération linéaire , nous supposerons que $(z,v_i) \neq 0$, $i=1,\ldots,p$, condition suffisante pour que la méthode soit bien définie. Voici quelques résultats sur cette méthode.

THEOREME 4

Si on applique l'algorithme (1.15) à la suite $\{s_n\}$ obtenue par l'itération $s_{n+1} = A\, s_n + b$, où la matrice A satisfait les hypothèses (H_1) alors :

$$(p_1) \qquad u_k^{(n)} - s \sim \sum_{i=k+1}^{p} \lambda_i^{n+k}\, y_i \quad ; \quad k=0,\ldots,p-1$$

$$(p_2) \qquad \lim_{n\to\infty} \frac{(y,u_{k+1}^{(n)}-s)}{(y,u_k^{(n)}-s)} = 0 \quad ; \quad k=0,\ldots,p-1$$

$$(p_3) \qquad \lim_{n\to\infty} \frac{(y,\Delta u_{k+1}^{(n)})}{(y,\Delta u_k^{(n)})} = 0 \quad ; \quad k=0,\ldots,p-1$$

$\forall\, y \in R^p$ tel que $(y,v_i) \neq 0$; $i=1,\ldots,p$ et

$y_i = a_i\, v_i$ où $s_0 - s = a_1 v_1 + a_2 v_2 + \ldots + a_p v_p$.

DEMONSTRATION

Du fait que $s_n - s = A^n(s_0 - s)$ et que $u_0^{(n)} = s_n$, on a :

$$(1.16) \qquad u_0^{(n)} - s = \sum_{i=1}^{p} \lambda_i^n\, y_i$$

Supposons que (p_1) ait été démontrée jusqu'à k .

Démontrons qu'elle est encore vraie pour $k+1$.

$$(1.17)\quad \begin{cases} \Delta u_k^{(n)} \sim \sum_{i=k+1}^{p} \lambda_i^{n+k} (\lambda_i - 1) y_i \\ \Delta u_k^{(n)} \sim \lambda_{k+1}^{n+k} \left[(\lambda_{k+1} - 1) y_{k+1} + \sum_{i=k+2}^{p} \left(\frac{\lambda_i}{\lambda_{k+1}}\right)^{n+k} (\lambda_i - 1) y_i \right] \end{cases}$$

Mais comme par hypothèse $\dfrac{|\lambda_i|}{|\lambda_{k+1}|} < 1$, $i=k+2,\ldots,p$, on a :

$$(1.18)\quad \begin{cases} \Delta u_k^{(n)} \sim \lambda_{k+1}^{n+k} (\lambda_{k+1} - 1) y_{k+1} \\ \Delta^2 u_k^{(n)} \sim \lambda_{k+1}^{n+k} (\lambda_{k+1} - 1)^2 y_{k+1} \\ u_{k+1}^{(n)} - s \sim \lambda_{k+1}^{n+k+1} y_{k+1} + \sum_{i=k+2}^{p} \lambda_i^{n+k+1} y_i - \dfrac{\lambda_{k+1}}{(\lambda_{k+1} - 1)} \lambda_{k+1}^{n+k} (\lambda_{k+1} - 1) y_{k+1} \\ u_{k+1}^{(n)} - s \sim \sum_{i=k+2}^{p} \lambda_i^{n+k+1} y_i \end{cases}$$

(p_1) étant démontrée, pour (p_2) et (p_3) on aura respectivement :

$$(1.19)\quad \begin{cases} \dfrac{(y, u_{k+1}^{(n)} - s)}{(y, u_k^{(n+1)} - s)} \sim \dfrac{\sum_{i=k+2}^{p} \left(\frac{\lambda_i}{\lambda_{k+1}}\right)^{n+k+1} (y, y_i)}{(y, y_{k+1}) + \sum_{i=k+2}^{p} \left(\frac{\lambda_i}{\lambda_{k+1}}\right)^{n+k+1} (y, y_i)} \\ \lim_{n \to \infty} \dfrac{(y, u_{k+1}^{(n)} - s)}{(y, u_k^{(n+1)} - s)} = 0 \\ \dfrac{(y, \Delta u_{k+1}^{(n)})}{(y, \Delta u_k^{(n+1)})} = \dfrac{\sum_{i=k+2}^{p} \left(\frac{\lambda_i}{\lambda_{k+1}}\right)^{n+k+1} (\lambda_i - 1)(y, y_i)}{(\lambda_{k+1} - 1)(y, y_{k+1}) + \sum_{i=k+2}^{p} \left(\frac{\lambda_i}{\lambda_{k+1}}\right)^{n+k+1} (\lambda_i - 1)(y, y_i)} \\ \lim \dfrac{(y, \Delta u_{k+1}^{(n)})}{(y, \Delta u_k^{(n+1)})} = 0 \end{cases}$$

D'après les hypothèses $(y,y_{k+1}) \neq 0$,

$$\lambda_{k+1} \neq 1 \quad \text{et} \quad \left|\frac{\lambda_i}{\lambda_{k+1}}\right| < 1 \quad , \quad i=k+2,\ldots,p \ .$$

On a donc démontré que pour les suites obtenues par une itération linéaire, avec une matrice A qui satisfait les hypothèses (H_1) , pour tout $1 < k \leq p$, la suite $\{u_k^{(n)}\}$ converge plus vite que $\{u_{k-1}^{(n)}\}$ au sens des deux critères différents d'accélération de la convergence $((p_2)$ et $(p_3))$.

I-3 APPLICATION AU CALCUL DES ELEMENTS PROPRES D'UNE MATRICE

Le procédé (1.15) peut aussi être utilisé pour le calcul simultané des éléments propres d'une matrice A satisfaisant les hypothèses (H_1) , en l'appliquant à la suite $\{s_n\}$ tel que $s_{n+1} = A\, s_n$. Pour ce cas, on a le théorème suivant, que nous donnons sans démonstration, (voir [9]) .

THEOREME 5

Si on applique l'algorithme (1.15) à la suite $\{s_n\}$ obtenue par l'itération $s_{n+1} = A\, s_n$, avec A satisfaisant les hypothèses (H_1) alors

$$(1.20) \quad \begin{cases} \lim\limits_{n\to\infty} \dfrac{(y,u_k^{(n+1)})}{(y,u_k^{(n)})} = \lambda_{k+1} \quad ; \quad k=0,1,\ldots,p-1 \\ \lim\limits_{n\to\infty} \dfrac{u_k^{(n)}}{(y,u_k^{(n)})} = v_{k+1} \quad ; \quad k=0,1,\ldots,p-1 \ . \end{cases}$$

Si on ne veut calculer que les valeurs propres de A , on peut utiliser une variante de la méthode précédente ; le procédé Δ^2 d'Aitken itéré, a la suite (z,s_n) . On aura un tableau de scalaires $\{\delta_k^n\}$ défini

par :

$$(1.21)\quad \begin{cases} \delta_0^{(n)} = (z, s_n) \quad , \quad n=0,1,\ldots \\ \delta_{k+1}^{(n)} = \delta_k^{(n+1)} - \dfrac{\Delta\delta_k^{(n)} . \Delta\delta_k^{(n+1)}}{\Delta^2 \delta_k^{(n)}} \end{cases}$$

Cette méthode a l'avantage, par rapport à l'algorithme (1.15), d'éviter la mise en mémoire de vecteurs.

On peut démontrer, que les suites $\{\delta_k^{(n+1)}/\delta_k^{(n)}\}$ (k fixé) convergent vers λ_{k+1} , pour $k=0,1-\ldots,p-1$.

Ces méthodes sont intéressantes surtout quand on a besoin d'approcher seulement quelques valeurs propres (et éventuellement les vecteurs propres respectifs). C'est-à-dire, quand il s'agit d'approcher $\lambda_1,\lambda_2,\ldots,\lambda_i$, avec i petit, par rapport à p . Pour i grand (et p aussi), la propagation des erreurs d'arrondi pose souvent des problèmes numériques.

EXEMPLE NUMERIQUE

Nous avons utilisé l'algorithme (1.20) pour approcher les valeurs propres de plus grand module des matrices suivantes :

$$A = \begin{pmatrix} 0.67 & 0.13 & 0.12 & 0.11 \\ 0.13 & 0.96 & 0.14 & 0.13 \\ 0.12 & 0.14 & 0.31 & 0.16 \\ 0.11 & 0.13 & 0.16 & 0.15 \end{pmatrix}$$

Ses valeurs propres sont

$\lambda_1 = 1.092388$ $\qquad$ $\lambda_2 = 0.638491$

$\lambda_3 = 0.311148$ $\qquad$ $\lambda_4 = 0.047971$

On obtient, pour les trois premières valeurs propres :

ITERATION	APPROX (λ_1)	APPROX (λ_2)	APPROX (λ_3)
9	1.09187	0.632921	0.546851
13	1.09233	0.638872	0.132411
18	1.09238	0.638566	0.447138
22	1.09239	0.638502	0.395406
26	1.09239	0.638493	0.383158
30	1.09239	0.638492	0.304593

Pour la matrice

$$A = \begin{pmatrix} 2 & 4 & 0 & 0 & 0 & 0 \\ -5 & 13 & -2 & 0 & 0 & 0 \\ -13 & 29 & -15 & 5 & 0 & 0 \\ -25 & 61 & -48 & 18 & 0 & 0 \\ -36 & 80 & -38 & -26 & 32 & -6 \\ -34 & 24 & 140 & -250 & 155 & 29 \end{pmatrix}$$

ayant comme valeurs propres

$\lambda_1 = 6 \quad \lambda_2 = 5 \quad \lambda_3 = 4 \quad \lambda_4 = 3 \quad \lambda_5 = 2 \quad \lambda_6 = 1$

Nous avons obtenu, pour les quatre premières valeurs propres

ITERATION	APPROX (λ_1)	APPROX (λ_2)	APPROX (λ_3)	APPROX (λ_4)
14	6.03303	4.94588	9.00122	1.54532
18	6.01563	4.98295	3.57011	1.20547
22	6.00748	4.99357	3.75755	3.95633
26	6.00359	4.99715	4.00550	3.17028
30	6.00173	4.99866	4.11617	3.05451

REMARQUES : Cette méthode à l'avantage, par rapport aux méthodes de déflation, d'effectuer le calcul des valeurs propres de façon simultanée. Comme celles-ci, elle est intéressante seulement quand on ne veut calculer qu'un nombre réduit des valeurs propres.

Dans les exemples ici présentés, nous n'avons pas pu obtenir des renseignements sur les dernières valeurs propres, à cause de la propagation des erreurs.

I-4 RESULTATS NUMERIQUES

Dans ce paragraphe, nous présentons des résultats numériques sur des différentes utilisations de la méthode (1.10), pour accélérer la convergence de la procédure Gauss-Seidel dans la résolution par la méthode des différences finies d'un problème aux dérivées partielles avec des conditions aux limites. Les variantes utilisées sont basées sur les remarques suivantes :

(a) On n'a pas intérêt à calculer tous les éléments de la suite $\{u_n\}$, pour n=1,..., mais à effectuer ce calcul seulement de temps en temps, selon qu'on est éloigné ou pas de la solution désirée.

(b) Ayant calculé un certain u_n , meilleure approximation de la solution s que le s_{n+1} correspondant, on a tout intérêt à réinitialiser le procédé itératif, avec u_n , plus tôt qu'a continuer le calcul de la suite $\{s_n\}$

(c) Si on a initialisé une itération linéaire avec un s_0 tel que $s_0-s = \varepsilon v_1 + a_2 v_2 + \ldots + a_p v_p$ avec ε assez petit par rapport aux a_i il faut un certain nombre d'itérations avant que la dominance de la valeur propre de plus grand module λ_1 se manifeste dans le vecteur s_n-s .

Nous avons donc été amenés à utiliser un procédé par étapes, de la façon suivante :

On se donne une approximation initiale u_0 et une suite $\{\ell_i\}$ d'entiers positifs plus grands que s , avec i=1,2,...
On commence avec i=1 et à l'étape i on fait :

(1.22)
1. $s_i^{(o)} = u_{i-1}$
2. $s_i^{(j+1)} = A\, s_i^{(j)} + b \quad , \quad j=0,1,\ldots,\ell_i$
   (test d'arrêt pour chaque j)
3. Application de (1.10) pour calculer une nouvelle approximation initiale
   $$u_i = s_i^{(\ell_i+1)} + \omega_i^{(\ell_i)} * (s_i^{(\ell_i+1)} - s_i^{(\ell_i)})$$
4. Retour à 1 , pour l'étape suivante.

Il se pose encore le problème de savoir quelle est la meilleure façon de choisir les ℓ_i . N'ayant pas une solution optimale, pour cette question, nous avons programmé deux versions, qui se différentient précisément dans la façon de choisir les ℓ_i (a priori ou a posteriori).

VARIANTE A (ℓ_i choisis a posteriori)

Nous avons déjà remarqué que la méthode (1.10) revient a approcher la valeur propre de plus grand module λ_1 , par le rapport (1.9). En conséquence, pour $\delta > 0$ convenablement choisi, nous déterminons ℓ_i par :

Pour $j=1,\ldots$; ℓ_i est le premier j tel que :

(1.23)
$|\rho_{j+1} - \rho_j| < \delta$, avec ρ_j défini par

$$\rho_j = \frac{(z, s_i^{(j+1)} - s_i^{(j)})}{(z, s_i^{(j)} - s_i^{(j-1)})}$$

Des résultats satisfaisants, ont été obtenus par l'application de cette variante sur le problème considéré ci-dessous, avec $\delta = 0.01$, comme le montrent les résulats du tableau 2 .

VARIANTE B (ℓ_i choisis a priori)

La suite $\{\ell_i\}$ est choisie à l'avance, sans aucun critère particulier. Par exemple, nous avons utilisé les formules de récurrence :

$$(1.24)\quad \begin{cases} \ell_{i+1} = 2\,\ell_i & \text{ou} \\ \ell_{i+1} = \ell_i + \ell' & i=1,2,\ldots \end{cases}$$

avec ℓ_1 et ℓ' donnés. Le fait d'augmenter à chaque étape le nombre d'itéré avant de réinitialiser correspond au souci de tenir compte de la remarque (c), parce que, à chaque fois que l'on réinitialise, le vecteur $s_i^{(o)}$ est tel que :

$$(1.25)\quad s_i^{(o)} - s = \varepsilon\, v_1 + a_2 v_2 + \ldots + a_p v_p$$

avec ε très petit puisque $s_i^{(o)} = u_{i-1}$ a été obtenu en voulant éliminer l'influence de λ_1 dans l'erreur $u_{i-1} - s$.

Nous avons fait des essais, avec cette variante, avec $\ell_1 = 5$, 10 et 15 et avec $\ell' = 10$ (voir tableaux 3, 4, 5 et 6). Les gains obtenus avec diverses suites $\{\ell_i\}$ ainsi définies, sont considérables et assez similaires.

LE PROBLEME TRAITE

Soit a résoudre le problème aux dérivées partielles :

$$(1.26)\quad \begin{cases} \dfrac{\partial^2 u(x,y)}{\partial^2 x^2} + \dfrac{\partial^2 u(x,y)}{\partial\, y^2} = 0 \quad \text{sur} \quad D \subset R^2 \\ u(x,y) = f(x,y) \quad \text{sur} \quad \Gamma \quad \text{frontière de} \quad D\ , \end{cases}$$

par la méthode des différences finies. Pour celà , on discrétise le domaine D et on obtient un domaine discrêt D_{hk} , ou h est le pas dans l'axe des x et k le pas dans l'axe des y . On veut obtenir une approximation la solution de (1.26), aux points intérieures de D_{nh} $(\mathring{D}_{hh})$.

En approchant les dérivées de (1.26) par des différences, on se ramène à un problème algébrique : un système d'équations linéaires, ayant p équations et p inconnues (les valeurs approchés de u aux points de $\overset{\circ}{D}_{nh}$) .
Le système à résoudre peut se mettre sous la forme d'un problème de point fixe :

(1.27) $\quad s = A\,s + b$

où A est la matrice de Gauss-Seidel associée.
Sans qu'on soit obligé d'expliciter A , on peut résoudre ce système, avec la méthode de Gauss-Seidel, qui consiste (pour un s_0 donné), à générer la suite s_n par la formule :

(1.28) $\quad s_{n+1} = A\,s_n + b$

Il est bien connu (voir [22][23] et [24]) que pour le problème traité, cette suite converge vers s et s approche la solution du problème (1.26).

Pour obtenir les résultats numériques que nous présentons dans ce paragraphe nous avons utilisé les données suivantes :

Le domaine D est celui de la figure 1.1.
Comme fonction f , nous avons utilisé une fonction harmonique, de façon a connaître la sólution exacte sur D : $u(x,y) = f(x,y)$. Ainsi, nous avons pu comparer les résultats approchés obtenus, avec la solution exacte du problème aux dérivées partielles.

(1.29) $\quad f(x,y) = \cos x \ \mathrm{sh}\ y$

Le domaine discrêt obtenu (grille) a comme caractéristiques :

$$(1.30)\quad \begin{cases} h = k = 0.01 \\ p = \text{nombre de points intérieurs} \in \overset{\circ}{D}_{hk} = 426 \\ t = \text{nombre de points frontière} \in \Gamma_{hk} = 156 \\ q = \text{nombre total de points de } D_{hk} = 582 \end{cases}$$

Comme test d'arrêt, nous avons utilisé :

$$(1.31)\quad \max_{1\leq i\leq p} \left| \frac{x_i^{(n)} - x_i^{(n-1)}}{x_i^{(n-1)}} \right| < \varepsilon$$

où ε est la précision demandée et $x^{(n)}$ et $x^{(n-1)}$ sont deux itérés successifs obtenus par l'itération $x^{(n)} = A\, x^{(n-1)} + b$, (deux itérés d'une même étape, pour les variantes A et B de (1.10)).

Dans tous les cas nous avons utilisé la même approximation initiale s_0 (où u_0) définie par :

$$(1.32)\quad \text{i-ème composante de } s_0 = \frac{t+i}{p} = \frac{156+i}{426}$$

Pour les variantes A et B de (1.10), le vecteur z a été choisi arbitrairement :

$$(1.33)\quad z_i = \begin{cases} 1 & \text{si} \quad i = 8j+1 \quad , \quad j = 0,1,\ldots,53 \\ 0 & \text{autrement} \end{cases}$$

Les résultats ont été obtenus en simple précision sur ordinateur IBM-360/67 (système CP/CMS). Nous remarquons d'autre part, que pour la méthode (1.10) il faut un vecteur de plus en mémoire que pour la méthode de Gauss-Seidel, pour laquelle il suffit d'avoir un seul vecteur en mémoire.

Dans les tableaux qui suivent, les résultats sont présentés en fonction de ε ; on donne, pour $\varepsilon = 10^{-2}, 10^{-3}, \ldots, 10^{-6}$, le nombre d'itérations et le temps de calcul requis pour satisfaire le test (1.31). On présente aussi une colonne (ER), avec les erreurs relatives, par rapport à la solution du problème aux dérivées partielles de façon a montrer qu'on a convergé vers la bonne solution.
Pour les variantes A et B de (1.10) on donne aussi le gains en itérations et en temps et les pourcentages respectifs, par rapport à la méthode de Gauss-Seidel.

Figure 1.1. : DOMAINE D

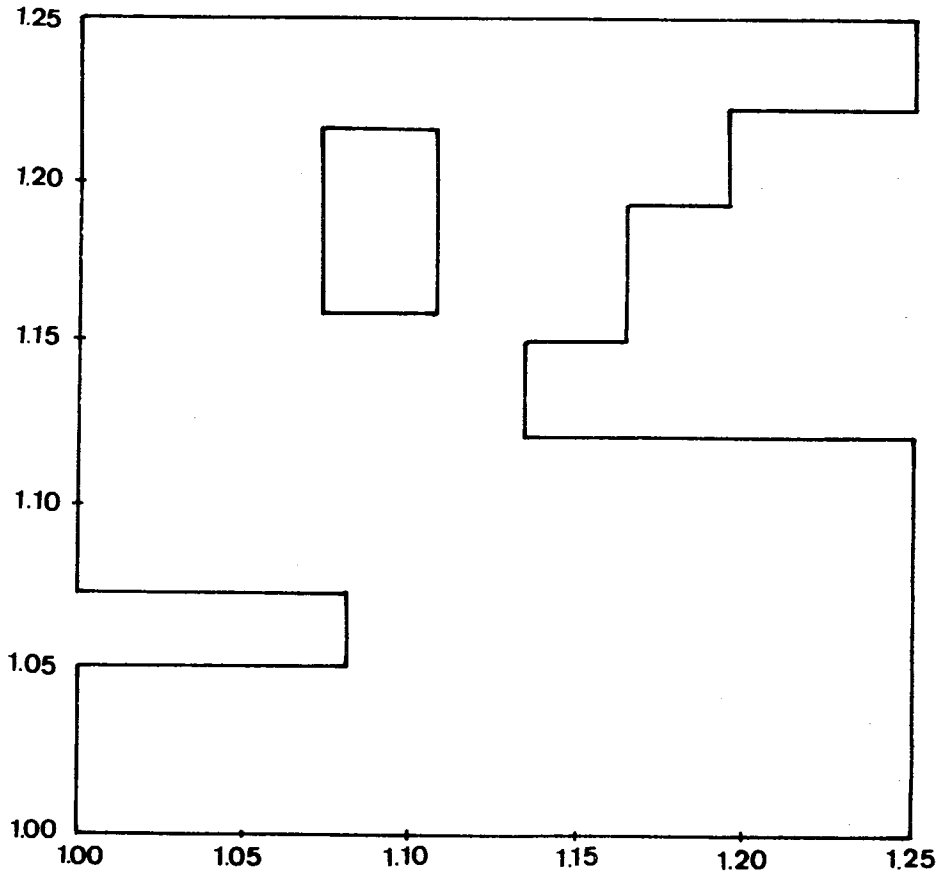


TABLEAU 1

METHODE	ε	N. ITER.	TEMPS (2)	ER (1)
Gauss-Seidel (6.5)	10^{-2}	23	1.152	0.2
	10^{-3}	72	3.568	0.19 E-1
	10^{-4}	120	5.704	0.19 E-2
	10^{-5}	167	7.925	0.17 E-3
	10^{-6}	222	10.729	0.15 E-4

TABLEAU 2

METHODE	ε	N. ITER.	GAIN/GS	%
VARIANTE A $\delta = 0.01$	10^{-2}	18	5	21.73
	10^{-3}	44	28	38.88
	10^{-4}	49	71	59.16
	10^{-5}	73	84	56.28
	10^{-6}	122	100	45.04
ER (1)	ε	TEMPS (2)	GAIN/GS	%
0.165	10^{-2}	0.900	0.252	21.87
0.58 E-2	10^{-3}	2.229	1.339	37.52
0.96 E-3	10^{-4}	2.437	3.267	57.27
0.78 E-4	10^{-5}	3.663	4.262	53.78
0.39 E-4	10^{-6}	6.102	4.627	43.12

(1) $ER = \max_i \left|\frac{x_i - \bar{x}_i}{\bar{x}_i}\right|$ où x est la solution approchée obtenue et $\bar{x}$ est la solution exacte du problème aux dérivées partielles, aux points de la maille.

(2) TEMPS mesuré en secondes, avec la procédure TIMER de la bibliothèque FORTRAN.

TABLEAU 3

METHODE	ε	N. ITER	GAIN/GS	%
VARIANTE B $\ell_1 = 10$ $\ell_i = \ell_{i-1} + 10$	10^{-2}	18	5	21.73
	10^{-3}	39	33	45.83
	10^{-4}	62	58	48.33
	10^{-5}	82	85	50.89
	10^{-6}	109	113	50.90
ER	ε	TEMPS	GAIN/GS	%
0.16	10^{-2}	0.963	0.189	16.40
0.78 E-2	10^{-3}	1.992	1.576	44.17
0.13 E-2	10^{-4}	3.112	2.592	45.44
0.91 E-4	10^{-5}	4.101	3.824	48.25
0.30 E-4	10^{-6}	5.401	5.328	49.66

TABLEAU 4

METHODE	ε	N. ITER	GAIN/GS	%
VARIANTE B $\ell_1 = 10$ $\ell_i = 2\ell_{i-1}$	10^{-2}	18	5	21.73
	10^{-3}	39	33	45.83
	10^{-4}	62	58	48.33
	10^{-5}	84	84	49.70
	10^{-6}	123	99	44.59
ER	ε	TEMPS	GAIN/GS	%
0.16	10^{-2}	0.960	0.192	16.66
0.78 E-2	10^{-3}	2.051	1.517	42.51
0.13 E-2	10^{-4}	3.118	2.586	46.54
0.97 E-4	10^{-5}	4.219	3.706	46.76
0.26 E-4	10^{-6}	6.225	4.504	41.98

TABLEAU 5

METHODE	ε	N. ITER	GAIN/GS	%
VARIANTE B $\ell_1 = 5$ $\ell_{i+1}=2\ell_i$	10^{-2}	20	3	13.04
	10^{-3}	49	23	31.94
	10^{-4}	80	40	33.33
	10^{-5}	104	63	37.72
	10^{-6}	165	57	25.67
ER	ε	TEMPS	GAIN/GS	%
0.62 E-1	10^{-2}	1.079	0.073	6.33
0.16 E-1	10^{-3}	2.458	1.110	31.10
0.72 E-3	10^{-4}	4.008	1.696	29.73
0.20 E-3	10^{-5}	5.324	2.601	32.82
0.31 E-4	10^{-6}	8.418	2.311	21.54

TABLEAU 6

METHODE	ε	N. ITER	GAIN/GS	%
VARIANTE B $\ell_1 = 15$ $\ell_{i+1}=2\ell_i$	10^{-2}	22	1	4.34
	10^{-3}	46	26	36.11
	10^{-4}	77	43	35.83
	10^{-5}	84	83	49.70
	10^{-6}	112	110	49.55
ER	ε	TEMPS	GAIN/GS	%
0.11	10^{-2}	1.157	0	0
0.10 E-1	10^{-3}	2.309	1.259	35.28
0.19 E-2	10^{-4}	3.823	1.881	32.97
0.76 E-4	10^{-5}	4.169	3.756	47.39
0.24 E-4	10^{-6}	5.541	5.188	48.35

CONCLUSIONS

Les résultats numériques que nous venons de présenter montrent qu'avec un algorithme très simple à mettre en pratique et, même, sans avoir des critères optimaux pour déterminer les paramètres le définissant, on peut obtenir des gains considérables. Pour l'exemple traité nous avons obtenu jusqu'à 59 % de gain en nombre d'itérations et 57 % en temps de calcul. La moyenne pour tous les tableaux est de 38 % pour le nombre d'itérations et de 35 % pour le temps de calcul. Ces résultats montrent d'une part, qu'on a une liberté assez considérable dans le choix des paramètres et d'autre part, que le temps de calcul nécessaire pour la mise en oeuvre de la procédure d'accélération de la convergence est très petit, par rapport au temps total de calcul (le gain en temps étant très proche du gain en nombre d'itérations).

A la vue de ces résultats, on peut conclure qu'on a tout intérêt à utiliser régulièrement les méthodes d'accélération de la convergence ici proposées et à les perfectionner afin d'améliorer encore les méthodes d'itérations linéaires pour la résolution des systèmes d'équations.

CHAPITRE II

GENERALISATIONS DE L'ε-ALGORITHME A DES SUITES DE VECTEURS

II-1 L'ε-ALGORITHME VECTORIEL

Soit $\{s_n\}$ une suite de vecteur de C^p . Les règles de l'ε-algorithme vectoriel, proposé par Wynn (Voir [1]), pour accélérer cette suite sont les suivantes :

$$(2.1)\quad \begin{cases} \varepsilon_{-1}^{(n)} = 0 \quad , \quad \varepsilon_0^{(n)} = s_n \quad ; \quad n=0,1,\ldots \\ \varepsilon_{k+1}^{(n)} = \varepsilon_{k-1}^{(n+1)} + (\Delta\varepsilon_k^{(n)})^{-1} \quad ; \quad n;k=0,1,\ldots \end{cases}$$

où l'inverse d'un vecteur $y \in C^p$ est définie par

$$(2.2)\quad y^{-1} = \frac{\bar{y}}{(y,y)}$$

où $\bar{y}$ dénote le vecteur conjugué de y et (y,y) est le produit scalaire dans C^p :

$$(2.3)\quad (y,y) = \sum_{i=1}^{p} y_i \bar{y}_i$$

Nous remarquons que cet inverse peut être remplacé par :

$$(2.4)\quad y^{-1} = \frac{y}{\|y\|^2}$$

où $\|.\|$ dénote une norme de vecteurs, comme l'a proposé Brezinski (Voir [1]), pour accélérer la convergence de suites dans un espace de

Banach. Le théorème fondamental dans la théorie de l'ε-algorithme vectoriel est le suivant, (voir [3] et [4]).

THEOREME 7

Si on applique l'ε-algorithme vectoriel à une suite $\{s_n\}$ de vecteurs de C^p qui converge vers s et qui vérifie :

$$(2.5) \qquad \sum_{i=0}^{p} a_i (s_{n+i} - s) = 0 \quad , \quad \forall\, n > N$$

où les a_i sont des scalaires réels, avec $a_p \neq 0$ alors :

$$(2.6) \quad \begin{cases} \varepsilon_{2p}^{(n)} = s & \forall\, n > N \quad \text{si} \quad \sum_{i=1}^{p} a_i \neq 0 \\ \varepsilon_{2p}^{(n)} = 0 & \forall\, n > N \quad \text{si} \quad \sum_{i=1}^{p} a_i = 0 \end{cases}$$

Nous remarquons que ce résultat n'est plus vrai pour une norme différente de la norme euclidienne. Pour les suites de vecteurs obtenues par une itération linéaire on a :

THEOREME 8

Si on applique l'ε-algorithme vectoriel a une suite de vecteurs $\{s_n\}$; produits par $s_{n+1} = A\, s_n + b$, avec s_o donné, où A est une matrice carrée, réelle telle que $I-A$ soit inversible, alors :

$$(2.7) \qquad \varepsilon_{2m}^{(n)} = s \qquad \forall\, n$$

où $s = (I-A)^{-1} b$ et où m est le degré du polynôme minimal de A pour le vecteur $s-s_0$. Nous ne donnons pas la démonstration de ce théorème (voir [4], par exemple) ; nous rappelons seulement que le polynôme minimal d'une matrice A , pour un vecteur y est le polynôme de plus bas degré

$$p(t) = \sum_{i=0}^{m} a_i t^i \quad , \quad \text{tel qu'on ait} \quad \Big(\sum_{i=0}^{m} a_i A^i\Big) y = 0 \ .$$

Brézinski (voir [5]) a généralisé ce résultat au cas A singulière et(ou) I-A singulière, ainsi que pour les suites produites par

$s_n = \sum_{i=1}^{k} A_i\, s_{n-i} + b$; avec $k \geq 1$ et $s_0, s_1, \ldots, s_k$ donnés. Pour

des résultats numériques concernant l'application du théorème 8 et d'autres applications, voir [5] et [6]. On voit donc, que l'ε-algorithme vectoriel fournit une méthode directe pour la résolution de systèmes d'équations linéaires. Néanmoins, dans la pratique elle est peu compétitive avec d'autres méthodes directes bien connues, comme la méthode de Gauss par exemple. C'est plutôt en tant qu'accélérateur de la convergence d'une suite donnée, que l'ε-algorithme vectoriel a été proposé. Il se pose alors le problème de savoir sous quelles conditions on peut espérer accélérer la convergence en passant d'une colonne paire à la colonne paire suivante. Les deux théorèmes qui suivent nous serviront de base pour donner une réponse à cette question dans le cas des suites produites par une itération linéaire.

THEOREME 9 (voir [5])

Si l'application de l'ε-algorithme vectoriel à $\{s_n\}$ et à $\{a\, s_n + b\}$, où $a \in R$ et où b est un vecteur de même dimension que s_n , fournit respectivement les vecteurs $\varepsilon_k^{(n)}$ et $\bar{\varepsilon}_k^{(n)}$ alors :

$$(2.8) \quad \begin{cases} \bar{\varepsilon}_{2k}^{(n)} = a\, \varepsilon_{2k}^{(n)} + b \\ \\ \bar{\varepsilon}_{2k+1}^{(n)} = \dfrac{\varepsilon_{2k}^{(n)}}{a} \end{cases}$$

THEOREME 10 (voir [9])

Soit $s_0 \in R^p$ un vecteur arbitraire, mais tel que $(s_0, v_i) \neq 0$, $i=1,\ldots,p$ et soit la suite $\{s_n\}$ de vecteurs produits par $s_{n+1} = A\, s_n$; $n=0,1,\ldots$ avec A satisfaisant les hypothèses (H_1). Par l'application de l'ε-algorithme vectoriel à cette suite on obtient :

$$\varepsilon_{2k}^{(n)} \sim \sum_{i=k+1}^{p} \lambda_i^{n+k}\, y_i \quad ; \quad k=0,\ldots,p-1$$

(2.9)

$$\varepsilon_{2k+1}^{(n)} \sim \frac{1}{\lambda_{k+1}^{n+k}} \frac{z_{k+1}}{\|z_{k+1}\|^2} \quad ; \quad k=0,\ldots,p-1$$

avec $y_i = a_i v_i$ et $z_i = (\lambda_i - 1) y_i$, et a_i, v_i définis par

$$s_0 = a_1 v_1 + a_2 v_2 + \ldots + a_p v_p$$

THEOREME 11

Si on applique l'ε-algorithme vectoriel à la suite $\{s_n\}$ produite par l'itération $s_{n+1} = A\, s_n + b$, où A est une matrice carrée, réelle, satisfaisant les hypothèses (H_1) alors ; pour q fixé on a :

$$\lim_{n\to\infty} \frac{(\varepsilon_{2k+2}^{(n)} - s, \varepsilon_{2k+2}^{(n)} - s)}{(\varepsilon_{2k}^{(n+q)} - s, \varepsilon_{2k}^{(n+q)} - s)} = 0 \qquad (2.10)$$

DEMONSTRATION

Soit $\bar{\varepsilon}_{2k}^{(n)}$ et $\varepsilon_{2k}^{(n)}$ respectivement les quantités obtenues en appliquant l'ε-algorithme vectoriel aux suites $u_n = s_n - s$ et s_n. D'après le théorème 9, on doit avoir :

$$\varepsilon_{2k}^{(n)} = s + \bar{\varepsilon}_{2k}^{(n)} \quad . \qquad (2.11)$$

Mais puisque $u_n = A^n u_0$, le théorème 10 nous dit que avec

$$u_0 = a_1 v_1 + \ldots + a_p v_p \quad , \quad y_i = a_i v_i$$

$$(2.12) \qquad \bar{\varepsilon}_{2k}^{(n)} \sim \sum_{i=k+1}^{p} \lambda_i^{n+k} y_i \qquad ; \qquad i=1,\ldots,p-1$$

et en conséquence :

$$(2.13) \qquad \varepsilon_{2k}^{(n)} - s \sim \sum_{i=k+1}^{p} \lambda_i^{n+k} y_i$$

Et en utilisant le fait que $|\lambda_{k+1}| > |\lambda_i|$, $i=k+2,\ldots,p$, on a :

$$(2.14) \quad \begin{cases} \varepsilon_{2k}^{(n)} - s \sim \lambda_{k+1}^{n+k} y_{k+1} \\ \dfrac{(\varepsilon_{2k+2}^{(n)}-s, \varepsilon_{2k+2}^{(n)}-s)}{(\varepsilon_{2k}^{(n+q)}-s, \varepsilon_{2k}^{(n+q)}-s)} \sim \dfrac{1}{\lambda_{k+1}^{2q}} \left(\dfrac{\lambda_{k+2}}{\lambda_{k+1}}\right)^{2(n+k+1)} \dfrac{(y_{k+2}, y_{k+2})}{(y_{k+1}, y_{k+1})} \end{cases}$$

ce qui entraine

$$(2.15) \qquad \lim_{n\to\infty} \frac{(\varepsilon_{2k+2}^{(n)}-s, \varepsilon_{2k+2}^{(n)}-s)}{(\varepsilon_{2k}^{(n+q)}-s, \varepsilon_{2k}^{(n+q)}-s)} = 0$$

On a donc démontré que chaque colonne paire de l'ε-algorithme vectoriel converge plus vite que la colonne paire précédente. Nous faisons remarquer aussi qu'on n'a pas fait l'hypothèse $\rho(A) < 1$, c'est à dire, que même si la suite $\{s_n\}$ est divergente, l'ε-algorithme vectoriel conserve ses propriétés. En principe, si seulement quelques valeurs propres sont de module supérieur à 1 :

$|\lambda_1| > |\lambda_2| \ldots |\lambda_i| > 1 > |\lambda_{i+1}| \ldots > |\lambda_p|$ alors, les suites $\{\varepsilon_{2k}^{(n)}\}$ seront divergentes, pour $k=0,\ldots,i$ et convergentes pour $k=i+1,\ldots,p$. Dans la pratique néanmoins, on peut avoir des problèmes de dépassement de capacité, pour n assez grand, parce que les éléments de $\{s_n\}$ croissent comme λ_1^n.

Voici, maintenant, un algorithme qui montre l'application de l'ε-algorithme vectoriel à la résolution de systèmes d'équations non linéaires (voir [5]). Soit à résoudre $x = f(x)$; où $f : R^p \to R^p$ est F-différentiable, dans un voisinage d'une solution x que nous cherchons Soit x_0 donné et à la (n+1)ième itération

$$s_0 = x_n$$

$$s_k = f(s_{k-1}) \quad ; \quad k=1,\ldots,2m-r$$

(2.16)

Application de l'ε-algorithme vectoriel aux vecteurs $s_0,\ldots,s_{2m-r}$, afin de calculer $\varepsilon_{2(m-r)}^{(r)}$

$$x_{n+1} = \varepsilon_{2(m-r)}^{(r)}$$

m est le degré du polynôme minimal de $f'(x)$ pour le vecteur $x_n - x$ et r la multiplicité éventuelle de la racine $\lambda = 0$ pour ce polynôme minimal. C'est un algorithme qui ne nécessite ni calculs de dérivées, ni inversion de matrices et qui, néanmoins, est à convergence quadratique :

THEOREME 12 (voir [5])

Soit $f : R^p \to R^p$ tel qu'il existe $x \in R^p$ qui vérifie $x = f(x)$, tel que f soit différentiable au sens de Fréchet dans un voisinage de x et tel que $I-f'(x)$ soit inversible. Alors, il existe un voisinage V de x tel que pour tout $x_0 \in V$ l'algorithme précédent converge vers x et ceci au moins quadratiquement, c'est-à-dire :

$$\|x_{n+1}-x\| = O(\|x_n-x\|^2) \ . \tag{2.17}$$

Des applications de cet algorithme ont été étudiées en [7] et [8]. Pour le problème de la propagation des erreurs d'arrondi, voir [17].
Une autre application de cette méthode a été étudiée par Brezinski (voir [9]); c'est le calcul simultané des éléments propres d'une matrice, par une généralisation de la méthode de la puissance :

(2.18)
- s_0 donné tel que $(s_0, v_i) \neq 0$, $i=1,\ldots,p$
- calcul de la suite $s_{n+1} = A\, s_n$, $n=0,1,\ldots$
- application de l'ε-algorithme vectoriel à $\{s_n\}$
- calcul des rapports $a_p^{(n)}$, $b_k^{(n)}$; où

$$a_k^{(n)} = \frac{(y, \varepsilon_{2k}^{(n+1)})}{(y, \varepsilon_{2k}^{(n)})}$$

$$b_k^{(n)} = \frac{(y, \varepsilon_{2k+1}^{(n)})}{(y, \varepsilon_{2k+1}^{(n)})}$$

Le théorème suivant, dont la démonstration est basée sur le théorème 10 montre les possibilités d'application de cet algorithme.

THEOREME 13 (voir [9])

Soit la suite $\{s_n\}$ de vecteurs produits par $s_{n+1} = A\, s_n$, $n=0,1,\ldots$; avec A satisfaisant les hypothèses (H_1). Par l'application de l'algorithme précédent à cette suite, on obtient :

(2.19)
$$\lim_{n\to\infty} a_k^{(n)} = \lim_{n\to\infty} b_k^{(n)} = \lambda_{k+1} \quad ; \quad k=0,1,\ldots,p-1$$

$$\lim_{n\to\infty} \frac{\varepsilon_{2k}^{(n)}}{(y, \varepsilon_{2k}^{(n)})} = \lim_{n\to\infty} (y, \varepsilon_{2k+1}^{(n)}) \varepsilon_{2k}^{(n)} = v_{k+1} \quad ; \quad k=0,1,\ldots,p-1$$

$$a_k^{(n)} = \lambda_{k+1} + O\left[\left(\frac{\lambda_{k+2}}{\lambda_{k+1}}\right)^{n+k+1}\right]$$

Nous remarquons que ce résultat est à rapprocher avec ceux obtenus au paragraphe I-3. Une variante, qui ne permet d'obtenir que les valeurs propres consiste à appliquer l'ε-algorithme scalaire à la suite $\{(y,s_m)\}$ et à considérer les rapports $\{\varepsilon_{2k}^{(n+1)}/\varepsilon_{2k}^{(n)}\}$. On évite ainsi la mise en mémoire de vecteurs.

II-2 L'ε-ALGORITHME TOPOLOGIQUE

Brezinski (voir [10]) a généralisé de deux façons différentes la transformation de Shanks, la table de Padé et l'ε-algorithme dans un espace vectoriel topologique. Pour ces généralisations de l'ε-algorithme des théorèmes similaires au théorème 7 concernant l'ε-algorithme vectoriel ont été démontrés avec,en plus, l'avantage que les coefficients a_i peuvent éventuellement être des nombres complexes. En conséquence, tous les résultats concernant l'ε-algorithme vectoriel sont aussi valables pour les deux versions de l'ε-algorithme topologique.

Prenons, par exemple, la première version de l'ε-algorithme topologique ; voici les règles qui la définissent pour une suite de vecteurs de R^p .

$$(2.20)\quad \begin{cases} \varepsilon_{-1}^{(n)} = 0 \in R^p \qquad \varepsilon_0^{(n)} = s_n \in R^p & , \quad n=0,1,\ldots \\ \varepsilon_{k+1}^{(n)} = \varepsilon_{k-1}^{(n+1)} + [\Delta\varepsilon_k^n]^{-1} & , \quad n,k=0,1,\ldots \end{cases}$$

avec

$$(2.21)\quad \begin{cases} [\Delta\varepsilon_{2k}^{(n)}]^{-1} = \dfrac{y}{(y,\Delta\varepsilon_{2k}^{(n)})} \\ [\Delta\varepsilon_{2k+1}^{(n)}]^{-1} = \dfrac{\Delta\varepsilon_{2k}^{(n)}}{(\Delta\varepsilon_{2k+1}^{(n)},\Delta\varepsilon_{2k}^{(n)})} \end{cases}$$

où $y \in R^p$ est un vecteur quelconque différent de zéro.

La deuxième version est différente de celle-ci, seulement dans la définition de l'inverse de $\Delta\varepsilon_{2k+1}^{(n)}$. La deuxième équation de (2.21) est remplacé par :

$$(2.22) \qquad [\Delta\varepsilon_{2k+1}^{(n)}]^{-1} = \frac{\Delta\varepsilon_{2k}^{(n+1)}}{(\Delta\varepsilon_{2k+1}^{(n)}, \Delta\varepsilon_{2k}^{(n+1)})}$$

Pour la première version, la suite $\varepsilon_2^{(n)}$ est identique a la suite $\{u_n\}$ définie par l'algorithme (1.10) et l'on peut envisager que l'algorithme (1.15) possède, avec cette version de l'ε-algorithme topologique une relation similaire à celle qui existe entre la méthode du Δ^2 d'Aitken vectoriel itérée ($\varepsilon_2^{(n)}$ vectoriel itéré) et l'ε-algorithme vectoriel.

Nous faisons remarquer aussi que, dans la définition des deux versions de l'ε-algorithme topologique, il intervient un vecteur y arbitraire. C'est aussi le cas pour les algorithmes étudiés au chapitre I. Des recherches plus approfondies sur cette question, pourraient conduire à la caractérisation d'un y optimal, pour certaines familles de suites. Disons seulement que du point de vue coût du calcul, on a intérêt à utiliser un y avec plusieurs composantes nulles, un vecteur de base, par exemple.

En ce qui concerne la résolution des systèmes d'équations linéaires, on peut démontrer (voir [20]) que le premier ε-algorithme topologique, (sous certaines conditions sur le choix de y et de s_0) fournit une suite identique à celle obtenue par l'application de la méthode du gradient conjugué (ou du gradient bi-conjugué, dans le cas non symétrique).

CHAPITRE III

ALGORITHMES D'ACCELERATION DE LA CONVERGENCE DES SUITES DE NOMBRES

III-1 LES ALGORITHMES DE LOSANGE

Brezinski (voir [5]) a étudié sous un même contexte les algorithmes d'accélération de la convergence de suites de nombres, qui se caractérisent par le fait qu'ils relient des quantités situés aux quatre sommets d'un losange, dans un tableau à double entrée.

Soit $\{s_n\}$ la suite que l'on veut accélérer et $\{x_n\}$ une suite de paramètres, x_n , $s_n \in R$.
Un algorithme de losange est défini par les relations suivantes :

$$(3.1)\quad \begin{cases} \theta_{-1}^{(n)} = 0 \qquad \theta_0^{(n)} = s_n & ; \quad n=0,1,\ldots \\ \theta_{k+1}^{(n)} = \theta_{-1}^{(n)} + D_k^{(n)} & ; \quad n,k=0,1,\ldots \end{cases}$$

avec, par exemple :

$$(3.2)\quad D_k^{(n)} = \frac{1}{\Delta\theta_k^{(n)}}$$

pour l'ε-algorithme.

$$(3.3)\quad D_k^{(n)} = \frac{\Delta x_n}{\Delta\theta_k^{(n)}}$$

pour la première généralisation de l'ε-algorithme.

$$(3.4) \qquad D_k^{(n)} = \frac{\Delta x_{n+k}}{\Delta \theta_k^{(n)}}$$

pour la deuxième généralisation de l'ε-algorithme.

$$(3.5) \qquad D_k^{(n)} = \frac{(x_{n+k+1} - x_n)}{\Delta\, \theta_k^{(n)}}$$

pour le ρ-algorithme, (l'opérateur Δ porte toujours sur l'indice n).

THEOREME 14 (voir [5])

Supposons que $\lim_{n\to\infty} \theta_{2k+2}^{(n)} = \lim_{n\to\infty} \theta_{2k}^{(n)}$.

Une condition nécessaire et suffisante pour que $\{\theta_{2k+2}^{(n)}\}$ converge plus vite que $\{\theta_{2k}^{(n+1)}\}$, dans le sens que $\lim_{n\to\infty} \frac{\Delta\theta_{2k+2}^{(n)}}{\Delta\theta_{2k}^{(n+1)}} = 0$, pour k fixé est que :

$$(3.6) \qquad \lim_{n\to\infty} \frac{\Delta D_{2k+1}^{(n)}}{\Delta \theta_{2k}^{(n+1)}} = -1$$

Ce théorème a l'intérêt de donner les conditions dans lesquelles on peut espérer accélérer la convergence en passant de la colonne $2k$ à la colonne $2k+2$, dans le tableau correspondant à un algorithme de losange. Il a aussi conduit à l'introduction d'un paramètre d'accélération dans la formule définissant $\theta_{2k+2}^{(n)}$, de façon à satisfaire le critère d'accélération de la convergence même si (3.6) n'est pas vérifié. Celà définit un nouvel algorithme de losange, associé à chaque choix particulier de $D_{2k+1}^{(n)}$. Voici ses règles de calcul :

$$(3.7) \qquad \left\{ \begin{array}{l} \theta_{-1}^{(n)} = 0 \qquad\qquad \theta_0^{(n)} = s_n \quad ; \quad n=0,1,\ldots \\ \theta_{2k+1}^{(n)} = \theta_{2k-1}^{(n)} + D_{2k}^{(n)} \\ \theta_{2k+2}^{(n)} = \theta_{2k+1}^{(n)} + \omega_k D_{2k+1}^{(n)} \end{array} \right\} \quad k,n=0,1,\ldots$$

THEOREME 15 (voir [5])

Supposons que $\lim_{n\to\infty} \theta_{2k+2}^{(n)} = \lim_{n\to\infty} \theta_{2k}^{(n)}$.

Une condition nécessaire et suffisante pour que $\{\theta_{2k+2}^{(n)}\}$ converge plus vite que $\{\theta_{2k}^{(n+1)}\}$ est de prendre :

$$(3.8) \qquad \omega_k = - \lim_{n\to\infty} \frac{\Delta\theta_{2k}^{(n+1)}}{\Delta D_{2k+1}^{(n)}}$$

Bien sûr, l'algorithme (3.7) n'est utilisable que dans la mesure où l'on a accès à la limite qui définit ω_k . Voici quelques exemples où on peut le déterminer (voir [5]).

(e_1) Pour la suite $s_n = s + \alpha \lambda^n$; $\omega_0 = 1$ et on retrouve alors le procédé Δ^2 d'Aitken et en conséquence $\theta_2^{(0)} = s$

(e_2) Pour la suite $s_n = s + \frac{1}{(n+1)}$, on a $\omega_0 = 2$ et on retrouve alors la deuxième colonne du ρ-algorithme, c'est-à-dire :

$$\theta_2^{(n)} = \rho_2^{(n)} = s \quad , \quad \forall\, n \ .$$

(e_3) Pour les suites à convergence logarithmique :

$s_n = s + dn$, avec $d_{n+1} = d_n + a_1 d_n^p + \ldots$

on a $\omega_0 = p$.

(e_4) Pour la suite de sommes partielles d'une série $s_n = \sum_{i=0}^{n} a_i$; telle que $\lim_{n\to 0} \frac{a_{n+1}}{a_n} \neq 1$ on trouve aussi $\omega_0 = 1$.

(e_5) Pour la suite $s_n = \sum_{i=0}^{n} \frac{1}{n^\alpha}$, $\omega_0 = \frac{\alpha}{\alpha-1}$.

Mais malheureusement très souvent le calcul du paramètre ω_k est soit très coûteux, soit impossible à réaliser. D'où l'idée d'approcher ω_k

par :

$$(3.9) \qquad \omega_k^{(n)} = - \frac{\Delta\theta_{2k}^{(n+1)}}{\Delta D_{2k+1}^{(n)}}$$

ce qui définit le θ-algorithme que nous étudions au prochain paragraphe. Une étude systématique des algorithmes de losange a été faite par Petit [18]

III-2 LE θ-ALGORITHME

On peut donc associer à chaque algorithme de losange (défini par la façon de déterminer $D_k^{(n)}$) un autre algorithme, appelé le θ-algorithme, qui n'est pas de losange et dont les règles sont les suivantes :

$$(3.10) \quad \left\{ \begin{array}{l} \theta_{-1}^{(n)} = 0 \qquad\qquad \theta_0^{(n)} = s_n \quad ; \quad n=0,1,\ldots \\ \theta_{2k+1}^{(n)} = \theta_{2k-1}^{(n+1)} + D_{2k}^{(n)} \\ \theta_{2k+2}^{(n)} = \dfrac{D_{2k+1}^{(n+1)}\theta_{2k}^{(n+1)} - D_{2k+1}^{(n)}\theta_{2k}^{(n+2)}}{D_{2k+1}^{(n+1)} - D_{2k+1}^{(n)}} \end{array} \right\} \; n,k=0,1,\ldots$$

Plus particulièrement, nous appellerons θ-algorithme celui défini par le choix $D_k^{(n)} = \dfrac{1}{\Delta\theta_k^{(n)}} \cdot \theta_{2k+2}^{(n)}$ dans (3.10) est alors donné par :

$$(3.11) \qquad \theta_{2k+2}^{(n)} = \frac{\theta_{2k}^{(n+2)}\Delta\theta_{2k+1}^{(n+1)} - \theta_{2k}^{(n+1)}\Delta\theta_{2k+1}^{(n)}}{\Delta^2 \theta_{2k+1}^{(n)}}$$

THEOREME 16 (voir [5])

Une condition nécessaire et suffisante pour que $\lim_{n\to\infty} \theta_{2k+2}^{(n)} = s$ est qu'il existe $\alpha < 1 < \beta$ tels que :

$$(3.12) \qquad \frac{D_{2k+1}^{(n+1)}}{D_{2k+1}^{(n)}} \notin [\alpha,\beta] \quad ; \quad \forall\, n \,.$$

THEOREME 17 (voir [5])

Supposons que $\lim_{n\to\infty} \theta_{2k}^{(n)} = \lim_{n\to\infty} \theta_{2k+2}^{(n)}$. Si

$\lim_{n\to\infty} \frac{\theta_{2k}^{(n+1)} - s}{D_{2k+1}^{(n)}}$ et $\lim_{n\to\infty} \frac{\Delta\theta_{2k}^{(n+1)}}{\Delta D_{2k+1}^{(n)}}$ existent, sont finies et sont égales,

alors $\{\theta_{2k+2}^{(n)}\}$ converge plus vite que $\{\theta_{2k}^{(n+1)}\}$, en ce sens que

$$\lim_{n\to\infty} \frac{\theta_{2k+2}^{(n)} - s}{\theta_{2k}^{(n+1)} - s} = 0 .$$

Une comparaison des résultats numériques a été faite, entre l'ε-algorithme, le ρ-algorithme et le θ-algorithme, sur quatorze suites différentes, (voir [6]). Les résultats obtenus montrent que :

- Etant donné une suite à accélérer ; on obtient des bons résultats soit avec l'ε-algorithme, soit avec le ρ-algorithme, mais pas avec les deux.

- Le θ-algorithme se comporte toujours comme celui qui donne les meilleurs résultats et parfois mieux que les deux.

On peut donc penser que le θ-algorithme réunit les bonnes propriétés de l'ε-algorithme et du ρ-algorithme, qui sont, parmi les algorithmes d'accélération de la convergence de suites de nombres, les plus puissants. Les deux théorèmes suivants donnent une idée plus claire de la raison de ces résultats et nous font conjecturer que le θ-algorithme peut être utilisé avec succès sur une grande famille de suites.

THEOREME 18

Si on applique le θ-algorithme à la suite $\{s_n\}$, telle que $s_n = 1 + \alpha \lambda^n$, $\forall\, n \geq N$ alors on obtient :

$$\theta_2^{(n)} = s \quad ; \quad \forall\, n \geq N \tag{3.13}$$

DEMONSTRATION

Du fait qu'on sait que $\varepsilon_2^{(n)} = s$, $\forall\, n \geq N$ (voir [5]), il suffit de démontrer que $\omega_0^{(n)} = 1$, $\forall\, n \geq N$. Ce que nous ferons en utilisant la définition même de $\omega_0^{(n)}$.

$$(3.14)\quad \begin{cases} \omega_0^{(n)} = -\dfrac{\Delta s_{n+1}}{\Delta D^{(n)}} \\ \Delta D_1^{(n)} = \dfrac{1}{\Delta \theta_1^{(n+1)}} - \dfrac{1}{\Delta \theta_1^{(n)}} \\ \Delta D_1^{(n)} = \dfrac{1}{\dfrac{1}{\Delta s_{n+2}} - \dfrac{1}{\Delta s_{n+1}}} - \dfrac{1}{\dfrac{1}{\Delta s_{n+1}} - \dfrac{1}{\Delta s_n}} \\ \Delta D_1^{(n)} = \dfrac{\Delta s_{n+1} \Delta s_{n+2}}{\Delta^2 s_{n+1}} + \dfrac{\Delta s_n \Delta s_{n+1}}{\Delta^2 s_n} \\ \omega_0^{(n)} = \dfrac{1}{\dfrac{\Delta s_{n+1}}{\Delta^2 s_{n+1}} - \dfrac{\Delta s_n}{\Delta^2 s_n}} \end{cases}$$

Mais, pour la suite $s_n = s + \alpha\,\lambda^n$, $\forall\, n \geq N$ on a :

$$(3.15)\quad \begin{cases} \Delta s_n = \alpha\ \lambda^n(\lambda-1) & , \quad \forall\, n \geq N \\ \Delta^2 s_n = \alpha\,\lambda^n(\lambda-1)^2 & , \quad \forall\, n \geq N \end{cases}$$

et en conséquence :

$$(3.16)\quad \begin{cases} \omega_0^{(n)} = \dfrac{1}{\dfrac{\lambda}{\lambda-1} - \dfrac{1}{\lambda-1}} \\ \omega_0^{(n)} = 1 \end{cases}$$

THEOREME 19

Si on applique le θ-algorithme à la suite $s_n = s + \dfrac{\alpha}{(n+1)}$, $\forall\, n \geq N$, alors on obtient :

$$(3.17)\quad \theta_2^{(n)} = s \quad , \quad \forall\, n \geq N\ .$$

DEMONSTRATION

On sait aussi qu'avec la forme simplifiée du ρ-algorithme ($x_n = n$, $\forall\, n$) on a $\rho_2^{(n)} = s$, $\forall\, n \geq N$, pour la suite en question. Il suffit donc de démontrer $\omega_0^{(n)} = 2$, $\forall\, n \geq N$.

Or, pour la suite $s_n = s + \dfrac{\alpha}{(n+1)}$, $\forall\, n \geq N$, on a :

$$(3.18)\quad \begin{cases} \Delta s_n = -\dfrac{\alpha}{(n+1)(n+2)} \\ \Delta^2 s_n = \dfrac{2\,\alpha}{(n+1)(n+2)(n+3)} \end{cases}$$

d'où, en utilisant la dernière équation de (3.14) :

$$\omega_0^{(n)} = \frac{1}{-\frac{(n+2)}{2} + \frac{(n+3)}{2}}$$

(3.19)

$$\omega_0^{(n)} = 2 \qquad , \qquad \forall\, n \geq N$$

Des résultats plus complets sur cette question ont été obtenus indépendamment par Cordellier (voir [19]).

III-3 RESULTATS NUMERIQUES

Nous présentons ici quelques résultats numériques obtenus dans [6], pour comparer la forme simplifié du ρ-algorithme (avec $x_n = n$), l'ε-algorithme et le θ-algorithme. Les quantités $\varepsilon(n)$, $\rho(n)$, $\theta(n)$ des tableaux correspondent à la dernière approximation obtenue en utilisant $s_0, s_1, \ldots, s_n$, avec l'algorithme correspondant. C'est-à-dire :

$$(3.20) \qquad \varepsilon(n) = \varepsilon_{2p_2}^{q_2} \ , \quad \rho(n) = \rho_{2p_2}^{q_2} \ , \quad \theta(n) = \theta_{2p_3}^{q_3}$$

avec pour $i=2,3$:

$$(3.21) \qquad n = i\, p_i + q_i \qquad 0 \leq q_i < i$$

Du point de vue du temps de calcul, le nombre d'opérations ne change pas de façon considérable d'un algorithme à l'autre.

$s = 1$, $s_n = s_{n-1} * \frac{n-0.5}{n-0.4}$ $s = 0$

n	s_n	θ(n)	ε(n)	ρ(n)
3	0.751201923076	- 0.403717463 (D-14)	0.710227272727	0.639204545454
5	0.714458350752	- 0.339666135 (D-13)	0.650391275391	0.556084540459
8	0.681975340679	0.206234108 (D-13)	0.597002210740	0.481086776250
12	0.655035448160	- 0.207165381 (D-13)	0.553129134227	0.429537229004
17	0.632699124550	0.300563069 (D-08)	0.523465766015	0.391663248106
23	0.613910932032	0.462520922 (D-09)	0.491906236612	0.375450948864

$s_n = \sum_{k=1}^{n} \log(k+1)\log(k+2)$, $n > 0$ $s = 0.684724788564$

n	s_n	θ(n)	ε(n)	ρ(n)
3	0.502570318068	0.687007922474	0.572971912351	0.684057386298
5	0.551259264946	0.685302859717	0.623757639521	0.684723115290
8	0.589438608789	0.684724044034	0.653717139179	0.684724787794
12	0.615741029538	0.684724788257	0.667808740172	0.684724788564
17	0.633435240598	0.684724788216	0.675019602282	0.684724788562
23	0.645505750701	0.684724788334	0.678369644239	0.684724788564

$$s_n = \sum_{k=0}^{n} (-1)^k \, k!$$

$s = 0.5963473612$

n	s_n	θ(n)	ε(n)	ρ(n)
3	- 0.4000000 (D 01)	0.615384615384	0.50000000	- 0.100000000 (D 01)
5	- 0.1000000 (D 03)	0.645161290322	0.571428571428	- 0.5407166122377 (D 01)
8	0.35900000 (D 05)	0.597646560936	0.598802395209	0.398485198977 (D 02)
12	0.442386620 (D 09)	0.59634818696	0.59616623707	0.145532336594 (D 04)
17	- 0.335990918 (D 15)	0.596347297411	0.596214683896	- 0.671696266553 (D 06)
23	- 0.24776789 (D 23)	0.596347383127	0.596327267089	- 0.983169594460 (D 09)

$$s_n = 1 + 3 \exp\left(-1.4(1.1)^{n-1}\right)$$

$s = 1$

n	s_n	θ(n)	ε(n)	ρ (n)
3	1.551348760260	1.408081289325	0.209746435967	- 0.113185588832 (01)
5	1.386305137043	1.212479371541	0.1527743331186 (01)	0.309329263489 (01)
8	1.196009598045	1.000286411811	0.207067688222 (01)	0.316460719711 (01)
12	1.055257505597	0.9989165974428	0.239349011309 (01)	0.307755281503 (01)
17	1.004823044592	1.000000432643	0.168940190898 (01)	0.279435102738 (01)
23	1.000039708092	1.000000037261	0.554014627845	0.222282330947 (01)

CHAPITRE IV

ALGORITHMES DE LOSANGE POUR LES SUITES DE VECTEURS

IV-1 GENERALISATION DES ALGORITHMES DE LOSANGE

Soit $\{s_n\}$ une suite de vecteurs de R^p et $\{x_n\}$ une suites de nombres. Le ρ-algorithme et les deux généralisations de l'ε-algorithme peuvent être définis aussi pour les suites de vecteurs, au moyen d'un algorithme général, dont les ε-algorithmes vectoriel et topologique sont des cas particuliers. Les règles de calcul de cet algorithme sont les suivantes

$$(4.1)\quad \begin{cases} \theta_{-1}^{(n)} = 0 \in R^p & \theta_0^{(n)} = s_n \quad ; \quad n=0,1,\ldots \\ \theta_{k+1}^{(n)} = \theta_{k-1}^{(n+1)} + D_k^{(n)} & ; \quad n,k=0,1,\ldots \end{cases}$$

Voici ce qu'on obtient avec quelques choix particuliers de $D_k^{(n)}$:

$$(4.2)\quad D_k^{(n)} = \frac{\Delta\theta_k^{(n)}}{(\Delta\theta_k^{(n)}, \Delta\theta_k^{(n)})}$$

pour l'ε-algorithme vectoriel.

$$(4.3) \qquad D_{2k}^{(n)} = \frac{y}{(y,\Delta\theta_{2k}^{(n)})} \quad , \quad D_{2k+1}^{(n)} = \frac{\Delta\theta_{2k}^{(n)}}{(\Delta\theta_{2k+1}^{(n)},\Delta\theta_{2k}^{(n)})}$$

avec $y \in R^p$ arbitraire non nul, pour la première version de l'ε-algorithme topologique

$$(4.4) \qquad D_{2k}^{(n)} = \frac{y}{(y,\Delta\theta_{2k}^{(n)})} \quad , \quad D_{2k+1}^{(n)} = \frac{\Delta\theta_{2k}^{(n+1)}}{(\Delta\theta_{2k+1}^{(n)},\Delta\theta_{2k}^{(n+1)})}$$

avec $y \in R^p$ arbitraire non nul, pour la deuxième version de l'ε-algorithme topologique

$$(4.5) \qquad D_{k}^{(n)} = \frac{\Delta x_n}{(\Delta\theta_{k}^{(n)},\Delta\theta_{k}^{(n)})} \quad \Delta\theta_{k}^{(n)}$$

Pour la version vectorielle de la première généralisation de l'ε-algorithme

$$(4.6) \qquad D_{k}^{(n)} = \frac{\Delta x_{n+k}}{(\Delta\theta_{k}^{(n)},\Delta\theta_{k}^{(n)})} \quad \Delta\theta_{k}^{(n)}$$

Pour la version vectorielle de la deuxième généralisation de l'ε-algorithme

$$(4.7) \qquad D_{k}^{(n)} = \frac{x_{n+k+1}-x_n}{(\Delta\theta_{k}^{(n)},\Delta\theta_{k}^{(n)})} \quad \Delta\theta_{k}^{(n)}$$

Pour le ρ-algorithme vectoriel.

On se pose une fois de plus le problème de savoir sous quelles conditions on accélère la convergence passant d'une colonne paire à la suivante.

THEOREME 20

Supposons que $\lim_{n\to\infty} \theta_{2k+2}^{(n)} = \lim_{n\to\infty} \theta_{2k}^{(n)} = s$.

Une condition nécessaire et suffisante pour que $\{\theta_{2k+2}^{(n)}\}$ converge plus vite que $\{\theta_{2k}^{(n+1)}\}$ pour k fixé, dans le sens que

$$(4.8) \qquad \lim_{n\to\infty} \frac{(\Delta\theta_{2k+2}^{(n)}, \Delta\theta_{2k+2}^{(n)})}{(\Delta\theta_{2k}^{(n+1)}, \Delta\theta_{2k}^{(n+1)})} = 0$$

est que :

$$(4.9) \qquad \lim_{n\to\infty} \frac{(\Delta D_{2k+1}^{(n)}, \Delta D_{2k+1}^{(n)}) + 2\,(\Delta\theta_{2k}^{(n+1)}, \Delta D_{2k+1}^{(n)})}{(\Delta\theta_{2k}^{(n+1)}, \Delta\theta_{2k}^{(n+1)})} = -1$$

DEMONSTRATION

En utilisant (4.1), on trouve :

$$(4.10) \quad \begin{cases} (\Delta\theta_{2k+2}^{(n)}, \Delta\theta_{2k+2}^{(n)}) = (\Delta\theta_{2k}^{(n+1)}, \Delta\theta_{2k}^{(n+1)}) + (\Delta D_{2k+1}^{(n)}, \Delta D_{2k+1}^{(n)}) + 2(\Delta\theta_{2k}^{(n+1)}, \Delta D_{2k+1}^{(n)}) \\ \dfrac{(\Delta\theta_{2k+2}^{(n)}, \Delta\theta_{2k+2}^{(n)})}{(\Delta\theta_{2k}^{(n+1)}, \Delta\theta_{2k}^{(n+1)})} = 1 + \dfrac{(\Delta D_{2k+1}^{(n)}, \Delta D_{2k+1}^{(n)}) + 2(\Delta\theta_{2k}^{(n+1)}, \Delta D_{2k+1}^{(n)})}{(\Delta\theta_{2k}^{(n+1)}, \Delta\theta_{2k}^{(n+1)})} \end{cases}$$

En prenant la limite quand n tend vers l'infini, on constate facilement que (4.8) entraîne (4.9) et réciproquement.

Notre but étant l'accélération des suites obtenues par des itérations linéaires, on s'est posé le problème de savoir, pour cette sorte de suites, quels choix éventuels de $D_k^{(n)}$ peuvent conduire à satisfaire (4.9). Néanmoins, (4.9) est assez difficile à manipuler et l'on a dû plutôt envisager d'exploiter directement (4.8). Voici le résultat correspondant.

THEOREME 21

Si on applique l'ε-algorithme vectoriel à la suite $\{s_n\}$, produite par l'itération $s_{n+1} = A\, s_n + b$, où A est une matrice carrée, réelle satisfaisant les hypothèses (H_1) alors :

$$(4.11) \quad \lim_{n\to\infty} \frac{(\Delta\varepsilon_{2k+2}^{(n)}, \Delta\varepsilon_{2k+2}^{(n)})}{(\Delta\varepsilon_{2k}^{(n)}, \Delta\varepsilon_{2k}^{(n)})} = 0 \quad ; \quad k=0,1,\ldots,p-1 \ .$$

DEMONSTRATION

Soient $\bar{\varepsilon}_{2k}^{(n)}$ et $\varepsilon_{2k}^{(n)}$ les quantités obtenues en appliquant l'ε-algorithme vectoriel aux suites $u_n = s_n - s$ et s_n respectivement. D'après le théorème 9 on doit avoir :

$$(4.12) \quad \varepsilon_{2k}^{(n)} = s + \bar{\varepsilon}_{2k}^{(n)}$$

Mais du fait que $u_n = A^n u_0$ et en utilisant le théorème 10 on a :

$$(4.13) \quad \bar{\varepsilon}_{2k}^{(n)} \sim \sum_{i=k+1}^{p} \lambda_i^{n+k} y_i \quad ; \quad i=1,\ldots,p-1$$

et en conséquence :

$$
(4.14)\quad
\begin{cases}
\varepsilon_{2k}^{(n)} \sim s + \sum_{i=k+1}^{p} \lambda_i^{n+k} y_i \\
\Delta\varepsilon_{2k}^{(n)} \sim \sum_{i=k+1}^{p} \lambda_i^{n+k}(\lambda_i-1)y_i \\
\Delta\varepsilon_{2k}^{(n)} \sim \lambda_{k+1}^{n+k}\left[(\lambda_{k+1}-1)y_{k+1} + \Sigma \dfrac{\lambda_i^{n+k}(\lambda_i-1)y_i}{\lambda_{k+1}^{n+k}}\right] \\
\Delta\varepsilon_{2k}^{(n)} \sim \lambda_{k+1}^{n+k}(\lambda_{k+1}-1)y_{k+1} \\
(\Delta\varepsilon_{2k}^{(n)}, \Delta\varepsilon_{2k}^{(n)}) \sim \lambda_{k+1}^{2(n+k)}(\lambda_{k+1}-1)^2(y_{k+1}, y_{k+1}) \\
\dfrac{(\Delta\varepsilon_{2k+2}^{(n)}, \Delta\varepsilon_{2k+2}^{(n)})}{(\Delta\varepsilon_{2k}^{(n+1)}, \Delta\varepsilon_{2k}^{(n+1)})} \sim \left(\dfrac{\lambda_{k+2}}{\lambda_{k+1}}\right)^{2n+k+1} \dfrac{(\lambda_{k+2}-1)(y_{k+2}, y_{k+2})}{(\lambda_{k+1}-1)(y_{k+1}, y_{k+1})}
\end{cases}
$$

et en utilisant à nouveau les hypothèses (H_1) :

$\lambda_i \neq 1$ et $|\lambda_i| > |\lambda_{i+1}|$, $i=1,\ldots,p-1$, on trouve

$$
(4.15)\quad \lim_{n\to\infty} \frac{(\Delta\varepsilon_{2k+2}^{(n)}, \Delta\varepsilon_{2k+2}^{(n)})}{(\Delta\varepsilon_{2k}^{(n+1)}, \Delta\varepsilon_{2k}^{(n+1)})} = 0 \quad ; \quad k=0,\ldots,p-1 \ .
$$

Ce résultat est similaire à celui du théorème ([11]). Mais il montre aussi que parmi les algorithmes de losange, le choix de $D_k^{(n)}$, défini, par (4.2) est d'un intérêt particulier. D'autre part, il met en question, l'utilité d'un paramètre d'accélération, pour les suites obtenues par des itérations linéaires. Nous abordons cette question au paragraphe suivant.

IV-2 INTRODUCTION D'UN PARAMETRE D'ACCELERATION

Le théorème 21 montre aussi que pour la classe de suites qui nous intéresse l'ε-algorithme vectoriel satisfait (4.8), pour $k=0,1,\ldots,p-1$.

Cette condition étant satisfaite, l'introduction d'un paramètre d'accélération ne trouve pas de justification. Mais, quoique notre but principal dans cette partie soit l'accélération des suites produites par des itérations linéaires, nous traitons ici cette question, qui peut être importante pour d'autres suites de vecteurs.

Nous envisagerons donc l'introduction d'un paramètre d'accélération et, d'une façon sommaire, le θ-algorithme vectoriel.
L'algorithme (4.1), après introduction d'un paramètre d'accélération dans le calcul des quantités $\theta_{2k}^{(n)}$ devient :

$$(4.16)\quad \left\{ \begin{array}{l} \theta_{-1}^{(n)} = 0 \in R^p \qquad \theta_0^{(n)} = s_n \quad ; \quad n=0,1,\ldots \\ \theta_{2k+1}^{(n)} = \theta_{2k-1}^{(n+1)} + D_{2k}^{(n)} \\ \theta_{2k+2}^{(n)} = \theta_{2k}^{(n+1)} + \omega_k D_{2k+1}^{(n)} \end{array} \right\} \quad k,n=0,1,\ldots$$

On pourrait, par exemple, envisager de caractériser ω_k, à partir du théorème 20, mais du fait que :

$$(4.17)\quad \frac{(\Delta\theta_{2k+2}^{(n)}, \Delta\theta_{2k+2}^{(n)})}{(\Delta\theta_{2k}^{(n+1)}, \Delta\theta_{2k}^{(n+1)})} = \frac{(\Delta D_{2k+1}^{(n)}, \Delta D_{2k+1}^{(n)})}{(\Delta\theta_{2k}^{(n+1)}, \Delta\theta_{2k}^{(n+1)})} \omega_k^2 + 2 \frac{(\Delta\theta_{2k}^{(n+1)}, \Delta D_{2k+1}^{(n)})}{(\Delta\theta_{2k}^{(n+1)}, \Delta\theta_{2k}^{(n+1)})} \omega_k + 1$$

la définition de ω_k deviendrait très compliquée, ce qui posserait des problèmes pour son utilisation. Nous suivrons plus tôt un procédé similaire à celui qui a été utilisé par Brezinski (voir [10]) pour l'introduction d'un paramètre d'accélération dans l'ε-algorithme topologique.

THEOREME 22

Supposons que $\lim_{n\to\infty} \theta_{2k+2}^{(n)} = \lim_{n\to\infty} \theta_{2k}^{(n)}$. Une condition nécessaire et suffisante pour que $\{\theta_{2k+2}^{(n)}\}$ converge plus vite que $\{\theta_{2k}^{(n)}\}$, dans le sens que :

$$(4.18) \qquad \lim_{n\to\infty} \frac{(y,\Delta\theta_{2k+2}^{(n)})}{(y,\Delta\theta_{2k}^{(n+1)})} = 0$$

pour $y \in R^p$ arbitraire différent de zéro, est de prendre

$$(4.19) \qquad \omega_k = - \lim_{n\to\infty} \frac{(y,\Delta\theta_{2k}^{(n+1)})}{(y,\Delta D_{2k+1}^{(n)})}$$

DEMONSTRATION

Elle dérive des équations (4.16) :

$$\Delta\theta_{2k+2}^{(n)} = \Delta\theta_{2k}^{(n+1)} + \omega_k \, \Delta D_{2k+1}^{(n)}$$

$$(4.20) \qquad (y,\Delta\theta_{2k+2}^{(n)}) = (y,\Delta\theta_{2k}^{(n+1)}) + \omega_k (y,\Delta D_{2k+1}^{(n)})$$

$$\frac{(y,\Delta\theta_{2k+2}^{(n)})}{(y,\Delta\theta_{2k}^{(n+1)})} = 1 + \omega_k \frac{(y,\Delta D_{2k+1}^{(n)})}{(y,\Delta\theta_{2k}^{(n+1)})}$$

Le théorème suivant est une conséquence directe du théorème 21 et du critère utilisé pour déterminer ω_k .

THEOREME 23

Si on applique l'algorithme (4.16), avec ω_k définie par (4.19) et $D_k^{(n)} = \dfrac{\Delta\theta_k^{(n)}}{(\Delta\theta_k^{(n)}, \Delta\theta_k^{(n)})}$, $n,k=0,1,\ldots$ à la suite $\{s_n\}$, obtenue par l'itération $s_{n+1} = As_n + b$, où A satisfait les hypothèses (H_1) , alors $\omega_k = 1$, $k=0,\ldots,p-1$.

Les résultats précédents montrent que pour les suites produites par une itération linéaire, l'ε-algorithme vectoriel est optimal, dans la classe des algorithmes de losange.

Nous abordons maintenant le cas où l'on ne connait pas la limite qui définit ω_k . On peut alors remplacer ω_k par une suite $\{\omega_k^{(n)}\}$ qui converge vers ω_k , comme on le fait pour définir le θ-algorithme scalaire. Pour y convenablement choisi, nous définissons la suite $\{\omega_k^{(n)}\}$ par la formule

$$(4.21) \qquad \omega_k^{(n)} = \frac{-(y, \Delta\theta_{2k}^{(n+1)})}{(y, \Delta D_{2k+1}^{(n)})}$$

et nous appelons θ-algorithme vectoriel, celui défini par les règles suivantes :

$$(4.22) \quad \left\{ \begin{array}{l} \theta_{-1}^{(n)} = 0 \in R^p \quad , \quad \theta_0^{(n)} = s_n \quad ; \quad n=0,1,\ldots \\ \theta_{2k+1}^{(n)} = \theta_{2k-1}^{(n)} + D_{2k}^{(n)} \\ \theta_{2k+1}^{(n)} = \theta_{2k}^{(n+1)} + \omega_k^{(n)} \, D_{2k+1}^{(n)} \end{array} \right\} \; k,n=0,1,\ldots$$

Nous terminons ce chapitre en donnant deux propriétés de cet algorithme, sur sa convergence et sa vitesse de convergence. Elles dérivent tout de suite de la définition (4.22). Malheureusement, les hypothèses requises ne sont pas toujours vérifiées et quand elles le sont, il est en général difficile de le démontrer.

THEOREME 24

Supposons que $\lim_{n\to\infty} \theta_{2k}^{(n)} = s$.

Une condition nécessaire et suffisante pour que $\lim_{n\to\infty} \theta_{2k+2}^{(n)} = s$ est que

$$(4.23) \quad \lim_{n\to\infty} \omega_k^{(n)} D_{2k+1}^{(n)} = 0 \in R^p$$

Si de plus, $\lim_{n\to\infty} \omega_k^{(n)}$ et $\lim_{n\to\infty} \frac{(y,\theta_{2k}^{(n+1)}-s)}{(y,D_{2k+1}^{(n)})}$ existent et sont égales,

alors :

$$(4.24) \quad \lim_{n\to\infty} \frac{(y,\theta_{2k+2}^{(n)}-s)}{(y,\theta_{2k}^{(n+1)}-s)} = 0$$

CHAPITRE V

ETUDE DE L'EFFICACITE DES METHODES ITERATIVES

V-1 LES CRITERES D'EFFICACITE ET SES LIMITATIONS

Jusqu'à présent nous avons comparé la vitesse de convergence de deux suites. Néanmoins, dans la résolution itérative d'un problème donné, la notion de vitesse de convergence n'est pas suffisante pour choisir entre deux méthodes parce qu'elle ne fait pas intervenir le travail nécessaire pour l'évaluation de chaque terme de la suite.

C'est dans le but d'avoir une méthode précise de classement de processus itératifs que la notion d'indice d'efficacité à été introduite.

Plusieurs auteurs (voir [13], [14],[26]) ont étudié l'efficacité des méthodes itératives pour résoudre des systèmes d'équations non linéaires. Nous allons maintenant donner un aperçu des critères utilisés.

Soit à résoudre le système d'équations non linéaires $f(x) = 0$, ayant s comme solution. Considérons une méthode donnée qui pour un s_0 donné engendre la suite s_n qui converge vers s .

Supposons, en plus, que les limites (utilisées) ci-dessous existent. Les quantités E_0 et E sont respectivement les indices d'efficacité définis par Ostrowski (voir [14]) et par Brent (voir [13]) :

$$(5.1)\quad \begin{cases} E_0 = \lim\limits_{n\to\infty} \sqrt[W_{n+1}]{\dfrac{\log \|s_{n+1}-s\|}{\log \|s_n-s\|}} \\ E = \lim\limits_{n\to\infty} \dfrac{1}{W_{n+1}} \log\left(\dfrac{\log \|s_{n+1}-s\|}{\log \|s_n-s\|}\right) \end{cases}$$

où W_{i+1} est le "travail" exigé pour passer de s_n à s_{n+1}, dont l'unité représente une évaluation de f (ou de l'une de ses dérivées partielles). Supposons, en plus que $W = \lim\limits_{n\to\infty} W_i$ alors, si l'ordre de convergence r de la méthode existe alors E_0 et E satisfont :

$$(5.2)\quad \begin{cases} r = \lim\limits_{n\to\infty} \dfrac{\log \|s_{n+1}-s\|}{\log \|s_n-s\|} \\ E_0 = r^{1/W} \\ E = \log r^{1/W} \end{cases}$$

Mais ces indices, s'ils sont bien adaptés pour comparer des algorithmes de résolution de systèmes d'équations dont l'ordre de convergence est plus grand que un, ne sont plus applicables quand il s'agit d'un processus itératif à convergence linéaire ($r = 1$). Des équations (5.2) on déduit que pour toute méthode à convergence linéaire, $E_0 = 1$ et $E = 0$. Il est donc nécessaire de disposer d'un critère d'efficacité permettant de comparer les méthodes quelque soit leur ordre de convergence.

D'autre part (voir [15],[16],[27]) plusieurs méthodes peuvent se mettre sous la forme d'itérations de point fixe $s_{n+1} = F(s_n)$; s est donc la solution de $s = F(s)$ (on suppose son existence et unicité). Si F est différentiable au sens de Fréchet dans un voisinage de la solution alors une mesure de l'efficacité est donnée par le rayon spectral de $F'(s)$. En effet au voisinage de la solution la méthode peut être approchée (voir [27]) par une itération linéaire du type

$$(5.3) \qquad s_{n+1}-s = F'(s)\,(s_n-s)$$

et dans ce cas on a, en général (voir [27]) $r = 1$ et

$$(5.4) \qquad \lim_{n\to\infty} \frac{\|s_{n+1}-s\|}{\|s_n-s\|} = \rho(F'(s))$$

pour toute norme de R^p .

Mais $\rho(F'(s))$ n'est une mesure d'efficacité, que si les méthodes à comparer nécessitent à peu près la même quantité de "travail", pour passer d'un itéré au suivant.

Revenons à notre objectif dans ce chapitre : comparer l'efficacité des méthodes simultanées d'accélération de la convergence étudiées dans les chapitres précédents. Malheureusement, nous ne pouvons utiliser aucune des deux approches ci-dessus, parce que, d'une part, les suites générées sont très souvent à convergence linéaire et, d'autre part il n'est pas possible de mettre chaque suite sous la forme d'une itération de point fixe $x_{n+1} = F(x_n)$. Au paragraphe suivant nous allons donc essayer de surmonter cet inconvénient en étudiant la signification de certaines quantités, ce qui nous conduira à la définition d'un indice général d'efficacité.

V-2 UN INDICE GENERAL D'EFFICACITE

Soit $\{s_n\}$ une suite convergente vers s. Supposons l'existence de r et c finis, $c \neq 0$ tels que :

$$(5.5) \qquad \limsup_{n\to\infty} \frac{\|s_{n+1}-s\|}{\|s_n-s\|^r} = c$$

r est alors appelé ordre de convergence de $\{s_n\}$ et c sa constante asymptotique d'erreur.

Posons $e_n = -\log_{10} \varphi_\infty (s_n - s)$ (norme du max). e_n représente le nombre de chiffres décimaux exacts de $\varphi_\infty(s_n-s)$ (on peut aussi parler de chiffres binaires, en utilisant le logarithme en base 2, etc...). Pour n suffisamment grand, on a approximativement :

$$(5.6) \quad \begin{cases} \|s_{n+1}-s\| \simeq c \|s_n-s\|^r \\ \\ e_{n+1} \simeq r\, e_n + d \end{cases}$$

avec $d = -\log_{10} c$. Cela signifie, que lorsqu'on passe de s_n à s_{n+1}, on multiplie par r le nombre de chiffres exacts de $\varphi_\infty(s_n-s)$ et on en ajoute d en plus.
Supposons maintenant que, ayant effectué un certain "travail" k_0, on ait D_0 chiffres exacts. Nous voulons estimer le nombre D_k de chiffres exacts, obtenus après k unités de "travail" supplémentaires. Supposons, pour simplifier, que le passage de s_k à s_{k+1} nécessite W unités de travail, pour tout k et qu'au départ, $k_0 = 0$ et $D_0 = 1$. Alors, d'après ce qui précède on a :

$$(5.7) \quad \begin{cases} D_W = r + d \\ \\ D_{2W} = r\, D_W + d \end{cases}$$

et de façon générale

$$(5.8) \qquad D_{kw} = r\, D_{(k-1)w} + d = r^k + d(1+r+\ldots+r^{k-1})$$

Supposons donc que D_n vérifie une loi de la forme :

$$(5.9) \qquad D_{n+1} = r^{1/w} D_n + b \quad , \quad D_0 = 1$$

et essayons de déterminer b :

$$(5.10) \quad \left\{ \begin{array}{l} D_0 = 1 \\ D_1 = r^{1/w} + b \\ D_2 = r^{2/w} + b\,(1+r^{1/w}) \\ \vdots \\ D_{n+1} = r^{1/w}\,[r^{n/w} + b(1+r^{1/w}+r^{2/w}+\ldots+r^{(n-1)/w})] + b \end{array} \right.$$

et ainsi de suite, d'où :

$$(5.11) \qquad D_n = r^{n/w} + b(1+r^{1/w}+\ldots+r^{(n-1)/w})$$

Mais étant donné que $D_w = r + d$, on obtient :

$$(5.12) \qquad b = \frac{d}{1+r^{1/w}+\ldots+r^{(w-1)/w}}$$

En résumé, on a donc une définition par récurrence

$$(5.13) \quad \left\{ \begin{array}{l} D_0 = 1 \\ D_{n+1} = r^{1/w} D_n + \dfrac{d}{1+r^{1/w}+\ldots+r^{(w-1)/w}} \end{array} \right.$$

et une définition directe :

$$(5.14) \quad D_n = r^{n/w} + d\,\frac{1+r^{1/w}+\ldots+r^{(n-1)/w}}{1+r^{1/w}+\ldots+r^{(w-1)/w}}$$

Regardons les propriétés asymptotiques de D_n :

$$(5.15) \quad E_0 = \lim_{n\to\infty} (D_n)^{1/n} = r^{1/w}$$

C'est le facteur par lequel on multiplie le nombre de chiffres significatifs, par unité de travail. La limite de D_n/n , quand n tend vers l'infini a aussi un sens intuitif.

$$(5.16) \quad \begin{aligned} E_a &= \lim_{n} D_n/n = \infty \qquad \text{si} \quad r > 1 \\ E_a &= \lim_{n\to\infty} D_n/n = d/w \qquad \text{si} \quad r = 1 \end{aligned}$$

C'est, asymptotiquement, le nombre de chiffres exacts ajoutés par unité de travail : $E_a = \lim_{n\to\infty} D_{n+1} - D_n$.

On peut déjà remarquer que si pour deux suites $\{x_n\}$ et $\{s_n\}$ on a

$$(5.17) \quad 1 \neq E_0\{x_n\} \geq E\{s_n\}$$

alors

$$(5.18) \quad \infty = E_a\{x_n\} \geq E_a\{s_n\}$$

Si, par contre :

$$(5.19) \quad E_0\{x_n\} = E_0\{s_n\} = 1$$

Et si $E_a\{x_n\} > E_a\{s_n\}$ alors on peut affirmer que la méthode générant la suite $\{x_n\}$ est plus efficace que celle qui génère $\{s_n\}$. Mais cette démarche comporte encore quelques difficultés : l'existence de deux indices différents et le fait qu'on ne définit pas l'efficacité de chaque méthode séparément, mais seulement une façon de comparer deux méthodes entres elles. Cela vient du fait que E_0 et E_a ne sont pas comparables (à la rigueur, on pourrait dire qu'ils ne sont pas exprimés dans les mêmes "unités").
Nous définissons donc un indice général d'efficacité, par la relation

$$(5.20) \quad E_g = \lim_{n\to\infty} (D_n)^{1/n} \cdot \exp(-n/D_n)$$

avec D_n défini par (5.14) . E_g est bien défini, si et seulement si c et r existent et sont finis. Cet indice à les propriétés suivantes

(i) Si $r > 1$ alors $\{n/D_n\}$ tend vers zéro et $\{\exp(-n/D_n)\}$ tend vers 1 . En conséquence :

$$(5.21) \quad E_g = E_0 = r^{1/w} \qquad \text{si} \quad r > 1$$

(ii) Si $r = 1$ alors $\{(D_n)^{1/n}\}$ tend vers 1 et $\lim_{c\to 0} E_g = 1$ (convergence super-linéaire)

$$(5.22) \quad \lim_{c\to 1} E_g = 0 \quad \text{(convergence logarithmique)}$$

Et finalement, si $c \in (0,1)$ alors :

$$(5.23) \quad E_g = \exp(-w/d) \ , \ E_g \in (0,1)$$

(iii) Si $E_g\{u_n\} > E_g\{x_n\}$ alors $\{u_n\}$ est plus efficace que $\{x_n\}$.

Il est à remarquer que, d'après cette définition de l'efficacité, toute méthode à convergence linéaire est moins efficace que celles dont l'ordre de convergence est supérieur à 1 . On ne peut pas comparer entre elles des méthodes à convergence super-linéaire ($r = 1$, $c = 0$) , ni des méthodes

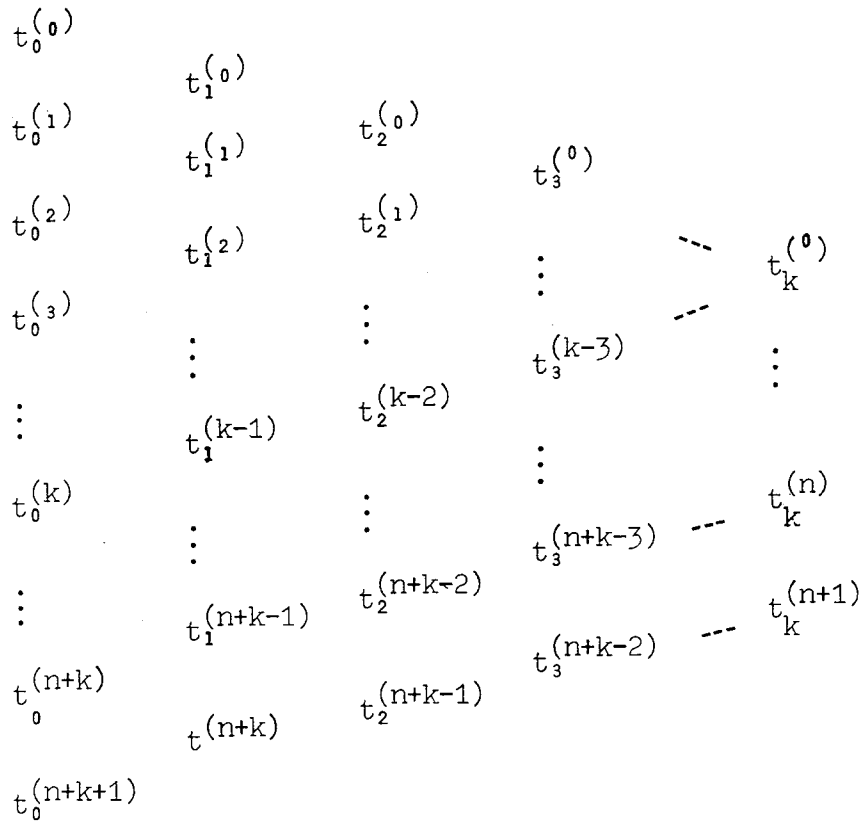
à convergence logarithmique ($r = 1$, $c = 1$) . Les quantités r , c et E_g étant des valeurs asymptotiques, on peut, dans des cas particuliers et lors des premières itérations, avoir des résultats contraires à ceux que les indices d'efficacité E_g peuvent nous faire espérer.

V-3 EFFICACITE DES ALGORITHMES ETUDIES

Dans ce paragraphe, nous nous proposons de calculer l'indice d'efficacité E_g de certains algorithmes étudiés dans les chapitres précédents. Plus précisément nous ferons ces calculs pour la méthode définie au chapitre I (règles (1.15)) et pour l'ε-algorithme vectoriel. En fait, nous identifions chaque colonne du tableau (k fixé) à double entrée défini par l'une de ces méthodes, avec un algorithme que nous appelons par le nom de la méthode, indexé par le numéro de la colonne. Ainsi, par exemple, nous parlons des "algorithmes" $\varepsilon_0, \varepsilon_2, \ldots, \varepsilon_{2k}$, qui associent à une suite $\{s_n\}$ les suites $\{\varepsilon_0^{(n)}\}, \{\varepsilon_2^{(n)}\}, \ldots, \{\varepsilon_{2k}^{(n)}\}$.

Soit une méthode qui génère le tableau $\{t_k^{(n)}\}$. Nous disons que le travail nécessaire pour passer de $t_k^{(n)}$ à $t_k^{(n+1)}$, pour k fixé, est celui que l'on doit effectuer pour calculer $t_0^{(n+k+1)}, t_1^{(n+k)}, \ldots, t_k^{(n+1)}$.

FIGURE 1.2.



Celà suppose que l'on range en mémoire certaines quantités, avec lesquelles ayant aussi calculé $t_0^{(n+k+1)}$, on peut calculer $t_1^{(n+k)}, t_2^{(n+k-1)}, \ldots, t_k^{(n+1)}$. Dans les cas qui nous intéressent, $t_0^{(n)} = s_n$, $\forall$ n et le coût de calcul de $t_i^{(n+k+1-i)}$ est le même, pour $i=1,\ldots,k$.

En conséquence, le 'travail' pour passer de $t_k^{(n)}$ à $t_k^{(n+1)}$ est égal au 'travail' pour passer de s_{n+k} à s_{n+k+1}, plus k fois le travail nécessaire, pour calculer un élément du tableau $t_i^{(n)}$, avec $i > 0$.

En ce qui concerne les ordres de convergence et les constantes asymptotiques d'erreur on peut déduire facilement d'après le théorème 4 (relation (p)) que la méthode (1.15) satisfait :

$$(5.24) \qquad \lim_{n\to\infty} \frac{\|u_k^{(n+1)}-s\|}{\|u_k^{(n)}-s\|} = |\lambda_{k+1}| \qquad , \quad k=0,1,\ldots,p-1 \ ,$$

dans le cas où $u^{(n)} = s_n$ et où la suite $\{s_n\}$ a été obtenue par l'itération $s_{n+1} = A\, s_n + b$, avec A satisfaisant les hypothèses (H_1). Sous ces mêmes hypothèses on peut démontrer, d'après le théorème 11 (relation (2.13)), que pour l'ε-algorithme vectoriel :

$$(5.25)\qquad \lim_{n\to\infty} \frac{\|\varepsilon_{2k}^{(n+1)} - s\|}{\|\varepsilon_{2k}^{(n)} - s\|} = |\lambda_{k+1}| \qquad , \quad k=0,1,\ldots,p-1 \ .$$

C'est-à-dire, que l'ordre de convergence et la constante asymptotique d'erreur des algorithmes u_k et ε_{2k} sont donnés respectivement par $r_k = 1$, $c_k = |\lambda_{k+1}|$, pour $k=0,1,\ldots,p-1$.

Supposons, en plus des hypothèses (H_1) que $|\lambda_1| < 1$, afin d'assurer la convergence des suites $\{u_k^{(n)}\}$ et $\{\varepsilon_{2k}^{(n)}\}$ pour tout k fixé , $k=0,1,\ldots,p-1$.

Soient w_s le "travail" nécessaire, pour passer de s_n à s_{n+1} . w_u et w_ε respectivement, le travail nécessaire, pour déterminer un élément $u_i^{(n)}$ et un élément $\varepsilon_i^{(n)}$, avec $i > 0$. Alors, de tout ce qui précède nous déduisons :

$$(5.26)\qquad \begin{cases} E_g\{u_k^{(n)}\} = \exp\left(\dfrac{w_s + k\, w_u}{\log_{10} |\lambda_{k+1}|}\right) \\[2ex] E_g\{\varepsilon_{2k}^{(n)}\} = \exp\left(\dfrac{w_s + 2k\, w_\varepsilon}{\log_{10} |\lambda_{k+1}|}\right) \end{cases}$$

Appelons $\alpha_k = \dfrac{\log_{10} |\lambda_1|}{\log_{10} |\lambda_{k+1}|} < 1$. $k=1,\ldots,p-1$. Alors :

$$(5.27)\quad \begin{cases} E_g\{s_n\} = \exp\left(\dfrac{w_s}{\log_{10}|\lambda_1|}\right) \\ E_g\{u_k^{(n)}\} = (E_g\{s_n\})^{\alpha k} \exp\left(\dfrac{k\, w_u}{\log_{10}|\lambda_{k+1}|}\right) \\ E_g\{\varepsilon_{2k}^{(n)}\} = (E_g\{s_n\})^{\alpha k} \exp\left(\dfrac{2k\, w_\varepsilon}{\log_{10}|\lambda_{k+1}|}\right) \end{cases}$$

Essayons de déterminer w_u et w_ε, en fonction de p (dimension) et du coût des opérations élémentaires. Les λ_i et w_s dépendent évidemment de la matrice A. Soit α le coût d'une addition ou d'une soustraction (supposé le même). $\beta\alpha$ le coût d'une multiplication. Nous considérerons (pour nous ramener à un seul paramètre) que le coût d'une division est égal à $\beta\alpha$ ce qui n'est, en général, pas vrai mais nous supposons que les calculs du type $\frac{a.v}{b}$, avec $a, b \in R$ (a éventuellement égal à 1), $v \in R^p$ se font dans l'ordre défini par $(\frac{a}{b}).v$. Le nombre de divisions effectuées, dans les méthodes ici étudiées est donc négligeable (p suffisemment grand), par rapport au nombre de multiplications ou d'additions. Pour l'ε-algorithme vectoriel on a, en utilisant la définition (2.1) :

$$(5.28)\quad w_\varepsilon = (3p-1 + \beta(2p+1))\alpha$$

et pour la méthode (1.15) :

$$(5.29)\quad w_u = (3p + \beta(2p+1))\alpha$$

Nous pouvons déjà constater que la méthode (1.15) est plus efficace que l'ε-algorithme vectoriel, pour accélérer des itérations linéaires, parce qu'on a (on suppose les hypothèses H_1 et $|\lambda_1| < 1$) :

$$(5.30)\quad \frac{E_g\{\varepsilon_{2k}^{(n)}\}}{E_g\{u_k^{(n)}\}} = \exp\left(\frac{k\alpha(3p+\beta(2p+1)-2)}{\log_{10}|\lambda_{k+1}|}\right) < 1$$

pour $k=1,\ldots,p-1$ (u_k plus efficace que ε_{2k}), et ce résultat est vérifié pour toute matrice A, satisfaisant les hypothèses (H_1) et $|\lambda_1| < 1$, indépendemment de sa forme et des valeurs des λ_i. Mais il faut préciser que la façon de mettre en oeuvre l'algorithme se manifeste directement sur les valeurs de w et de E_g.
Brezinski (voir [28]) a proposé une variante de l'ε-algorithme scalaire qui nécessite trois fois moins de temps de calcul que les règles (3.1) (3.2).Un tel résultat n'existe pas pour l'ε-algorithme vectoriel, mais on peut aussi utiliser l'ε-algorithme scalaire sur chaque suite de composantes (on a alors le même ordre de convergence et la même constante asymptotique d'erreur). L'utilisation d'une telle variante pourrait changer le résultat que nous venons d'obtenir sur l'efficacité de l'ε-algorithme par rapport à la méthode (1.15) dans la résolution itérative des systèmes d'équations. D'autre part, dans le calcul de w_u par (5.29), nous n'avons supposé aucune condition sur le vecteur z, mais puisqu'il est arbitraire, on peut le choisir convenablement, de façon a obtenir un w_u, le plus petit possible.

Pour pouvoir comparer l'efficacité des algorithmes u_i et u_j où ε_{2i} et ε_{2j} ; il faut avoir des renseignements sur certaines valeurs propres de A. Par exemple, pour déterminer si on a intérêt à calculer la suite $\{u^{(n)}\}$ à partir de $\{s_n\}$, il faut connaître $|\lambda_1|$ et $|\lambda_2|$. La méthode proposée au paragraphe I.3 peut alors être utile, parce qu'elle fourni un moyen, pour approcher simultanément les premières valeurs propres d'une matrice. Voici un exemple, pour illustrer l'utilisation de l'indice d'efficacité E_g, dans un cas très simple, où l'on connait les premières valeurs propres.

Considérons une matrice $(1000,1000)$, ayant 5000 éléments non nuls. On a alors :

(5.31) $w_s = 5000\ \alpha\ (1+\ )$

Supposons, en plus

$$\alpha = 10^{-6}$$

(5.32) $$\beta = 3$$

$$\lambda_i = \frac{0.9 + 0.1\ (i-1)}{i} \quad , \quad i=1,\ldots$$

k	$\lvert\lambda_{k+1}\rvert$	$E\{u_k^{(n)}\}$	$E\{\varepsilon_{2k}^{(n)}\}$
0	0.900000	0.827105	0.827105
1	0.500000	0.977889	0.946653
2	0.366667	0.986145	0.945713
3	0.300000	0.989057	0.940388

REMARQUES ET CONCLUSIONS

Dans l'étude des processus itératifs, il est d'une importance fondamentale d'avoir une méthode générale et convenable pour classer les algorithmes suivant leur efficacité. C'est dans le but de faire un pas dans cette voie que nous avons défini l'indice E_g qui généralise l'indice d'efficacité E_0 d'Ostrowski. Néanmoins, plusieurs problèmes restent à résoudre dans ce domaine. Peut être le plus important est celui de pouvoir tenir compte de la place mémoire nécessaire pour utiliser l'algorithme. Il y a certainement des relations entre cette variable, le coût d'une itération et la vitesse de convergence, mais elles sont difficiles à chiffrer. Disons seulement, que dans la pratique, le choix d'un algorithme doit être fait en fonction d'un compromis entre son indice d'efficacité et la place mémoire qu'il demande. Voici d'autres remarques importantes :

(i) Pour une itération linéaire, avec $\rho(A) = 0$ et pour toute autre méthode directe, r et c ne sont pas définis. On pourrait alors convenir une valeur de E_g appropriée; $E_g = 1$, par exemple.

(ii) La notion de vitesse de convergence peut donner des résultats différents de ceux obtenus avec l'indice d'efficacité, même si le "travail" ne change pas d'une suite à l'autre.

Prenons par exemple les suites $u_n = \lambda_1^{b^n}$ et $v_n = \lambda_2^{b^n}$ avec $0 < \lambda_1 < \lambda_2 < 1$ et $b > 1$. Elles sont d'ordre b et leur constante asymptotique d'erreur vaut 1 . Le "travail" par itération est le même pour les deux suites et on a :

$$E_g\{u_n\} = E_g\{v_n\} = b$$

$$\lim_{n\to\infty} \frac{u_n}{v_n} \tag{5.33}$$

On voit, que $\{u_n\}$ converge "plus vite" que $\{v_n\}$, alors que leur indice d'efficacité est le même. Celà vient du fait que dans les deux cas on multiplie par b le nombre de chiffres exactes par itération, mais que, au départ $(n = 0)$ on a $-\log_{10}(\lambda_1)$ chiffres décimaux pour la suite $\{u_n\}$ et $-\log_{10}(\lambda_2)$ chiffres décimaux pour la suite $\{v_n\}$.

(iii) Nous avons déjà remarqué qu'il existe des variantes de l' -algorithme qui nécessitent moins d'opérations que les règles usuelles. C'est aussi le cas pour d'autres algorithmes. Il se pose alors le problème de la complexité du calcul (voir [29]) , c'est-à-dire, de déterminer l'algorithme qui nécessite un nombre minimum d'opérations pour effectuer un calcul donné. (Pour passer de $\varepsilon_k^{(n)}$ à $\varepsilon_k^{(n+1)}$, par exemple). Celà veut dire, qu'il existe une relation très importante entre l'efficacité d'un processus itératif et la complexité des calculs intermédiaires qu'il entraine.

REFERENCES

[1] P. WYNN
Acceleration techniques for iterated vector and matrix problems.
Math. of Comp. (16) (1962) 301-322.

[2] C. BREZINSKI
Accélération de la convergence de suites dans un espace de Banach.
C.R. Acad. Sc. Paris, 278 A (1974) 351-354.

[3] P. WYNN
U pour une conjecture concerning a method for solving lineair equations and certains others matters.
MRC Technical summary report 626, 1966.

[4] E. GEKELER
On the solution of systems of equations by the ε-algorithm de Wynn.
Math. of Comp. 26 (1972), 427-436.

[5] C. BREZINSKI
Accélération de la convergence en Analyse Numérique.
Lectures Notes in Mathematics, n° 584, Springer-Verlag (1977).

[6] C. ESPINOZA
Applications de l'ε-algorithme à des suites non scalaires et comparaison de quelques résultats numériques obtenus avec les ε , ρ et θ-algorithmes .
Rapport D.E.A., Université de Lille (1974).

[7] R. ALT

Méthodes A-stables pour l'intégration des systèmes différentiels mal conditionnés.
Thèse Troisième Cycle, Paris (1971).

[8] C. BREZINSKI, A.C. RIEU

The solution of systems of equations using the ε-algorithm and an application to boundary value problems.
Maths. of Comp. V.28 (1974), 731-741.

[9] C. BREZINSKI

Computation of the eigenelements of a matrix by the ε-algorithm.
Linear Algebra, 11 (1975), 7-20.

[10] C. BREZINSKI

Généralisation de la transformations de Shanks, de la table de Padé et de l'ε-algorithme.
Calcolo, 12 (1975), 317-360.

[11] C. LEMARECHAL

Une méthode de résolution de certains systèmes non linéaires bien posés.
C.R. Acad. Sc. Paris, 272 A, (1971), 605-607.

[12] N. GASTINEL

Analyse Numérique Linéaire.
Hermann, Paris (1966).

[13] R. BRENT

The computational complexity of iterative methods for systems of non linear equations.
Proc. Symposium on the complexity of computation 61-72
Miller and Thatcher Ed. Plemun Press, New-York (1972).

[14] A. OSTROWSKI
Solution of equations and systems of equations.
Academic Press, New-York (1960).

[15] R. VOIGT
Orders of convergence for iterative processes.
S.I.A.M. J. Num. Anal., 8, (1971) 222-243.

[16] R. VARGA
Matrix iterative analysis.
Prentice Hall, Englewood Cliffs, New Jersey (1962).

[17] C. BREZINSKI
Numerical stability of a quadratic method for solving systems of non linear equations.
Computing, 14, (1975), 205-211

[18] D. PETIT
Etude de procédés d'accélération de la convergence .
Thèse troisième cycle, Université de Lille (1977) (à paraître).

[19] F. CORDELLIER
Caractérisation des suites que la première étape du θ-algorithme transforme en suites constantes.
C.R. Acad. Sc. Paris, 284 A (1977), 389-392.

[20] C. BREZINSKI
Padé approximants and orthogonal polynomials.
Conference on rational approximation, Tampa (1976).

[21] GERMAIN-BONNE B.
La méthode de Mann. Résultat sur une itération convexe.
Séminaire d'Analyse Numérique, Université de Lille (1973).

[22] F. ROBERT
Matrices nonnégatives et normes vectorielles.
Cours D.E.A., Université de Grenoble 1 (1973).

[24] N. GASTINEL
Itérations - Méthodes discrètes.
Cours C3, Maîtrise d'Informatique, Université Grenoble 1 (1971).

[25] Z. MAHJOUB
Expérimentation de stratégies itératives chaotiques sur des problèmes de point fixe à grand nombre de variables.
Thèse de Docteur Ingénieur, Université de Grenoble 1 (1977) (à paraître).

[26] S. RACHIDI
Méthode des sécantes pour la résolution d'équations algébriques non linéaires.
Thèse troisième cycle, Université de Grenoble 1 (1976).

[27] J. ORTEGA and W. RHEINBOLDT
Iterative solution of nonlinear equations un several variables.
Academic Press, New-York (1970).

[28] C. BREZINSKI
Computation of Padé approximations and continued fractions.
Journal of computational and applied mathematics, volume 2, n° 2 (1976), 113-123.

[29] J.C. LAFON
Complexité de l'évaluation de plusieurs formes bilinéaires et des principaux calculs matriciels.
Thèse, Université de Grenoble 1 (1976).

DEUXIEME PARTIE

ACCELERATION DE LA CONVERGENCE DES METHODES DE PROJECTION

INTRODUCTION

Dans cette partie nous abordons le problème de l'accélération de la convergence sous un angle différent de celui de la première partie ; au lieu de transformer la suite générée par un certain algorithme en une autre qui converge "plus vite", nous allons essayer de modifier les algorithmes eux-mêmes afin qu'ils génèrent des suites qui convergent "plus vite" que celles obtenues avec l'algorithme originale.

Les méthodes envisagées sont celles de projection pour la résolution itératives des systèmes d'équations linéaires. On les étudie dans le contexte général défini par Householder et Bauer (voir [1]) et l'on propose une méthode pour transformer certains de ces algorithmes.

Dans le premier chapitre nous définissons les méthodes de projection et nous rappelons brièvement chaque cas particulier. On fixe ensuite certains paramètres, afin que chaque méthode soit identifiée par la façon de déterminer, à chaque itération n, la variété linéaire dans laquelle on projète x_n. Nous donnons un critère général pour déterminer cette variété à partir de celle définie par une méthode donnée. On démontre que si celle-ci est convergente alors sa transformée converge au moins aussi vite qu'elle. D'autre part, on met en évidence que la transformée de la méthode de la plus profonde descente n'est autre que la méthode du gradient conjugué.

Le second chapitre est consacré aux applications de cette transformation. On considère séparément la cas A symétrique définie positive. On développe quelques variantes intéressantes et on donne des résultats numériques qui illustrent l'intérêt de la transformation proposée. Une démarche similaire est faite pour les méthodes de résolution des systèmes où la matrice A n'a pas de caractéristiques particulières.

NOTATIONS ET DEFINITIONS

(i) Dans cette partie, on considère R^p comme un espace vectoriel (sur R), muni du produit scalaire

$$(x,y) = \sum_{i=1}^{p} x_i\, y_i$$

et on utilisera plusieurs types de normes de R^p :

Soit G une matrice (p,p) , symétrique et définie positive nous désignons par $\|.\|_G$ la norme elliptique définie par :

$$\|x\|_G = [(x,Gx)]^{1/2}$$

D'autre part, nous appelons φ_k la norme de Holder

$$\varphi_k(x) = [\sum_{i=1}^{p} |x_i|]^{1/k} \quad , \quad k=1,\ldots$$

$$\varphi_\infty(x) = \max_{1\leq i\leq p} |x_i|$$

On notera $\|.\|$ le cas particulier $G = I(k=2)$:

$\|x\| = \|x\|_I = \varphi_2(x)$

(ii) Dans la résolution itérative d'un système d'équations linéaires

$$Ax = b$$

avec A(p,p) non singulière, nous appelons :

(1) $\omega = A^{-1}b$ la solution

(2) x_k le $k^{ième}$ itéré obtenu à partir d'un x_o donné

(3) $s_k = x_k - \omega$ l'erreur à l'itération k

(4) $r_k = A\, s_k = A\, x_k - b$ le résidu à l'itération k

(iii) Les méthodes que nous considérons dans cette partie sont aussi appelées méthodes de descente car elles consistent à minimiser une certaine fonctionnelle $g(x)$ qui prend son minimum à la solution $x = \omega$ du système à résoudre et on a $g(x_{n+1}) \leq g(x_n)$, $n=0,1,\ldots$. En général il est difficile d'obtenir des résultats asymptotiques sur la diminution de g à chaque itération, mais on peut démontrer l'existence d'une borne supérieure du rapport $g(x_{n+1}) / g(x_n)$, $n=0,1,\ldots$. Nous utiliserons donc le critère d'accélération de la convergence suivant :

Soit l'algorithme F qu'on se propose d'accélérer et tel que pour tout x_o donné on ait :

$$\frac{g(x_{n+1})}{g(x_n)} \leq \rho < 1 \qquad , \qquad n=0,1,\ldots$$

Soit l'algorithme T obtenu par une transformation de F . Pour une approximation initiale quelconque y_o , soit $\{y_n\}$ la suite générée par l'algorithme T . Nous dirons que T accélère la convergence de F s'il existe ρ' tel que

$$\frac{g(y_{n+1})}{g(y_n)} \leq \rho' < \rho \qquad , \qquad n=0,1,\ldots$$

Nous dirons qu'une méthode de transformation d'algorithmes définie dans un ensemble $\mathcal{E}$ est une méthode d'accélération de la convergence, si à tout $F \in \mathcal{E}$, elle associe un algorithme T tel que $\rho' \leq \rho$ et s'il existe un sous-ensemble non vide $\mathcal{P} \subset \mathcal{E}$, tel que pour $F \in \mathcal{P}$, sa transformée T est telle que $\rho' < \rho$.

CHAPITRE I

METHODES DE PROJECTION GENERALISEES

I.1. DEFINITION ET PROPRIETES

Soit A une matrice (p,p) non singulière et b un vecteur non nul de R^p. Nous envisageons l'utilisation des méthodes itératives de projection pour résoudre le système d'équations linéaires

$$A x = b \tag{1.1}$$

Soit G une matrice symétrique et définie positive.
Une méthode de projection pour la résolution de (1.1) consiste à définir, à chaque itération n, une matrice H_n de type (p,q), $1 \leq q \leq p$ et à calculer x_{n+1} tel qu'il appartienne à l'intersection des deux variétés linéaires

$$\left\{ \begin{array}{l} V_n = \{x \in R^p \mid x = x_n + H_n \wedge \ , \ \wedge \in R^q\} \\ W_n = \{x \in R^p \mid H_n^T G(x-\omega) = 0 \in R^q\} \end{array} \right. \tag{1.2}$$

c'est-à-dire, qu'ayant déterminé H_n, on calcule $\wedge_n$ en résolvant un système d'équations linéaires

$$(H_n^T G H_n)\wedge = - H_n^T G s_n \in R^q \tag{1.3}$$

Nous supposerons que les matrices (q,q) $H_n^T G H_n$ sont inversibles pour que le système (1.3) ait une solution unique. Ces méthodes ont les deux propriétés suivantes :

$$(1.4) \quad \begin{cases} (p_1) : \|s_{n+1}\|_G = \min\limits_{x \in V_n} \|x-\omega\|_G \\ (p_2) : H_n^T G s_{n+1} = 0 \in R^q \end{cases}$$

(p_2) dérive du fait que $x_{n+1} \in W_n$. Pour (p_1) on a :

$$(1.5) \quad \|s_n + H_n \wedge\|_G^2 = \|s_n\|_G^2 + 2 \wedge^T H_n^T G s_n + \|H_n \wedge\|_G^2$$

Le minimum est alors atteint pour $\wedge_n = - (H_n^T G H_n)^{-1} H_n^T G s_n$.
On obtient :

$$(1.6) \quad \begin{cases} \|s_{n+1}\|_G^2 = \|s_n\|_G^2 - \|H_n \wedge_n\|_G^2 \\ \|s_{n+1}\|_G^2 \leq \|s_n\|_G^2 \end{cases}$$

Ce sont donc des méthodes de descente pour minimiser la fonctionnelle quadratique

$$(1.7) \quad g(x) = \|x-\omega\|_G^2$$

x_{n+1} est la projection orthogonale de x_n (par rapport à G), sur la variété linéaire W_n .

Mais, pour mettre en oeuvre la méthode il faut pouvoir calculer $\wedge_n$. Or, d'après (1.3) $\wedge_n$ dépend de s_n qui n'est pas connu. C'est le choix de G et des H_n qui peut permettre de surmonter cet obstacle. Entre autres, trois possibilités peuvent être envisagées :

1. $G = A$ si A est symétrique et définie positive.

2. $G = I$ et $H_n = A^T Y_n$ (Y_n donné) pour A quelconque.

3. $G = A^T A$, ce qui correspond à la première possibilité, pour résoudre le système aux moindres carrés

(1.8) $A^T A x = A^T b$

Dans ces cas, il existe toujours Y_n connu tel que

(1.9) $H_n^T G s_n = Y_n^T r_n$

Donc, dans tout ce qui suit, nous utiliserons le vecteur $H_n^T G s_n$ que l'on peut calculer.

G étant fixée, on peut choisir q suivant les caractéristiques de A , de la place mémoire dont on dispose, etc.... Dans ces conditions, on peut dire que chaque méthode particulière est définie par la façon dont on détermine les H_n . Ceci nous permet de caractériser la classe de méthodes que nous voulons accélérer. Nous allons résumer les règles qui définissent ces algorithmes.

DEFINITION 1

G et q étant fixés, on désigne par $\mathcal{C}$, la classe des méthodes définies par :

(1.10)
$$\begin{cases} x_o \text{ donné et à l'itération } n : \\ r_n = A x_n - b \\ H_n = H(n, r_n) \in \mathcal{M}_{p,q} \\ \wedge_n = - (H_n^T G H_n)^{-1} (H_n^T G s_n) \\ x_{n+1} = x_n + H_n \wedge_n \end{cases}$$

H étant une fonction de $N \times R^p$ dans $\mathcal{M}_{p,q}$. Elle caractérise l'algorithme. La plupart des algorithmes couramment utilisés appartiennent à cette classe. Voici quelques exemples (voir [2]).

(i) A SYMETRIQUE DEFINIE POSITIVE ; $G = A$, $q = 1$

$$(1.11)\quad \begin{cases} n+1 = i \text{ (modulo } p) \\ H(n,r_n) = e_i \end{cases}$$

pour la méthode de Gauss-Seidel

$$(1.12)\quad \begin{cases} |r_n^{(i)}| = \varphi_\infty(r_n) \qquad\qquad (b) \\ H(n,r_n) = e_i \quad \text{(le plus grand i satisfaisant (b))} \end{cases}$$

pour la méthode de Southwell.

$$(1.13)\quad H(n,r_n) = r_n$$

pour la méthode de la plus profonde descente.

(ii) A SYMETRIQUE DEFINIE POSITIVE ; $G = A$, $q > 1$

$$(1.14)\quad \begin{cases} t = t(n) \quad , \quad t : N \to \{0,1,\ldots,p-q\} \\ j^{\text{ème}} \text{ colonne de } H_n = e_{t+j} \quad , \quad j=1,\ldots,q \end{cases}$$

Méthode de Gauss-Seidel par blocs ; cas particulier, parce qu'en général $q = q(n)$ (voir [3],[7]).

$$(1.15)\quad j^{\text{ème}} \text{ colonne de } H_n = A^{(j-1)}r_n \quad , \quad j=1,\ldots,q$$

Pour la variante à q pas de la méthode de la plus profonde descente (voir [4]).

(iii) A QUELCONQUE ; $G = I$, $q = 1$

$$\begin{matrix} n+1 = i \ \text{(modulo p)} \\ H(n,r_n) = A^T e_i \end{matrix} \qquad (1.16)$$

pour la méthode de Kaczmarz
Soit Ψ : $R^p \to R$ une norme de R^p . Supposons que pour tout $x \in R^p$ il existe z_x tel que $z_x^T . x = \Psi(x)$.

$$H_n(n,r_n) = A^T z_{r_n} \qquad (1.17)$$

définit la méthode basée sur une décomposition d'une norme.

(iv) A QUELCONQUE ; $G = I$, $q > 1$

Supposons maintenant qu'il existe $Z_x \in \mathcal{M}_{p,q}$ tel que

$$x^T Z_x (Z_x A A^T Z_x)^{-1} Z_x^T x \ \text{trace} \ (Z_x^T A A^T Z_x) = \Psi^2(x) \qquad (1.18)$$

alors

$$H_n(n,r_n) = Z_{r_n} \qquad (1.19)$$

définit la méthode basée sur une sur-décomposition d'une norme (voir [5]).
Soit Ψ une norme vectorielle de taille q sur R^p . Supposons que pour tout $x \in R^p$, il existe $Z_x \in \mathcal{M}_{p,q}$ tel que $Z_x^T x = \Psi(x)$. Alors :

$$H(n,r_n) = A^T Z_{r_n} \qquad (1.20)$$

définit la méthode basée sur une décomposition d'une norme vectorielle de taille q sur R^p (voir [6]).

Il est évident que nous n'avons pas cité toutes les méthodes mais seulement les plus connues et (par souci de clarté) nous les avons

défini, telles qu'on les trouve dans la bibliographie. Mais on aurait pu les exposer dans le contexte utilisé ici et rapprocher ainsi, par exemple, la méthode de Gauss-Seidel de celle de Kaczmarz, la méthode de Southwell de celle associée à une décomposition de φ_∞, la plus profonde descente de la méthode associée à une décomposition de φ_2 (méthode du gradient) etc...

I.2 UNE METHODE D'ACCELERATION

Soit une méthode donnée, dans la classe $\mathcal{C}$. Nous proposons d'accélérer sa convergence en utilisant un critère supplémentaire pour déterminer les matrices H_n. La méthode ainsi obtenue ne sera plus de classe $\mathcal{C}$ parce que les H_n ne dépendront pas seulement de n et r_n mais aussi de H_{n-1}. Pour une méthode de classe $\mathcal{C}$, définie par une certaine fonction $H = H(n,r_n)$, nous proposons d'obtenir H_n tel que :

$$(1.21)\quad \begin{cases} H_n = H(n,r_n) + H_{n-1}\,\Gamma_n \quad , \quad \Gamma_n \in \mathcal{M}_{qq} \\ H_{n-1}^T\, G\, H_n = 0 \in \mathcal{M}_{qq} \end{cases}$$

Pour calculer x_n on a dû calculer $\Lambda_{n-1} = -(H_{n-1}^T\, G\, H_{n-1})^{-1}\, H_{n-1}^T\, G\, s_{n-1}$; en conséquence $H_{n-1}^T\, G\, H_{n-1}$ est inversible et Γ_n est définie par :

$$(1.22)\quad \Gamma_n = -(H_{n-1}^T\, G\, H_{n-1})^{-1}\, H_{n-1}^T\, G\, Z_n$$

avec $Z_n = H(n,r_n)$, notation que nous utiliserons dans tout ce qui suit. On a alors :

DEFINITION 2

Soit une méthode c de la classe $\mathcal{C}$, utilisant les directions $Z_n = H(n,r_n)$. Nous appelons méthode c accélérée l'itération suivante :

$$(1.23)\quad \begin{cases} x_o \text{ donné}, \quad H_o = H(0,r_o) \text{ et à l'itération } n : \\ \Lambda_n = -(H_n^T G H_n)^{-1} H_n^T G s_n \\ x_{n+1} = x_n + H_n \Lambda_n \\ \Gamma_{n+1} = -(H_n^T G H_n)^{-1} H_n^T G Z_{n+1} \\ H_{n+1} = Z_{n+1} + H_n \Gamma_{n+1} \end{cases}$$

En général, on peut dire que l'augmentation du coût de calcul à chaque itération est donnée par :

(i) Inversion de $H_n G H_n \in \mathcal{M}_{qq}$, au lieu de résoudre le système d'équations linéaires (1.3) (même matrice)

(ii) Calcul de la matrice $H_n^T G Z_{n+1} \in \mathcal{M}_{q,q}$.

(iii) Deux produits matriciels avec des matrices (q,q)

(iv) Addition de deux matrices (q,q)

L'encombrement mémoire est multiplié par deux (pour q fixé) parce qu'il faut stocker H_n , nécessaire à l'itération n+1.
Le lemme suivant peut faciliter certains calculs et nous conduit à un résultat sur le taux de convergence de la méthode.

LEMME 1

$$(1.24)\quad \begin{cases} \Lambda_n = -(H_n^T G H_n)^{-1} Z_n^T G s_n \\ H_n^T G H_n = Z_n^T G Z_n - \Gamma_n^T (H_{n-1} G H_{n-1}) \Gamma_n \end{cases}$$

DEMONSTRATION

$$(1.25)\quad H_n^T G s_n = Z_n^T G s_n + \Gamma_n^T H_{n-1}^T G s_n$$

mais d'après la propriété (p_2) de (1.4) des méthodes de projection on a $H_{n-1}^T G s_n = 0$, $\forall\, n \geq 1$. On obtient la première identité, en utilisant

la définition (1.23) de Λ_n . Pour la deuxième identité on a :

$$(1.26) \quad H_n^T G H_n = H_n^T G Z_n + H_n^T G H_{n-1} \Gamma_n$$

or , $H_n^T G H_{n-1} = (H_{n-1}^T G H_n)^T = 0$, par construction du procédé. Alors :

$$(1.27) \quad H_n^T G H_n = Z_n^T G Z_n + \Gamma_n^T H_{n-1}^T G Z_n$$

et (1.25) suit d'après la définition de Γ_n , qui entraîne $H_{n-1}^T G Z_n = - (H_{n-1}^T G H_{n-1})\Gamma_n$. -

THEOREME 1

Si la méthode originale est convergente alors la méthode accélérée converge et ceci au moins aussi vite que la méthode originale.

DEMONSTRATION

$$(1.28) \begin{cases} \dfrac{\|s_{n+1}\|_G^2}{\|s_n\|_G^2} = 1 - \dfrac{s_n^T G H_n (H_n^T G H_n)^{-1} H_n^T G s_n}{\|s_n\|_G^2} \\ \dfrac{\|s_{n+1}\|_G^2}{\|s_n\|_G^2} \leq 1 - \dfrac{\|H^T G s_n\|^2}{\|s_n\|_G^2 \, \rho(H_n^T G H_n)} \\ \dfrac{\|s_{n+1}\|_G}{\|s_n\|_G} \leq \tau_n = \sqrt{1 - \dfrac{\|H_n^T G s_n\|^2}{\|s_n\|_G^2 \, \rho(H_n^T G H_n)}} \end{cases}$$

Pour que la méthode converge vers ω , il est suffisant qu'il existe $\tau < 1$ tel que $\tau_n \leq \tau$, $\forall$ n . D'autre part , τ_n donne une mesure de la vitesse de convergence. Pour la méthode à accélérer (celle qui utilise les directions Z_n) , on a :

$$(1.29)\qquad \tau_n = \sqrt{1 - \frac{\|Z_n^T G s_n\|^2}{\|s_n\|_G^2 \rho(Z_n^T G Z_n)}}$$

Pour la méthode accélérée, on a, en utilisant le fait que $H_n^T G s_n = Z_n^T G s_n$:

$$(1.30)\qquad \tau_n' = \sqrt{1 - \frac{\|Z_n^T G s_n\|^2}{\|s_n\|_G^2 \rho(H_n^T G H_n)}}$$

Démontrer $\tau_n' \leq \tau_n$ est alors équivalent à démontrer que $\rho(H_n^T G H_n) \leq \rho(Z_n^T G Z_n)$. Or nous avons, en utilisant le lemme 1 :

$$(1.31)\qquad \begin{cases} H_n^T G H_n = Z_n^T G Z_n - \Gamma_n^T (H_{n-1}^T G H_{n-1})\Gamma_n \\ \underset{\|x\|=1}{\text{Sup}}\ x^T(H_n^T G H_n)x \leq \underset{\|x\|=1}{\text{Sup}}\ x^T(Z_n^T G Z_n)x - \underset{\|x\|=1}{\text{Inf}}\ x^T(\ {}^T H_{n-1}^T G H_{n-1}\Gamma_n)x \end{cases}$$

Mais, G étant symétrique et définie positive, la matrice $\Gamma_n^T H_{n-1}^T G H_{n-1} \Gamma_n \in \mathcal{M}_{q,q}$ est semi-définie positive et en conséquence

$$(1.32)\qquad \rho(H_n^T G H_n) \leq \rho(Z_n^T G Z_n) - \xi_k$$

avec $\xi_k \geq 0$ et ceci achève la démonstration du théorème. Dans le prochain paragraphe nous abordons le problème de la convergence de la méthode.

I.3 CONVERGENCE DE LA METHODE

Nous voulons maintenant donner des conditions pour que la méthode soit convergente. Pour cela nous donnons d'abord une définition qui est aussi une hypothèse suffisante pour démontrer la convergence de la méthode originale et donc celle de la méthode accélérée.

DEFINITION 3

Nous dirons que les matrices H_n sont liées au gradient, s'il existe une norme φ sur R^p telle que :

$$(1.32) \quad \|H_n^T G s_n\| \geq \varphi(r_n) \quad , \quad \forall n$$

Si, en plus il existe une constante $M > 0$ telle que

$$(1.33) \quad \rho(H_n^T G H_n) \leq M \quad , \quad \forall n$$

alors nous dirons que les matrices H_n sont bornées.

THEOREME 2

Supposons que les matrices $Z_n = H(n,r_n)$ définissant les méthodes (1.10) et (1.23) soient liées au gradient et bornées et que les matrices $H_n^T G H_n$ soient inversibles, alors ces méthodes convergent vers pour toute approximation initiale $x_o \in R^p$. Leurs taux respectifs de convergence sont liés par la relation

$$(1.34) \quad \tau' \leq \tau = \sqrt{1 - \frac{1}{M.S^2_{\varphi\Psi}(A^{-1})}} < 1$$

où Ψ est la norme $\|.\|_G$, φ la norme de R^p de (1.32), M la borne supérieure (1.33) de $\rho(H_n^T G H_n)$ et

$$(1.35) \quad S_{\varphi\Psi}(A^{-1}) = \max_{x \in R^p} \frac{\Psi(A^{-1} x)}{\varphi(x)} > 0$$

DEMONSTRATION

Au théorème 1 nous avons démontré que $\tau'_n \leq \tau_n$, $\forall n$. Il nous reste seulement à démontrer que $\tau_n \leq \tau$. Or, d'après la définition 3 on a :

$$(1.36)\qquad \frac{\|H_n^T G\, s_n\|^2}{\|s_n\|_G^2\, \rho(H_n^T G H_n)} \geq \frac{\varphi^2(r_n)}{M.\Psi^2(A^{-1} r_n)}$$

et en utilisant (1.29) et (1.35) :

$$(1.37)\qquad \tau_n \leq \tau \leq \sqrt{1 - \frac{1}{M.S^2_{\varphi\Psi}(A^{-1})}}$$

Ce théorème montre, en particulier, la convergence des méthodes obtenues par une décomposition d'une norme.

COROLLAIRE 2.1. (voir [6])

Si les matrices $Z_n = H(n,r_n)$ sont bornées et telles que :

$$(1.38)\qquad Z_n^T G\, s_n = \Psi(r_n)$$

où Ψ est une norme de taille q sur R^p, alors les méthodes (1.10) et (1.23) convergent vers ω, pour tout $x_o \in R^p$.

DEMONSTRATION

Elle dérive tout de suite du théorème 2, parce que :

$$(1.39)\qquad \|Z_n^T G\, s_n\| = \varphi(r_n)$$

où $\varphi(x) = \|\Psi(x)\|$ est une norme sur R^p.

COROLLAIRE 2.2. (voir [5])

Si les $Z_n = H(n,r_n)$ sont bornées et définies par une sur-décomposition d'une norme, c'est-à-dire, si

$$(1.40)\qquad s_n^T G\, Z_n (Z_n^T G\, Z_n)^{-1} Z_n^T G\, s_n \text{ trace } (Z_n^T G\, Z_n) = \Psi^2(r_n)$$

où Ψ est une norme quelconque de R^p, alors les méthodes (1.10) et (1.23) convergent vers ω, pour toute approximation initiale $x_o \in R^p$.

DEMONSTRATION

C'est encore une conséquence du théorème 2, parce que :

$$(1.41)\quad \begin{cases} \Psi^2(r_n) \leq \dfrac{\|Z_n^T G\, s_n\|^2 q\, \rho(Z_n^T G\, Z_n)}{\rho(Z_n^T G\, Z_n)} \\ \\ \|Z_n^T G\, s_n\| \geq \dfrac{1}{\sqrt{q}}\ \Psi(r_n) = \varphi(r_n) \end{cases}$$

Au lieu d'utiliser des directions liées au gradient, on peut être amené à utiliser un ensemble de k matrices $Y_i(p,q)$, $i=1,\ldots,k$, dont l'union de ses colonnes engendre R^p. La méthode (1.10) consiste, dans ce cas, à faire des projections, de façon cyclique, sur les variétés linéaires :

$$(1.42)\quad L_i \ :\ Y_i^T G\,(x-\omega) = 0 \in R^q \quad , \quad i=1,\ldots,k$$

En général, on ne peut pas déterminer un taux de convergence pour cette sorte de directions, comme on l'a fait pour celles liées au gradient, mais il est toujours possible de démontrer sa convergence. Nous suivrons pour celà une démarche similaire à celle utilisée en [2] pour démontrer la convergence de la méthode de Kacgmarz ; on utilise le fait que le seul point commun à toutes les L_i est ω et on met en évidence que $\{x_n\}$ est l'union de k sous-suites convergentes vers une même limite, commune à toutes les L_i.

THEOREME 3

Soient Y_i , i=1,2,...,k un ensemble de matrices (p,q) dont l' union des colonnes engendre R^p . Si les matrices $Z_n = H(n,r_n)$ sont telles que :

$$(1.43)\quad \begin{cases} n+1 = i \pmod p \\ H(n,r_n) = Y_i \end{cases}$$

alors les méthodes (1.10) et (1.23) convergent vers ω , pour toute approximation initiale $x_o \in R^p$.

DEMONSTRATION

Pour la méthode (1.10), on a par définition

$$(1.44)\quad x_{n+1} \in L_i \quad ; \quad n+1 = i \pmod p$$

Nous montrerons d'abord que ceci est aussi vrai pour la méthode (1.23). La suite de la démonstration est alors la même pour les deux méthodes. On a :

$$(1.45)\quad \begin{cases} Z_n^T G\, s_{n+1} = H_n^T G\, s_{n+1} - \Gamma_n^T H_{n-1}^T G\, s_{n+1} \\ Z_n^T G\, s_{n+1} = H_n^T G\, s_{n+1} - \Gamma_n^T G_{n-1}^T G\, s_n - \Gamma_n^T H_{n-1}^T G\, H_n \wedge_n \end{cases}$$

Les deux premiers termes de droite son nuls à cause de (1.4), le troisième à cause de (1.21). Donc :

$$(1.46)\quad Y_i^T G\,(x_{n+1}-\omega) = 0$$

et (1.44) est satisfait par la méthode (1.23). Pour les deux méthodes, étant donné que $\|s_\ell\|_G$ est non croissante il existe toujours, pour ℓ donné, un t , tel que :

$$(1.47)\quad \begin{cases} k\,t \le \ell \le k(t+1) \\ \|s_{k(t+1)}\|_G \le \|s_\ell\|_G \le \|s_{kt}\|_G \end{cases}$$

Considérons donc la suite $x_k, x_{2k}, x_{3k}, \ldots$ qui appartient à L_k. Elle satisfait :

(i) les nombres $\|x_{kt}-\omega\|_G$ ont une limite $z \ge 0$ si $t \to \infty$.

(ii) z est aussi la limite des nombres $\|x_\ell-\omega\|_G$ si $\ell \to \infty$

Il existe donc une sous-suite x_{kt_i} $(i \to \infty)$ convergente vers un point y_k et telle que :

$$(1.48)\quad \|y_k-\omega\|_G = z$$

Mais, alors, les points x_{kt_i+1} qui sont les projections des x_{kt_i} et qui appartiennent à L_1 ont aussi une limite y_1 et on aura :

$$(1.49)\quad \begin{cases} \|y_1-\omega\|_G = \lim\limits_{i\to\infty} \|x_{kt_i+1}-\omega\|_G = z \\ \|y_1-\omega\|_G \le \|y_k-\omega\|_G = z \end{cases}$$

l'égalité n'ayant lieu que si y_1 est confondu avec y_k et en conséquence $y_k \in L_1$. On peut faire le même raisonnement sur les suites $\{x_{kt_i+j}\}$, $j=2,\ldots,k-1$. On prouvera successivement que leurs limites $y_2, y_3, \ldots, y_{k-1}$ sont confondues avec y_1 et y_k. Mais étant donné que le seul point commun aux L_i, $i=1,\ldots,k$ est ω, cette limite ne peut pas être autre que ω et de (1.47) on obtient :

$$(1.50)\quad \lim_{n\to\infty} \|x_\ell-\omega\| = 0$$

Ce paragraphe nous a servi pour donner une approche générale sur la convergence des méthodes de projection et à montrer que la méthode accélérée converge sous les mêmes hypothèses que la méthode originale au moins avec le

même taux de convergence. Dans le prochaine paragraphe nous étudions un cas particulier : la transformée des méthodes associées à la décomposition de φ_2 .

I.4. TRANSFORMEE DE LA METHODE ASSOCIEE A LA DECOMPOSITION DE φ_2

Voyons maintenant ce qu'on obtient quand on applique la transformation proposée à la méthode de la plus profonde descente où à celle du gradient. Dans ce cas :

$$(1.51)\quad \begin{cases} q = 1 \\ H(n,r_n) = C^T r_n \quad , \quad \forall\, n \end{cases}$$

avec $G = A$, $C = I$ pour la méthode de la plus profonde descente et $G = I$, $C = A$ pour le gradient. En changeant de notations (pour simplifier) et en utilisant le fait que $C\,G = A$, on explicite la méthode (1.23) par :

$$(1.52)\quad \begin{cases} x_o \ \text{donné} \quad , \quad u_o = r_o \\ \lambda_n = -\dfrac{\|r_n\|^2}{\|C^T U_n\|_G^2} \\ x_{n+1} = x_n + \lambda_n C^T U_n \\ \gamma_n = \dfrac{\|r_{n+1}\|^2}{\|r_n\|^2} \\ U_{n+1} = r_{n+1} + \gamma_n U_n \end{cases}$$

On voit tout de suite que ce n'est autre que la méthode du gradient conjuguée (voir [2]). Puisque la méthode du gradient conjugué converge en un nombre fini de pas il est certain qu'elle converge plus vite que celle de la plus profonde descente où du gradient . Nous avons donc une transformation telle que :

(i) La méthode transformée converge dans tous les cas où la méthode originale converge et au moins à la même vitesse.

(ii) Pour un sous-ensemble non vide d'algorithmes de la classe qu'on vise à accélérer, la transformée converge plus vite que la méthode originale.

Nous pouvons donc affirmer que la transformation proposée est une méthode d'accélération de la convergence pour les algorithmes de la classe $\mathcal{C}$.

I.5. MINIMISATION D'UNE FONCTIONNELLE NON QUADRATIQUE

Soit g : $R^p \to R$ deux fois continuellement différentiable dans R^p . Nous cherchons un minimum global de g , c'est-à-dire une solution ω de

$$(1.53) \quad r(x) = \text{grad}\,(g(x)) = 0 \in R^p$$

Pour résoudre ce problème, on envisage une méthode itérative du type

$$(1.54) \quad x_{n+1} = x_n + H_n \wedge_n$$

où $H_n \in \mathcal{M}_{p,q}$ est une matrice de directions et $\wedge_n \in R^q$ est un vecteur contenant le déplacement dans chacune de ces directions $(1 \leq q \leq n)$. Ayant calculé x_n et H_n , la valeur des éléments de $\wedge_n$ sont déterminés, en résolvant le système d'équations non linéaires a q inconnues

$$(1.55) \quad H_n^T . r(x_n + H_n \wedge) = 0 \in R^q$$

dont nous supposons qu'il a une solution. Dans un voisinage de x_n , $r(x_n + H_n \wedge)$ est approché par :

$$(1.56) \quad r(x_n + H_n \wedge) \simeq r(x_n) + A(x_n) H_n \wedge$$

où $A(x_n)$ est l'hessienne de g au point x_n . Une approximation de $\wedge_n$

est alors donnée par

$$(1.57) \quad \wedge_n = - (H_n^T A(x_n) H_n)^{-1} H_n^T r(x_n)$$

Le cas où H_n est formée de q vecteurs de base a été étudié dans un cadre non linéaire (voir [7] par exemple). La matrice $H_n^T A(x_n) H_n$ est alors une sous-matrice de $A(x_n)$ et $H_n^T r(x_n)$ est un sous-vecteur de $r(x_n)$ de dimension q. Mais on pourrait aussi envisager des H_n définis différemment. Par exemple, comme on l'a proposé en [6] dans un cadre linéaire, on décompose $r(x_n)$ en $R = (R_1,\ldots,R_q)^T$ où $R_i \in R^{\lambda_i}$, $\lambda_1+\lambda_2,+\ldots+\lambda_q = p$, et on utilise la norme vectorielle suivante :

$\Psi(R) = (\varphi_\infty(R_1), \varphi_\infty(R_2),\ldots, \varphi_\infty(R_q))^T$. On a pour H_n :

$$(1.58) \quad H_n = (\varepsilon_{i_1},\varepsilon_{i_2},\ldots,\varepsilon_{i_q})$$

où i_j, $j=1,\ldots,q$ désigne le numéro (compris entre $\lambda_1+\lambda_2+\ldots+\lambda_{j-1}+1$ et $\lambda_1+\lambda_2+\ldots+\lambda_j$ de la plus forte composante r_{ij} de R_i. $\varepsilon_{i_j} = \text{signe}\,(r_{ij})\, e_{i_j}$.

Cette méthode à l'avantage que les matrices H_n sont liées au gradient et néanmoins, le calcul de $H_n A(x_n) H_n$ demande à peu près le même travail que pour les méthodes utilisant des H_n formées par des vecteurs de base. Il est a remarquer que l'utilisation de $H_n = I$, $\forall\, n$ $(q=p)$ conduit à la méthode de Newton :

$$(1.59) \quad x_{n+1} = x_n - [A(x_n)]^{-1} r(x_n)$$

Il peut arriver que le calcul de $A(x_n)$ ne soit pas onéreux par rapport à son inversion et qu'on ait plutôt intérêt à utiliser $1 \leq q < n$ afin de réduire la taille du système a résoudre. A ce moment-là, il est important de bien choisir la matrice H_n, pour profiter au maximum du calcul de $A(x_n)$. C'est dans ce but que nous généralisons la méthode précédemment étudié pour les systèmes linéaires : Ayant calculé x_n et une certaine matrice $Z_n = H(n,r_n)$, nous proposons de déterminer H_n telle que :

$$(1.60)\quad \begin{cases} H_n = Z_n + H_{n-1}\,\Gamma_n \quad , \quad \Gamma_n \in \mathcal{M}_{q,q} \\ H_{n-1}^T\, A(x_n)\, H_n = 0 \ \in \mathcal{M}_{q,q} \end{cases}$$

Explicitons, comme illustration le cas $q = 1$, $Z_n = r(x_n)$.
Il existe, en général, $c_n > 0$ tel que :

$$(1.61)\quad g(x_n)-g(x_{n+1}) = c_n \frac{\|r(x_n)\|^2}{H_n^T A(x_n) H_n} + \xi_n$$

H_n est alors déterminé de façon a minimiser $H_n^T A(x_n) H_n$ dans l'ensemble $V_n = \{H \in R^p \ , \ H = r(x_n) + \gamma\, H_{n-1},\ \gamma \in R\}$ on a alors :

$$(1.62)\quad \begin{cases} H_n = r(x_n) + \gamma_n\, H_{n-1} \\ \gamma_n = \dfrac{r(x_n)^T\, A(x_n)\, H_{n-1}}{H_{n-1}^T\, A(x_n)\, H_{n-1}} \end{cases}$$

On peut encore approcher γ_n , en approchant $A(x_n)\, H_{n-1}$:

$$(1.63)\quad \begin{cases} A(x_n)\, H_{n-1} \simeq - \dfrac{r(x_n) - r(x_{n-1})}{\Lambda_n} \\ \gamma_n' = \dfrac{\|r(x_n)\|^2 - (r(x_n), r(x_{n-1}))}{\|r(x_{n-1})\|^2} \end{cases}$$

où $H_{n-1}^T\, r(x_n)$ et $H_{n-2}^T\, r(x_{n-1})$ s'annulent d'après (1.55). Dans le cas (1.62) on retrouve la méthode du gradient conjugué de Daniel (voir [8]), alors que (1.63) conduit, si on néglige le terme $(r(x_n), r(x_{n-1}))$ à la méthode de Fletcher-Rives (voir [8]).
Des expériences numériques et des études théoriques manquent pour cette sorte de méthodes, sauf pour des H_n très particuliers ou $q=1$. Le cas $q > 1$, avec des H_n liées au gradient devrait conduire à des résultats intéressants. Dans le prochain chapître nous étudions quelques applications pour les systèmes linéaires. La troisième partie sera consacrée à une famille de systèmes non linéaires et à une méthode pour sa résolution qui entre dans le contexte présenté ici, avec $q > 1$.

CHAPITRE II

APPLICATIONS DE LA METHODE

Il est certain que lorsque l'encombrement mémoire et la propagation des erreurs d'arrondi le permettent, on a tout intérêt à utiliser des méthodes directes. Celles-ci sont, en général, plus économiques en temps de calcul que les méthodes itératives. Mais s'il faut appliquer une méthode itérative alors on doit prendre en considération les critères suivants :

(i) Choisir q de façon à exploiter au maximum la mémoire disponible et les caractéristiques du système à résoudre.

(ii) Utiliser des directions H_n qui n'augmentent pas beaucoup le coût de chaque itération mais qui, néanmoins, assurent une diminution considérable de la fonctionnelle à minimiser.

(iii) Profiter des avantages des méthodes directes pour inverser les matrices $H_n^T G H_n \in \mathcal{M}_{q,q}$

Dans ce chapitre, nous explicitons quelques méthodes qui visent surtout la deuxième remarque ci-dessus.

II.1. SYSTEMES SYMETRIQUES DEFINIS POSITIFS.

II.1.1. Variante a q-pas de la plus profonde descente

Au premier chapitre nous avons présenté la variante à q-pas de la méthode de la plus profonde descente, comme un exemple d'utilisation de $q > 1$. Puisque pour $q = 1$ on retrouve la méthode du gradient conjugué, on peut penser que l'application de la méthode d'accélération dans le cas $q > 1$ n'est pas dépourvue d'intérêt car on sait (voir [4]) que la méthode originale converge plus vite que la méthode de la plus profonde descente. En fait, elle correspond à effectuer q pas de la métohde du gradient conjugué à chaque itération.

La convergence de la version accélérée découle du fait que les matrices Z_n sont liées au gradient. En effet, soit :

$$(2.1) \qquad Z_n = \frac{1}{\|r_n\|} (r_n, Ar_n, \ldots, A^{q-1} r_n)$$

où $\frac{1}{\|r_n\|}$ est un facteur de normalisation. Alors :

$$(2.2) \quad \begin{cases} Z_n^T r_n = \frac{1}{\|r_n\|} (\|r_n\|^2, \|r_n\|_A^2, \ldots, \|r_n\|_{A^{q-1}}^2)^T \\ Z_n^T A Z_n = \frac{1}{\|r_n\|^2} \begin{bmatrix} r_n^T A r_n & r_n^T A^2 r_n & \ldots & r_n^T A^q r_n \\ r_n^T A^2 r_n & r_n^T A^3 r_n & \ldots & r_n^T A^{q+1} r_n \\ \vdots & & & \\ r_n^T A^q r_n & r_n^T A^{q+1} r_n & \ldots & r_n^T A^{2q-1} r_n \end{bmatrix} \end{cases}$$

A étant symétrique, et définie positive, il existe $\alpha, \beta > 0$ tels que pour tout n et pour $i=0,1,2,\ldots,2q$ on ait :

$$(2.3) \qquad \alpha \|r_n\|^2 \leq r_n^T A^i r_n \leq \beta \|r_n\|^2$$

On a alors :

$$(2.4)\quad \begin{cases} \|Z_n^T r_n\| \geq \sqrt{q}\ \alpha \|r_n\| \\ \rho(Z_n^T A Z_n) \leq \max\limits_{1\leq i\leq 2q-1} \dfrac{r_n^T A^{2q-1} r_n}{\|r_n\|} \\ \rho(Z_n^T A Z_n) \leq \beta \end{cases}$$

Les Z_n sont donc liées au gradient, d'où la convergence de la méthode accélérée. Il est à remarquer que les matrices $Z_n^T A Z_n$ sont des matrices de Hankel ce qui non seulement facilite le calcul de ses éléments (2q-1 produits scalaires) mais aussi leur inversion : le nombre d'opérations arithmétiques est un $O(q^2)$ (voir [9]).
Ces méthodes sont surtout intéressantes dans leur généralisation pour la minimisation d'une fonctionnelle non quadratique, parce que, bien qu'on calcule à chaque itération la première et la deuxième dérivées, on minimise suivant plusieurs directions privilégiées.

II.1.2. Un algorithme nécessitant peu de multiplications par itération

Nous explicitons la méthode accélérée associée à une décomposition de φ_1 , dans le cas A symétrique et définie positive. Cet exemple à l'intérêt de montrer comment la méthode proposée ici transforme un algorithme qui n'est jamais utilisé parce que très lent, en un autre qui peut même concurrencer ceux utilisés couramment. Nous nous limitons au cas $q = 1$. Les règles de l'algorithme accéléré sont :

$$
(2.5)\quad \left\{
\begin{array}{l}
x_o \text{ donné} \qquad u_{-1} = 0 \qquad \alpha_o = 0 \\
\alpha_n = - \dfrac{z_n^T A U_{n-1}}{U_{n-1}^T A U_{n-1}} \quad \text{avec} \quad \varphi_1(r_n) = z_n^T . r_n \\
U_n = z_n + \alpha_n U_{n-1} \\
A U_n = A z_n + \alpha_n A U_{n-1} \\
x_{n+1} = x_n + \lambda_n U_n \quad \text{avec} \quad \lambda_n = - \dfrac{\varphi_1(r_n)}{U_n^T A U_n} \\
r_{n+1} = r_n + \lambda_n A U_n
\end{array}
\right.
$$

Coût de chaque itération = (3p+3) multiplications + (p'+7p) additions + 3p tests + 3 divisions. p' est le nombre d'éléments non nuls de A . L'économie de calcul vient du fait que les composantes de z_n sont ± 1 , ce qui simplifie le calcul de $A z_n$.

Afin de mesurer la vitesse de convergence de cette méthode par rapport à une méthode bien connue, sur un problème modèle, nous l'avons comparé avec la méthode de Gauss-Seidel, A étant la matrice du potentiel (100,100). La valeur de ω (solution) a été donnée par avance : $\omega_i = 1$, $i=1,\ldots,100$. Le tableau 2.1 correspond à $(x_o)_i = 0$, $i=1,\ldots,100$. Pour le tableau 2.2. on a utilisé une approximation initiale plus proche de la solution : $(x_o)_i = 0{,}9999$, $i=1,\ldots,100$. La comparaison est faite sur les valeurs de $\varphi_1(x_n - \omega)$.

Tableau 2.1.

n	Méthode Gauss-Seidel $\varphi_1(x_n-\omega)$		Méthode (2.5) $\varphi_1(x_n-\omega)$	
0	.1	D 03	.1	D 03
1	.84055	D 02	.10666	D 03
4	.59196	D 02	.85478	D 02
20	.15409	D 02	.15478	D 02
40	.29783	D 01	.19207	D 01
80	.43123	D-01	.28408	D-01
94	.10992	D-01	.62654	D-02
100	.67730	D-02	.26547	D-02

Tableau 2.2.

n	Méthode Gauss-Seidel $\varphi_1(x_n-\omega)$		Méthode (2.5) $\varphi_1(x_n-\omega)$	
0	.1	D-01	.1	D-01
1	.84069	D-02	.10668	D-01
4	.59206	D-02	.85492	D-02
20	.15411	D-02	.15478	D-02
40	.29788	D-03	.19200	D-03
79	.11850	D-04	.20325	D-05
90	.47714	D-05	.81911	D-06
100	.20868	D-05	.26024	D-06

Coût d'une itération Gauss-Seidel = p' multiplications + (p'+p) additions + p division.

Nous n'avons pas comparé par rapport à la méthode originale qui converge beaucoup plus lentement que celle de Gauss-Seidel. Il est à noter, d'ailleurs, l'augmentation de $\varphi(x_n - \omega)$ lors de la première itération, qui coïncide avec une itération de la méthode originale. D'autres remarques sont :

(i) Dans le tableau 2.1. la méthode de Gauss-Seidel utilise 99 itérations pour passer de 0.84055 D 02 à 0.67730 D-02 tandis que la méthode (2.5) utilise 90 itérations seulement pour passer de 0.85478 D 02 à 0.62654 D-02.

(ii) Dans le tableau 2.2. on retrouve le même phénomène mais encore plus accentué. La méthode de Gauss-Seidel utilise 99 itérations pour gagner autant que ce qu'on gagne avec 75 itérations du type (2.5).

(iii) La méthode (2.5) est particulièrement intéressante quand p' est assez grand parce que le nombre de multiplications à effectuer ne dépend pas de p' mais de p .

II.2. SYSTEMES QUELCONQUES

II.2.1. Différences par rapport au cas A symetrique définie positive

Si on est conduit à utiliser une méthode itérative pour résoudre (1.1) avec A n'ayant pas des bonnes conditions (symétrique définie positive ou diagonale dominante, par exemple) alors on est restreint aux méthodes de projection avec $G = A^T A$ (ce qui revient à résoudre le système aux moindres carrés) ou avec $G = I$. Mais dans les deux cas ces méthodes convergent très lentement et, en plus, chaque itération côute à peu près le double par rapport au cas $G = A$. Par exemple, des efforts considérables ont été réalisés pour rendre possible l'application de la méthode du gradient conjugué aux systèmes symétriques mais non définis positifs (ou non définis négatifs) (voir [11]).

Ceci étant, nous attachons une grande importance au fait de pouvoir accélérer les méthodes de résolution des systèmes quelconques. Des méthodes utilisant $q > 1$ peuvent aussi être envisagées, en particulier par analogie avec II.1.1. et aux méthodes basées sur une décomposition d'une norme, le cas :

$$(2.6) \qquad Z_n = (A^T r_n, (A^T)^2 r_n, \ldots, (A^T)^q r_n)$$

est à considérer. On minimise sur une variété de dimension q (en général) et la méthode converge parce que $A^T r_n$ est liée au gradient. Mais, en général, les quantités $r_n^T A^i s_n$, $i=2,\ldots,q$ n'ont de rapport avec aucune norme de r_n . On peut alors utiliser les normes de Holder :

$$(2.7) \qquad Z_n = (z_{n1}, z_{n2}, \ldots, z_{nq})$$

où les vecteurs z_{nj} , $j=1,\ldots,q$, décomposent la norme $\varphi_j(r_n)$.

$$(2.8) \qquad z_{nj}^{(i)} = \frac{\text{signe } ((r_n^{(i)})^j)(r_n^{(i)})^{j-1}}{\varphi_j(r_n)^{j-1}}$$

où $r_n^{(i)}$ et $z_{nj}^{(i)}$ représentent respectivement la $i^{\text{ème}}$ composante des vecteurs r_n et $z_{n,j}$; $j=1,\ldots,p$.

Ce qui est intéressant dans les méthodes utilisant $q > 1$ pour des matrices quelconques est que l'augmentation du coût d'une itération par rapport au cas symétrique défini positif, diminue lorsque q augmente et peut même devenir égal. Pour A quelconque, le meilleur moyen de calculer le résidu est d'utiliser

$$(2.9) \qquad r_n = A x_n - b$$

Si A est symétrique et définie positive, au cours de l'itération précédente on a calculé $A H_{n-1}$ et on peut alors utiliser :

$$(2.10) \qquad r_n = r_{n-1} + (A H_n) \wedge_n$$

ce qui entraine qp multiplications et qp additions. Pour $q = 1$ il

est évident que (2.10) est beaucoup moins chère que (2.9), mais pour q assez grand et A creuse, le coût de (2.10) peut égaler celui de (2.9) et même le dépasser. Le coût d'une itération est alors le même pour les deux cas. En plus, Rheid (voir [10]) affirme que (2.9) est numériquement plus intéressant que (2.10).

II.2.2. Un algorithme utilisant la dominance diagonale

Nous présentons maintenant un algorithme pour résoudre le système aux moindres carrés (1.8), dans le cas où A est strictement à dominance diagonale. Il a la particularité d'utiliser cette propriété de A dans un contexte de minimisation. C'est un algorithme similaire à l'algorithme (2.5) ; chaque itération demande seulement le produit d'une matrice par un vecteur, et de plus ce vecteur a comme éléments 1 où -1 . Pourtant, il correspond au cas $q = 1$, $G = A^T A$, qui demande, en général deux produits d'une matrice par un vecteur à chaque itération. Voici les règles qui le caractérisent :

$$
(2.11)\quad
\begin{cases}
x_o \text{ donné} \qquad U_{-1} = 0 \qquad\qquad \alpha_o = 0 \\
\alpha_n = -\dfrac{(Az_n, AU_{n-1})}{\|AU_{n-1}\|^2} \quad , \quad \varphi_1(r_n) = z_n^T . r_n \qquad (n \geq 1) \\
U_n = z_n + \alpha_n U_{n-1} \\
AU_n = Az_n + \alpha_n AU_{n-1} \\
x_{n+1} = x_n + \lambda_n U_n \quad , \quad \lambda_n = -\dfrac{(r_n, Az_n)}{\|AU_n\|^2} \\
r_{n+1} = r_n + \lambda_n AU_n
\end{cases}
$$

Quoique les z_n soient obtenus par une décomposition de la norme φ_1 de r_n la convergence de la méthode peut ne pas être assurée pour A quelconque, étant donné que r_n n'est pas le résidu du système à résoudre mais $A^T r_n$.

THEOREME 4

Si A a sa diagonale positive et est strictement à dominance diagonale, alors, les z_n de (2.11) sont liés au gradient et la méthode converge.

DEMONSTRATION

Par hypothèse, il existe $\tau > 0$ tel que $a_{ii} \geq \tau + \Sigma |a_{ij}|$ pour $i=1,\ldots,p$.
On a alors :

$$(2.12) \quad \begin{cases} |(Az_n)_i| \geq \tau \quad , \quad \forall\, n \text{ et } i=1,\ldots,p \\ z_n^T A z_n = \varphi(Az_n) \end{cases}$$

c'est-à-dire, que toute composante $(Az_n)_i$ de Az_n à le même signe que $(z_n)_i$ et , en conséquence, le même signe que $(r_n)_i$. Donc :

$$(2.13) \quad \begin{cases} |z_n^T G s_n| = (A z_n, r_n) \\ |z_n^T G s_n| = \sum_{i=1}^{p} |(r_n)_i| \cdot |(A z_n)_i| \\ |z_n^T G s_n| \geq \tau \varphi_1(r_n) \end{cases}$$

En plus, les $z_n^T G z_n = \|A z_n\|$ sont bornés parce qu'il y a seulement un nombre fini de z_n différents. La convergence de la méthode dérive alors du théorème 2.

II.2.3. Résultats numériques

Plusieurs variantes peuvent être envisagées en faisant varier q et la façon de déterminer les Z_n . C'est à l'utilisateur d'en choisir une d'après les caractéristiques de son problème. Notre but principal étant d'accélérer les méthodes existantes (les plus connues), nous présentons seulement des résultats numériques qui montrent ce qu'on peut gagner en accélérant ces méthodes et dans quelle mesure les méthodes obtenues fournissent des meilleurs résultats que d'autres plus utilisées.

Nous comparons les méthodes basées sur une décomposition d'une norme (φ_1 et φ_∞) dans le cas $q = 1$ et $G = I$. Des résultats similaires sont à espérer pour d'autres valeurs de q et pour d'autres façons de déterminer les directions. Voici les règles des méthodes à comparer :

METHODE I (φ) (non accélérée)

$$(2.14)\quad \begin{cases} x_o \text{ donné et à l'itération } n : \\ \lambda_n = - \dfrac{\varphi(r_n)}{\|A^T z_n\|^2} \quad , \qquad \varphi(r_n) = z_n^T . r_n \\ x_{n+1} = x_n + \lambda_n A^T z_n \end{cases}$$

Coût d'une itération = (p'+2p) multiplications + (2p'+3p) additions + 1 division.

METHODE II (φ) (accélérée)

$$(2.15)\quad \begin{cases} x_o \text{ donné} \quad , \quad U_{-1} = 0 \quad , \quad \alpha_o = 0 \\ \alpha_n = - \dfrac{(A^T z_n \, , \, A^T U_{n-1})}{\|A^T U_{n-1}\|^2} \qquad (n \geq 1) \\ A^T U_n = A^T z_n + \alpha_n A^T U_{n-1} \\ x_{n+1} = x_n + \lambda_n A^T U_n \quad , \quad \lambda_n = - \dfrac{\varphi(r_n)}{\|A^T U_n\|^2} \end{cases}$$

Coût d'une itération = (p'+4p) multiplications + (2p'+5p) additions + 2 divisions.

METHODE KACZMARZ

x_o donné et à l'itération n

$y_o = x_n$

Pour i = 1 jusqu'à p

faire

(2.16)
$$\lambda_i = - \frac{(Ay_{i-1}-b)_i}{\|A^T e_i\|^2}$$

$$y_i = y_{i-1} + \lambda_i A^T e_i$$

fin faire

$x_{n+1} = y_p$

Coût d'une itération = 2p' multiplications + 2p' additions + p divisions.

Dans tous les cas ω est connue : $\omega_i = 1$, i=1,...,p et on initialise avec $x_o = 0$. On compare les valeurs de $\varphi_1(x_n-\omega)$. Dans le tableau 2.3. on compare les méthodes basées sur une décomposition de φ_∞ et de φ_1 , entre elles et avec leur version accélérée correspondante. La matrice A se caractérise par le fait d'être très mal conditionnée :

$$A = \begin{vmatrix} 2 & 1 & 3 & 4 \\ 1 & -3 & 1 & 5 \\ 3 & 1 & 6 & -2 \\ 4 & 5 & -2 & -1 \end{vmatrix}$$

Tableau 2.3.

n	MET. I (φ_∞)	MET. II (φ_∞)	MET. I (φ_1)	MET. II (φ_1)
2	.13333 D 01	.64478 D 00	.16402 D 00	.85173 D-01
6	.47797 D 00	.17278 D 00	.94783 D-01	.63787 D-07
10	.19996 D 00	.47765 D-02	.50822 D-01	.23878 D-11
14	.19841 D 00	.12525 D-02	.27458 D-01	.11657 D-14
20	.18025 D 00	.18078 D-06	.10371 D-01	.0

Comme deuxième exemple, tableau 3.4., on compare la méthode basée sur une décomposition de φ_1 , avec sa version accélérée et avec la méthode de Kaczmarz. A étant la matrice (22,22) de la figure 2.1. obtenue par la discrétisation d'un problème de déplacement des marées par la méthode des éléments finis (voir [12]). Cette matrice est aussi très mal conditionnée et n'est ni symétrique ni diagonale dominante.

Tableau 2.4.

n	MET. I (φ_1)	MET. II (φ_1)	MET. KAC.
10 × 22	.11892 D 02	.36444 D 00	.14748 D 01
20 × 22	.99670 D 01	.12200 D-01	.16333 D 00
30 × 22	.69984 D 01	.45326 D-03	.18080 D-01
40 × 22	.53777 D 01	.14914 D-04	.20007 D-02
50 × 22	.42360 D 01	.49380 D-06	.22134 D-03
60 × 22	.34217 D 01	.14420 D-07	.24481 D-04
70 × 22	.28079 D 01	.45022 D-09	.27065 D-05
80 × 22	.22976 D 01	.16737 D-10	.29910 D-06
90 × 22	.18907 D 01	.41780 D-12	.33041 D-07

CONCLUSION

Les résultats numériques que nous venons de présenter montrent que la méthode d'accélération ici proposée peut fournir des résultats très intéressants surtout quand on l'applique aux méthodes utilisant des directions liées au gradient.

Nous espérons d'autre part, avoir mis en évidence la richesse des possibilités offertes par les méthodes de projection dans la résolution des systèmes d'équations linéaires. La plupart des méthodes ici étudiées se généralisent facilement pour la résolution des systèmes non linéaires et pour la minimisation de fonctionnelles non quadratiques. On pourrait également essayer d'accélérer la convergence de ces méthodes en utilisant les algorithmes étudiés dans la première partie. Mais il est difficile d'obtenir des résultats théoriques, parce que l'erreur à l'itération n, ne se met pas, en général sous la forme d'une somme d'exponentielles $\sum_{i=1}^{p} a_i \lambda_i$, comme c'est le cas pour les itérations linéaires, quand A satisfait certaines hypothèses.

REFERENCES

[1] A. HOUSEHOLDER and F. BAUER
On certain iterative methods for solving linear systems.
Num. Math. 2 (1960), 55-59

[2] N. GASTINEL
Analyse Numérique Linéaire. Hermann, Paris (1966).

[3] F. ROBERT
Matrices nonnégatives et normes vectorielles
Cours D.E.A., Université de Grenoble (1973).

[4] L. KANTOROVICH and G. AKILOV
Functional analysis in normed spaces
Pergamon Press, Oxford (1964).

[5] N. GASTINEL
Sur-décomposition d'une norme vectorielle et procédé itératif.
Num. Math. 5, (1963) 142-151.

[6] F. ROBERT
Décomposition d'une norme vectorielle et procédé itératif.
Note interne. Laboratoire de Calcul, Université de Grenoble (1964).

[7] S. SCHECHTER
Relaxation methods for convex problems.
SIAM J. Numer. Anal., Vol.5, n° 3, (1968) 601-612.

[8] J. ORTEGA and W. RHEINBOLDT
Iterative solution of nonlinear equations in several variables.
Academic Press (1970).

[9] W. TRENCH
An algorithm for the inversion of finite Hankel matrices.
J. Soc. Indust. Appl. Math., Vol.13, (1965) 1102-1107.

[10] J. RHEID
Large sparse sets of linear equations. Academic Press (1971).

[11] R. FLETCHER
Conjugate gradient methods for indefinite systems.
Procedings of Dundee conference.
Lectures notes in Mathematics, N° 506, Springer-Verlag (1976).

[12] C. Le PROVOST and A. PONCET
Finite element method for spectral modelling of tides.
Int. J. of Num. Math. Eng. (à paraître).

[13] J. BEUNEU
Résolution des systèmes d'équations linéaires par la méthode des compensations.
Publication 69. Laboratoire de Calcul, Université de Lille (1976).

TROISIEME PARTIE

CONVERGENCE D'UNE METHODE DE RESOLUTION DES EQUATIONS D'EQUILIBRE DANS LES RESEAUX THERMIQUES.

INTRODUCTION

L'objet principal de cette partie est de démontrer la convergence de la méthode de conservation de débit anticipée par appui sur couronne, proposée en 1 pour accélérer la convergence de la méthode Gauss-Seidel dans la résolution d'un problème non linéaire d'écoulement dans un réseau.

Dans ce but, nous présentons au paragraphe I.1 les principales approches qui ont été données à ce problème et les résultats correspondants d'existence et unicité de la solution et de convergence des méthodes Gauss-Seidel et Jacobi. Dans le paragraphe I.2 nous formulons la MCAC et on met en évidence certaines de ses caractéristiques, en particulier, que la convergence de cette méthode n'est pas monotone, comme c'est le cas pour ce problème avec la méthode de Jacobi ou Gauss-Seidel (pour certains points de départ). Cette caractéristique pose des difficultés pour démontrer la convergence de la méthode parce qu'on ne peut pas utiliser la théorie de convergence monotone qui à servi pour démontrer la convergence des méthodes Gauss-Seidel et Jacobi (voir [2], [3], et [4]). On note aussi que, quand le système d'équations à résoudre est défini par un opérateur gradient alors la MCAC est une méthode de minimisation de la fonctionnelle dont cet opérateur est la dérivée.

Cette remarque nous conduit à étudier le problème plutôt par des principes variationnels.

Dans le paragraphe II.1, on détermine la fonctionnelle à minimiser et on montre que toute méthode de minimisation utilisant des directions essentiellement périodiques est convergente, la MCAC y comprise. Dans le paragraphe II.2 nous généralisons la MCAC en donnant d'autres façons de choisir des directions afin de conserver une des caractéristiques importantes de la MCAC, qui est de minimiser (implicitement) dans des sous-espaces de dimension plus grand que un.

Comme résultat numérique nous présentons une courbe qui nous a été fournie par les responsables de l'équipe METHODES de la Section de Physique de Neutrons Rapides du C.E.N.-Cadarache où l'on montre, sur un exemple pratique, l'amélioration obtenue par l'application de cette méthode, par rapport à la méthode de sur-relaxation.

En annexe, nous donnons un algorithme de marquage pour la mise en niveaux d'un graphe connexe. La MCAC et sa généralisation font l'hypothèse que le graphe associé au réseau est partitionné en plusieurs niveaux (ou couronnes) selon le plus petit nombre d'arcs existants entre chaque noeud et les noeuds périphériques.

CHAPITRE I

PRESENTATION DU PROBLEME

I.1. DEFINITION ET PROPRIETES DES RESEAUX THERMIQUES

Un réseau thermique connexe est défini par un ensemble $N = \{1,2,\ldots,n\}$ de noeuds et un ensemble $L = \{(i,k)\} \subset N \times N$ d'arcs liant ces noeuds . Toute paire de noeuds i , k peuvent être reliés par un cycle d'arcs $(i_m,i_{m+1}) \in L$ avec $i_o = i$ et $i_M = k$. Pour chaque arc $(i,k) \in L$ on suppose donnée une fonction conductance $\varphi_{ik}(u,v)$. Par exemple, si u,v sont respectivement les températures aux noeuds i,k alors $\varphi_{ik}(u,v)$ est le flux de chaleur entre les noeuds i et k . On suppose, d'autre part, que les fonctions φ_{ik} sont continues et satisfont :

(i) $\varphi_{ik}(s,t) = 0$ si $(i,k) \notin L$

(ii) $\varphi_{ik}(s,t)$ strictement croissante en s et strictement décroissante en t pour tout $(i,k) \in L$

(iii) $\varphi_{ik}(s,t) + \varphi_{ki}(t,s)$ croissante en s et en t.

(iv) φ_{ik} et φ_{ki} sont non bornées quand s ou t tendent vers l'infini, pour tout $(i,k) \in L$.

Pour toute fonction réelle u définie sur N : $u = (u_1,\ldots,u_n)$ on définit le flux au noeud i par

$$(1.1) \qquad f_i(u) = \sum_{k \in N} \varphi_{ik}(u_i,u_k)$$

Ces précisions étant apportées , on peut considérer deux approches différentes du problème. Nous allons maintenant en donner un résumé.

I.1.1. Première approche (Birkhoff and Kellogg [7])

On considère deux types de noeuds : noeuds intérieurs et noeuds frontières. On cherche la solution du système d'équations (en général non linéaires).

$$(1.2)\quad \begin{cases} f_i(u) = q_i & i \text{ un noeud intérieur} \\ u_i = F_i(f_i(u)) & i \text{ un noeud frontière} \end{cases}$$

où les q_i sont des constantes réelles données et les fonctions F_i sont un ensemble de fonctions continues et décroissantes.
Pour l'ordre composante à composante dans R^n, on appelle sur-solution de (1.2) le vecteur v si

$$(1.3)\quad \begin{cases} f_i(v) \geq q_i & i \text{ un noeud intérieur} \\ v_i \geq F_i(f_i(v)) & i \text{ noeud frontière} \end{cases}$$

Si les inégalités sont inversées alors on dit que v est une sous-solution.
Birkhoff and Kellogg (voir [7])ont obtenu les résultats suivants :

(i) Si N à au moins un nceud frontière alors (1.2) à une solution unique

(ii) Si U est la solution de (1.2) et v et ω sont respectivement une sur et une sous-solution alors $\omega \leq u \leq v$

(iii) Si l'on génère la suite $u^{(k)}$ par la méthode de Jacobi alors $u^{(0)} \geq u^{(1)} \geq \ldots \geq u$ converge vers u si $u^{(0)}$ est une sur-solution. De même, $u^{(0)} \leq u^{(1)} \leq \ldots \leq u$ converge vers u si $u^{(0)}$ est une sous-solution.

Porsching (voir [4]) a ajouté les résultats suivants :

(i) Des méthodes pratiques pour calculer des sur et des sous-solutions (en un nombre fini de pas)

(ii) Il obtient pour la méthode de Gauss-Seidel le même résultat (iii) que Birkhoff and Kellogg pour celle de Jacobi

(iii) Pour le même point de départ (une sous ou sur-solution) la méthode de Gauss-Seidel converge au moins aussi vite que celle de Jacobi.

(iv) Pour tout $x^{(o)}$ donné il existe une sur-solution $\omega^{(o)}$ et une sous-solution $y^{(o)}$ telles que, en appliquant Gauss-Seidel ou Jacobi respectivement aux points de départ $y^{(o)}$, $x^{(o)}$ et $\omega^{(o)}$ on ait :

$$y^{(k)} \leq x^{(k)} \leq \omega^{(k)} \quad , \quad k=1,... \tag{1.4}$$

La convergence de $\{x^{(k)}\}$ découle alors de celle de $\{y^{(k)}\}$ et $\{\omega^{(k)}\}$ vers u (convergence globale).

I.1.2. Deuxième approche (Birkhoff and Dias [5])

Dans plusieurs cas, comme dans les réseaux électriques (courant continu) et hydrauliques les φ_{ik} satisfont

$$\varphi_{ik}(u,v) = -\varphi_{ki}(u,v) = c_{ik}(u-v) \tag{1.5}$$

Pour ce type de problèmes, Birkhoff et Diaz (voir [5] et [6]) ont donné une approche particulière qui n'entre pas tout à fait dans le cadre défini par (1.2). Ils posent le problème en utilisant un ensemble L' d'arcs orientés : $L' = \{1,...,\ell\}$ obtenus au moyen d'une certaine fonction surjective $a : L \to L'$, telle que :

$$\begin{cases} a((i,k)) = a((k,i)) \\ a((i,k)) \neq a((j,t)) \end{cases} \tag{1.6}$$

si $(i,k) \neq (j,t)$ ou $(k,i) \neq (j,t)$. On complète la définition du graphe avec une matrice d'incidence $E = (e_{kj})$ ayant n lignes et ℓ colonnes et telle que :

$$e_{kj} = \begin{cases} 1 & \text{si le noeud k est le noeud initial de l'arc j} \\ -1 & \text{si le noeud k est le noeud final de l'arc j} \\ 0 & \text{si l'arc j n'est pas incident au noeud k} \end{cases} \tag{1.7}$$

Le graphe étant connexe, chaque ligne de E a au moins un élément non nul et chaque colonne a exactement deux éléments non nuls. On écrit, en plus $j = (i,k)$ où i est le noeud initial de l'arc j et k le noeud final.

On appelle Δu_j la "chute de potentiel" dans l'arc j pour un état $u = (u_1,\ldots,u_n)$ donné :

$$(1.8) \qquad \Delta u_j = u_i - u_k$$

En tenant compte de (1.5), les fonctions φ_{ik} sont remplacées par des fonctions à une seule variable $c_j(\Delta u_j)$. Alors les fonctions f_i définies par (1.1) deviennent

$$(1.9) \qquad f_i(u) = \sum_{j=1}^{\ell} e_{ij}\, c_j(\Delta u_j)$$

On considère une nouvelle partition de N en trois types de noeuds. Pour cela on se donne les sous-ensembles ∂N et $\partial^* N$, appelés respectivement frontière et ensemble résiduel et tels que $\partial^* N \subset \partial N \subset N$. On veut résoudre le système d'équations non linéaires suivant :

$$(1.10) \quad \begin{cases} \text{(i)} & f_i(u) = 0 & \text{si } i \in N-\partial N \\ \text{(ii)} & f_i(u) = G_i(u_i) & \text{si } i \in \partial^* N \\ \text{(iii)} & u_i \text{ connu} & \text{si } i \in \partial N-\partial^* N \end{cases}$$

Par analogie avec la théorie du potentiel, on appelle les conditions (iii) conditions de Dirichlet, celles de (ii) avec G_i = constante : conditions de Neumann et on parle des problèmes mixtes quand il existe d'autres fonctions G_i.

Puisque, par hypothèse, les G_i sont continues si, en plus, elles sont strictement décroissantes et surjectives alors il existe $F_i = G_i^{-1}$ et (ii) peut se mettre sous la forme $u_i = F(f_i(u))$. C'est aussi le cas pour les u_i connus avec F_i = constante. Donc, lorsque certaines fonctions G_i sont constantes $(f_i(u) = q_i)$ et les autres strictement décroissantes et surjectives alors (1.10) est un cas particulier de (1.2). Lorsqu'il existe des G_i n'ayant pas ces caractéristiques, on a le résultat suivant :

THEOREME 1 (voir [6])

Si les fonctions c_j et G_i sont continues, avec les c_j strictement croissantes et les G_i strictement décroissantes et si les fonctions c_j et G_i prennent les deux signes alors le problème (1.10) à une solution unique.

Dans ce qui suit, nous utiliserons cette approche qui est la plus adaptée pour un traitement variationnel du problème.

I.2. METHODE DE CONSERVATION DE DEBIT ANTICIPEE (MCAC)

En [1], il a été proposé une méthode, dite d'accélération de la convergence de la procédure Gauss-Seidel non linéaire pour résoudre (1.10). Elle peut aussi être formulée de la façon suivante :
Nous appelons noeuds de niveau o les noeuds de l'ensemble $N_o = \partial N - \partial^* N$ (u_i connus) . L'ensemble $O_1 = N - N_o$ est appelé cercle 1. Il contient tous les noeuds où les u_i ne sont pas connus. On définit aussi les niveaux 1,2,...,k et les cercles 2,...,k par le récurrence

$$(1.11)\quad \begin{cases} N_o = \partial N - \partial^* N & O_o = N \\ O_j = O_{j-1} - N_{j-1} & \\ N_j = \{i \in O_j \mid \exists\ t \in N_{j-1} \text{ avec } (i,t) \in L\} & \end{cases}$$

pour $j=1,\ldots,k$; k étant le premier entier tel que $O_k = N_k$.

Supposons N_o non vide et k plus grand que un. Soit

$$(1.12)\quad m = n - \text{card}\,(N_o)$$

On a à résoudre un système d'équations non linéaires à m inconnues (les u_i non donnés) et m équations que nous écrivons sous la forme $\hat{f}(u) = 0$, avec

$$(1.13)\quad \begin{cases} \hat{f}_i(u) = f_i(u) & \text{si} \quad i \in N - \partial N \\ \hat{f}_i(u) = f_i(u) - G_i(u_i) & \text{si} \quad i \in \partial^* N \end{cases}$$

Considérons, en plus les vecteurs $z^{(j)}$, $j=1,..,k$:

$$(1.14)\quad z_i^{(j)} = \begin{cases} 1 & \text{si} \quad i \in O_j \\ 0 & \text{si} \quad i \notin O_j \end{cases}$$

DEFINITION 1

La MCAC pour résoudre (1.10) consiste à réaliser à chaque itération :

(i) Une itération de la méthode Gauss-Seidel non linéaire, pour obtenir $u^{(\ell_o)}$ à partir de $u^{(\ell-1)}$.

(ii) Pour $j=1,\ldots,k$ calculer λ_j tel que :

$$u^{(\ell_j)} = u^{(\ell_{j-1})} + \lambda_j z^{(j)}$$

$$\sum_{i \in O_j} \hat{f}_i(u^{(\ell_j)}) = 0$$

(iii) $u^{(\ell)} = u^{(\ell_k)}$

Quand les fonctions G_i sont des constantes alors une des caractéristiques de cette méthode est que :

$$(1.15) \quad \sum_{O_k} \hat{f}_i(u^{(\ell)}) = \sum_{O_{k-1}} \hat{f}_i(u^{(\ell)}) = \ldots = \sum_{O_1} \hat{f}_i(u^{(\ell)}) = 0$$

celà est dû aux relations (1.5) qui entraînent que si l'on modifie des composantes appartenant à un cercle donné O_j, les sommes sur $O_{j-1}, O_{j-2}, \ldots, O_1$ ne sont pas modifiées.
Considérons donc la matrice $Z(m,k)$, ayant comme colonnes les vecteurs $z^{(j)}$, $j=1,\ldots,k$. La partie (ii) de la définition 1 consiste alors à calculer le vecteur $\wedge^{(\ell)} \in R^k$, tel que :

$$(1.16) \quad \begin{cases} u^{(\ell_k)} = u^{(\ell_o)} + Z \wedge^{(\ell)} \\ Z^T \hat{f}(u^{(\ell_k)}) = 0 \in R^k \end{cases}$$

et on voit que dans le cas où $\hat{f}$ est un opérateur gradient (1.16) revient à déterminer $\wedge^{(\ell)}$ par le principe de Curry (voir [8]) de façon à minimiser la fonctionnelle dont $\hat{f}$ est la dérivée, dans le sous-espace engendré par Z. C'est donc un cas particulier des méthodes étudiées dans la deuxième partie.

Nous pensons que c'est cette propriété qui rend la méthode très performante (voir figure 3.1).
Une autre caractéristique de cette méthode, valable sous l'hypothèse que $\hat{f}$ est un opérateur gradient, est qu'elle consiste à utiliser une suite de directions essentiellement périodiques pour minimiser la fonctionnelle dont $\hat{f}$ est la dérivée. e_i étant le $i^{ème}$ vecteur de base, cette suite est formée par les vecteurs $e_1, e_2, \ldots, e_m, z^{(1)}, z^{(2)}, \ldots, z^{(k)}$ pris de façon cyclique. Elle satisfait la définition suivante

DEFINITION 2 (voir [8])

Une suite $\{p^{(\ell)}\} \subset R^m$ qui contient seulement un nombre fini de vecteurs distincts est essentiellement périodique s'il existe un entier $t \geq n$ et un indice ℓ_o tel que pour tout $\ell \geq \ell_o$ les t vecteurs $p^{(\ell+1)}, \ldots, p^{(\ell+t)}$ engendrent R^n.

Comme nous l'avons vu d'une façon sommaire au paragraphe précédent, la convergence des méthodes Jacobi et Gauss-Seidel pour ce type de problèmes a été démontrée en utilisant le fait que ces méthodes convergent de façon monotone quand l'approximation initiale est une sous ou une sur-solution. Cette propriété est aussi utilisée pour démontrer la convergence globale (pour tout approximation initiale). Par contre, pour la MCAC, on n'a pas cette propriété. Supposons, par exemple, que $u^{(o)}$ soit une sous-solution c'est-à-dire que $\hat{f}_i(u^{(o)}) \leq 0$ pour tout $i=1,\ldots,m$. D'après (1.15), $u^{(1)}$ satisfait

$$(1.17) \qquad \sum_{i=1}^{m} \hat{f}_i(u^{(1)}) = 0$$

et par conséquent, soit $u^{(1)}$ est la solution de (1.10), soit $u^{(1)}$ n'est pas une sous-solution puisqu'il est nécessaire que certains $\hat{f}_i$ soient strictement positifs.
Ceci étant donné, dans le prochain chapitre nous montrerons que $\hat{f}$ est un opérateur gradient et que la fonctionnelle dont $\hat{f}$ est la dérivée satisfait les conditions nécessaires pour que les méthodes utilisant des directions essentiellement périodiques convergent globalement vers la solution de (1.10).

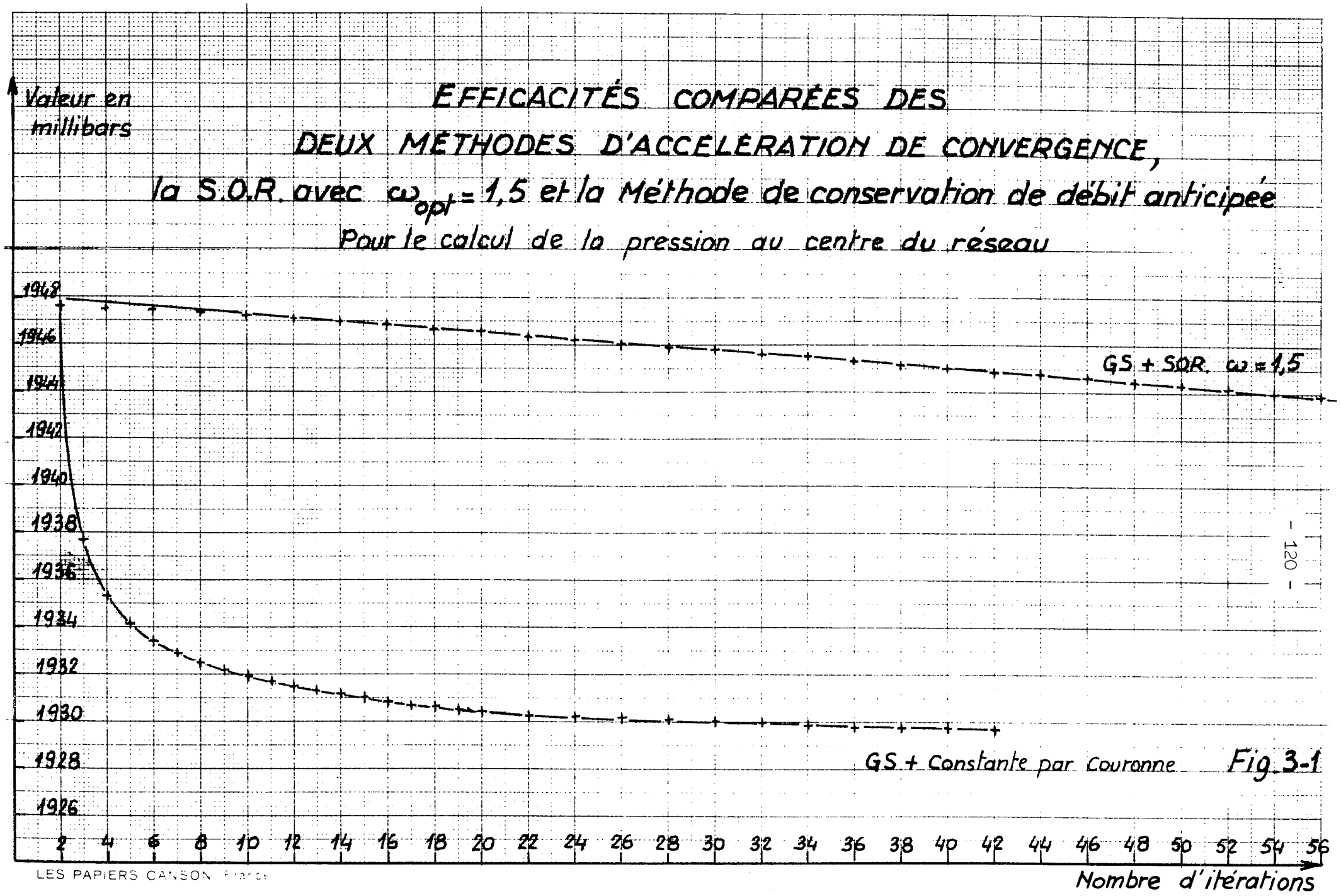
La figure 3.1 montre les résultats obtenus avec la MCAC et avec la méthode de sur-relaxation avec $\omega = 1.5$, sur un exemple pratique ayant 205 inconnues et 17 niveaux (ou couronnes). Les fonctions G_i sont des constantes et les c_j sont définies par :

$$(1.18) \quad c_j(\Delta u_j) = \text{signe}\,(\Delta u_j)\left[\left(k_{j1}^2 + k_{j2}/|\Delta u_j|\right)^{1/2} - k_{j1}\right]$$

où les k_{j1} et k_{j2} sont des constantes positives.
Un autre problème à 2501 inconnues et 25 couronnes a été résolu au C.E.N.-Cadarache, avec une précision de trois milibars :

$$(1.19) \quad \max_i \left|u_i^{(\ell)} - u_i^{(\ell-1)}\right| < 3 \text{ mbars.}$$

Le temps de résolution a été de deux minutes et douze secondes d'IBM 360/91.



CHAPITRE II

ETUDE VARIATIONNELLE ET CONVERGENCE DES METHODES DE MINIMISATION

II.1. CARACTERISATION ET PROPRIETES DE LA FONCTIONNELLE A MINIMISER

Dans ce paragraphe nous explicitons g telle que $\hat{f} = \text{grad}\,(g)$ et, dans le but de démontrer que toute méthode de minimisation de g utilisant des directions essentiellement périodiques est convergente, nous montrerons que g est continuement G-différentiable, strictement convexe et telle que $\lim_{\|x\|\to\infty} g(x) = \infty$.

Soit $g(u)$ la fonctionnelle suivante :

$$(2.1) \qquad g(u) = \sum_{L'} \int_0^{\Delta u_j} c_j(s)ds - \sum_{\partial^* N} \int_0^{u_i} G_i(s)ds$$

D'après (1.9), (1.10), (1.13) et par définition de g on a :

$$(2.2) \qquad \hat{f}(u) = \text{grad}\,(g)$$

Pour que g soit continuement G-différentiable, l'existence des dérivées $\frac{\partial g(u)}{\partial u_i}$ n'est pas suffisante, nous supposerons alors que les fonctions c_j et G_i sont continues, ce qui entraine la continuité de $\hat{f}$. g est alors F-différentiable (voir [9] , théorème 3.7.1) et en conséquence G-différentiable (voir [8]).

Cette fonctionnelle a été définie et utilisée en [6] pour démontrer le théorème 1. Ces auteurs ont aussi donné un théorème similaire au suivant mais qui impose, en plus, aux fonctions G_i d'être strictement décroissantes.

THEOREME 2

Si les fonctions c_j sont continues et strictement croissantes et les fonctions G_i sont décroissantes alors $g(u)$ est strictement convexe.

DEMONSTRATION

g étant une fonctionnelle continue, il suffit de démontrer que

$$(2.3) \qquad g(x+y) + g(x-y) - 2g(x) > 0 \qquad \forall\, x,y \in R^m - \{0\}\ ,\ x \neq y$$

Nous complétons les vecteurs x et y par :

$$(2.4) \quad \begin{cases} y_{m+1} = y_{m+2} = \ldots = y_n = 0 \\ x_{m+1} = u_{m+1}\ , \ldots\ , x_n = u_n \qquad (u_i \text{ connus}) \ . \end{cases}$$

On a ainsi $\Delta(x+y)_j = \Delta x_j + \Delta y_j$ et $\Delta(x-y)_j = \Delta x_j - \Delta y_j$.
Le graphe étant connexe et supposant, en plus, que l'ensemble $\partial N - \partial^* N$ est non vide (existence de certains u_i connus), cela entraîne qu'il existe au moins un $\Delta y_j \neq 0$, parce que $y \neq 0$ et qu'il existe des différences du type $\Delta y_j = y_i - 0$.
Ceci étant donné, on a par définition :

$$\int_0^{\Delta x_j+\Delta y_j} c_j(s)ds + \int_0^{\Delta x_j-\Delta y_j} c_j(s)ds - 2\int_0^{\Delta x_j} c_j(s)ds =$$

$$= \int_0^{\Delta x_j} c_j(s)ds + \int_{x_j}^{\Delta x_j+\Delta y_j} c_j(s)ds + \int_0^{\Delta x_j} c_j(s)ds + \int_{x_j}^{\Delta x_j-\Delta y_j} c_j(s)ds -$$

$$- 2\int_0^{\Delta x_j} c_j(s)ds = \int_0^{\Delta y_j} c_j(\Delta x_j+t)dt - \int_0^{\Delta y_j} c_j(\Delta x_j-t)dt =$$

$$= \int_0^{\Delta y_j} [c_j(\Delta x_j+t) - c_j(\Delta x_j-t)]dt \geq 0$$

avec les inégalité stricte au moins pour un j parce que les c_j sont strictement croissantes et qu'il existe au moins un $\Delta y_j \neq 0$. On a alors

$$(2.5) \qquad \sum_{L'} \int_0^{\Delta(x+y)_j} c_j(s)ds + \sum_{L'} \int_0^{\Delta(x-y)_j} c_j(s)ds - 2 \sum_{L'} \int_0^{\Delta x_j} c_j(s)ds > 0$$

Pour démontrer la convexité stricte de g il nous reste seulement à démontrer que :

$$(2.6) \quad \sum_{\partial^* N} \int_0^{x_i+y_i} G_i(s)ds + \sum_{\partial^* N} \int_0^{x_i-y_i} G_i(s)ds - 2 \sum_{\partial^* N} \int_0^{x_i} G_i(s)ds \leq 0$$

Pour celà on utilise la décroissance des G_i :

$$\int_0^{x_i+y_i} G_i(s)ds + \int_0^{x_i-y_i} G_i(s)ds - 2\int_0^{x_i} G_i(s)ds =$$

$$= \int_{x_i}^{x_i+y_i} G_i(s)ds + \int_{x_i}^{x_i-y_i} G_i(s)ds =$$

$$= \int_0^{y_i} G_i(x_i+t)dt - \int_0^{y_i} G_i(x_i-t)dt =$$

$$= \int_0^{y_i} [G_i(x_i+t) - G_i(x_i-t)\, dt] \leq 0$$

Pour assurer la convergence des méthodes utilisant une suite de directions essentiellement périodiques il faut encore démontrer que $\lim_{\|u\|\to\infty} g(u) = \infty$.

Cette condition est aussi suffisante pour assurer l'existence d'un minimum unique (voir [8] théorème 4.3.4). Mais g étant continue G-différentiable et strictement convexe, on peut montrer que c'est aussi une condition nécessaire en utilisant des résultats classiques (théorèmes 4.3.1 et 4.3.2 de [8] et 8.7 de [10]). Donc, dans le cas où l'on connait l'existence d'une solution unique de (1.10) alors la convexité de g assure l'existence d'un seul minimum global et le problème est résolu.
L'existence et l'unicité de la solution de (1.10) a été démontrée sous les hypothèses :

(i) Les fonctions c_j continues, strictement croissantes et surjectives, et les fonctions G_i continues avec certaines G_i strictement décroissantes et surjectives et les autres constantes, par Birkhoff and Kellogg (résultats exposés au paragraphe I,1.1.).

(ii) Les fonctions c_j et G_i satisfaisant les hypothèses du théorème 1 (Birkhoff [6]).

Le théorème suivant à l'intérêt d'apporter une démonstration différente et unifiée sous la seule hypothèse de décroissance des G_i .

THEOREME 3

Si les fonctions c_j sont continues, strictement croissantes et surjectives et les fonctions G_i sont continues et décroissantes alors $\lim_{\|x\|\to\infty} g(x) = \infty$.

DEMONSTRATION

D'après le théorème 4.3.2 de [8], pour que $\lim_{\|x\|\to\infty} g(x) = \infty$ il faut et il suffit que tous les ensembles de niveau de g soient bornés. Soit β quelconque et considérons l'ensemble de niveau L_β et son cône de récession $0^+ L_\beta$:

$$L_\beta = \{x \in R^m \mid g(x) \leq \beta\}$$

$$0^+ L_\beta = \{y \in R^m \mid x + \lambda y \in L_\beta \quad \forall\, \lambda \geq 0 \,,\, x \in L_\beta\} \tag{2.7}$$

$$0^+ L_\beta = \{y \in R^m \mid g(x+\lambda y) \leq \beta \quad \forall\, \lambda \geq 0 \,,\, x \in L_\beta\}$$

Supposons L_β non vide. g étant continue et convexe, L_β est convexe et fermé. Pour démontrer que L_β est aussi borné nous utiliserons le théorème 8.4 de [10] : "Un ensemble fermé non vide et convexe $L \subset R^m$ est borné si et seulement si son cône de récession $0^+ L$ contient seulement le vecteur nul".
Supposons qu'il existe $y \neq 0$ tel que $y \in 0^+ L_\beta$ et $x \in L_\beta$. Alors :

$$g(x+\lambda y) \leq \beta \qquad \forall\, \lambda \geq 0 \,. \tag{2.8}$$

Comme pour la démonstration du théorème 2, nous complétons le vecteur x par les u_i connus et le vecteur y par des zéros, de façon à avoir $\Delta(x+\lambda y)_j = \Delta x_j + \lambda \Delta y_j$, $\forall\, j \in L$ et au moins un $\Delta y_j \neq 0$. Les fonctions G_i étant décroissantes, pour $y_i \neq 0$ et pour tous les G_i qui prennent les deux signes ou tels que signe $(G_i(0)) \neq$ signe (y_i) , il existe $\lambda' \geq 0$ tel que :

$$(2.9) \qquad \int_0^{x_i+\lambda y_i} G_i(s)ds \leq 0 \qquad \forall\, \lambda \geq \lambda'$$

Pour les G_i tels que signe $(G_i(0))$ = signe (y_i) on a la majoration suivante :

$$(2.10) \qquad \left|\int_0^{x_i+\lambda y_i} G_i(s)ds\right| \leq \left|\int_0^{x_i} G_i(s)ds\right| + \lambda\, G_i(0)y_i \qquad \forall\, \lambda \geq \lambda^*$$

où λ^* est tel que $x_i + \lambda^* y_i$ a le même signe que y_i , $\forall$ i .

On pose $\lambda_o = \max(\lambda', \lambda^*)$, $M_1 = \sum_{\partial^* N} \left|\int_0^{x_i} G_i(s)ds\right|$ et

$M_2 = \sum_{\partial^* N} |G_i(0)y_i|$. Alors :

$$(2.11) \qquad \sum_{L'} \int_0^{\Delta x_j+\lambda\Delta y_j} c_j(s)ds \leq \beta + M_1 + \lambda M_2 \quad , \quad \forall\, \lambda \geq \lambda_o$$

Posons $L_1 = L' - \{j \mid \Delta y_j = 0\}$ et $M_3 = \beta + M_1 - \sum_{L'} \int_0^{\Delta x_j} c_j(s)ds$. Alors

$$(2.12) \qquad \sum_{L_1} \int_0^{\lambda\Delta y_j} c_j(\Delta x_j+t)dt \leq \lambda\, M_2 + M_3 \quad , \quad \forall\, \lambda \geq \lambda_o$$

Mais les fonctions c_j étant continues, strictement croissantes et surjectives, il existe $\lambda_1 \geq \lambda_o$ tel que :

$$(2.13) \qquad \begin{cases} \displaystyle\int_0^{\lambda_1\Delta y_j} c_j(\Delta x_j+t)dt > \max(0, M_3) \quad , \quad \forall\, j \in L_1 \\ \displaystyle|c_j(\Delta x_j+\lambda_1\Delta y_j)| > \frac{2\,M_2}{|\Delta y_j|} \quad , \quad \forall\, j \in L_1 \end{cases}$$

En prenant $\lambda = 2\,\lambda_1 \geq 0$, on a pour tout $j \in L_1$:

$$(2.14) \qquad \begin{cases} \displaystyle\int_0^{\lambda\Delta y_j} c_j(\Delta x_j+t)dt = \int_0^{\lambda_1\Delta y_j} c_j(\Delta x_j+t)dt + \int_{\lambda_1\Delta y_j}^{2\lambda_1\Delta y_j} c_j(\Delta x_j+t)dt \\ \displaystyle\int_0^{\lambda\Delta y_j} c_j(\Delta x_j+t)dt > M_3 + \lambda M_2 \\ \displaystyle\sum_{L_1} \int_0^{\lambda\Delta y_j} c_j(\Delta x_j+t)dt > M_3 + \lambda M_2 \end{cases}$$

Ceci contredit (2.12) qui a été obtenu en supposant qu'il existe $y \neq 0$ tel que $y \in 0^+ L_\beta$. Cette hypothèse est donc fausse et en conséquence L_β est borné, ce qui achève la démonstration.

Les hypothèses de ce théorème sont plus faibles que celles de Birkhoff and Kellogg (voir [7]) qui imposent aux fonctions G_i d'être inversibles. Par rapport aux hypothèses du théorème 1, celles du théorème 3 sont plus fortes sur les c_j mais plus faibles sur les G_i . Dans le cas d'un réseau électrique, le fait d'imposer aux c_j d'être surjectives revient à exclure l'existence de courants saturés.

Nous sommes maintenant en mesure d'identifier la famille de problèmes pour laquelle la méthode suivante converge globalement.

DEFINITION 3

Une méthode de minimisation utilisant des directions essentiellement périodiques est définie par l'itération suivante :

$$x^{(\ell+1)} = x^{\ell} - \omega_\ell \alpha_\ell p^{(\ell)} \quad , \quad \ell=0,1,... \tag{2.15}$$

avec x_o donné , $\{p^{(\ell)}\}$ une suite de vecteurs essentiellement périodiques α_ℓ déterminé par un algorithme de minimisation dans une direction et ω_ℓ étant un paramètre de relaxation.

THEOREME 4

Si les fonctions c_j et G_i du problème (1.10) satisfont les hypothèses du théorème 1 ou celles du théorème 3 alors toute méthode de minimisation utilisant des directions essentiellement périodiques avec ω_ℓ et α_ℓ tels que :

$$\begin{aligned} & 0 < \varepsilon \leq \omega_\ell \leq 1 \\ & g(x^{(\ell)} - \alpha_\ell p^{(\ell)}) = \min \{g(x^{(\ell)} - \alpha p^{(\ell)}) \mid -\infty < \alpha < \infty\} \end{aligned} \tag{2.16}$$

converge vers la solution unique de (1.10), pour toute approximation initiale $x_o \in R^m$.

DEMONSTRATION

D'après les théorèmes 1, 2 et 3 la fonctionnelle g est continuement différentiable, strictement convexe et telle que $\lim_{\|x\| \to \infty} g(x) = \infty$.

Alors, le théorème 14.6.6. de [8] sur la convergence globale des méthodes utilisant des directions essentiellement périodiques s'applique.

COROLLAIRE 4.1.

Si les fonctions c_j et G_j de (1.10) satisfont les hypothèses du théorème 1 ou celles du théorème 3 alors la MCAC converge vers la solution unique de (1.10), pour tout $x_o \in R^m$.

DEMONSTRATION

D'après les résultats du paragraphe I.2 la MCAC est une méthode de minimisation utilisant des directions essentiellement périodiques, avec $\omega_\ell = 1$. La convexité stricte de g entraine que la définition (2.16) de α_ℓ coïncide avec le principe de Curry utilisé pour la MCAC :

$$(2.17) \quad g'(x_\ell - \alpha_\ell p^{(\ell)}).p^{(\ell)} = 0$$

En conséquence, le théorème 4 s'applique.

II.2. GENERALISATION DE LA MCAC

La courbe de la figure (3.1) montre que lors des premières itérations la MCAC converge très vite mais qu'ensuite l'accélération qu'elle apporte à la méthode de Gauss-Seidel est presque nulle. Nous pensons que ceci est dû au fait que les directions de minimisation de la MCAC sont privilégiées mais que au bout d'un certain nombre d'itérations g a été suffisamment minimisé dans ces directions et en conséquence le gain est moindre. D'autre part, dans le cas de la figure 3.1, les fonctions G_i sont constantes et (1.15) est alors vérifié. Puisque la convergence est démontrée pour tout ensemble de directions essentiellement périodiques, nous proposons d'autres façons de déterminer des directions essentiellement périodiques qui satisfont (1.15) quand les G_i sont des constantes.

Pour celà nous considérons (de façon artificielle) un ou plusieurs noeuds du niveau 1 comme des noeuds où la valeur de u est connue (appartenant à v_o) . On définit ainsi d'autres niveaux et d'autres cercles, dont les $z^{(j)}$ correspondants satisfont (1.15). Soient, par exemple $t \in N_1$, et :

$$N_o^{(t)} = \partial N - \partial^* N + \{t\} \quad , \quad O_o^{(t)} = N$$

$$O_j^{(t)} = O_{j-1}^{(t)} - N_{j-1}^{(t)}$$

$$N_j^{(t)} = \{i \in O_j^{(t)} \mid \exists\, \ell \in N_{j-1} \text{ avec } (i,\ell) \in L\}$$

pour $j=1,\ldots,k$. $k = k(t)$ étant le premier entier pour lequel $O_k^{(t)} = N_k^{(t)}$. Aux cercles $O_j^{(t)}$ on peut associer des vecteurs $z^{(t_j)}$ par des relations du type (1.14) . En faisant cette démarche pour tout $t \in N_1$, on génère une suite des directions essentiellement périodiques {directions utilisées par la MCAC} $\cup$ $\{z^{t_j}\}$, pour tout $t \in N_1$ et $j=1,\ldots,k(t)$. On fait alors des minimisations sur plusieurs sous-espaces de dimension plus grande que 1 . (1.15) est toujours satisfait parce que d'après (1.5) et (1.9) on a :

$$(2.19) \qquad \sum_I \hat{f}_i(u) = \sum_{i \in N_j} \sum_{q \in L_i} e_{iq}\, c_q(\Delta u_q) - \sum_J G_i(u_i)$$

où $I = O_j^{(t)}$, $J = O_j^{(t)} \cap \partial^* N$, $L_i = \{q = (i,\ell) \mid \ell \notin O_j^{(t)}\}$.

Cette relation est intéressante parce qu'elle facilite les calculs. En plus, quand les G_i sont des constantes et qu'on modifie u par l'adjonction de $\lambda\, z^{(t_{j+1})}$, avec $\lambda \in R$, on ne modifie aucune composante de $N_j^{(t)}$ ni de $c(O_j^{(t)})$ et en conséquence :

$$(2.20) \qquad \sum_I \hat{f}_i(u) = \sum_I \hat{f}_i(u + \lambda z^{(t_{j+1})})$$

ce qui montre que (1.15) est satisfait.

L'ordre dans lequel les indices $t \in N_1$ sont utilisés peut être défini à priori, de façon a modifier toutes les composantes le plus souvent possible, par exemple. Ou encore, on pourrait définir t par

$$(2.21) \qquad |f_t| = \min_{i \in L_o} |f_i|$$

où $L_o = N_1 - \{i \text{ déjà utilisés}\}$. C'est un choix logique, compte tenu que l'on considère t comme un noeud où u_t est connu.

A N N E X E A

ALGORITHME POUR LA MISE EN NIVEAUX D'UN GRAPHE CONNEXE

Dans ce qui précède nous avons supposé que le graphe associé au réseau peut être mis en niveaux et qu'on connait les ensembles N_j , $j=0,\ldots,k$. Dans la pratique, c'est souvent le cas, mais il peut arriver que le graphe puisse être mis en niveaux mais que néanmoins on ne connaisse pas à priori k ni les ensembles N_j , $j=0,\ldots,k$. Nous donnons donc un algorithme pour résoudre ce problème. Il a seulement la prétention de fournir aux non spécialistes des graphes un moyen pour savoir dans quelle mesure il peut être intéressant d'utiliser la MCAC (pour k assez grand) et pour faciliter cette utilisatinn. On suppose connu l'ensemble N_o et on initialise la variable j à 0 et le vecteur $v \in R^n$ à :

$$v_i = \begin{cases} -1 & \text{si} \quad i \notin N_o \\ 0 & \text{si} \quad i \in N_o \end{cases}$$

Les étapes de l'algorithme sont les suivantes :

(1) Pour i=1 jusqu'à n faire
  si $v_i = -1$
    alors
      allez en (2)
  fsi
fin faire

k = j

Pour j=1 jusqu'à k faire
  $N_j = \{i \mid v_i = j\}$
fin faire

fin de l'algorithme

(2) Pour tout i tel que $v_i = j$ faire
  Pour t=1 jusqu'à n faire
    si $(i,t) \in L$ et $v_t = -1$
      alors
        $v_t = j+1$
    fsi
  fin faire
fin faire

j = j+1

allez en (1)

REFERENCES

[1] P. BERNA et M. MALAMAS
Présentation de nouvelles méthodes pour la résolution itérative de problèmes non linéaires.
Colloque d'Analyse Numérique, Port-Bail (1976).

[2] W. RHEINBOLDT
On M-functions and their application to nonlinear Gauss-Seidel iterations and to network flows.
J. Math. Anal. Appl. 32, (1970) 274-306.

[3] G. MINTY
Monotone networks.
Proc. Roy. Soc. Ser.A, 257, (1960) 194-212

[4] T. PORSCHING
"Jacobi and Gauss-Seidel methods for nonlinear network problems.
SIAM J. Numer. Anal. 6, (1969) 437-448.

[5] G. BIRKHOFF and J. DIAZ
Nonlinear network problems.
Quart. Appl. Math. 13, (1965) 431-443.

[6] G. BIRKHOFF
A variational principle for nonlinear networks.
Quart. Appl. Math. 21, (1963) 160-162.

[7] G. BIRKHOFF and B. KELLOGG
Solution of equilibrium equations in thermal networks.
Proc. Symp. on Generalized networks, vol. 16, Microwave Research Institute Symposia Series.
J. Fox. Ed. Brooklyn Polytechnic Press, New-York (1966).

[8] J. ORTEGA and W. RHEINBOLDT
Iterative solution of nonlinear equations in several variables.
Academic Press, New-York (1970).

[9] H. CARTAN
Calcul différentiel. Hermann , Paris (1967).

[10] R. ROCKAFELLAR
Convex Analysis.
Princeton University Press, New-Jersey (1970).

195 D 12	-.7175 D 10	0	0	0	-.7400 D 10	0	0	-.6127 D 11	-.5182 D 11	0	0	.2652 D 11	0	0	0	-.2652 D 11	0	0	.1638 D 05	-.4096 D 04	0
175 D 10	.1190 D 12	0	0	0	0	.7763 D 10	0	0	-.5578 D 11	-.5646 D 11	-.2652 D 11	.2560 D 03	0	0	0	0	.2652 D 11	0	0	.2048 D 05	.8320 D 04
0	0	.1198 D 12	-.7175 D 10	0	0	0	-.6679 D 10	0	-.5551 D 11	-.5844 D 11	0	0	0	.2652 D 11	0	0	0	.2652 D 11	0	-.2048 D 05	-.1229 D 05
0	0	-.7175 D 10	.1195 D 12	-.7400 D 10	0	0	0	-.6127 D 11	-.5182 D 11	0	0	0	-.2652 D 11	0	.2652 D 11	0	0	0	.4096 D 04	-.1638 D 05	0
0	0	0	0	.2926 D 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	.3197 D 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	.3093 D 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	.3283 D 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0
127 D 11	0	0	-.6127 D 11	-.6117 D 11	-.6117 D 11	0	0	.2488 D 11	-.1779 D 11	0	-.1638 D 05	0	0	0	-.2488 D 05	.2458 D 05	0	0	-.4096 D 04	.2048 D 05	0
182 D 11	-.5578 D 11	-.5551 D 11	-.5182 D 11	0	0	0	0	-.1779 D 11	.2349 D 12	-.1807 D 11	.8192 D 04	.2048 D 05	.1638 D 05	.8192 D 04	0	0	0	0	-.1638 D 05	.4096 D 04	.4096 D 04
0	-.5646 D 11	-.5844 D 11	0	0	0	.6397 D 11	-.6593 D 11	0	-.1807 D 11	.2492 D 12	0	.4096 D 04	.1229 D 05	0	0	0	.4096 D 04	.2458 D 05	0	-.4096 D 04	0
0	-.2652 D 11	0	0	0	.2652 D 11	0	0	-.1638 D 05	.4096 D 04	0	.1195 D 12	.7175 D 10	0	0	0	-.7400 D 10	0	0	-.6127 D 11	-.5182 D 11	0
652 D 11	-.2560 D 03	0	0	0	0	.2652 D 11	0	0	-.2048 D 05	-.8192 D 04	.7175 D 10	.1190 D 12	0	0	0	0	-.7763 D 10	0	0	-.5578 D 11	-.5646 D 11
0	0	0	-.2652 D 11	0	0	0	.2652 D 11	0	.2048 D 05	.1229 D05	0	0	.1198 D 12	-.7175 D10	0	0	0	.6679 D 10	0	-.5551 D 11	-.5844 D 11
0	0	.2652 D 11	0	-.2652 D 11	0	0	0	-.4096 D 04	.1638 D 05	0	0	0	-.7175 D 10	.1195 D 12	-.7400 D 10	0	0	0	-.6127 D 11	-.5182 D 11	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	.2926 D 11	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	.3197 D 11	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	.3093 D 11	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	.3283 D 11	0	0	0
1638 D 05	0	0	0	.2458 D 05	-.2458 D 05	0	0	.4096 D 04	.2048 D 05	0	-.6127 D 11	0	0	-.6127 D 11	-.6117 D 11	-.6117 D 11	0	0	.3283 D 12	-.1779 D 11	0
3192 D 04	.2048 D 05	-.1638 D 05	-.8192 D 04	0	0	0	0	.1638 D 05	-.4096 D 04	-.4096 D 04	-.5182 D 11	.5178 D 11	-.5551 D 11	-.5182 D 11	0	0	0	0	-.1779 D 11	.2349 D 12	-.1807 D 11
0	.4096 D 04	-.1229 D 05	0	0	0	.4096 D 04	.2458 D 05	0	.4096 D 04	0	0	.5646 D 11	-.5844 D 11	0	0	0	-.6397 D 11	.6593 D 11	0	-.1807 D 11	.2492 D 12

-.7175 D 10	0	0	0	-.7400 D 10	0	0	-.6127 D 11	-.5182 D 11	0	0	.2652 D 11	0	0	0	-.2652 D 11	0	0	.1638 D 05	-.4096 D 04	0
.1190 D 12	0	0	0	0	.7763 D 10	0	0	-.5578 D 11	-.5646 D 11	-.2652 D 11	.2560 D 03	0	0	0	0	.2652 D 11	0	0	.2048 D 05	.8120 D 04
0	.1198 D 12	-.7175 D 10	0	0	0	-.6679 D 10	0	-.5551 D 11	-.5844 D 11	0	0	0	.2652 D 11	0	0	0	.2652 D 11	0	-.2048 D 05	-.1229 D 05
0	-.7175 D 10	.1195 D 12	-.7400 D 10	0	0	0	-.6127 D 11	-.5182 D 11	0	0	0	-.2652 D 11	0	.2652 D 11	0	0	0	.4096 D 04	-.1638 D 05	0
0	0	0	.2926 D 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	.3197 D 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	.3093 D 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	.3283 D 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	-.6127 D 11	-.6117 D 11	-.6117 D 11	0	0	.2488 D 11	-.1779 D 11	0	-.1638 D 05	0	0	0	-.2458 D 05	.2458 D 05	0	0	-.4096 D 04	.2048 D 05	0
-.5578 D 11	-.5551 D 11	-.5182 D 11	0	0	0	0	-.1779 D 11	.2349 D 12	-.1807 D 11	.8192 D 04	.2048 D 05	.1638 D 05	.8192 D 04	0	0	0	0	-.1638 D 05	.4096 D 04	.4096 D 04
-.5646 D 11	-.5844 D 11	0	0	0	.6397 D 11	-.6593 D 11	0	-.1807 D 11	.2492 D 12	0	.4096 D 04	.1229 D 05	0	0	0	.4096 D 04	.2458 D 05	0	-.4096 D 04	0
-.2652 D 11	0	0	0	.2652 D 11	0	0	-.1638 D 05	.4096 D 04	0	.1195 D 12	.7175 D 10	0	0	0	-.7400 D 10	0	0	-.6127 D 11	-.5182 D 11	0
-.2560 D 03	0	0	0	0	.2652 D 11	0	0	-.2048 D 05	-.8192 D 04	.7175 D 10	.1190 D 12	0	0	0	0	-.7763 D 10	0	0	-.5578 D 11	-.5646 D 11
0	0	-.2652 D 11	0	0	0	.2652 D 11	0	.2048 D 05	.1229 005	0	0	.1198 D 12	-.7175 D10	0	0	0	.6679 D 10	0	-.5551 D 11	-.5844 D 11
0	.2652 D 11	0	-.2652 D 11	0	0	0	-.4096 D 04	.1638 D 05	0	0	0	-.7175 D 10	.1195 D 12	-.7400 D 10	0	0	0	-.6127 D 11	-.5182 D 11	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	.2926 D 11	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	.3197 D 11	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	.3093 D 11	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	.3283 D 11	0	0	0
0	0	0	.2458 D 05	-.2458 D 05	0	0	.4096 D 04	.2048 D 05	0	-.6127 D 11	0	0	-.6127 D 11	-.6117 D 11	-.6117 D 11	0	0	.3283 D 12	-.1779 D 11	0
.2048 D 05	-.1638 D 05	-.8192 D 04	0	0	0	0	.1638 D 05	-.4096 D 04	-.4096 D 04	-.5182 D 11	.5178 D 11	-.5551 D 11	-.5182 D 11	0	0	0	0	-.1779 D 11	.2349 D 12	-.1807 D 11
.4096 D 04	-.1229 D 05	0	0	0	.4096 D 04	.2458 D 05	0	.4096 D 04	0	0	.5646 D 11	-.5844 D 11	0	0	0	-.6397 D 11	.6593 D 11	0	-.1807 D 11	.2492 D 12

Dernière page d'une thèse

VU

Grenoble, le 2 mai 1977

Le Président de la thèse

Vu, et permis d'imprimer,

Grenoble, le

Le Président de l'Université
Scientifique et Médicale