K. JbiLOU

METHODES D'EXTRAPOLATION et DE PROJECTION.
APPLICATIONS AUX SUITES DE VECTEURS.

Exemplaire provisoire

Corrections non encore effectuées

Lille 17 mars 1988

TABLE DES MATIERES

III - Analyse de la convergence.

IV - Applications.

CHAPITRE III : UNE VERSION SIMPLIFIEE DU H-ALGORITHME :

I - Présentation de l'algorithme.

II - Quelques choix des $f_k(n)$

III - Applications

IV - Méthodes de calcul des valeurs et vecteurs propres.

V - Résolution des systèmes linéaires

VI - Exemples numériques.

CHAPITRE IV : ALGORITHME GENERAL DE PROJECTION ET ACCELERATION DE SUITES DE VECTEURS :

I - Présentation et propriétés.

1 - Presentation

2 - Quelques choix des $g_i(n)$

3 - Propriétés.

II - ALgorithmes de calcul pour n fixé

1 - CRPA ; m indépendant de n

2 - RPA ; m quelconque

3 - ALgorithme (II.3)

III - Etude du cas : $g_i(n) = \Delta S_{m+i-1}$; $m = 0$

1 - propriétés

2 - Le $(S-\beta)$ - algorithme.

3 - Règles particulières.

4 - Propriétés de convergence et d'accélération de la convergence.

IV - Applications :

1 - Suites de la forme : $S_n \sim S + \sum_{i=1}^{N} \lambda_i y_i$

2 - Calcul de valeurs et vecteurs propres.

3 - Accélération de la convergence des suites (IV.1).

V - Résolution des systèmes linéaires.

VI - Résolution des systèmes non-linéaires.

CHAPITRE V : ACCELERATION DES METHODES DE PROJECTION-MINIMISATION ET METHODES DE PROJECTION OBLIQUE :

I - Méthodes de projection-minimisation :

1 - Notations et définitions

2 - Présentation.

3 - Algorithme de calcul : RPA.

4 - Propriétés.

II - Transformation des méthodes de projection.

1- Definition du procédé.

2- Propriétés.

3- Accélération de la convergence.

III - Etude du cas $q = 1$:

1- Méthode générale.

2- Méthodes basées sur une décomposition d'une norme.

3- Etude unifiée des méthodes de gradient conjugué et de résidu conjugué.

4- Modification de la méthode (II.3)

IV - Méthodes de projection - oblique:

1°- Présentation.

2- Algorithme de calcul: RPA

3- Méthodes à convergence finie.

- INTRODUCTION -

Ce travail est consacré à l'étude de certains algorithmes récursifs d'extrapolation et de projection dans $\mathbb{C}^p$.

Après avoir rappelé le RPA [5] de C. BREZINSKI, nous proposons dans le premier chapitre deux méthodes directes de résolution des systèmes linéaires basées sur cet algorithme. Nous donnerons ensuite la connection entre la première méthode et la décomposition L.U de GAUSS, ce qui nous permettra de proposer une méthode pour trouver l'inverse d'une matrice symétrique.

Le second chapitre sera consacré à l'étude de la transformation E-vectorielle [4]. Nous étudierons les problèmes de convergence et d'accélération de la convergence et nous appliquerons cette transformation à trois classes de suites de vecteurs.

Dans le troisième chapitre, nous proposerons un algorithme récursif (T) dérivé du H-algorithme [7] qui, dans certains cas, nous permettra de retrouver d'autres algorithmes déjà connus. Nous appliquerons cet algorithme pour accélérer la convergence des suites produites par des itérations linéaires ainsi qu'au calcul

des valeurs et vecteurs propres.

Nous introduisons dans le quatrième chapitre une transformation $S_{k,l}$ vectorielle regroupant la plupart des méthodes connues et nous donnerons des algorithmes de calcul basés sur le RPA et le CRPA [5]. Nous nous intéresserons ensuite à une transformation particulière résultant de $S_{k,l}$ dont on donnera un algorithme de calcul appelé le $(S-\beta)$ ainsi que des propriétés algébriques. Le dernier sera étudié plus profondément et appliqué ensuite au calcul des éléments propres d'une matrice et à la résolution des systèmes linéaires et non-linéaires.

Le chapitre V sera consacré à l'accélération des méthodes de projection-minimisation. Après la présentation générale de ces méthodes nous donnerons quelques propriétés et un algorithme de calcul.

Nous définirons un procédé qui associera à chaque méthode de projection-minimisation sa transformée et nous montrerons que cette dernière converge au moins aussi vite que la méthode originale. Nous retrouvons par ce procédé des méthodes très connues comme le gradient conjugué ou le résidu conjugué généralisé. Nous montrerons enfin dans le dernier paragraphe que la plupart des méthodes d'extrapolation sont des méthodes de projection-oblique, lorsqu'elles sont appliquées aux suites produites par des itérations linéaires.

- Notations et definitions -

Soient (u_n) et (v_n) deux suites de nombres réels convergent vers zéro.

Definition 01 :

Si pour tout $\varepsilon > 0$, il existe un entier N tel que pour tout $n > N$ on ait : $|V_n| < \varepsilon . |u_n|$, alors on écrit : $V_n = o(u_n)$

Theorème 01 [3] :

S'ils existent un entier N et deux réels a et b verifiant : $a < 1 < b$ tel que pour tout $n > N$, on ait : $\frac{u_{n+1}}{u_n} \notin [a, b]$ et si $\lim_{n \to \infty} \frac{V_n}{u_n} = c$ alors $\lim_{n \to \infty} \frac{\Delta V_n}{\Delta u_n} = c$

Nous rappelons que $\Delta u_n = u_{n+1} - u_n$.

Chaque fois que nous appliquerons l'opérateur Δ à une suite $T_K^{(n)}$, ça sera sur l'indice n (l'indice superieur).

Soient maintenant (u_n) et (v_n) deux suites de $\mathbb{C}^p$ et soit $L = \{\phi_1, ---, \phi_K\}$ un ensemble de k fonctionnelles linéaires bornées sur $\mathbb{C}^p$, alors :

le symbole $u_n \sim v_n$ signifie que $\forall \phi \in L$:

$$\langle \phi, u_n \rangle \sim \langle \phi, v_n \rangle$$

<u>Théorème 02</u>:

si $k = p$ et si $\phi_1, \dots, \phi_p$ sont linéairement indépendants alors, $u_n \sim v_n \Longrightarrow \|u_n\| \sim \|v_n\|$

<u>dem</u>:

$u_n \sim v_n \Longrightarrow \langle \phi, u_n \rangle \sim \langle \phi, v_n \rangle$ pour tout $\phi \in L$. Et on montre dans ce cas que: $\|u_n - v_n\| = o(\|v_n\|)$, [p54]. Or :

$$\frac{1}{\|v_n\|} \left| \|u_n\| - \|v_n\| \right| \leq \frac{\|u_n - v_n\|}{\|v_n\|} \xrightarrow[n \to \infty]{} 0$$

donc:

$$\lim_{n \to \infty} \frac{\|u_n\|}{\|v_n\|} = 1 \quad , \text{ soit } \quad \|u_n\| \sim \|v_n\| . \quad \square$$

<u>Definition 03</u>:

Soient (x_n) et (y_n) deux suites de $\mathbb{C}^p$ convergeant vers x^* et y^*.

i) On dit que (x_n) converge plus vite que (y_n) par rapport à la fonctionnelle linéaire ϕ, sur $\mathbb{C}^p$, si:

$$\lim_{n \to \infty} \frac{\langle \phi, x_n - x^* \rangle}{\langle \phi, y_n - y^* \rangle} = 0$$

ii) On dit que (x_n) converge plus vite que (y_n) par rapport à la norme $\| \ \|$, si:

$$\lim_{n \to \infty} \frac{\|x_n - x^*\|}{\|y_n - y^*\|} = 0 \quad : \text{ convergence forte.}$$

- Chapitre I -

APPLICATION DU RPA A LA RESOLUTION DES SYSTEMES LINEAIRES ET AUX TRANSFORMATIONS DE SUITES DE VECTEURS

Introduction :

Après un rappel du RPA, nous donnerons deux règles particulières qui permettent d'éviter les singularités. Nous proposerons ensuite deux méthodes (A) et (B) de résolution des systèmes linéaires basées sur le RPA. Nous donnerons la connection entre la méthode (A) et la décomposition LU de Gauss, ce qui nous permettra de proposer un algorithme pour l'inverse d'une matrice symétrique.

Nous appliquerons enfin la méthode (A) aux transformations de suites de vecteurs et nous expliquerons l'intérêt d'une telle utilisation.

- RPA et règles particulières :

I.1 - RPA :

Soit E un espace vectoriel sur K ($\mathbb{R}$ ou $\mathbb{C}$) et E^* son dual. $y \in E$, $x_i \in E$ et $z_i \in E^*$.

on pose,

$$N_K = \begin{vmatrix} y & x_1 & \cdots & x_K \\ \langle z_1, y\rangle & \langle z_1, x_1\rangle & \cdots & \langle z_1, x_K\rangle \\ \cdots & \cdots & \cdots & \cdots \\ \langle z_K, y\rangle & \langle z_K, x_1\rangle & \cdots & \langle z_K, x_K\rangle \end{vmatrix}$$

$$D_K = \begin{vmatrix} \langle z_1, x_1\rangle & \cdots & \langle z_1, x_K\rangle \\ \cdots & \cdots & \cdots \\ \langle z_K, x_1\rangle & \cdots & \langle z_K, x_K\rangle \end{vmatrix}$$

$$N_{K,i} = \begin{vmatrix} x_i & x_1 & \cdots & x_K \\ \langle z_1, x_i\rangle & \langle z_1, x_1\rangle & \cdots & \langle z_1, x_K\rangle \\ \cdots & \cdots & \cdots & \cdots \\ \langle z_K, x_i\rangle & \langle z_K, x_1\rangle & \cdots & \langle z_K, x_K\rangle \end{vmatrix}$$

On pose :

$$E_k = N_k / D_k \quad \text{et} \quad g_{k,i} = N_{k,i} / D_k$$

On suppose que $\forall k$, $D_k \neq 0$.

Il a été montré dans [10] que E_k peut être calculé récursivement par l'algorithme suivant :

$$(1.1) \quad \left\| \begin{array}{ll} E_0 = y \quad ; \; g_{0,i} = x_i & i \geq 1 \\ E_k = E_{k-1} - \dfrac{\langle z_k, E_{k-1} \rangle}{\langle z_k, g_{k-1,k} \rangle} \cdot g_{k-1,k} \; , & k > 0 \\ g_{k,i} = g_{k-1,i} - \dfrac{\langle z_k, g_{k-1,i} \rangle}{\langle z_k, g_{k-1,k} \rangle} \cdot g_{k-1,k} \; , & i > k > 0 \end{array} \right.$$

Cet algorithme dû à C. BREZINSKI [5] a été appelé : "recursif projection algorithm" et en abrégé : RPA.

Le calcul de E_k par le RPA nécessite, dans le cas où $E = \mathbb{C}^N$:

* le stockage de $2k+1$ vecteurs de $\mathbb{C}^N$.
* $\frac{k}{2}(k+5)$ produits scalaires.
* $\frac{N.k}{2}(k+3)$ multiplications.
* $\frac{N.k}{2}(k+3)$ additions et soustractions.
* $2Nk$ divisions.

Soit un total de l'ordre de $2Nk^2$, lorsque N est grand devant k.

1.2 – Polynôme d'interpolation généralisée :

Considérons le problème suivant :

On cherche les $a_i \in \mathbb{K}$ tels que :

$$(2.1)\quad \begin{cases} P_k = a_1 x_1 + \cdots + a_k x_k \\ \langle z_j, P_k \rangle = \omega_j \ ; \ j = 1, \ldots, k. \end{cases}$$

où les $x_i \in E$, et les $\omega_j \in \mathbb{K}$.

Lorsque $\omega_j = \langle z_j, y \rangle$, il a été montré dans [5] que $P_k = y - E_k$ est l'unique solution du problème (2.1). P_k se calcule donc récursivement à partir de (1.1) par :

$$(\underline{RIA})\quad \left\| \begin{array}{l} P_0 = 0 \\ P_k = P_{k-1} + \dfrac{\omega_k - \langle z_k, P_{k-1} \rangle}{\langle z_k, g_{k-1,k} \rangle} \cdot g_{k-1,k} \qquad ; k \geq 1. \end{array} \right.$$

où les $g_{k,i}$ se calculent par (1.1).

Cet algorithme sera appelé le RIA : "recursif interpolation algorithm".

Nous allons donner tout d'abord deux règles particulières pour le RPA et le RIA qui nous permettront de continuer les calculs lorsqu'on rencontre des singularités : dénominateur nul ou très voisin de zéro.

I.3 - Règles particulières :

propriété 1 :

$$E_{k+m} = \frac{\begin{vmatrix} E_k & g_{k,k+1} & \cdots & g_{k,k+m} \\ \langle z_{k+1}, E_k\rangle & \langle z_{k+1}, g_{k,k+1}\rangle & \cdots & \langle z_{k+1}, g_{k,k+m}\rangle \\ \hline \langle z_{k+m}, E_k\rangle & \langle z_{k+m}, g_{k,k+1}\rangle & \cdots & \langle z_{k+m}, g_{k,k+m}\rangle \end{vmatrix}}{\begin{vmatrix} \langle z_{k+1}, g_{k,k+1}\rangle & \cdots & \langle z_{k+1}, g_{k,k+m}\rangle \\ \hline \langle z_{k+m}, g_{k,k+1}\rangle & \cdots & \langle z_{k+m}, g_{k,k+m}\rangle \end{vmatrix}}$$

Dém. analogue à celle faite dans [6] pour le E-algorithme scalaire. □

Nous avons la même expression pour $g_{k+m,i}$ en remplaçant dans le numérateur E_k par $g_{k,i}$.

Si $m=1$, nous obtenons l'algorithme (1.1)

Si $k=0$, nous retrouvons l'expression initiale de $E_m = N_m / D_m$.

Et en appliquant la formule de Schur [8], nous avons :

$$E_{k+m} = E_k - \left(g_{k,k+1}, \cdots, g_{k,k+m}\right) \cdot \begin{pmatrix} \langle z_{k+1}, g_{k,k+1}\rangle & \cdots & \langle z_{k+1}, g_{k,k+m}\rangle \\ \hline \langle z_{k+m}, g_{k,k+1}\rangle & \cdots & \langle z_{k+m}, g_{k,k+m}\rangle \end{pmatrix}^{-1} \cdot \begin{pmatrix} \langle z_{k+1}, E_k\rangle \\ \vdots \\ \langle z_{k+m}, E_k\rangle \end{pmatrix}$$

et une expression analogue pour $g_{k+m,i}$ en remplaçant E_k par $g_{k,i}$.

Propriété 2 :

$$P_{k+m} = \frac{\begin{vmatrix} P_k & g_{k,k+1} & \cdots & g_{k,k+m} \\ \omega_{k+1} - \langle z_{k+1}, P_k \rangle & \langle z_{k+1}, g_{k,k+1} \rangle & \cdots & \langle z_{k+1}, g_{k,k+m} \rangle \\ \cdots & \cdots & \cdots & \cdots \\ \omega_{k+m} - \langle z_{k+m}, P_k \rangle & \langle z_{k+m}, g_{k,k+1} \rangle & \cdots & \langle z_{k+m}, g_{k,k+m} \rangle \end{vmatrix}}{\begin{vmatrix} \langle z_{k+1}, g_{k,k+1} \rangle & \cdots & \langle z_{k+1}, g_{k,k+m} \rangle \\ \cdots & \cdots & \cdots \\ \langle z_{k+m}, g_{k,k+1} \rangle & \cdots & \langle z_{k+m}, g_{k,k+m} \rangle \end{vmatrix}}$$

dem :

$$P_{k+m} = y - E_{k+m}$$

$$= -\frac{1}{D_{k,m}} \cdot \begin{vmatrix} P_k & g_{k,k+1} & \cdots & g_{k,k+m} \\ \langle z_{k+1}, E_k \rangle & \langle z_{k+1}, g_{k,k+1} \rangle & \cdots & \langle z_{k+1}, g_{k,k+m} \rangle \\ \cdots & \cdots & \cdots & \cdots \\ \langle z_{k+m}, E_k \rangle & \langle z_{k+m}, g_{k,k+m} \rangle & \cdots & \langle z_{k+m}, g_{k,k+m} \rangle \end{vmatrix}$$

Or : $E_k = y - P_k$

$$\langle z_{k+i}, E_k \rangle = \langle z_{k+i}, y \rangle - \langle z_{k+i}, P_k \rangle$$

$$= \omega_{k+i} - \langle z_{k+i}, P_k \rangle \qquad i = 1, \ldots, m \quad \blacksquare$$

En appliquant la formule de Schur [8], nous obtenons :

$$P_{k+m} = P_k - \left(g_{k,k+1}, \ldots, g_{k,k+m} \right) \cdot \begin{bmatrix} \langle z_{k+1}, g_{k,k+1} \rangle & \cdots & \langle z_{k+1}, g_{k,k+m} \rangle \\ \cdots & \cdots & \cdots \\ \langle z_{k+m}, g_{k,k+1} \rangle & \cdots & \langle z_{k+m}, g_{k,k+m} \rangle \end{bmatrix}^{-1} \begin{bmatrix} \omega_{k+1} - \langle z_{k+1}, P_k \rangle \\ \vdots \\ \omega_{k+m} - \langle z_{k+m}, P_k \rangle \end{bmatrix}$$

Les deux règles particulières du RPA et du RIA nous permettent d'éviter les singularités que l'on peut rencontrer lors du calcul de E_k et de P_k.

I. Méthodes directes de résolution des systèmes linéaires :

Soit à résoudre le système régulier d'ordre n suivant :

$$(\text{II}.1) \quad A\,x = b$$

où :

$$A = [a_{ij}]_{\substack{1 \le i \le n \\ 1 \le j \le n}} \quad ; \quad b = (b_1, \dots, b_n)^T$$

et

$x = (x_1, \dots, x_n)^T$ est la solution de (II.1).

Nous proposerons par la suite deux méthodes directes pour résoudre (II.1), basées sur le RPA. Nous donnerons la connection entre la première et la décomposition LU de Gauss, ce qui nous permettera de proposer une méthode pour inverser une matrice symétrique.

II.1 Méthode (A):

On considère le système (II.1) et on pose:

$$z_i = (a_{i1}, \dots, a_{in}, -b_i)$$

$$z_i^T \in \mathbb{R}^{n+1}; \quad i = 1, \dots, n.$$

Reprenons l'expression de $g_{k,i}$, avec:

$$x_i = e_i \; ; i = 1, \dots, n+1$$

où $\{e_i\}_{i=1,\dots,n+1}$ est la base canonique de $\mathbb{R}^{n+1}$.

$$g_{k,i} = \begin{vmatrix} e_i & e_1 & \dots & e_k \\ \langle z_1, e_i \rangle & \langle z_1, e_1 \rangle & \dots & \langle z_1, e_k \rangle \\ - & - & - & - \\ \langle z_k, e_i \rangle & \langle z_k, e_1 \rangle & \dots & \langle z_k, e_k \rangle \end{vmatrix} / D_k .$$

$g_{k,i} \in \mathbb{R}^{n+1}$ et s'écrit sous la forme:

$$g_{k,i} = e_i + \alpha_{1i} e_1 + \dots + \alpha_{ki} e_k .$$

et comme:

$$\langle z_j, g_{k,i} \rangle = 0 \quad ; j = 1, \dots, k$$

nous obtenons le système:

$$(\mathrm{II}.2)\quad \begin{cases} a_{11}\alpha_{1i} + \cdots + a_{1k}\alpha_{ki} = -a_{1i} \\ a_{21}\alpha_{1i} + \cdots + a_{2k}\alpha_{ki} = -a_{2i} \\ \cdots\cdots\cdots\cdots\cdots\cdots \\ a_{k1}\alpha_{1i} + \cdots + a_{kk}\alpha_{ki} = -a_{ki} \end{cases} \quad ; i=k+1,\dots,n+1$$

avec:

$$a_{j,n+1} = -b_j \quad ; j = 1,\dots,n.$$

posons :

$$\varphi_{k,i} = (\alpha_{1i},\dots,\alpha_{ki})^T \in \mathbb{R}^k \quad ; i=k+1,\dots,n+1.$$

donc:

$$g_{k,i} = \begin{pmatrix} \varphi_{k,i} \\ 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \leftarrow i^e \text{ composante.}$$

Le système (II.2) s'écrit : $A_k.\varphi_{ki} = -(a_{1i},\dots,a_{ki})^T$

où $A_k = [a_{ij}]_{\substack{1\le i\le k \\ 1\le j\le k}}$.

Lorsque $k=n$; $i=n+1$, le problème (II.2) n'est autre que le problème initial (II.1) et on a:

$$\varphi_{n,n+1} = \mathcal{X}$$

et

$$g_{n,n+1} = \begin{pmatrix} \mathcal{X} \\ 1 \end{pmatrix}$$

qui s'obtient récursivement par l'algorithme :

$$\text{II.3)} \quad \left\| \begin{array}{ll} g_{0,i} = e_i & ; i=1,---,n+1. \\ \\ g_{k,i} = g_{k-1,i} - \dfrac{\langle z_k, g_{k-1,i}\rangle}{\langle z_k, g_{k-1,k}\rangle} \cdot g_{k-1,k} & ; k=1,---,n \\ & \quad i=k+1,---,n+1. \end{array} \right.$$

La résolution du système (II.1) par l'algorithme précédent nécéssite le stockage de $\frac{3n^2+n}{2}$ scalaires, $\frac{n(n+1)}{3}.\left(n-\frac{5}{2}\right)$ multiplications et $\frac{n(n+1)}{3}.\left(n-\frac{3}{2}\right)$ additions et soustractions, soit lorsque n est grand un total d'opérations arithmétiques de l'ordre de $\frac{2}{3}n^3$.

Nous allons donner, maintenant, la connection entre méthode (A) et la décomposition L.U de Gauss et ceci nous permettera de donner un algorithme d'inversion d'une matrice symétrique.

2 - Connection de (A) avec la décomposition L.U :

Soit G la matrice d'ordre n dont les colonnes sont formées par les n premières composantes des vecteurs $g_{k-1,k}$; $k=1,---,n$, que l'on suppose exister. Nous verrons plus loin la condition sur A pour que cette condition soit vérifiée.

Alors:

$$G = \begin{pmatrix} 1 & \alpha_{12} & \alpha_{13} & \cdots & \alpha_{1n} \\ & 1 & \alpha_{23} & \cdots & \alpha_{2n} \\ & & 1 & \cdots & \alpha_{3n} \\ & 0 & & \ddots & \vdots \\ & & & & 1 \end{pmatrix}$$

G est triangulaire supérieure à diagonale unité.

Posons: $C = A.G$

alors,

$$C = \begin{pmatrix} a_{11} & a_{11}\alpha_{12}+a_{12} & \cdots & a_{11}\alpha_{1n}+a_{12}\alpha_{2n}+\cdots+a_{1n} \\ a_{12} & a_{21}\alpha_{12}+a_{22} & \cdots & a_{21}\alpha_{1n}+a_{22}\alpha_{2n}+\cdots+a_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{1n} & a_{n1}\alpha_{12}+a_{n2} & \cdots & a_{n1}\alpha_{1n}+a_{n2}\alpha_{2n}+\cdots+a_{nn} \end{pmatrix}$$

Or d'après (II.2) nous avons :

$$a_{j1}\alpha_{1i}+a_{j2}\alpha_{2i}+\cdots+a_{jk}\alpha_{ki}+a_{ji}=0 \quad ; \; i>j>0 \; ; \; k=1,\ldots,n.$$

C est donc triangulaire inférieure dont les termes non nuls sont :

$$c_{pq} = a_{p1}\alpha_{1q}+a_{p2}\alpha_{2q}+\cdots+a_{p,q-1}\alpha_{q-1,q}+a_{pq} \quad ; \; p \geq q$$

$$C = AG \implies A = C.G^{-1}$$

où

$B = G^{-1}$ est triangulaire supérieure à diagonale unité, la décomposition $A = C.B$ est donc unique.

Posons:

$D = \mathrm{diag}(c_{11}, \dots, c_{nn})$ et supposons que $c_{jj} \neq 0$, $j = 1, \dots, n$.

Soit H la matrice triangulaire inférieure à diagonale unité définie par: $h_{ij} = \dfrac{c_{ij}}{c_{jj}}$; $i > j$.

alors

$$A = H.DB$$

si l'on pose $L = H$ et $U = DB$, nous retrouvons la décomposition LU de Gauss.

Considerons le cas où A est symétrique:

$$A = HDB \implies A^T = B^T D H^T$$

donc:

$$A = B^T D H^T$$

or H^T est triangulaire supérieure à diagonale unité et $B^T D$ est triangulaire inferieure. Mais comme la décomposition $A = C.B$ est unique, alors:

$$B = H^T \quad \text{soit} \quad G^{-1} = H^T$$

donc:

$$A = H D H^T$$

et $A^{-1} = (H^T)^{-1} . D^{-1} . H^{-1}$ et comme $H^T = G^{-1}$, alors:

$$(\text{I}.4) \quad \boxed{A^{-1} = G . D^{-1} . G^T}$$

Le calcul de A^{-1} par cette méthode nécessite, lorsque n est grand, $\frac{n^3}{2}$ multiplications et $\frac{n^3}{2}$ additions.

Nous avons signalé que cette méthode n'est valable que si les $g_{k-1,k}$, $k=1,\dots,n$, existent, soit si :

$$\begin{vmatrix} a_{11} & \cdots & a_{1k} \\ \vdots & & \vdots \\ a_{k1} & \cdots & a_{kk} \end{vmatrix} \neq 0 \quad \text{pour } k=1,\dots,n.$$

<u>Remarque</u>:

Après que ce travail ait été réalisé, nous avons retrouvé un article de F. SLOBODA [25] dans lequel cette méthode (I.4) a été décrite, en partant d'un algorithme similaire au RPA, et où il a été donné une méthode de résolution des systèmes linéaires identique à la méthode de bordage.

<u>Exemples numériques</u>:

1°) Soit A la matrice d'ordre 4 donnée par:

$$A = \begin{pmatrix} 10 & 9 & 8 & 7 \\ 9 & 8 & -3 & -4 \\ 8 & -3 & 10 & 0 \\ 7 & -4 & 0 & -10 \end{pmatrix}$$

A est une matrice très mal-conditionnée, dont l'inverse est.

$$A^{-1} = \begin{pmatrix} -870 & 860 & 954 & -953 \\ 860 & -850 & -943 & 942 \\ 954 & -943 & -1046 & 1045 \\ -953 & 942 & 1045 & -1044 \end{pmatrix}$$

Le conditionnement de A est : $\mathrm{Cond}A = \|A\|_\infty . \|A^{-1}\|_\infty = 135592$

L'algorithme (II.4) nous donne les résultats suivants :

MATRICE G

[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]

MATRICE C = A.G

[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]

MATRICE INVERSE

[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]

$A * A^{-1}$

[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]

2°) Soit A la matrice d'ordre 4 definie par :

$$A = \begin{pmatrix} 1 & 0.42 & 0.54 & 0.66 \\ 0.42 & 1 & 0.32 & 0.44 \\ 0.54 & 0.32 & 1 & 0.22 \\ 0.66 & 0.44 & 0.22 & 1 \end{pmatrix}$$

L'algorithme (II.4) nous donne les résultats suivants :

MATRICE G

[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]

MATRICE C = A.G

[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]

MATRICE INVERSE

[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]

$A * A^{-1}$

[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]

3 - Méthode (B) :

On considère le même système (II.1). Et soit $\{e_i\}_{i=1,\dots,n}$ la base canonique de $\mathbb{R}^n$.

La solution x du système (II.1) s'écrit sous la forme:

$$x = x_1 e_1 + \dots + x_n e_n$$

où $x_1, \dots, x_n$ vérifient:

$$(3.1) \quad \begin{cases} a_{11} x_1 + \dots + a_{1n} x_n = b_1 \\ \vdots \qquad\qquad \vdots \qquad \vdots \\ a_{n1} x_1 + \dots + a_{nn} x_n = b_n \end{cases}$$

La méthode (B) consiste à résoudre à chaque étape k, $k = 1, \dots, n$, le système d'ordre k :

$$(3.2) \qquad A_k . X_k = B_k$$

où :

$$A_k = [a_{ij}]_{\substack{1 \le i \le k \\ 1 \le j \le k}} \quad ; \quad B_k = (b_1, \dots, b_k)^T$$

Posons:

$$X_k = (x_1^{(k)}, \dots, x_k^{(k)})^T \quad ; \quad P_k = (x_1^{(k)}, \dots, x_k^{(k)}, 0 \dots, 0)^T \in \mathbb{R}^n$$

et :

$$A_{ik} = (a_{i1}, \dots, a_{ik}, 0 \dots, 0) \quad , \quad i = 1, \dots, k.$$

(3.2) est alors équivalent à :

$$\langle A_{ik}, P_k \rangle = b_i \quad ; i = 1, ---, k$$

Résoudre (3.2) revient donc à résoudre le problème d'interpolation suivant :

$$(3.3) \begin{cases} P_k = x_1^{(k)} e_1 + --- + x_k^{(k)} e_k \\ \langle A_{ik}, P_k \rangle = b_i \quad ; i = 1, ---, k \end{cases}$$

Et P_k se calcule récursivement par le RIA :

$$\boxed{\begin{array}{l} P_0 = 0 \quad ; \; g_{0,i} = e_i \quad ; i = 1, ---, n \\[2ex] P_k = P_{k-1} + \dfrac{b_k - \langle A_{kk}, P_{k-1} \rangle}{\langle A_{kk}, g_{k-1,k} \rangle} \cdot g_{k-1,k} \quad ; k = 1, ---, n \\[2ex] g_{k,i} = g_{k-1,i} - \dfrac{\langle A_{kk}, g_{k-1,i} \rangle}{\langle A_{kk}, g_{k-1,k} \rangle} \cdot g_{k-1,k} \; ; \quad i > k > 0 \end{array}}$$

Lorsque $k = n$, le problème (3.3) est équivalent au problème initial (II.1) et $P_n = X$.

Si à l'étape k, A_k est singulière, le calcul de P_k par (3.4) est impossible. Dans ce cas on calculera P_{k+1} directement à partir de P_{k-1} en utilisant la règle particulière du (RIA).

La résolution de (II.1) par (B) nécessite le stockage de $2n^2+3n$ scalaires, $\frac{1}{3}(n^3+n+2)$ multiplications et autant d'additions et de soustractions.

I.4 – Connection avec la méthode de Bordage:

On considère le système régulier d'ordre n suivant:

$$(4.1) \qquad A_n . X_n = F_n$$

$$F_n = (F_{n-1}, f_n)^T \quad \text{avec} \quad F_{n-1} = (f_1, ---, f_{n-1})$$

La solution $X_n = P_n$ de (4.1) par (B) est donnée par:

$$(4.2) \qquad P_n = P_{n-1} + \frac{f_n - \langle A_{nn}, P_{n-1} \rangle}{\langle A_{nn}, g_{n-1,n} \rangle} \cdot g_{n-1,n}$$

avec:

$$P_{n-1} = \begin{pmatrix} X_{n-1} \\ 0 \end{pmatrix} \; ; \; A_{nn} = (a_{n1}, ---, a_{nn})$$

et

$$g_{n-1,n} = e_n + \alpha_{1n} e_1 + --- + \alpha_{n-1,n} e_{n-1}$$

Posons:

$$Q_{n-1} = (\alpha_{1n}, ---, \alpha_{n-1,n})^T \text{ et } v_n = (a_{n1}, ---, a_{n,n-1})$$

alors d'après la méthode (A), nous avons:

$$A_{n-1} . Q_{n-1} = -(a_{1n}, ---, a_{n-1,n})^T$$

d'autre part, comme:

$$g_{n-1,n} = (Q_{n-1}^T, 1)^T \text{ et } A_{nn} = (v_n, a_{nn}) \text{, alors :}$$

$$\langle A_{nn}, P_{n-1} \rangle = \langle v_n, X_{n-1} \rangle \text{ qu'on notera } v_n . X_{n-1}$$

et:

$$\langle A_{nn}, g_{n-1,n} \rangle = v_n . Q_{n-1} + a_{nn}$$

Et en reportant dans (4.2), nous obtenons :

$$X_n = \begin{pmatrix} X_{n-1} \\ 0 \end{pmatrix} + \frac{f_n - v_n \cdot X_{n-1}}{a_{nn} + v_n \cdot Q_{n-1}} \cdot \begin{pmatrix} Q_{n-1} \\ 1 \end{pmatrix}$$

Ce n'est autre que l'expression obtenue par la méthode de bordage [14, page 167].

<u>Remarque</u> : La méthode (A) est plus interessante que la méthode (B). Elle est tout d'abord "plus" économique : $\frac{1}{3}\left(n^3 - \frac{3}{2}n^2 - \frac{5}{2}\right)$ multiplications et le stockage de $\frac{3}{2}n(n+1)$ scalaires pour la méthode (A) contre $\frac{1}{3}(n^3+n^2+2n)$ multiplications et le stockage de $2n(n+1)$ scalaires. Mais elle est surtout plus stable numériquement comme nous le verrons sur des exemples numériques.

<u>Exemples numériques</u> :

Nous avons pris comme exemple une matrice mal-conditionnée : la matrice de Hilbert d'ordre 8 :

$a_{ij} = 1/(i+j-1)$; $i,j = 1, \dots, 8$.

Le second membre est choisit de telle sorte que la solution soit le vecteur dont chaque composante est 1.

Les méthodes (A), (B) et Gauss nous donnent les résultats suivants :

GAUSS		METHODE (B)		METHODE (A)
=.99999999950+00		X1=.99999999950+00		X1=.99999999990+00
=.10000000025D+01		X2=.10000000025D+01		X2=.10000000003D+01
=.99999987940+00		X3=.99999987940+00		X3=.99999995180+00
=.10000001730D+01		X4=.10000001730D+01		X4=.10000000271D+01
=.99999953700+00		X5=.99999953700+00		X5=.99999992510D+00
=.10000000594D+01		X6=.10000000594D+01		X6=.10000001076D+01
=.99999953102D+00		X7=.99999953102D+00		X7=.99999992240D+00
=.10000001323D+01		X8=.10000001323D+01		X8=.10000000220D+01

On voit que (B) et Gauss nous donnent les mêmes résultats alors que la méthode (A) nous fait gagner un chiffre en plus sur chaque composante.

III. Application aux transformations de suites :

Considerons le problème d'interpolation généralisée suivant :

$$\begin{cases} P_k^{(n)} = a_0 x_0^{(n)} + \dots + a_k x_k^{(n)} \\ \langle z_i, P_k^{(n)} \rangle = \omega_i \qquad i = 0, \dots, k. \end{cases}$$

où $x_i^{(n)} \in E$; $z_i \in E^*$ et $a_i \in K$

avec : $\omega_0 = 1$; $\omega_i = 0$ pour $i \geq 1$

$\langle z_0, x_i^{(n)} \rangle = b_i^{(n)} \qquad i \geq 0$

$\langle z_i, x_j^{(n)} \rangle = c_{ij}^{(n)} \qquad ; i \geq 1 \quad ; j \geq 0$

Les a_i sont donc solution du système :

$$\begin{cases} a_0 b_0^{(n)} + \dots + a_k b_k^{(n)} = 1 \\ a_0 c_{10}^{(n)} + \dots + a_k c_{1k}^{(n)} = 0 \\ \dots\dots\dots\dots\dots\dots \\ a_0 c_{k0}^{(n)} + \dots + a_k c_{kk}^{(n)} = 0 \end{cases}$$

Le système sera résolu, pour chaque n fixé, par la méthode (A) du paragraphe II.

$P_k^{(n)}$ peut être écrit sous-forme d'un rapport de deux déterminants :

$$P_k^{(n)} = \frac{\begin{vmatrix} x_0^{(n)} & x_1^{(n)} & \cdots & x_k^{(n)} \\ c_{10}^{(n)} & c_{11}^{(n)} & \cdots & c_{1k}^{(n)} \\ \cdots & \cdots & \cdots & \cdots \\ c_{k0}^{(n)} & c_{k1}^{(n)} & \cdots & c_{kk}^{(n)} \end{vmatrix}}{\begin{vmatrix} b_0^{(n)} & b_1^{(n)} & \cdots & b_k^{(n)} \\ c_{10}^{(n)} & c_{11}^{(n)} & \cdots & c_{1k}^{(n)} \\ \cdots & \cdots & \cdots & \cdots \\ c_{k0}^{(n)} & c_{k1}^{(n)} & \cdots & c_{kk}^{(n)} \end{vmatrix}} \quad ; k \geq 0$$

La plus part des transformations de suites vectorielles sont de la forme précédente. On prends le cas où $E = \mathbb{C}^p$.

- E-algorithme topologique [4], avec :

$x_0^{(n)} = S_n$; $x_i^{(n)} = g_i(n)$; $b_0^{(n)} = 1$ et $b_i^{(n)} = 0$ pour $i \geq 1$

$c_{ij}^{(n)} = \langle y, \Delta g_j(n+i-1) \rangle$; $c_{i0}^{(n)} = \langle y, \Delta S_{n+i-1} \rangle$

- Algorithme générale de projection (chapitre IV) :

$x_0^{(n)} = S_n$; $x_i^{(n)} = g_i(n)$, $b_0^{(n)} = 1$ et $b_i^{(n)} = 0$; $i \geq 1$

$c_{i0}^{(n)} = \langle \phi_i, \Delta S_n \rangle$ et $c_{ij}^{(n)} = \langle \phi_i, \Delta g_j(n) \rangle$; $j \geq 1$; $i \geq 1$

où $\{\phi_1, \ldots, \phi_k\}$ est un système libre de E^*.

- MMPE [26], [22] :

$x_i^{(n)} = S_{n+i}$; $i \geq 0$, $b_i^{(n)} = 1$

$c_{ij}^{(n)} = \langle \phi_i, \Delta S_{n+j} \rangle$ $i \geq 1$ et $j \geq 0$

. MPE [25], [12] :

$$x_i^{(n)} = S_{n+i} \quad ; \; b_i^{(n)} = 1 \qquad i \geq 0$$

$$c_{ij}^{(n)} = \langle \Delta S_{n+i-1}, \Delta S_{n+j} \rangle \quad ; \; i \geq 1 \; ; \; j \geq 0$$

. RRE [25], [23] :

$$x_i^{(n)} = S_{n+i} \quad ; \; b_i^{(n)} = 1 \qquad i \geq 0$$

$$c_{ij}^{(n)} = \langle \Delta^2 S_{n+i-1}, \Delta S_{n+j} \rangle \quad ; \; i \geq 1 \; ; \; j \geq 0$$

$P_k^{(n)}$ est placé dans un tableau à double entrée où k désigne le numéro d'une colonne et n celui d'une diagonale descendante :

$$\begin{array}{llll}
P_0^{(0)} & & & \\
P_0^{(1)} & P_1^{(0)} & & \\
P_0^{(2)} & P_1^{(1)} & P_2^{(0)} & \\
P_0^{(3)} & P_1^{(2)} & P_2^{(1)} & P_3^{(0)} \\
\vdots & \vdots & \vdots & \vdots \quad \ddots
\end{array}$$

Dans beaucoup de problèmes (résolution des systèmes linéaires et non linéaires) on n'a pas besoin de toutes les quantités $P_k^{(n)}$ mais uniquement de la diagonale descendante $P_k^{(0)}$ ou $P_k^{(r)}$, r connu. La méthode (A) nous permet de calculer cette diagonale en calculant chaque terme sur celui-ci à partir du terme précédent.

Si, ensuite nous avons besoin des autres quantités $P_k^{(n)}$, on les calculera à partir de la première diagonale descendante: $P_0^{(0)}, P_1^{(0)}$--- en utilisant les formes progressives de chaque algorithme. Cette manière de procéder permet d'éviter l'instabilité numérique souvent rencontrée lors de l'implémentation des algorithmes par leurs formes normales (voir [9]).

Si sur une diagonale descendante on ne peut pas calculer une certaine quantité, alors on passe directement à la suivante en utilisant la dernière calculée sur cette diagonale. Supposons par exemple que $P_i^{(0)}$ non calculable: dénominateur très voisin de zéro, alors la méthode (A) nous permet de calculer $P_{i+1}^{(0)}$ à partir de $P_{i-1}^{(0)}$ en utilisant les règles particulières du RIA.

Le calcul de $P_p^{(0)}$ par la méthode (A) nécessite:

$\frac{1}{3}p(p-1)(p-\frac{5}{2})$ multiplications et $\frac{1}{3}p(p+1)(p-\frac{3}{2})$ additions. Et il faudra ajouter à ceci le nombre d'opérations nécessaires pour la construction de la matrice du système correspondant.

A titre de comparaison, le calcul de $P_p^{(0)}$ par le H-algorithme, lorsque $c_{ij}^{(n)} = f_i(n+j)$ et $b_i^{(n)} = 1$, nécessite $\frac{5}{6}p^3$ multiplications et $\frac{5}{3}p^3$ additions. On voit donc que l'utilisation de la méthode (A), lorsqu'il s'agit de ne calculer qu'une diagonale descendante, est plus économique.

- Chapitre II -

Etude de la transformation E vectorielle

Introduction :

Nous nous consacrerons ce chapitre à l'étude de la convergence et de l'accélération de la convergence de la transformation E-vectorielle de C. BREZINSKI [4]. Le résultat essentiel sera donné dans le théorème 1, ce qui nous permettra d'appliquer cette transformation à certaines classes de suites de vecteurs. Ceci nous amènera à proposer une généralisation du procédé de Richardson au cas vectoriel.

I- Présentation de la méthode et propriétés.

I.1- Présentation :

Soit B un espace vectoriel, normé, sur $\mathbb{K}$ ($\mathbb{R}$ ou $\mathbb{C}$) et B^* son dual. La norme sur B sera notée : $\| \ \|$.

Nous considérons dans B les suites (S_n) qui sont de la forme :

$$(I.1) \quad S_n = S + \sum_{i=1}^{k} a_i \, g_i(n) \quad ; \ k \in \mathbb{N}^*$$

où : $g_i(n) \in B$; $a_i \in \mathbb{K}$; $i = 1, \dots, k$.

Il s'agit de calculer S à partir de $S_n, \dots, S_{n+k}$; $g_i(n), \dots, g_i(n+k)$; $i = 1, \dots, k$. Pour cela nous allons écrire (I.1) pour $n, \dots, n+k-1$, et nous obtenons :

$$S_{n+j} = S + \sum_{i=1}^{k} a_i \, g_i(n+j) \quad ; \ j = 0, \dots, k-1 .$$

et :

$$(I.2) \quad \Delta S_{n+j} = \sum_{i=1}^{k} a_i \, \Delta g_i(n+j) \quad ; \ j = 0, \dots, k-1$$

Prenons un élément z de B^* et appliquons le aux deux membres de (I.2) ; nous avons alors :

$$\langle z, \Delta S_{n+j} \rangle = \sum_{i=1}^{k} a_i \langle z, \Delta g_i(n+j) \rangle \quad ; \ j = 0, \dots, k-1 .$$

Ce qui nous donne un système linéaire de k équations à k inconnues : $a_1, \dots, a_k$.

$$(I.3) \quad \begin{pmatrix} \langle z, \Delta g_1(n)\rangle & \cdots & \langle z, \Delta g_k(n)\rangle \\ \vdots & & \vdots \\ \langle z, \Delta g_1(n+k-1)\rangle & \cdots & \langle z, \Delta g_k(n+k-1)\rangle \end{pmatrix} * \begin{pmatrix} a_1 \\ \vdots \\ a_k \end{pmatrix} = \begin{pmatrix} \langle z, \Delta S_n\rangle \\ \vdots \\ \langle z, \Delta S_{n+k-1}\rangle \end{pmatrix}$$

et :

$$S = S_n - a_1 g_1(n) - \dots - a_k g_k(n).$$

Posons :

$$a = (a_1, \dots, a_k)^T \quad ; \quad u = \left(\langle z, \Delta S_n\rangle, \dots, \langle z, \Delta S_{n+k-1}\rangle\right)$$

$$g(n) = \left(g_1(n), \dots, g_k(n)\right) \quad \text{et} \quad g(n) * a = a_1 g_1(n) + \dots + a_k g_k(n)$$

alors :

Le système (I.3) est équivalent à : $A_n . a = u$ où

$$A_n = \left[\langle z, \Delta g_i(n+j-1)\rangle \right]_{\substack{1 \le i \le k \\ 1 \le j \le k}}$$

On supposera que $\det A_n \neq 0$ pour tout (k, n). Donc :

$$S = S_n - g(n) * A_n^{-1} . u$$

Et d'après la formule de Schur [8], on a :

$$S = \frac{\begin{vmatrix} S_n & g_1(n) & \cdots & g_k(n) \\ \langle z, \Delta S_n \rangle & \langle z, \Delta g_1(n) \rangle & \cdots & \langle z, \Delta g_k(n) \rangle \\ \cdots & \cdots & \cdots & \cdots \\ \langle z, \Delta S_{n+k-1} \rangle & \langle z, \Delta g_1(n+k-1) \rangle & \cdots & \langle z, \Delta g_k(n+k-1) \rangle \end{vmatrix}}{\begin{vmatrix} \langle z, \Delta g_1(n) \rangle & \cdots & \langle z, \Delta g_k(n) \rangle \\ \cdots & \cdots & \cdots \\ \langle z, \Delta g_1(n+k-1) \rangle & \cdots & \langle z, \Delta g_k(n+k-1) \rangle \end{vmatrix}}$$

En général, la suite (S_n) n'a pas toujours exactement la forme (I.1), nous noterons alors le rapport de déterminants précédent : $E_k(S_n)$ qui sera considéré comme approximation de la limite de (S_n) lorsque celle-ci est convergente.

$E_k(S_n)$ peut être calculée récursivement par le E-algorithme topologique [4] de C. BREZINSKI :

$$\left\| \begin{array}{ll} E_0^{(n)} = S_n \quad ; \quad g_{0i}^{(n)} = g_i(n) & \\ E_k^{(n)} = E_{k-1}^{(n)} - \dfrac{\langle z, \Delta E_{k-1}^{(n)} \rangle}{\langle z, \Delta g_{k-1,k}^{(n)} \rangle} \cdot g_{k-1,k}^{(n)} & ; k \geq 1. \\ g_{k,i}^{(n)} = g_{k-1,i}^{(n)} - \dfrac{\langle z, \Delta g_{k-1,i}^{(n)} \rangle}{\langle z, \Delta g_{k-1,k}^{(n)} \rangle} \cdot g_{k-1,k}^{(n)} & ; i > k > 0. \end{array} \right.$$

2 - Propriétés :

propriété 1 [4] :

$$\forall k \geq 0 \qquad E_k(S_n) = E_k^{(n)}$$

$$E_k(g_i(n)) = g_{k,i}^{(n)} \qquad ; i > k$$

propriété 2 [4] :

i) si $S_n = S + \sum_{i=1}^{k} a_i g_i(n)$ alors : $E_k^{(n)} = S$

ii) si $S_n = S + \sum_{i \geq 1} a_i g_i(n)$ alors : $\forall k \geq 1$, $E_k^{(n)} = S + \sum_{i \geq k+1} a_i g_{k,i}^{(n)}$

- Comme $E_k^{(n)}$ dépend de S_n et des $g_i(n)$ on note aussi :

$$E_k^{(n)} = E_k\left(S_n ; g_1(n), \dots, g_k(n)\right).$$

- Nous avons alors les résultats suivants :

propriété 3 :

$$\forall a_i \neq 0 \quad , i = 1, \dots, k$$

$$E_k\left(S_n ; a_1 g_1(n), \dots, a_k g_k(n)\right) = E_k\left(S_n ; g_1(n), \dots, g_k(n)\right).$$

dem :

A partir de l'expression des quantités précédentes sous-forme de rapport de déterminants. □

Supposons maintenant que l'une des trois conditions suivantes est vérifiée :

C) $\begin{cases} \text{i) les } g_i \text{ sont indépendants de } S_n. \\ \text{ii) } \forall i, \exists a_i \neq 0 \text{ tel que : } g_i(n\,;\, aS_n+b) = a_i g_i(n, S_n) \\ \quad \text{où } a \text{ et } b \text{ sont deux constantes tel que } a \neq 0,\ b \in B. \\ \text{iii) la transformation } E_k \text{ est appliquée avec les} \\ \quad \text{mêmes } g_i. \end{cases}$

alors :

propriété 4 :

Si l'une des trois conditions précédentes est vérifiée alors : $\forall a \neq 0$ et $\forall b \in B$:

$$E_k\left(aS_n+b\,;\, g_1(n), \dots, g_k(n)\right) = a\, E_k\left(S_n\,;\, g_1(n), \dots, g_k(n)\right) + b.$$

dem : facile à vérifier à partir des expressions des quantités précédentes sous-forme de rapports de déterminants □

Nous allons donner maintenant deux algorithmes de calcul de $E_k^{(N)}$ où N est fixé et k variable. Le cas où n et k sont variables tous les deux est traité dans [18].

II Algorithmes de calcul de $E_k^{(n)}$ pour n fixé :

1 RPA :

Soit E l'ensemble des suites d'éléments de B.

posons $y = (S_n)$ et $x_i = (g_i(n))_n$; $i = 1, ---, k$.

Soit $n = N$ fixé.

On definit les $z_i \in E^*$ par :

si $V \in E$: $\langle z_i, V \rangle = \langle z, \Delta V_{N+i-1} \rangle$; $i = 1, ---, k$

Soit E_k l'élément de E obtenu par application du RPA à $y, x_1, ---, x_k$; On verifie facilement que le N-ième terme de E_k : $(E_k)_N$ est identique au vecteur $E_k^{(N)}$.

Le calcul de $E_k^{(N)}$ par le RPA nécéssite un total d'opérations de l'ordre de : $k^2 p$, dans le cas où $B = \mathbb{C}^p$.

2 Méthode (A) :

Soit $n = N$ fixé.

$$E_k^{(N)} = S_N + \sum_{i=1}^{k} a_{k,i}^{(N)} \times g_i(N)$$

où :

$a_{k,1}^{(N)}, ---, a_{k,k}^{(N)}$ sont des scalaires solutions du système d'ordre k suivant :

$$\begin{pmatrix} \langle z, \Delta g_1(N)\rangle & \cdots & \langle z, \Delta g_k(N)\rangle \\ \hline \langle z, \Delta g_1(N+k-1)\rangle & \cdots & \langle z, \Delta g_k(N+k-1)\rangle \end{pmatrix} \times \begin{pmatrix} a_{k,1}^{(N)} \\ \vdots \\ a_{k,k}^{(N)} \end{pmatrix} = \begin{pmatrix} \langle z, \Delta S_1\rangle \\ \vdots \\ \langle z, \Delta S_{N+k-1}\rangle \end{pmatrix}$$

Nous résolvons ensuite le système par la méthode (A) exposée au chap I.

Le calcul de $E_k^{(N)}$ par cette méthode nécessite, lorsque $B = \mathbb{C}^p$:

* $\frac{1}{3}k\left(k^2 - \frac{3}{2}k - \frac{5}{2}\right) + kp$ multiplications
* $\frac{1}{3}k\left(k^2 - \frac{1}{2}k - \frac{3}{2}\right) + kp$ additions et soustractions

Il faudra ajouter à cela le nombre d'opérations nécessaires pour construire la matrice du système précédent. Pour minimiser ce calcul on prendra un vecteur z contenant beaucoup de zéros.

<u>ar que :</u>

Pour le calcul de $E_K^{(n)}$ avec K fixé et n variable on pourra consulter [15].

I Analyse de la convergence :

Nous nous interesserons uniquement au problème de la convergence forte ; pour la convergence faible les résultats seront ceux du E-algorithme scalaire.

Propriétés :

Si $\|S_n - S\| = o(1)$

si $\forall k$: $\dfrac{\|g^{(n)}_{k-1,k}\|}{|\langle z, \Delta g^{(n)}_{k-1,k}\rangle|}$ est majorée par une constante M

alors :

$$\|E^{(n)}_k - S\| = o(1) \qquad (n \to \infty)$$

Dem :

posons $R^{(n)}_k = E^{(n)}_k - S$ alors :

$$R^{(n)}_k = R^{(n)}_{k-1} - \frac{\langle z, \Delta R^{(n)}_{k-1}\rangle}{\langle z, \Delta g^{(n)}_{k-1,k}\rangle} \cdot g^{(n)}_{k-1,k}$$

faisons une recurrence sur k :

- $k = 0$; $R^{(n)}_0 = S_n - S$ donc $\|R^{(n)}_0\| = o(1)$
- Supposons que $\|R^{(n)}_{k-1}\| = o(1)$ $(n \to \infty)$

On a :

$$\|R^{(n)}_k\| \leq \|R^{(n)}_{k-1}\| + \|z\| . \|\Delta R^{(n)}_{k-1}\| \times \frac{\|g^{(n)}_{k-1,k}\|}{|\langle z, \Delta g^{(n)}_{k-1,k}\rangle|}$$

or : $\|R_{k-1}^{(n)}\| = o(1) \quad (n \to \infty) \Rightarrow \|\Delta R_{k-1}^{(n)}\| = \|R_{k-1}^{(n+1)} - R_{k-1}^{(n)}\|$

$$= o(1) \quad (n \to \infty).$$

d'autre part:

$$\frac{\|g_{k-1,k}^{(n)}\|}{|\langle z, g_{k-1,k}^{(n)}\rangle|} \leq M$$

donc : $\|E_k^{(n)} - S\| = o(1) \quad (n \to \infty)$ ∎

Théorème 1 :

On suppose que S_n converge en norme vers S.

On pose $r_n = S_n - S$ et $R_k^{(n)} = E_k^{(n)} - S$

Si :

i) $\lim_{n \to \infty} \frac{\langle z, g_i(n+1)\rangle}{\langle z, g_i(n)\rangle} = b_i \neq 1$ et $b_i \neq b_j$ pour $i \neq j$

et

$\forall i \in \{1, \dots, k\}$. $\frac{\|g_i(n)\|}{|\langle z, g_i(n)\rangle|}$ majorée par M_0 pour $n \geq N_0$

ii) il existe $j_0 \in \{1, \dots, k\}$ tel que:

$$\rho_n = \frac{\langle z, r_{n+1}\rangle}{\langle z, r_n\rangle} \xrightarrow[n \to \infty]{} b_{j_0}$$

iii) $\left\| r_n - g_{j_0}(n) \times \frac{\langle z, \Delta r_n\rangle}{\langle z, \Delta g_{j_0}(n)\rangle} \right\| = o(\|r_n\|) \quad (n \to \infty)$

alors :

$\forall k \geq 1$: $\|E_k^{(n)} - S\| = o(\|S_n - S\|) \quad (n \to \infty)$.

<u>marque 1</u>:

Certaines conditions du theorème paraissent assez fortes, mais nous verrons qu'elles sont satisfaites pour certaines classes de suites vectorielles que nous étudierons.

<u>m</u>: $R_k^{(n)} = E_k^{(n)} - S$ et peut s'écrire sous la forme:

$$R_k^{(n)} = \frac{\begin{vmatrix} \lambda_n & g_1(n) & \cdots & g_k(n) \\ \langle z, \Delta S_n \rangle & \langle z, \Delta g_1(n) \rangle & \cdots & \langle z, \Delta g_k(n) \rangle \\ \cdots & \cdots & \cdots & \cdots \\ \langle z, \Delta S_{n+k-1} \rangle & \langle z, \Delta g_1(n+k-1) \rangle & \cdots & \langle z, \Delta g_k(n+k-1) \rangle \end{vmatrix}}{\begin{vmatrix} \langle z, \Delta g_1(n) \rangle & \cdots & \langle z, \Delta g_k(n) \rangle \\ \cdots & \cdots & \cdots \\ \langle z, \Delta g_1(n+k-1) \rangle & \cdots & \langle z, \Delta g_k(n+k-1) \rangle \end{vmatrix}}$$

Divisons la première colonne du numérateur par $\langle z, \Delta S_n \rangle$, la seconde par $\langle z, \Delta g_1(n) \rangle, \ldots,$ la $(k+1)$ ième par $\langle z, \Delta g_k(n) \rangle$ et faisons de même pour le dénominateur. Nous obtenons alors l'expression suivante de $R_k^{(n)}$:

$$R_k^{(n)} = \langle z, \Delta S_n \rangle \times \begin{vmatrix} \dfrac{r_n}{\langle z, \Delta S_n \rangle} & \dfrac{g_1(n)}{\langle z, \Delta g_1(n) \rangle} & - - - & \dfrac{g_k(n)}{\langle z, \Delta g_k(n) \rangle} \\ 1 & 1 & - - - & 1 \\ \dfrac{\langle z, \Delta S_{n+1} \rangle}{\langle z, \Delta S_n \rangle} & \dfrac{\langle z, \Delta g_1(n+1) \rangle}{\langle z, \Delta g_1(n) \rangle} & - - - & \dfrac{\langle z, \Delta g_k(n+1) \rangle}{\langle z, \Delta g_k(n) \rangle} \\ - - - & - - - & - - - & - - - \\ \dfrac{\langle z, \Delta S_{n+k-1} \rangle}{\langle z, \Delta S_n \rangle} & \dfrac{\langle z, \Delta g_1(n+k-1) \rangle}{\langle z, \Delta g_1(n) \rangle} & - - - & \dfrac{\langle z, \Delta g_k(n+k-1) \rangle}{\langle z, \Delta g_k(n) \rangle} \end{vmatrix} \Big/ D_k^{(n)}$$

où $D_k^{(n)}$ est le déterminant obtenu du précédent en éliminant la première ligne et la première colonne.

<u>posons</u> :

$$b_i^{(n)} = \frac{\langle z, \Delta g_i(n+1) \rangle}{\langle z, \Delta g_i(n) \rangle} \quad ; i = 1, \dots, k.$$

et

$$\beta_n = \frac{\langle z, \Delta S_{n+1} \rangle}{\langle z, \Delta S_n \rangle}$$

pour $p \in \{1, \dots, k-1\}$, on a :

$$\frac{\langle z, \Delta g_i(n+p) \rangle}{\langle z, \Delta g_i(n) \rangle} = \frac{\langle z, \Delta g_i(n+p) \rangle}{\langle z, \Delta g_i(n+p-1) \rangle} \times \dots \times \frac{\langle z, \Delta g_i(n+1) \rangle}{\langle z, \Delta g_i(n) \rangle}$$

$$= b_i^{(n+p-1)} \times \dots \times b_i^{(n)} = b_{i,p}^{(n)}$$

et de la même manière :

$$\frac{\langle z, \Delta S_{n+p} \rangle}{\langle z, \Delta S_n \rangle} = \beta_{n+p-1} \times \dots \times \beta_n = \beta_{n,p}.$$

Or d'après i) $\displaystyle\lim_{n \to \infty} \frac{\langle z, g_i(n+1) \rangle}{\langle z, g_i(n) \rangle} = b_i \neq 1$

et d'après le theorème (01) nous avons aussi :

$$\lim_{n\to\infty} \frac{\langle z, \Delta g_i(n+1)\rangle}{\langle z, \Delta g_i(n)\rangle} = b_i$$

donc :

$$\lim_{n\to\infty} \frac{\langle z, \Delta g_i(n+p)\rangle}{\langle z, \Delta g_i(n)\rangle} = b_i^p \qquad p = 1, \dots, k-1$$

$$i = 1, \dots, k.$$

de la même manière, puisque $\lim_{n\to\infty} \frac{\langle z, r_{n+1}\rangle}{\langle z, r_n\rangle} = b_{j_0} \neq 1$

alors :

$$\lim_{n\to\infty} \frac{\langle z, \Delta S_{n+p}\rangle}{\langle z, \Delta S_n\rangle} = b_{j_0}^p \qquad ; \; p = 1, \dots, k-1.$$

Reprenons maintenant l'expression de $R_k^{(n)} = \langle z, \Delta S_n\rangle \times \frac{A_k^{(n)}}{D_k^{(n)}}$

d'après ce qui precède on a :

$$\lim_{n\to\infty} D_k^{(n)} = \prod_{i>j}^{k-1} (b_i - b_j)$$

Remplaçons dans $A_k^{(n)}$ la première colonne par sa difference avec la colonne j_0 :

$$A_k^{(n)} = \begin{vmatrix} \frac{r_n}{\langle z, \Delta S_n\rangle} - \frac{g_{j_0}(n)}{\langle z, \Delta g_{j_0}(n)\rangle} & \frac{g_1(n)}{\langle z, \Delta g_1(n)\rangle} & \cdots & \frac{g_k(n)}{\langle z, \Delta g_k(n)\rangle} \\ 0 & 1 & \cdots & 1 \\ \beta_{n1} - b_{j_0,1}^{(n)} & b_{1,1}^{(n)} & \cdots & b_{k,1}^{(n)} \\ \cdots & \cdots & \cdots & \cdots \\ \beta_{n,k-1} - b_{j_0,k-1}^{(n)} & b_{1,k-1}^{(n)} & \cdots & b_{k,k-1}^{(n)} \end{vmatrix}$$

notons $A_{k,p}^{(n)}$ le déterminant d'ordre k obtenu à partir de $A_k^{(n)}$ en supprimant la première ligne et la colonne p ; $(p=2,\dots,k+1)$.

Developpons $A_k^{(n)}$ suivant la première ligne :

$$A_k^{(n)} = \left(\frac{r_n}{\langle z, \Delta S_n\rangle} - \frac{g_{j_0}(n)}{\langle z, \Delta g_{j_0}(n)\rangle}\right) \times D_k^{(n)} + \sum_{p=1}^{k} (-1)^p \frac{g_p(n)}{\langle z, \Delta g_p(n)\rangle} \cdot A_{k,p}^{(n)}$$

Or :

puisque $\lim\limits_{n\to\infty} \beta_{n,j} = \lim\limits_{n\to\infty} b_{j_0,j}^{(n)} = b_{j_0}^{j}$; $j=1,\dots,k-1$

alors

$$\lim_{n\to\infty} A_{k,p}^{(n)} = 0$$

et :

$$R_k^{(n)} = \left(r_n - \langle z, \Delta S_n\rangle \times \frac{g_{j_0}(n)}{\langle z, \Delta g_{j_0}(n)\rangle}\right) + \frac{\langle z, \Delta S_n\rangle}{D_k^{(n)}} \times \sum_{p=1}^{k} (-1)^p \frac{g_p(n)}{\langle z, \Delta g_p(n)\rangle} \cdot A_{k,p}^{(n)}$$

Or :

$$\frac{\|R_k^{(n)}\|}{\|r_n\|} \leq \frac{1}{\|r_n\|} \times \left\| r_n - \langle z, \Delta S_n\rangle \cdot \frac{g_{j_0}(n)}{\langle z, \Delta g_{j_0}(n)\rangle} \right\| + \frac{|\langle z, \Delta r_n\rangle|}{\|r_n\|} \times \frac{1}{|D_k^{(n)}|} \times \sum_{p=1}^{k} \frac{\|g_p(n)\|}{|\langle z, \Delta g_p(n)\rangle|} \times |A_{k,p}^{(n)}|$$

Or :

$$\frac{\langle z, \Delta r_n\rangle}{\|r_n\|} = \frac{\langle z, r_n\rangle}{\|r_n\|} \times \left(\frac{\langle z, r_{n+1}\rangle}{\langle z, r_n\rangle} - 1\right)$$

et comme : $\dfrac{|\langle z, r_n\rangle|}{\|r_n\|} \leq \|z\|$ et $\dfrac{\langle z, r_{n+1}\rangle}{\langle z, r_n\rangle} - 1 \xrightarrow[n\to\infty]{} b_{j_0}$

alors :

il existe une constante $\alpha > 0$ et $n_0 \in \mathbb{N}^*$ tel que

$$\forall n \geq n_0 : \quad \frac{|\langle z, \Delta r_n\rangle|}{\|r_n\|} \leq \alpha.$$

et comme :

- $\lim_{n\to\infty} D_k^{(n)} = \prod_{i>j}^{k-1} (b_i - b_j)$; $\lim_{n\to\infty} A_{k,p}^{(n)} = 0$

- $\forall n \geq N_0$: $\dfrac{\|g_p(n)\|}{|\langle z, \Delta g_p(n)\rangle|} \leq M_0$ (Hyp iii)

alors (*) $\Longrightarrow$ $\lim_{n\to\infty} \dfrac{\|R_k^{(n)}\|}{\|r_n\|} \leq \lim_{n\to\infty} \left(\dfrac{1}{\|r_n\|} \times \left\| r_n - \langle z, \Delta S_n\rangle \times \dfrac{g_{j_0}(n)}{\langle z, \Delta g_{j_0}(n)\rangle} \right\| \right)$

et comme cette dernière limite est nulle d'après l'hypothèse iii)

On a :

$$\forall k \geq 1 : \quad \|E_k^{(n)} - S\| = o\left(\|S_n - S\|\right) \quad (n\to\infty) \quad \blacksquare$$

<u>Cas particulier : Δ^2 d'Aitken topologique :</u>

a) Considerons le cas $k=1$ et $g_1(n) = \Delta S_n$, alors :

$$E_1^{(n)} = T_n = S_n - \frac{\langle z, \Delta S_n\rangle}{\langle z, \Delta^2 S_n\rangle} \cdot \Delta S_n .$$

Propriété

Soit S_n une suite de B convergente vers S.

Si : i) $\lim_{n\to\infty} \dfrac{\langle z, r_{n+1}\rangle}{\langle z, r_n\rangle} = \rho \in [-1, 1[$

ii) $\|r_{n+1} - \rho r_n\| = o(\|r_n\|)$; alors :

$$\|T_n - S\| = o\left(\|S_n - S\|\right)$$

Dem: Montrons que les trois conditions du théorème 1 sont satisfaites :

1°) $g_1(n) = \Delta S_n$.

$$\frac{\langle z, g_1(n+1)\rangle}{\langle z, g_1(n)\rangle} = \frac{\langle z, \Delta r_{n+1}\rangle}{\langle z, \Delta r_n\rangle}$$

Or $\lim\limits_{n\to\infty} \frac{\langle z, r_{n+1}\rangle}{\langle z, r_n\rangle} = \rho \neq 1 \Rightarrow \lim\limits_{n\to\infty} \frac{\langle z, \Delta r_{n+1}\rangle}{\langle z, \Delta r_n\rangle} = \rho$.

2°) $\lim\limits_{n\to\infty} \frac{\langle z, r_{n+1}\rangle}{\langle z, r_n\rangle} = \lim\limits_{n\to\infty} \frac{\langle z, g_1(n+1)\rangle}{\langle z, g_1(n)\rangle} = b_1 = \rho \neq 1$.

3°) $\left\| r_n - g_1(n) \cdot \frac{\langle z, \Delta r_n\rangle}{\langle z, \Delta g_1(n)\rangle} \right\| \overset{?}{=} o\left(\|r_n\|\right) \quad (n\to\infty)$

posons :

$\frac{\langle z, \Delta r_{n+1}\rangle}{\langle z, \Delta r_n\rangle} = \rho_n = \rho + \alpha_n \quad$ où $\lim\limits_{n\to\infty} \alpha_n = 0$

alors :

$$r_n - g_1(n) \cdot \frac{\langle z, \Delta r_n\rangle}{\langle z, \Delta g_1(n)\rangle} = r_n - \frac{r_{n+1} - r_n}{\rho_n - 1}$$

$$= \frac{r_n \cdot \rho_n - r_n - r_{n+1} + r_n}{\rho_n - 1} = \frac{r_{n+1} - r_n \cdot \rho_n}{1 - \rho_n}$$

$$= \frac{r_{n+1} - r_n \rho - r_n \alpha_n}{1 - \rho_n}$$

$\Longrightarrow$

$$\left\| r_n - g_1(n) \cdot \frac{\langle z, \Delta r_n\rangle}{\langle z, \Delta g_1(n)\rangle} \right\| \leq \frac{1}{|1-\rho_n|} \left(\|r_{n+1} - \rho r_n\| + |\alpha_n| \, \|r_n\| \right)$$

et comme $\rho_n \xrightarrow[n\to\infty]{} \rho \neq 1$ et $\|r_{n+1} - \rho r_n\| = o\left(\|r_n\|\right)$ (hypothèse)

alors :

$$\left\| r_n - g_1(n) \cdot \frac{\langle z, \Delta r_n\rangle}{\langle z, \Delta g_1(n)\rangle} \right\| = o\left(\|r_n\|\right) \quad (n\to\infty) \quad \square$$

b) Considérons toujours le cas $k=1$ et prenons :

$$g_1(n) = \Delta S_n \cdot \frac{x_n}{\Delta x_n}$$

où

(x_n) est une suite de nombres réels, convergente vers zéro

et $\lim\limits_{n\to\infty} \frac{x_{n+1}}{x_n} = a \neq 1$

alors :

$$E_1 = S_n - \frac{\Delta S_n}{\frac{x_{n+1}}{x_n} \cdot \frac{\Delta x_n}{\Delta x_{n+1}} \cdot \frac{\langle z, \Delta S_{n+1}\rangle}{\langle z, \Delta S_n\rangle} - 1}$$

posons :

$$y_n = \frac{x_{n+1}}{x_n} - 1$$

donc :

$$E_1^{(n)} = S_n - \frac{\Delta S_n}{\frac{y_n}{y_{n+1}} \cdot \frac{\langle z, \Delta S_{n+1}\rangle}{\langle z, \Delta S_n\rangle} - 1}$$

Cette dernière transformation peut être considérée comme une modification du Δ^2 topologique.

Remarque 3 :

Dans le cas scalaire cette dernière transformation se réduit à :

$$E_1^{(n)} = \frac{S_n y_n \Delta S_{n+1} - S_{n+1} y_{n+1} \Delta S_n}{y_n \Delta S_{n+1} - y_{n+1} \Delta S_n}$$

qui n'est autre que la première colonne de la première généralisation de l'ε-algorithme scalaire [3].

IV. Applications :

1 - Considérons les suites (S_n) d'éléments de B qui sont de la forme :

$$S_n = S + \sum_{i \geq 1} \lambda_i^n \cdot y_i$$

où

$$\lambda_i \in K \quad \text{et} \quad y_i \in B \quad ; i \geq 1.$$

On notera (H_0) l'hypothèse suivante :

$$(H_0) \begin{cases} \lambda_1 \neq 0 \\ 1 > |\lambda_1| > |\lambda_2| \geq \cdots \geq |\lambda_i| \geq |\lambda_{i+1}| \geq \cdots \end{cases}$$

Remarquons que $(H_0) \Rightarrow (S_n)$ converge vers S.

Posons :

$$g_j(n) = \Delta S_{n+j-1} \cdot \langle z, \Delta S_{n+j-1} \rangle^{j-1} \; ; \; j = 1, \ldots, k$$

Nous avons le résultat suivant :

Propriété 7 :

Si on applique la transformation E_K à la suite (S_n) definie ci-dessus, verifiant (H_0) et si $\langle z, y_1 \rangle \neq 0$, alors :

$$\forall k \geq 1 \quad , \quad \| E_K^{(n)} - S \| = o\left(\| S_n - S \| \right) \quad (n \to \infty).$$

dem :

montrons que les trois conditions du théorème 1 sont vérifiées :

i) $g_j(n) = \Delta S_{n+j-1} \cdot \langle z, \Delta S_{n+j-1} \rangle^{j-1}$; $j = 1, \dots, k$

donc :

$$\frac{\langle z, g_j(n+1) \rangle}{\langle z, g_j(n) \rangle} = \left(\frac{\langle z, \Delta S_{n+j} \rangle}{\langle z, \Delta S_{n+j-1} \rangle} \right)^j$$

or :

$$S_{n+j} = S + \sum_{i \geq 1} \lambda_i^{n+j} \cdot y_i \Rightarrow \Delta S_{n+j} = \sum_{i \geq 1} \lambda_i^{n+j} (\lambda_i - 1) y_i$$

et

$$\frac{\langle z, \Delta S_{n+j} \rangle}{\langle z, \Delta S_{n+j-1} \rangle} = \frac{\lambda_1^{n+j} (\lambda_1 - 1) \cdot \langle z, y_1 \rangle \cdot [1 + o_j(1)]}{\lambda_1^{n+j-1} \cdot (\lambda_1 - 1) \cdot \langle z, y_1 \rangle \cdot [1 + o_{j-1}(1)]}$$

avec :

$$o_j(1) = \sum_{i \geq 2} \left(\frac{\lambda_i}{\lambda_1} \right)^{n+j} \frac{\lambda_i - 1}{\lambda - 1} \cdot \frac{\langle z, y_i \rangle}{\langle z, y_1 \rangle}$$

$$(H_0) \Longrightarrow \lim_{n \to \infty} o_j(1) = 0 \quad ; j = 1, \dots, k$$

donc :

$$\lim_{n \to \infty} \frac{\langle z, g_j(n+1) \rangle}{\langle z, g_j(n) \rangle} = \lambda_1^j = b_j \quad ; \quad j = 1, \dots, k$$

et

$b_i \neq b_j$ pour $i \neq j$ car $\lambda_1 \neq 0, 1, -1$.

* D'autre part :

$$\frac{g_j(n)}{\langle z, g_j(n) \rangle} = \frac{\Delta S_{n+j-1}}{\langle z, \Delta S_{n+j-1} \rangle} = \frac{\lambda_1^{n+j-1} \cdot (\lambda_1 - 1) \cdot [y_1 + o(1)]}{\lambda_1^{n+j-1} \cdot (\lambda_1 - 1) \langle z, y_1 \rangle \cdot [1 + o(1)]}$$

$$\Rightarrow \lim_{n \to \infty} \frac{\| g_j(n) \|}{|\langle z, g_j(n) \rangle|} = \frac{\| y_1 \|}{|\langle z, y_1 \rangle|}$$; la condition i) est donc vérifiée.

ii)

$$r_n = \sum_{i \geq 1} \lambda_i^n y_i \Longrightarrow \frac{\langle z, r_{n+1} \rangle}{\langle z, r_n \rangle} = \frac{\lambda_1^{n+1} \langle z, y_1 \rangle [1 + o(1)]}{\lambda_1^n \cdot \langle z, y_1 \rangle \cdot [1 + o(1)]}$$

Or $\lambda_1 \neq 0$ et $\langle z, y_1 \rangle \neq 0$, donc :

$$\lim_{n\to\infty} \frac{\langle z, r_{n+1}\rangle}{\langle z, r_n\rangle} = \lambda_1 = b_1 \quad \text{soit } \delta_0 = 1.$$

iii) $g_1(n) = \Delta S_n = \sum_{i\geq 1} \lambda_i^n (\lambda_i - 1) \cdot y_i$

posons :

$$A_n = r_n - g_1(n) \cdot \frac{\langle z, \Delta r_n\rangle}{\langle z, \Delta g_1(n)\rangle}$$

alors :

$$A_n = \sum_{i\geq 1} \lambda_i^n \cdot y_i - \left(\sum_{i\geq 1} \lambda_i^n (\lambda_i - 1) y_i\right) \times \frac{\sum_{i\geq 1} \lambda_i^n \cdot (\lambda_i - 1)\langle z, y_i\rangle}{\sum_{i\geq 1} \lambda_i^n \cdot (\lambda_i - 1)^2 \langle z, y_i\rangle}$$

$$A_n = \frac{\lambda_1^n \cdot [y_1 + a_n] \cdot \lambda_1^n (\lambda_1 - 1)^2 \langle z, y_1\rangle \times [1 + b_n] - \lambda_1^{2n} \cdot (\lambda_1 - 1)^2 \cdot \langle z, y_1\rangle \times [y_1 + c_n] \cdot [1 + d_n]}{\lambda_1^n (\lambda_1 - 1)^2 \cdot \langle z, y_1\rangle \cdot [1 + b_n]}$$

or $(H_0) \Rightarrow \lim_{n\to\infty} a_n = \lim_{n\to\infty} c_n = 0 \in E$ et $\lim_{n\to\infty} b_n = \lim_{n\to\infty} d_n = 0 \in K$

En simplifiant par : $\lambda_1^n (\lambda_1 - 1)^2 \langle z, y_1\rangle$, nous obtenons :

$$A_n = \frac{\lambda_1^n \cdot (y_1 + a_n)(1 + b_n) - \lambda_1^n (y_1 + c_n)(1 + d_n)}{1 + b_n}$$

donc,

$$\frac{\|A_n\|}{\|r_n\|} = \frac{|\lambda_1|^n}{|\lambda_1|^n} \times \frac{\|(y_1 + a_n)(1 + b_n) - (y_1 + c_n) \cdot (1 + d_n)\|}{\|y_1 + o(1)\| \times |1 + b_n|}$$

et puisque $\|y_1\| \neq 0$ alors : $\lim_{n\to\infty} \frac{\|A_n\|}{\|r_n\|} = 0$.

la condition iii) est alors verifiée, d'où :

$$\forall k \geq 1 : \quad \|E_k^{(n)} - S\| = o\left(\|S_n - S\|\right) \quad (n \to \infty) \blacksquare$$

2 Procédé de Richardson vectoriel :

Soit f une fonction de $\mathbb{R}$ dans B deux fois différentiable et telle que f'' soit majorée sur $\mathbb{R}$.

(x_n) est une suite auxilliaire de nombres réels tels que :

. $\lim_{n\to\infty} x_n = 0$ et $\lim_{n\to\infty} \frac{x_{n+1}}{x_n} = a$ avec $0<|a|<1$.

Soit (S_n) la suite d'éléments de B definie par :

$$S_n = f(x_n).$$

f continue $\Rightarrow \lim_{n\to\infty} S_n = f(0) = S.$

Par la suite, nous appliquerons E_k à cette suite (S_n), en prenant :

$$(*) \qquad g_i(n) = h_n \cdot x_n^i \qquad \text{avec} \quad h_n = \frac{\Delta S_n}{\Delta x_n} \in B.$$

Remarque 4 :

$$h_n = \frac{S_{n+1} - S_n}{x_{n+1} - x_n} \qquad \text{avec} \quad S_n = f(x_n).$$

Soit

$$h_n = \frac{f(x_{n+1}) - f(x_n)}{x_{n+1} - x_n}$$

f est differentiable, donc:

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

d'où:

$$\frac{f(x_{n+1}) - f(x_n)}{x_{n+1} - x_n} = f'(x_n) + O(|x_{n+1} - x_n|)$$

Mais comme: $\lim_{n\to\infty} x_n = 0$ et f' continue, alors:

$$\lim_{n\to\infty} \frac{f(x_{n+1}) - f(x_n)}{x_{n+1} - x_n} = f'(0) = h. \quad \text{Soit } \lim_{n\to\infty} h_n = h$$

Dans le cas où $B = \mathbb{R}$ et $g_i(n) = h x_n^i$ avec $h \in \mathbb{R} - \{0\}$, nous retrouvons le procédé classique de Richardson [3].

Appliquons maintenant la transformation E_k, $k \geq 1$, à la suite $(S_n)_n$ définie précédemment, où les $g_i(n)$ sont ceux definis par (*).

<u>Theorème</u> :

Soit (S_n) la suite de B definie par $S_n = f(x_n)$ avec:

i) $f : \mathbb{R} \to B$ de classe $\mathcal{C}^2$ tel que:

- il existe une constante M : $\|f''(x)\| \leq M$, $\forall x \in \mathbb{R}$
- $\langle z, f'(0) \rangle \neq 0$

ii) $\lim_{n\to\infty} x_n = 0$ et $\lim_{n\to\infty} \frac{x_{n+1}}{x_n} = a$; $0 < |a| < 1$

alors:

$$\forall k \geq 1 \; ; \quad \|E_k^{(n)} - S\| = o(\|S_n - S\|) \quad (n \to \infty).$$

vérifiées :

$$i)_* \quad \frac{\langle z, g_i(n+1)\rangle}{\langle z, g_i(n)\rangle} = \frac{\langle z, h_{n+1}\rangle}{\langle z, h_n\rangle} \cdot \left(\frac{x_{n+1}}{x_n}\right)^i \quad ; \quad i=1,\dots,k.$$

Or,

$$\lim_{n\to\infty} h_n = h = f'(0) \quad \text{et} \quad \lim_{n\to\infty} \frac{x_{n+1}}{x_n} = a \quad ; \text{ donc:}$$

$$\lim_{n\to\infty} \frac{\langle z, g_i(n+1)\rangle}{\langle z, g_i(n)\rangle} = a^i = b_i \quad \text{et } b_i \neq b_j \quad ; i=1,\dots,k$$

puisque $0<|a|<1$.

$$* \quad \frac{\|g_i(n)\|}{|\langle z, g_i(n)\rangle|} = \frac{\|h_n\|}{|\langle z, h_n\rangle|} \cdot \left|\frac{x_n^i}{x_n^i}\right|$$

donc:

$$\lim_{n\to\infty} \frac{\|g_i(n)\|}{|\langle z, g_i(n)\rangle|} = \frac{\|h\|}{|\langle z, h\rangle|}$$

; la condition i) du théorème 1 est satisfaite.

ii) $S_n = f(x_n)$ et $S = f(0)$, donc,

$$r_n = f(x_n) - f(0) = f'(0).x_n + \int_0^1 (1-t).f''(tx_n)x_n^2\,dt$$

posons.

$$M_n = \int_0^1 (1-t).f''(tx_n)\,dt$$

Puisque : $\forall z \in \mathbb{R} : \|f''(z)\| \leq M$ alors: $\|M_n\| \leq M/2, \forall n.$

$$r_n = f'(0).x_n + x_n^2 M_n.$$

$$r_{n+1} = f'(0).x_{n+1} + x_{n+1}^2 . M_{n+1}$$

donc:

$$\rho_n = \frac{\langle z, r_{n+1}\rangle}{\langle z, r_n\rangle} = \frac{\langle z, f'(0)\rangle x_{n+1} + \langle z, M_{n+1}\rangle . x_{n+1}^2}{\langle z, f'(0)\rangle x_n + \langle z, M_n\rangle . x_n^2}$$

$$\rho_n = \frac{\langle z, f'(0)\rangle \cdot \frac{x_{n+1}}{x_n} + \langle z, M_{n+1}\rangle \cdot \frac{x_{n+1}^2}{x_n}}{\langle z, f'(0)\rangle + \langle z, M_n\rangle \cdot x_n}$$

or : $\frac{x_{n+1}}{x_n} \xrightarrow[n\infty]{} a$; $x_n \xrightarrow[n\infty]{} 0$

donc :

$$\lim_{n\to\infty} \frac{x_{n+1}^2}{x_n} = 0 \quad \text{et alors : } \lim_{n\to\infty} \frac{\langle z, r_{n+1}\rangle}{\langle z, r_n\rangle} = a = b_1 \quad \text{soit } j_0 = 1.$$

) posons :

$$A_n = r_n - g_1(n) \cdot \frac{\langle z, \Delta r_n\rangle}{\langle z, \Delta g_1(n)\rangle}$$

on a :

$$r_n = f'(0) \cdot x_n + x_n^2 M_n$$

$$\Delta r_n = f'(0) \cdot \Delta x_n + x_{n+1}^2 M_{n+1} - x_n^2 M_n .$$

et, $\langle z, \Delta g_1(n)\rangle = \Delta x_n \cdot \langle z, h_{n+1} + \frac{x_n}{\Delta x_n} \cdot \Delta h_n\rangle$.

alors :

$$A_n = x_n \cdot \left[f'(0) + x_n M_n - h_n \cdot \frac{\langle z, f'(0)\rangle + \frac{x_{n+1}^2}{\Delta x_n}\langle z, M_{n+1}\rangle - \frac{x_n^2}{\Delta x_n} \cdot \langle z, M_n\rangle}{\langle z, h_{n+1} + \frac{x_n}{\Delta x_n} \cdot \Delta h_n\rangle} \right]$$

$A = x_n \times B_n$; où B_n est l'expression entre crochets.

or : $x_n \xrightarrow[n\infty]{} 0$; $\frac{x_{n+1}}{x_n} \xrightarrow[n\infty]{} a$; $\frac{x_n}{\Delta x_n} \xrightarrow[n\infty]{} \frac{1}{a-1}$; $\frac{x_n^2}{\Delta x_n} \xrightarrow[n\infty]{} 0$

$$\frac{x_{n+1}}{\Delta x_n} \xrightarrow[n\infty]{} \frac{a}{a-1} \quad \text{et} \quad \frac{x_{n+1}^2}{\Delta x_n} \xrightarrow[n\infty]{} 0$$

alors :

donc:

$$\lim_{n\to\infty} B_n = f'(0) - h \times \frac{\langle z, f'(0)\rangle}{\langle z, h\rangle}$$

$$= \frac{f'(0).\langle z, h\rangle - h.\langle z, f'(0)\rangle}{\langle z, h\rangle}$$

Or:

$$f'(0) = h \quad \text{et} \quad \langle z, h\rangle \neq 0 \quad \text{d'où} \quad \lim_{n\to\infty} B_n = 0$$

et comme:

$$\frac{\|A_n\|}{\|r_n\|} = \frac{\|B_n\|}{\|f'(0) + x_n r_n\|} \quad \text{alors} \quad \lim_{n\to\infty} \frac{\|A_n\|}{\|r_n\|} = 0$$

puisque $\|B_n\| \longrightarrow 0$ et $\|f'(0)\| \neq 0$.

Les trois conditions du théorème 1 sont vérifiées, donc:

$$\forall k \geq 1 : \quad \|E_k^{(n)} - S\| = o\left(\|S_n - S\|\right) \quad (n \to \infty) \blacksquare$$

. Considérons de manière générale les suites (S_n) de B de la forme: (3.1) $S_n = S + \sum_{i\geq 1} v_i . x_n^i$

où * $v_i \in B$

* (x_n) une suite de réels convergente vers zéro et $\lim_{n\to\infty} \frac{x_{n+1}}{x_n} = a$ avec $0 < |a| < 1$.

Prenons:

$$g_i(n) = h_n . x_n^i \quad \text{où} \quad h_n = \frac{\Delta S_n}{\Delta x_n}.$$

on montre facilement que $\lim_{n\to\infty} h_n = h = v_1$.

Nous obtenons le résultat suivant :

Théorème 3 :

Si on applique la transformation E_k $(k \geq 1)$ à la suite (S_n) definie par (3.1) et si $\langle z, v_1 \rangle \neq 0$, alors :

$$\forall k \geq 1 ; \qquad \| E_k^{(n)} - S \| = o\left(\| S_n - S \| \right) \quad (n \to \infty).$$

Dem :

On montre, comme dans le theorème 2, que les conditions du theorème 1 sont vérifiées et on a le résultat. □

- Chapitre II -

Une version simplifiée du tl-algorithme

Introduction :

Nous proposons dans ce chapitre une version simplifiée du H-algorithme [7] qu'on appelera "algorithme T". Certains choix des suites auxilliaires intervenant dans (T), nous permettera de retrouver des algorithmes connus. Nous étudierons particulièrement un de ces cas, que nous appliquerons aux suites produites par des itérations linéaires. Ceci nous amènera, comme pour l'ε-algorithme vectoriel, à proposer une méthode de calcul des éléments propres d'une matrice et à accélérer la convergence des suites précédentes.

I- Présentation de l'algorithme :

Soit (S_n) une suite d'éléments de $\mathbb{R}^p$. $\Phi_i \in \mathbb{R}^p$; $i \geq 1$.
$(f_i(n))_n$; $i \geq 1$ des suites de nombres réels.

On commence cette partie par un rappel du H-algorithme [7],

On pose :

$$H_k^{(n)} = \begin{vmatrix} S_n & S_{n+1} & \cdots & S_{n+k} \\ f_1(n) & f_1(n+1) & \cdots & f_1(n+k) \\ \cdots & \cdots & \cdots & \cdots \\ f_k(n) & f_k(n+1) & \cdots & f_k(n+k) \end{vmatrix} \Bigg/ \begin{vmatrix} 1 & 1 & \cdots & 1 \\ f_1(n) & f_1(n+1) & \cdots & f_1(n+k) \\ \cdots & \cdots & \cdots & \cdots \\ f_k(n) & f_k(n+1) & \cdots & f_k(n+k) \end{vmatrix}$$

$H_k^{(n)}$ se calcule récursivement par l'algorithme :

$$\begin{cases} H_0^{(n)} = S_n \quad ; \quad f_{0,i}^{(n)} = f_i(n) \\ H_k^{(n)} = H_{k-1}^{(n)} - \dfrac{\Delta H_{k-1}^{(n)}}{\Delta f_{k-1,k}^{(n)}} \cdot f_{k-1,k}^{(n)} \quad ; k \geq 1 \\ f_{k,i}^{(n)} = f_{k-1,i}^{(n)} - \dfrac{\Delta f_{k-1,i}^{(n)}}{\Delta f_{k-1,k}^{(n)}} \cdot f_{k-1,k}^{(n)} \quad ; i > k > 0 \end{cases}$$

$H_k^{(n)} \in \mathbb{R}^p$ alors que $f_{k,i}^{(n)} \in \mathbb{R}$.

Nous donnons une propriété qui nous servira par la suite :

Proposition 1 :

Si $\lim\limits_{n\to\infty} \dfrac{f_i(n+1)}{f_i(n)} = b_i \neq 1$ et si $b_i \neq b_j$ pour $i \neq j$, $i,j = 1,\dots,k$

alors :

$$\forall k \geq 1 : \quad \lim_{n\to\infty} \frac{f_{k,i}^{(n)}}{f_i(n)} = \frac{(b_i - b_1)\cdots(b_i - b_k)}{(1-b_1)\cdots(1-b_k)} \quad ; i > k.$$

Dem :

pour i fixé, faisons une récurrence sur k.

$k=1$:
$$\frac{f_{1,i}^{(n)}}{f_i(n)} = \frac{f_{0,i}^{(n)}}{f_i(n)} - \frac{\Delta f_{0,i}^{(n)}}{\Delta f_{0,1}^{(n)}} \cdot \frac{f_{0,1}^{(n)}}{f_{0,i}^{(n)}}$$

$$= 1 - \frac{\Delta f_i(n)}{f_i(n)} \times \frac{f_1(n)}{\Delta f_1(n)}$$

donc :

$$\lim_{n\to\infty} \frac{f_{1,i}^{(n)}}{f_i(n)} = 1 - \frac{b_i - 1}{b_1 - 1} = \frac{b_i - b_1}{1 - b_1}.$$

- Supposons que la proposition est vraie jusqu'à $k-1$.

Comme $\lim\limits_{n\to\infty} \dfrac{f_i(n+1)}{f_i(n)} = b_i \neq 1$ alors d'après le théorème (01) et l'hypothèse de récurrence, on a :

$$\lim_{n\to\infty} \frac{\Delta f_{k-1,i}^{(n)}}{\Delta f_i(n)} = \frac{(b_i - b_1)\cdots(b_i - b_{k-1})}{(1-b_1)\cdots(1-b_{k-1})}$$

Si, dans la règle auxiliaire du H-algorithme, on divise les deux membres par $f_i(n)$, on obtient :

$$\frac{f_{k,i}^{(n)}}{f_i(n)} = \frac{f_{k-1,i}^{(n)}}{f_i(n)} - \frac{\Delta f_{k-1,i}^{(n)}}{\Delta f_i(n)} \times \frac{\Delta f_i(n)}{f_i(n)} \times \frac{f_{k-1,k}^{(n)}}{f_k(n)} \times \frac{f_k(n)}{\Delta f_k(n)} \times \frac{\Delta f_k(n)}{\Delta f_{k-1,k}^{(n)}}$$

mais d'après ce qui précède, $\lim\limits_{n\to\infty} \frac{f_{k-1,i}^{(n)}}{f_i(n)} = \lim\limits_{n\to\infty} \frac{\Delta f_{k-1,i}^{(n)}}{\Delta f_i(n)}$. $\forall i \geq k-1$

donc:

$$\lim_{n\to\infty} \frac{f_{k,i}^{(n)}}{f_i(n)} = \frac{(b_i - b_1) --- (b_i - b_{k-1})}{(1-b_1) --- (1-b_{k-1})} \times \left[1 - \frac{b_i - 1}{b_k - 1} \right]$$

Soit:

$$\lim_{n\to\infty} \frac{f_{k,i}^{(n)}}{f_i(n)} = \frac{(b_i - b_1) \times --- \times (b_i - b_k)}{(1-b_1) \times --- \times (1-b_k)} \quad \blacksquare$$

Reprenons maintenant l'expression de $H_k^{(n)}$ donnée par le H-algorithme:

$$(*) \qquad H_k^{(n)} - H_{k-1}^{(n)} = \frac{\Delta H_{k-1}^{(n)}}{\Delta f_{k-1,k}^{(n)}} \cdot f_{k-1,k}^{(n)}$$

Or,

nous avons vu précédemment que: $\lim\limits_{n\to\infty} \left(\frac{f_{k-1,k}^{(n)}}{f_k(n)} \times \frac{\Delta f_k(n)}{\Delta f_{k-1,k}^{(n)}} \right) = 1$

donc:

$$\frac{f_{k-1,k}^{(n)}}{\Delta f_{k-1,k}^{(n)}} \sim \frac{f_k(n)}{\Delta f_k(n)} \qquad (n \to \infty)$$

Ceci nous amène à remplacer dans l'expression $(*)$, $\frac{f_{k-1,k}^{(n)}}{\Delta f_{k-1,k}^{(n)}}$ par: $\frac{f_k(n)}{\Delta f_k(n)}$.

Alors en notant $T_k^{(n)}$, au lieu de $H_k^{(n)}$, la nouvelle quantité obtenue, on a l'algorithme déduit suivant :

$$T) \quad \left\| \begin{array}{l} T_0^{(n)} = S_n \\ \\ T_k^{(n)} = T_{k-1}^{(n)} - \dfrac{f_k(n)}{\Delta f_k(n)} \cdot \Delta T_{k-1}^{(n)} \quad ; \quad k \geq 1. \end{array} \right.$$

Remarque 1 :

Dans le cas scalaire, de nombreuses transformations connues comme les procédés d'Overholt et de Richardson se déduisent de cet algorithme par des choix particuliers des suites $(f_k(n))_n$.

Quelques choix des $f_k(n)$:

II.1- Seconde variante du CRPA [5] :

Si on pose :

$$f_k(n) = \langle \Phi_k, T_{k-1}^{(n)} \rangle \qquad \forall k \geq 1$$

alors l'algorithme (T) n'est autre que la seconde variante du CRPA appliqué à $S_n, \ldots, S_{n+k}$ avec les formes linéaires $\Phi_1, \ldots, \Phi_k$.

Si on conserve le même choix en prenant comme initialisation : $T_0^{(n)} = \Delta S_n$, alors l'algorithme obtenu n'est autre que le β-algorithme étudié au chapitre IV.

2 – Second choix des $f_k(n)$:

Soit (S_n) une suite de $\mathbb{R}^p$ de la forme :

$$S_n = S + \sum_{i \geq 1} v_i f_i(n)$$

où :

$v_i \in \mathbb{R}^p$ et $f_i(n) \in \mathbb{R}$, avec $\lim_{n \to \infty} f_i(n) = 0$, $\forall i \geq 1$.

on suppose que les $f_i(n)$, $i \geq 1$ verifient les trois conditions :

$$c_1) \quad f_i(n+1) = \sum_{j \geq i} b_{ij} f_j(n) \quad , \forall i \geq 1.$$

$$c_2) \quad f_i(n) . f_j(n) = \sum_{l \geq i+j} a_{il}^{(j)} f_l(n) \quad \forall i, j \geq 1.$$

$$c_3) \quad \frac{f_j(n)}{f_i(n)} = \sum_{l \geq j-i} d_{il}^{(j)} f_l(n) \quad ; \forall i, j \quad , j > i \geq 1.$$

Nous allons montrer que :

$$\forall n : \quad T_1^{(n)} = S + v_{12} f_2(n) + v_{13} f_3(n) + \dots \text{ où } v_{1i} \in \mathbb{R}^p$$

En effet :

$$S_n = S + v_1 f_1(n) + v_2 f_2(n) + \dots$$

$$S_{n+1} = S + v_1 f_1(n+1) + v_2 f_2(n+1) + \dots$$

ord'après c_1) :

$$f_1(n+1) = b_{11} f_1(n) + b_{12} f_2(n) + \dots$$

$$f_2(n+1) = b_{22} f_2(n) + b_{23} . f_3(n) + \dots$$

$$S_{n+1} = S + v_1 b_{11} f_1(n) + \left(v_1 b_{12} + v_2 b_{22}\right) f_2(n) + \cdots$$

d' autre part $c_1) \Rightarrow$

$$\frac{f_1(n+1)}{f_1(n)} = b_{11} + b_{12} \frac{f_2(n)}{f_1(n)} + b_{13} \frac{f_3(n)}{f_1(n)} + \cdots$$

$$c_3) \Rightarrow \quad \frac{f_2(n)}{f_1(n)} = d_{11}^{(2)} f_1(n) + d_{12}^{(2)} f_2(n) + \cdots$$

et

$$\frac{f_3(n)}{f_1(n)} = d_{12}^{(3)} f_2(n) + \cdots$$

donc:

$$S_{n+1} - \frac{f_1(n+1)}{f_1(n)} . S_n = S + v_1 b_{11} f_1(n) + \left(v_1 b_{12} + v_2 b_{22}\right) f_2(n) + \cdots$$
$$- \left(b_{11} + b_{12} d_{11}^{(2)} f_1(n) + b_{12} d_{12}^{(2)} f_2(n) + b_{13} d_{12}^{(3)} f_2(n) + \cdots\right) \times$$
$$\left(S + v_1 f_1(n) + v_2 f_2(n) + v_3 f_3(n) + \cdots\right)$$

$$= S \times \left[1 - \left(b_{11} + b_{12} d_{11}^{(2)} f_1(n) + \left(b_{12} d_{12}^{(2)} + b_{13} d_{12}^{(3)}\right) f_2(n)\right)\right] +$$

$$\left(v_1 b_{12} + v_2 \left(b_{22} - b_{11}\right)\right) f_2(n) - v_1 b_{12} d_{11}^{(2)} \left(f_1(n)\right)^2 + \cdots$$

or $c_2) \Rightarrow$

$$\left(f_1(n)\right)^2 = a_{12}^{(1)} f_2(n) + \cdots$$

donc:

$$S_{n+1} - \frac{f_1(n+1)}{f_1(n)} S_n =$$

$$S \times \left[1 - \left(b_{11} + b_{12}\, d_{11}^{(2)} f_1(n) + \cdots\right)\right] + \left[v_1 b_{12} \left(1 - d_{11}^{(2)} c_{12}^{(1)}\right) + v_2 \left(b_{22} - b_{11}\right)\right] f_2(n) + \cdots$$

et:

$$T_1^{(n)} = \frac{S_{n+1} - \frac{f_1(n+1)}{f_1(n)} \cdot S_n}{1 - \frac{f_1(n+1)}{f_1(n)}}$$

donc:

$$T_1^{(n)} = \frac{\left[1 - \left(b_{11} + b_{12} d_{11}^{(2)} f_1(n) + \cdots\right)\right] \cdot S + \left[v_1 b_{12} \cdot \left(1 - d_{11}^{(2)} \cdot c_{12}^{(1)}\right) + v_2 \left(b_{22} - b_{11}\right)\right] \cdot f_2(n) + \cdots}{1 - \left(b_{11} + b_{12} \cdot d_{11}^{(2)} \cdot f_1(n) + \cdots\right)}$$

$$= S + \frac{1}{1 - b_{11}} \cdot \frac{\left[v_1 b_{12} \cdot \left(1 - d_{11}^{(2)} c_{12}^{(1)}\right) + v_2 \left(b_{22} - b_{11}\right)\right] f_2(n) + \cdots}{1 - \frac{b_{12} \cdot d_{11}^{(2)}}{1 - b_{11}} \cdot f_1(n) + \cdots}$$

$$= S + \frac{v_1 b_{12} \cdot \left(1 - d_{11}^{(2)} \cdot c_{12}^{(1)}\right) + v_2 \left(b_{22} - b_{11}\right)}{1 - b_{11}} f_2(n) + \cdots$$

d'où : si $b_{11} \neq 1$, alors:

$$T_1^{(n)} = S + \sum_{i \geq 2} v_{1i} \cdot f_2(n).$$

Ainsi de proche en proche on montre que:

$$T_k^{(n)} = S + v_{k,k+1} f_{k+1}(n) + \cdots \qquad \text{où } v_{k,i} \in \mathbb{R}^p ;\ i > k$$

En effet supposons que l'on ait à l'ordre k-1 :

$$T_{k-1}^{(n)} = S + v_{k-1,k} \cdot f_k(n) + v_{k-1,k+1} \cdot f_{k+1}(n) + \cdots$$

et:

$$T_k^{(n)} = \frac{T_{k-1}^{(n+1)} - \frac{f_k(n+1)}{f_k(n)} \cdot T_{k-1}^{(n)}}{1 - \frac{f_k(n+1)}{f_k(n)}}$$

En faisant jouer à $T_{k-1}^{(n)}$ le rôle de S_n et $f_k(n)$ celui de $f_1(n)$ dans la demonstration précédente, on montre par élimination de $f_k(n)$ que si $b_{kk} \neq 1$, alors:

$$T_k^{(n)} = S + v_{k,k+1} \cdot f_{k+1}(n) + \cdots$$

D'où la propriété:

<u>Propriété 2</u> :

Si $S_n = S + \sum_{i \geq 1} v_i \cdot f_i(n)$; $\forall n$

Si les $f_i(n)$ satisfont C_1), C_2) et C_3) et si $b_{ii} \neq 1$

alors:

$$T_k^{(n)} = S + v_{k,k+1} f_{k+1}(n) + \cdots \quad \forall n, \forall k \geq 1.$$

<u>Exemple</u> :

Soit (S_n) donnée par:

$$S_n = S + v_1 h_n + v_2 h_n^2 + v_3 h_n^3 + \cdots$$

où : $v_i \in \mathbb{R}^p$ $\forall i \geq 1$ et $h_n = \frac{h}{2^n}$

On prend : $f_i(n) = h_n^i$

les f_i verifient:

$$c_1) \quad f_i(n+1) = h_{n+1}^{i} = \left(\frac{h}{2^{n+1}}\right)^{i} = \frac{1}{2^i} \cdot f_i(n).$$

$$c_2) \quad f_i(n) \cdot f_j(n) = h_n^{i} \cdot h_n^{j} = h_n^{i+j} = f_{i+j}(n).$$

$$c_3) \quad \frac{f_j(n)}{f_i(n)} = \frac{h_n^{j}}{h_n^{i}} = h_n^{j-i} = f_{j-i}(n).$$

<u>donc</u>:

$$T_k^{(n)} = S + \sigma_{k/k+1} \cdot \left(\frac{h}{2^n}\right)^{k+1} + \cdots$$

- Le <u>Δ^2 topologique</u>:

on pose:

$$f_1(n) = \langle \phi, \Delta S_n \rangle \quad \text{où} \quad \phi \in B^{*}$$

donc:

$$T_1^{(n)} = S_n - \frac{\langle \phi, \Delta S_n \rangle}{\langle \phi, \Delta^2 S_n \rangle} \cdot \Delta S_n$$

C'est le Δ^2 d'Aitken topologique [3].

- Nous allons voir par la suite un quatrième choix des $f_i(n)$ qui sera considéré comme une généralisation du Δ^2 d'Aitken topologique.

4 Algorithme (T') :

La règle de l'algorithme (T) s'écrit sous la forme :

$$T_k^{(n)} = T_{k-1}^{(n)} + \alpha_k^{(n)} \cdot \Delta T_{k-1}^{(n)}$$

avec :

$$\alpha_k^{(n)} = - \frac{f_k(n)}{\Delta f_k(n)} .$$

On suppose que (S_n) converge vers S et que à l'iteration $k-1$: $\langle \phi_k , T_{k-1}^{(n)} - S \rangle = o(1)$ $(n \to \infty)$. Nous allons choisir la suite $(f_k(n))_n$ de telle sorte que : (*) $\langle \phi_k , T_k^{(n)} - S \rangle = o\left(\langle \phi_k , T_{k-1}^{(n)} - S \rangle \right)$ $(n \to \infty)$.

Supposons en plus que :

$$(**) \quad \lim_{n \to \infty} \frac{\langle \phi_k , T_{k-1}^{(n+1)} - S \rangle}{\langle \phi_k , T_{k-1}^{(n)} - S \rangle} = b'_k \neq 1 .$$

On a :

$$\frac{\langle \phi_k , T_k^{(n)} - S \rangle}{\langle \phi_k , T_{k-1}^{(n)} - S \rangle} = 1 + \alpha_k^{(n)} \cdot \frac{\langle \phi_k , \Delta T_{k-1}^{(n)} \rangle}{\langle \phi_k , T_{k-1}^{(n)} - S \rangle}$$

Pour que (*) soit satisfaite, il faut que :

$$\lim_{n \to \infty} \alpha_k^{(n)} = - \lim_{n \to \infty} \frac{\langle \phi_k , T_{k-1}^{(n)} - S \rangle}{\langle \phi_k , \Delta T_{k-1}^{(n)} \rangle} = - \lim_{n \to \infty} \frac{\langle \phi_k , \Delta T_{k-1}^{(n)} \rangle}{\langle \phi_k , \Delta^2 T_{k-1}^{(n)} \rangle}$$

(d'après (**) et le théorème (01))

Ceci nous conduit à prendre comme estimation de $\alpha_k^{(n)}$ la quantité :

$$\alpha_k^{(n)} = - \frac{\langle \phi_k, \Delta T_{k-1}^{(n)} \rangle}{\langle \phi_k, \Delta^2 T_{k-1}^{(n)} \rangle}$$

Ce qui revient à prendre : $f_k(n) = \langle \phi_k, \Delta T_{k-1}^{(n)} \rangle$.

Nous obtenons alors l'algorithme :

l'algorithme)
$$\begin{cases} T_0^{(n)} = S_n \\ T_k^{(n)} = T_{k-1}^{(n)} - \dfrac{\langle \phi_k, \Delta T_{k-1}^{(n)} \rangle}{\langle \phi_k, \Delta^2 T_{k-1}^{(n)} \rangle} \cdot \Delta T_{k-1}^{(n)} \quad ; \; k \geq 1. \end{cases}$$

Dans la suite nous appelerons cet algorithme le T'-algorithme, qui n'est autre que le δ^2 topologique itéré lorsque $\phi_k \equiv \phi, \forall k$.

Remarque :

Considerons deux transformations de suites vectorielles, t_1 et t_2 :

$$t_1 : S_n \longrightarrow t_1^{(n)}$$

$$t_2 : S_n \longrightarrow t_2^{(n)}$$

Soit T_n la transformation composite de t_1 et t_2, définie par.

$$T_n = (1 - a_n)\, t_1^{(n)} + a_n t_2^{(n)}$$

avec :

$$a_n = \begin{cases} - \dfrac{\langle \phi, \Delta t_1^{(n)} \rangle}{\langle \phi, \Delta t_2^{(n)} - \Delta t_1^{(n)} \rangle} & \text{si } \Delta t_1^{(n)} \neq \Delta t_2^{(n)} \\ 0 & \text{sinon} \end{cases}$$

où $\phi \in \mathbb{R}^p$

On note dans ce cas : $T_n = C\left(t_1^{(n)}, t_2^{(n)}\right)$

- Si $t_1^{(n)} = S_n$ et $t_2^{(n)} = S_{n+1}$ alors T_n est le Δ^2 d'Aitken topologique.
- Si $t_1^{(n)} = E_{k-1}^{(n)}$ et $t_2^{(n)} = t_1^{(n)} + \alpha_k \cdot g_{k-1,k}^{(n)}$ avec $\alpha_k \neq 0$, alors on obtient le E-algorithme topologique.
- Si $t_1^{(n)} = T_{k-1}^{(n)}$ et $t_2^{(n)} = T_{k-1}^{(n)}$ et $\phi = \phi_k$, alors nous obtenons l'algorithme $(\dot{T})$ précédent.

propriété 3 :

Si $\|S_n - S\| = o(1)$ et si pour $k \geq 1$: $\lim\limits_{n\to\infty} \dfrac{\langle \phi_k, T_{k-1}^{(n+1)} - S\rangle}{\langle \phi_k, T_{k-1}^{(n)} - S\rangle} = a_k \neq 1$

alors, $\forall k \geq 1$:

i) $\|T_k^{(n)} - S\| = o(1) \quad (n \to \infty)$

ii) $\langle \phi_k, T_k^{(n)} - S\rangle = o\left(\langle \phi_k, T_{k-1}^{(n)} - S\rangle\right) \quad (n \to \infty)$.

dem :

Il suffit d'écrire la règle de l'algorithme $(\dot{T})$ et d'utiliser le résultat du théorème (01) ∎

remarque 3 :

Pour une suite quelconque, il est difficile de vérifier les hypothèses précédentes qui dépendent de la limite S. Nous verrons, pour les classes de suites que nous étudierons, que ces hypothèses sont faciles à vérifier.

III Applications de l'algorithme (T') :

Considérons les suites (S_n) de $\mathbb{R}^p$ qui sont de la forme:

$$(\text{III}.1) \qquad S_n = S + \sum_{i \geq 1} \lambda_i^n . y_i$$

où

les λ_i sont des scalaires.

les y_i sont des vecteurs.

Soit (H) l'hypothèse suivante.

$$(H) \begin{cases} \lambda_i \neq 1 \ ; \ i \geq 1 \\ |\lambda_1| > |\lambda_2| > \dots > |\lambda_k| > |\lambda_{k+1}| > \dots \end{cases}$$

Nous allons appliquer l'algorithme (T') à la suite (S_n) et nous montrerons que sous certaines hypothèses, $T_k^{(n)}$ converge plus vite que S_n, $\forall k \geq 1$. Ce qui nous permet aussi de calculer les λ_i et les vecteurs y_i ; $i \geq 1$.

On suppose dans toute la suite que les formes linéaires ϕ_i, $i \geq 1$ sont bornées.

Nous rappelons que L est un ensemble de p fonctionnelles linéaires, bornées et indépendantes.

proposition 4 :

Si on applique l'algorithme (T) à une suite (S_n) de la forme (II.1), vérifiant (H) avec $|\lambda_1|<1$ et $\langle \phi_k, y_k\rangle \neq 0, \forall k$, alors :

$$\forall k \geq 0 \qquad T_k^{(n)} - S \sim \sum_{i\geq k+1} \lambda_i^{n+k} \cdot y_i \qquad (n\to\infty).$$

dem :

par récurrence sur k.

• $k=0$ $\qquad T_0^{(n)} - S = \sum_{i\geq 1} \lambda_i^n \cdot y_i$

• supposons que la proposition est vraie jusqu'à l'ordre k :

$$T_k^{(n)} - S \sim \sum_{i\geq k+1} \lambda_i^{n+k} \cdot y_i \qquad (n\to\infty).$$

donc : $\forall \phi \in L$, on a :

$$\langle \phi, T_k^{(n)} - S\rangle \sim \sum_{i\geq k+1} \lambda_i^{n+k} \cdot \langle \phi, y_i\rangle \qquad (n\to\infty)$$

donc :

$$\langle \phi, \Delta T_k^{(n)}\rangle \sim \sum_{i\geq k+1} \lambda_i^{n+k} (\lambda_i - 1) \cdot \langle \phi, y_i\rangle \qquad (n\to\infty)$$

et

$$\langle \phi, \Delta^2 T_k^{(n)}\rangle \sim \sum_{i\geq k+1} \lambda_i^{n+k} \cdot (\lambda_i - 1)^2 \cdot \langle \phi, y_i\rangle \qquad (n\to\infty).$$

or :

$$T_k^{(n)} = T_{k-1}^{(n+1)} - \frac{\langle \phi_{k+1}, \Delta T_k^{(n+1)}\rangle}{\langle \phi_{k+1}, \Delta^2 T_k^{(n)}\rangle} \cdot \Delta T_k^{(n)}$$

et comme :

$$\langle \phi_{k+1}, \Delta T_k^{(n+1)}\rangle \sim \lambda_{k+1}^{n+k+1} (\lambda_{k+1} - 1) \cdot \langle \phi_{k+1}, y_{k+1}\rangle \times \left[1 + \sum_{i\geq k+2} \left(\frac{\lambda_i}{\lambda_{k+1}}\right)^{n+k+1} \cdot \frac{\lambda_i - 1}{\lambda_{k+1} - 1} \cdot \frac{\langle \phi_{k+1}, y_i\rangle}{\langle \phi_{k+1}, y_{k+1}\rangle} \right.$$

$$(H) \Longrightarrow \left| \frac{\lambda_i}{\lambda_{k+1}} \right| < 1 \quad \text{pour } i \geq k+2.$$

d'où :

$$\langle \phi, T_{k+1}^{(n)} - S\rangle \sim \sum_{i\geq k+1} \lambda_i^{n+k+1} \langle \phi, y_i\rangle - \frac{\lambda_{k+1}^{n+k+1}(\lambda_{k+1}-1)\langle \phi_{k+1}, y_{k+1}\rangle}{\lambda_{k+1}^{n+k}(\lambda_{k+1}-1)\langle \phi_{k+1}, y_{k+1}\rangle} \cdot \lambda_{k+1}^{n+k}(\lambda_{k+1}-1)\langle \phi, y_{k+1}\rangle$$

Et en simplifiant, nous obtenons :

$$\langle \phi, T_{k+1}^{(n)} - S\rangle \sim \sum_{i\geq k+1} \lambda_i^{n+k+1} \cdot \langle \phi, y_i\rangle - \lambda_{k+1}^{n+k+1} \langle \phi, y_{k+1}\rangle$$

$$\Longrightarrow \quad \langle \phi, T_{k+1}^{(n)} - S\rangle \sim \sum_{i\geq k+2} \lambda_i^{n+k+1} \langle \phi, y_i\rangle \quad (n\to\infty)$$

Soit :

$$\forall k\geq 0 : \quad T_k^{(n)} - S \sim \sum_{i\geq k+1} \lambda_i^{n+k} y_i \quad (n\to\infty). \square$$

Ceci nous permet de donner le résultat suivant :

<u>Proposition 5</u> :

Si on applique l'algorithme (T') à une suite (S_n) de la forme (III.1), vérifiant (H) avec $|\lambda_1| < 1$ et $\langle \phi_k, y_k\rangle \neq 0$, $\forall k$ alors :

$\forall k \geq 1$:

i) $$\frac{\langle \phi_k, \Delta T_{k-1}^{(n+1)}\rangle}{\langle \phi_k, \Delta T_{k-1}^{(n)}\rangle} \sim \lambda_k + O\left(\left(\frac{\lambda_{k+1}}{\lambda_k}\right)^{n+k}\right) \quad (n\to\infty)$$

ii) $$\lim_{n\to\infty} \frac{\Delta T_{k-1}^{(n)}}{\langle \phi_k, \Delta T_{k-1}^{(n)}\rangle} = \frac{y_k}{\langle \phi_k, y_k\rangle}$$

iii) $$\langle \phi_k, T_k^{(n)} - S\rangle = o\left(\langle \phi_k, T_{k-1}^{(n)} - S\rangle\right) \quad (n\to\infty)$$

dem:

i) nous avons montré dans la propriété précédente que sous les même hypothèses, on a pour tout $\phi \in L$:

$$\langle \phi, \Delta T_{k-1}^{(n)} \rangle \sim \lambda_k^{n+k-1} \cdot (\lambda_k - 1) \cdot \langle \phi, y_k \rangle \cdot \left[1 + \sum_{i \geq k+1} \left(\frac{\lambda_i}{\lambda_k} \right)^{n+k-1} \cdot \frac{\lambda_i - 1}{\lambda_k - 1} \cdot \frac{\langle \phi, y_i \rangle}{\langle \phi, y_k \rangle} \right]$$

donc:

$$\frac{\langle \phi_k, \Delta T_{k-1}^{(n+1)} \rangle}{\langle \phi_k, \Delta T_{k-1}^{(n)} \rangle} \sim \frac{\lambda_k^{n+k} (\lambda_k - 1) \cdot \langle \phi_k, y_k \rangle \cdot \left[1 + \sum_{i \geq k+1} \left(\frac{\lambda_i}{\lambda_k} \right)^{n+k-1} \frac{\lambda_i - 1}{\lambda_k - 1} \cdot \frac{\langle \phi_k, y_i \rangle}{\langle \phi_k, y_k \rangle} \right]}{\lambda_k^{n+k-1} \cdot (\lambda_k - 1) \cdot \langle \phi_k, y_k \rangle}$$

Après simplification et utilisation de l'hypothèse (H), nous obtenons:

$$\frac{\langle \phi_k, \Delta T_{k-1}^{(n+1)} \rangle}{\langle \phi_k, \Delta T_{k-1}^{(n)} \rangle} \sim \lambda_k + O\left(\left(\frac{\lambda_{k+1}}{\lambda_k} \right)^{n+k} \right) \qquad (n \to \infty).$$

ii) pour tout $\phi \in L$, on a:

$$\langle \phi, \Delta T_{k-1}^{(n)} \rangle \sim \lambda_k^{n+k-1} \cdot (\lambda_k - 1) \cdot \langle \phi, y_k \rangle$$

d'où :

$$\lim_{n \to \infty} \frac{\langle \phi, \Delta T_{k-1}^{(n)} \rangle}{\langle \phi_k, \Delta T_{k-1}^{(n)} \rangle} = \frac{\langle \phi, y_k \rangle}{\langle \phi_k, y_k \rangle}$$

& en prenant pour ϕ respectivement une base de $\mathbb{R}^p$ on obtient:

$$\lim_{n \to \infty} \frac{\Delta T_{k-1}^{(n)}}{\langle \phi_k, \Delta T_{k-1}^{(n)} \rangle} = \frac{y_k}{\langle \phi_k, y_k \rangle}$$

iii) D'après les résultats précédents on a:

$$\frac{\langle \phi, T_k^{(n)} - S\rangle}{\langle \phi, T_{k-1}^{(n)} - S\rangle} \sim \left(\frac{\lambda_{k+1}}{\lambda_k}\right)^{n+k-1} \cdot \lambda_{k+1} \cdot \frac{\langle \phi, y_{k+1}\rangle}{\langle \phi, y_k\rangle}$$

donc $\forall \phi \in L$:

$$\langle \phi, T_k^{(n)} - S\rangle = o\left(\langle \phi, T_{k-1}^{(n)} - S\rangle\right) \quad (n \to \infty).$$ □

<u>proposition 6</u>:

Soit (S_n) une suite de $\mathbb{R}^p$ de la forme (III.1) et vérifiant les hypothèses de la propriété précédente, alors:

$$\forall k \geq 1 : \quad \|T_k^{(n)} - S\| = o\left(\|T_{k-1}^{(n)} - S\|\right) \quad (n \to \infty).$$

<u>dem.</u>

On utilise le ii) de la proposition 5 et le théorème (Q2) et on obtient le résultat précédent. □

Ces résultats nous permettront par la suite de proposer une méthode de calcul des valeurs et vecteurs propres d'une matrice.
Comme toutes les méthodes du même type, cette méthode est interessante lorsqu'il s'agit de ne calculer que quelques valeurs propres de plus grands modules.

IV Méthodes de calcul des valeurs et vecteurs propres :

Soit A une matrice carrée réelle d'ordre p.

$\lambda_i ; i=1,\dots,p$ sont les valeurs propres de A et les $v_i, i=1,\dots,p$ les vecteurs propres correspondants.

On supposera que $\rho(A) = \max_{i=1,p} |\lambda_i| < 1$ et si cette condition n'est pas vérifiée pour A, on cherchera les valeurs propres de $B = \frac{1}{a} A$ avec $a > \rho(A)$ et on en déduira ceux de A.

Nous considérons la suite (S_n) définie à partir de S_0 donné, par : $S_{n+1} = A.S_n$.

$\rho(A) < 1 \Rightarrow S = \lim_{n\to\infty} S_n = 0$.

Nous avons alors le résultat suivant :

Théorème 1 :

Soit A une matrice carrée d'ordre p dont les valeurs propres vérifient (H). Si on applique (T') à la suite (S_n) définie par $S_{n+1} = AS_n$ et si $\langle \phi_k, y_k \rangle \neq 0$, $k=1,\dots,p$, avec :

$y_i = a_i v_i$ et $S_0 = a_1 v_1 + \dots + a_p v_p$, alors $\forall k \in \{0,\dots,p-1\}$:

$$i)\ \frac{\langle \phi_{k+1}, T_k^{(n+1)} \rangle}{\langle \phi_{k+1}, T_k^{(n)} \rangle} = \lambda_{k+1} + O\left(\left(\frac{\lambda_{k+2}}{\lambda_{k+1}} \right)^{n+k+1} \right)$$

$$ii)\ \lim_{n\to\infty} \frac{T_k^{(n)}}{\langle \phi_{k+1}, T_k^{(n)} \rangle} = \frac{v_{k+1}}{\langle \phi_{k+1}, v_{k+1} \rangle}$$

dem.

$$S_n = A S_{n-1} = \dots = A^n S_0 \quad , \forall n$$

$$S_0 = a_1 v_1 + \dots + a_p v_p$$

donc,

$$S_n = \sum_{i=1}^{p} a_i \lambda_i^n v_i \quad , \text{soit en posant } y_i = a_i v_i \quad ; i=1 \dots p$$

$$S_n = \sum_{i=1}^{p} \lambda_i^n . y_i$$

(S_n) a donc la même forme que les suites (III.1), avec: $s=0$ et $\lambda_i = 0$ pour $i > p$. Alors en faisant une démonstration analogue à celle de la propriété (5), on montre facilement le théorème.

Remarque 4:

Dans le théorème précédent $s=0$ et λ_{k+1} a comme approximation: $\dfrac{\langle \Phi_{k+2}, T_k^{(n+1)} \rangle}{\langle \Phi_{k+1}, T_k^{(n)} \rangle}$. Par contre lorsque $s \neq 0$ (propriété 5) λ_{k+2} a comme approximation: $\dfrac{\langle \Phi_{k+2}, \Delta T_k^{(n+1)} \rangle}{\langle \Phi_{k+2}, \Delta T_k^{(n)} \rangle}$.

Posons: $a_k^{(n)} = \dfrac{\langle \Phi_{k+2}, T_k^{(n+1)} \rangle}{\langle \Phi_{k+2}, T_k^{(n)} \rangle}$ et supposons que les hypothèses du théorème précédent sont vérifiées. Nous allons utiliser l'ε-algorithme scalaire pour accélérer la convergence de $(a_k^{(n)})_n$ pour k fixé. $k \in \{0, \dots, p-1\}$.

Théorème 9:

Si on applique l'ε-algorithme scalaire à $(a_k^{(n)})_n$ avec k fixé, alors

pour q fixé : $\varepsilon_{2q}^{(n)} \sim \lambda_{k+1} + O\left(\left(\frac{\lambda_{k+q+2}}{\lambda_{k+1}}\right)^{n+k}\right) \quad (n \to \infty)$.

dem:

Dans [28], WYNN a montré que si on applique l'ε-algorithme scalaire à une suite (S_n) de la forme:

$$S_n \sim a + \sum_{i \geq 1} a_i b_i^n \quad \text{avec } 1 > |b_1| > |b_2| > \dots$$

alors $\forall q$:

$$\varepsilon_{2q}^{(n)} \sim a + O\left(b_{q+1}^n\right) \quad (n \to \infty).$$

Or on montre d'après le théorème 2 que:

$$a_k^{(n)} \sim \lambda_{k+1} + \sum_{i \geq 1} a_i \cdot \left(\frac{\lambda_{i+k+1}}{\lambda_{k+1}}\right)^{n+k+2} \text{ avec: } 1 > \left|\frac{\lambda_{k+2}}{\lambda_{k+1}}\right| > \left|\frac{\lambda_{k+3}}{\lambda_{k+1}}\right| > \dots$$

Si on applique le résultat de WYNN à $(a_k^{(n)})_n$, avec k fixé

On obtient: $\varepsilon_{2q}^{(n)} \sim \lambda_{k+1} + O\left(\left(\frac{\lambda_{k+q+2}}{\lambda_{k+1}}\right)^{n+k+1}\right) \quad (n \to \infty)$ ∎

Remarques:

L'algorithme (T') nécessite le même nombre de termes de la suite que l'ε-algorithme vectoriel [3] pour le calcul de λ_k avec la même erreur, mais sur les exemples traités il paraît plus stable numériquement.

I Résolution des systèmes linéaires.

Considerons le système : $B.S = b$ où B est une matice réelle et régulière d'ordre p et $b \in \mathbb{R}^p$.

Posons $A = I - B$.

On suppose que A satisfait l'hypothèse (H) et que $\rho(A) < 1$.

Soit (S_n) la suite de $\mathbb{R}^p$ définie à partir de S_0 par:

$$S_{n+1} = A.S_n + b.$$

λ_i ; $i = 1 \dots p$ sont les valeurs propres de A.

v_i ; $i = 1, \dots, p$ sont les vecteurs propres correspondants.

On a alors le resultat suivant :

Théorème 3:

Si on applique l'algorithme (T') à la suite (S_n) definie précédement et si $\langle \phi_k, y_k \rangle \neq 0$, $k = 1, \dots, p$ où $y_i = a_i v_i$ avec $S_0 - S = a_1 v_1 + \dots + a_p v_p$

alors $\forall k \in \{1, \dots, p\}$:

i) $\lim\limits_{n \to \infty} T_k^{(n)} = S$

ii) $\langle \phi_k, T_k^{(n)} - S \rangle = o\left(\langle \phi_k, T_{k-1}^{(n)} - S \rangle\right) \quad (n \to \infty)$

iii) $\| T_k^{(n)} - S \| = o\left(\| T_{k-1}^{(n)} - S \| \right) \quad (n \to \infty)$.

dem:

$$\left.\begin{array}{l} S_{n+1} = AS_n + b \\ S = AS + b \end{array}\right. \Rightarrow S_{n+1} - S = A(S_n - S) = \cdots = A^{n+1}(S_0 - S)$$

Or $S_0 - S = \sum_{i=1}^{p} a_i v_i$ donc $S_n = S + \sum_{i=1}^{p} \lambda_i^n y_i$ où $y_i = a_i v_i$

(S_n) a la forme (III.1). En appliquant les propriétés 4 et 5 on obtient i), ii) et iii) du théorème. □

VI Exemples numériques:

1°) Calcul des valeurs propres:

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

Les valeurs propres exactes sont:

$\lambda_1 = 1.092388$; $\lambda_3 = 0.311148$

$\lambda_2 = 0.638491$; $\lambda_4 = 0.047971$

On prends:

$$\Phi_j = (0, \ldots, 0, \underset{\substack{\uparrow \\ j\text{-ème composante}}}{j}, 0 \ldots 0)^T$$

Les résultats obtenus sont :

App (λ_1)	App (λ_2)	App (λ_3)
[illegible]	[illegible]	[illegible]

Cette méthode est intéressante lorsqu'il s'agit de calculer quelques valeurs propres uniquement.
Pour l'exemple précédent, nous n'avons pas obtenu d'estimations intéressantes pour λ_4.

* **Exemple 2 :**

Considérons le système $A'.x = b$ où

$$A' = \begin{pmatrix} 0.78 & -0.02 & -0.12 & -0.14 \\ -0.02 & 0.86 & -0.04 & 0.06 \\ -0.12 & -0.04 & 0.72 & -0.08 \\ -0.14 & 0.06 & -0.08 & 0.74 \end{pmatrix} ; \quad b = \begin{pmatrix} 0.5 \\ 0.86 \\ 0.48 \\ 0.58 \end{pmatrix}$$

On applique le procédé (T') à la suite (S_n) definie par :

$$(*) \begin{cases} S_{n+1} = AS_n + b \\ S_0 = 0 \\ A = I - A' \end{cases}$$

Les valeurs propres de A sont :

$\lambda_1 = 0.5$; $\lambda_2 = 0.26$; $\lambda_3 = 0.52$; $\lambda_4 = 0.54$

On prends pour Φ_j les formes :

$$\Phi_j = (0, \dots, 0, \underset{j}{1}, \dots 0) \quad ; \; j = 1, \dots, 4$$

On obtient les résultats suivants :

n	$\|T_1^{(n)} - S\|$	$\|T_2^{(n)} - S\|$	$\|T_3^{(n)} - S\|$	$\|T_4^{(n)} - S\|$
1	[illegible]	[illegible]	[illegible]	[illegible]
2	.61588707[illegible]D-01	.7278872[illegible]D-11	[illegible]	[illegible]
3	.16332[illegible]D-01	[illegible]	[illegible]	[illegible]
4	.38157[illegible]D-02	.18078[illegible]D-02	[illegible]	[illegible]
5	.85[illegible]D-03	.38074[illegible]D-04	.22[illegible]D-08	[illegible]
6	.21278[illegible]D-03	.33015[illegible]D-05	.12[illegible]D-07	.38867[illegible]D-12
7	.50580[illegible]D-04	.41[illegible]	.77719[illegible]	[illegible]
8	.12[illegible]D-04	.50[illegible]	[illegible]	[illegible]
9	.25[illegible]D-05	.31[illegible]	.27777[illegible]	[illegible]
10	.70[illegible]D-06	.73[illegible]	[illegible]	[illegible]
11	.12[illegible]D-06	[illegible]	[illegible]	[illegible]
12	.40[illegible]D-07	[illegible]	[illegible]	[illegible]
13	.57[illegible]D-08	[illegible]		

- Chapitre IV -

ALgorithme général de projection et accélération de la convergence de suites de vecteurs.

Introduction :

Nous introduisons, dans ce chapitre, une transformation vectorielle S_k, regroupant la plupart des méthodes d'extrapolation.

Après la présentation de cette transformation, nous donnerons quelques propriétés algébriques et donnerons trois algorithmes de calcul de $S_k^{(n)}$ pour n fixé. Nous nous limiterons, dans le troisième paragraphe, à une transformation particulière résultant de S_k. Nous proposerons, dans ce cas, un algorithme récursif appelé le $(S-\beta)$-algorithme dont nous étudierons les propriétés. Celui-ci sera appliqué aux suites produites par des itérations linéaires; ce qui nous permettra de donner des méthodes de calcul des éléments propres d'une matrice et des méthodes de résolution des systèmes linéaires et non linéaires.

I- Présentation de la méthode et propriétés :

1 Description de la méthode :

Soit B un espace vectoriel sur K ($\mathbb{R}$ ou $\mathbb{C}$), normé et B^* son dual. Soit $m \in \mathbb{N}$; $k \in \mathbb{N}$ avec $k \leq \dim B = p$.

$\{\phi_{m+j}\}_{j=1,\dots,k}$ est un système libre de B^*.

$(g_i(n))_n$; $i=1,\dots,k$ des suites d'éléments de B.

On considère dans B les suites (S_n) qui sont de la forme :

$$(I.1) \qquad S_n = s + \sum_{i=1}^{k} a_i g_i(n) \quad ; n \geq 0 \text{ et } a_i \in \mathbb{K}.$$

Connaissant S_n, S_{n+1} ; $g_i(n), g_i(n+1)$; $i=1,\dots,k$, on donnera l'expression de s.

Pour obtenir s dans le chapitre II, on a écrit l'équation (I.1) pour $n,\dots,n+k-1$, appliqué l'opérateur Δ aux deux membres de ces équations et multiplié par un élément ϕ de B^*.

Nous procéderons différemment dans ce qui suit, en prenant une seule équation (I.1). Nous appliquons ensuite l'opérateur Δ aux deux membres de (I.1) :

$$(I.2) \qquad \Delta S_n = \sum_{i=1}^{k} a_i \, \Delta g_i(n)$$

Multiplions ensuite (I.2) par ϕ_{m+j} ; $j = 1, \dots, k$

$$\langle \phi_{m+j}, \Delta S_n \rangle = \sum_{i=1}^{k} a_i \langle \phi_{m+j}, \Delta g_i(n) \rangle \quad ; j = 1, \dots, k$$

Ce qui donne sous-forme matricielle :

$$(I.3) \quad \begin{pmatrix} \langle \phi_{m+1}, \Delta g_1(n) \rangle & \dots & \langle \phi_{m+1}, \Delta g_k(n) \rangle \\ \dots & \dots & \dots \\ \langle \phi_{m+k}, \Delta g_1(n) \rangle & \dots & \langle \phi_{m+k}, \Delta g_k(n) \rangle \end{pmatrix} \cdot \begin{pmatrix} a_1 \\ \vdots \\ a_k \end{pmatrix} = \begin{pmatrix} \langle \phi_{m+1}, \Delta S_n \rangle \\ \vdots \\ \langle \phi_{m+k}, \Delta S_n \rangle \end{pmatrix}$$

et $\quad S = S_n - \sum_{i=1}^{k} a_i \, g_i(n)$

Si (S_n) n'a pas exactement la forme (I.1), nous remplacerons S par ${}_mS_k^{(n)}$ et : $\quad {}_mS_k^{(n)} = S_n - \sum_{i=1}^{k} a_i g_i(n)$ où les a_i sont solutions de (I.3).

Posons :

$$g(n) = \left(g_1(n), \dots, g_k(n) \right) \quad ; \quad a = \left(a_1, \dots, a_k \right)^T$$

$$u_n = \left(\langle \phi_{m+1}, \Delta S_n \rangle, \dots, \langle \phi_{m+k}, \Delta S_n \rangle \right)^T$$

(I.3) s'écrit alors : $\quad A_n \cdot a = - u_n \quad$ où A_n est la matrice de (I3).

Si on pose :

$$g(n) * a = \sum_{i=1}^{k} a_i \, g_i(n)$$

et si A_n est inversible, alors :

$${}_mS_k^{(n)} = S_n - g(n) * A_n^{-1} \cdot u_n$$

Ce qui nous donne, d'après le complement de Schur [8], ${}_mS_k^{(n)}$ sous

forme d'un rapport de deux déterminants :

$$
{}_mS_k^{(n)} = \frac{\begin{vmatrix} S_n & g_1(n) & \cdots & g_k(n) \\ \langle \phi_{m+1}, \Delta S_n \rangle & \langle \phi_{m+1}, \Delta g_1(n) \rangle & \cdots & \langle \phi_{m+1}, \Delta g_k(n) \rangle \\ \cdots & \cdots & \cdots & \cdots \\ \langle \phi_{m+k}, \Delta S_n \rangle & \langle \phi_{m+k}, \Delta g_1(n) \rangle & \cdots & \langle \phi_{m+k}, \Delta g_k(n) \rangle \end{vmatrix}}{\begin{vmatrix} \langle \phi_{m+1}, \Delta g_1(n) \rangle & \cdots & \langle \phi_{m+1}, \Delta g_k(n) \rangle \\ \cdots & \cdots & \cdots \\ \langle \phi_{m+k}, \Delta g_1(n) \rangle & \cdots & \langle \phi_{m+k}, \Delta g_k(n) \rangle \end{vmatrix}}
$$

<u>Remarque 1</u> :

m sera pris soit égale à zéro soit à n. Et nous noterons alors ${}_mS_k^{(n)} = S_k^{(n)}$.

I.2 <u>Quelques choix des $g_i(n)$</u> :

a) <u>$m = n$</u>

* $g_i(n) = \Delta S_{n+i-1}$ et $\phi_{n+j} = \Delta S_{n+j-1}$; $i, j = 1, \ldots, k$

c'est la MPE [26], [12].

* $g_i(n) = \Delta S_{n+i-1}$ et $\phi_{n+j} = \Delta^2 S_{n+j-1}$; $i, j = 1, \ldots, k$

c'est la RRE [26], [29].

* $g_i(n) = \Delta^q S_{n+i-1}$ et $\phi_{n+j} = \Delta S_{n+j-1}$; $i, j = 1, \ldots, k$ et $q \in \mathbb{N} - \{0\}$

nous obtenons une famille de procédés proposé par GERMAIN-BONNE [17].

b) $\underline{\underline{m=0}}$

* $g_i(n) = \Delta S_{n+j-1}$ et $\{\phi_j\}_{j=1,\dots,k}$ un système libre de $\mathcal{B}^*$, c'est la MMPE [26], [22].

* $S_n = t_1^{(n)}$; $g_i(n) = t_2^{(n+i-1)} - t_1^{(n+i-1)}$; $i=1,\dots,p$ et $\{\phi_j\}_{j=1,\dots,p}$ la base canonique de

$$t_1 : S_n \longrightarrow t_1^{(n)}$$
$$t_2 : S_n \longrightarrow t_2^{(n)}$$
deux transformations de suites.

C'est la seconde généralisation de la méthode composite vectorielle [11].

Dans toute la suite nous appelerons transformation $\mathcal{S}_k$, la transformation : $\mathcal{S}_k : S_n \longrightarrow S_k^{(n)}$.
où $S_k^{(n)}$ est donnée par (I.4).
Remarquons que le calcul de $S_k^{(n)}$ nécessite la connaissance de S_n, S_{n+1} ; $g_i(n)$, $g_i(n+1)$, $i=1,\dots,k$ et les $\{\phi_{m+j}\}_{j=1,\dots,k}$.

Avant de voir des algorithmes de calcul de $S_k^{(n)}$ pour n fixé, nous allons tout d'abord donner quelques propriétés de la transformation $\mathcal{S}_k$.

On notera Φ la matrice dont les colonnes sont $\phi_{m+1},\dots,\phi_{m+k}$.

I.3 - Propriétés :

Propriété 1 :

$S_k^{(n)}$ existe si et seulement si.

$\Phi^T . G_n$ est inversible. Où :

$\Phi = [\phi_{m+1}, \dots, \phi_{m+k}]$ et $G_n = [\Delta g_1(n), \dots, \Delta g_k(n)]$.

dém: $S_k^{(n)}$ existe ssi (I.3) admet une solution soit si $\det(\Phi^T . G_n) \neq 0$ □

Nous supposerons par la suite que cette condition est vérifiée pour tout (m, n, k).

Proposition 2 :

Si $S_n = S + \sum_{i=1}^{k} a_i g_i(n)$

alors : $S_k^{(n)} = S$

dém: Par construction de $S_k^{(n)}$ □.

Si les $g_i(n)$ satisfont une des trois conditions (C) (chap II) alors on a le résultat suivant :

Proposition 3 :

Si l'application de S_k aux suites S_n et $aS_n + b$, où $a \in \mathbb{K}$ et $b \in B$, fournit les quantités $S_k^{(n)}$ et $\tilde{S}_k^{(n)}$, alors :

$$\tilde{S}_k^{(n)} = a S_k^{(n)} + b .$$

dem:

$$\tilde{S}_k^{(n)} = \begin{vmatrix} a s_n + b & g_1(n) & \cdots & g_k(n) \\ a\langle \phi_{m+1}, \Delta s_n \rangle & \langle \phi_{m+1}, \Delta g_1(n) \rangle & \cdots & \langle \phi_{m+1}, \Delta g_k(n) \rangle \\ \text{———} & \text{———} & \text{———} & \text{———} \\ a\langle \phi_{m+k}, \Delta s_n \rangle & \langle \phi_{m+k}, \Delta g_1(n) \rangle & \cdots & \langle \phi_{m+k}, \Delta g_k(n) \rangle \end{vmatrix} \Big/ D_k^{(n)}$$

où $D_k^{(n)}$ est le déterminant obtenu en supprimant la première ligne et la première colonne. Et en décomposant la première colonne du numérateur on obtient: $\tilde{S}_k^{(n)} = a\, S_k^{(n)} + b$. $\square$.

Avant de voir en détail certaines transformations résultant de S_k par un choix particulier des $g_i(n)$, nous allons tout d'abord appliquer le GRPA et le RPA pour le calcul récursif de $S_k^{(n)}$ pour n fixé et k variable. Le premier sera appliqué pour m fixé indépendant de n et le second pour $m = n$ ou m indépendant de n fixé.

On notera :

$\langle\ ,\ \rangle_{2p}$ le produit scalaire usuel dans $\mathbb{R}^{2p}$.

$\langle\ ,\ \rangle_{p}$ le produit scalaire usuel dans $\mathbb{R}^{p}$.

Remarques:

Lorsque $g_i(n) = \Delta s_{n+i-1}$, nous donnerons un algorithme de calcul de $S_k^{(n)}$ avec k et n variables.

II. Algorithmes de calcul de $S_k^{(n)}$; n fixé :

II.1 CRPA ; m fixé indépendant de n :

On pose :

$$g_0(n) = s_n \quad ; \quad X_i = \begin{pmatrix} g_i(n) \\ g_i(n+1) \end{pmatrix} \; ; i = 0, \dots, k.$$

On définit sur $\mathbb{R}^{2p}$ les formes linéaires z_j par :

Si $V = \begin{pmatrix} V_1 \\ V_2 \end{pmatrix} \in \mathbb{R}^{2p}$ avec V_1 et V_2 deux vecteurs de $\mathbb{R}^p$:

$$\langle z_j, V \rangle_{2p} = \langle \phi_{m+j}, V_2 - V_1 \rangle_p \quad ; \; j = 1, \dots, k$$

donc :

$$\langle z_j, X_i \rangle_{2p} = \langle \phi_{m+j}, \Delta g_i(n) \rangle_p \quad ; j = 1, \dots, k \; ; \; i = 0, \dots, k.$$

Si $e_k^{(i)}$ est le vecteur de $\mathbb{R}^{2p}$ obtenue par application du CRPA à $X_0, \dots, X_k$ alors : $\left(S_k^{(n)}\right)_q = \left(e_k^{(i)}\right)_q \quad ; \; q = 1, \dots, p.$

$$(\text{II}.1) \quad \left\| \begin{array}{l} e_0^{(i)} = X_i \quad ; \; i = 1, \dots, k \\ e_k^{(i)} = e_{k-1}^{(i)} - \dfrac{\langle z_k, e_{k-1}^{(i)} \rangle}{\langle z_k, e_{k-1}^{(i+1)} \rangle} \cdot e_{k-1}^{(i+1)} \quad ; \; i \geq 0 \text{ et } k \geq 1. \end{array} \right.$$

Le calcul de $S_k^{(n)}$ pour n fixé par (II.1) nécessite un total d'opérations de l'ordre de $2pk^2$ (lorsque p est grand devant k).

Lorsque m est quelconque, pouvant donc dépendre de n, on ne peut pas appliquer le CRPA, mais par contre on peut appliquer le RPA :

I.2 RPA ; m quelconque :

On pose :

$$y = \begin{pmatrix} S_n \\ S_{n+1} \end{pmatrix} \quad ; \quad X_i = \begin{pmatrix} g_i(n) \\ g_i(n+1) \end{pmatrix} \quad ; i = 1, \dots, k.$$

Les formes z_{m+j} sur $\mathbb{R}^{2p}$ sont definies par : $\langle z_{m+j}, V \rangle_{2p} = \langle \Phi_{m+j}, V_2 - V_1 \rangle_p$.

Si on applique le RPA à y ; $X_1, \dots, X_k$ nous obtenons le vecteur E_k de $\mathbb{R}^{2p}$ donné par :

$$E_k = \begin{vmatrix} y & X_1 & \cdots & X_k \\ \langle z_{m+1}, y \rangle & \langle z_{m+1}, X_1 \rangle & \cdots & \langle z_{m+1}, X_k \rangle \\ \hline \langle z_{m+k}, y \rangle & \langle z_{m+k}, X_1 \rangle & \cdots & \langle z_{m+k}, X_k \rangle \end{vmatrix} \Bigg/ \begin{vmatrix} \langle z_{m+1}, X_1 \rangle & \cdots & \langle z_{m+1}, X_k \rangle \\ \hline \langle z_{m+k}, X_1 \rangle & \cdots & \langle z_{m+k}, X_k \rangle \end{vmatrix}$$

qui se calcule récursivement par :

$$(\mathrm{I}.2) \quad \left\| \begin{array}{l} E_0 = y \; ; \quad G_{0,i} = X_i \quad ; i = 1, \dots, k \\[2ex] E_k = E_{k-1} - \dfrac{\langle z_{m+k}, E_{k-1} \rangle}{\langle z_{m+k}, G_{k-1,k} \rangle} \cdot G_{k-1,k} \\[2ex] G_{k,i} = G_{k-1,i} - \dfrac{\langle z_{m+k}, G_{k-1,i} \rangle}{\langle z_{m+k}, G_{k-1,k} \rangle} \cdot G_{k-1,k} \cdot \qquad i > k > 0 \end{array} \right.$$

Or :

$$\langle z_{m+j}, X_i \rangle = \langle \Phi_{m+j}, \Delta g_i(n) \rangle \; ; \quad i = 1, \dots, k \quad ; j = 1, \dots, k.$$

$$\langle z_{m+j}, y \rangle = \langle \Phi_{m+j}, \Delta S_n \rangle \quad ; \quad j = 1, \dots, k$$

donc,

$$E_k = \begin{pmatrix} S_k^{(n)} \\ \bar{S}_k^{(n)} \end{pmatrix}$$

où

$\bar{S}_k^{(n)}$ est obtenue à partir de $S_k^{(n)}$ en remplaçant la première ligne du numérateur de cette dernière quantité par :

$$S_{n+1},\ g_1(n+1),\ \dots,\ g_k(n+1).$$

posons : $\alpha_k^{(n)} = \bar{S}_k^{(n)} - S_k^{(n)}$

alors :

$$\alpha_k^{(n)} = \begin{vmatrix} \Delta S_n & \Delta g_1(n) & \dots & \Delta g_k(n) \\ \langle \phi_{m+1}, \Delta S_n \rangle & \langle \phi_{m+1}, \Delta g_1(n) \rangle & \dots & \langle \phi_{m+1}, \Delta g_k(n) \rangle \\ \dots & \dots & \dots & \dots \\ \langle \phi_{m+k}, \Delta S_n \rangle & \langle \phi_{m+k}, \Delta g_1(n) \rangle & \dots & \langle \phi_{m+k}, \Delta g_k(n) \rangle \end{vmatrix} \Bigg/ \begin{vmatrix} \langle \phi_{m+1}, \Delta g_1(n) \rangle & \dots & \langle \phi_{m+1}, \Delta g_k(n) \rangle \\ \dots & \dots & \dots \\ \langle \phi_{m+k}, \Delta g_1(n) \rangle & \dots & \langle \phi_{m+k}, \Delta g_k(n) \rangle \end{vmatrix}$$

pour n fixé on peut calculer récursivement $\alpha_k^{(n)}$ en utilisant le RPA appliqué à $y' = \Delta S_n$; $X'_i = \Delta g_i(n)$; $i = 1, \dots, k$.

Notons $h_{k,i}$ le vecteur de $\mathbb{R}^p$ obtenu à partir de $\alpha_k^{(n)}$ en remplaçant dans celle-ci la première colonne du numérateur par : $\left(\Delta g_i(n), \langle \phi_{m+1}, \Delta g_i(n) \rangle, \dots, \langle \phi_{m+k}, \Delta g_i(n) \rangle \right)^T$.

Dans l'algorithme (II.2), $G_{k,i}$ peut s'écrire sous la forme :

$$G_{k,i} = \begin{pmatrix} g_{k,i} \\ \bar{g}_{k,i} \end{pmatrix} \qquad ; i > k > 0$$

où $g_{k,i}$ est obtenu à partir de l'expression de $S_k^{(n)}$ en remplaçant

Dans l'expression de cette dernière la première colonne du numérateur par: $\left(g_i(n), \langle \phi_{m+1}, \Delta g_i(n) \rangle, \ldots, \langle \phi_{m+k}, \Delta g_i(n) \rangle \right)^T$

alors:

$$h_{k,i} = \bar{g}_{k,i} - g_{k,i}$$

Reprenons l'algorithme (II.2) et considérons que les p. premières composantes des vecteurs E_k et $G_{k,i}$, on obtient alors l'algorithme:

$$\boxed{\begin{array}{ll}
S_0^{(n)} = S_n & ; \; g_{0,i} = g_i(n) \\
\alpha_0^{(n)} = \Delta S_n & ; \; h_{0,i} = \Delta g_i(n).
\end{array}}$$

$$(2.2) \quad \left.\begin{array}{l}
S_k^{(n)} = S_{k-1}^{(n)} - \dfrac{\langle \phi_{m+k}, \alpha_{k-1}^{(n)} \rangle}{\langle \phi_{m+k}, h_{k-1,k} \rangle} \cdot g_{k-1,k} \\[2ex]
\alpha_k^{(n)} = \alpha_{k-1}^{(n)} - \dfrac{\langle \phi_{m+k}, \alpha_{k-1}^{(n)} \rangle}{\langle \phi_{m+k}, h_{k-1,k} \rangle} \cdot h_{k-1,k}
\end{array}\right\} k > 0$$

$$\left.\begin{array}{l}
g_{k,i} = g_{k-1,i} - \dfrac{\langle \phi_{m+k}, h_{k-1,i} \rangle}{\langle \phi_{m+k}, h_{k-1,k} \rangle} \cdot g_{k-1,k} \\[2ex]
h_{k,i} = h_{k-1,i} - \dfrac{\langle \phi_{m+k}, h_{k-1,i} \rangle}{\langle \phi_{m+k}, h_{k-1,k} \rangle} \cdot h_{k-1,k}.
\end{array}\right\} i > k > 0.$$

Rque 3:

Si on pose $g_i(n) = \Delta S_{n+i-1}$, l'algorithme précédent n'est autre que celui donné par BENEU [2].

Le calcul de $S_k^{(n)}$ pour, n fixé, nécessite là aussi $2p(k^2+k)$ opérations arithmétiques.

I.3. Algorithme (I.3) :

Soit $n=N$ fixé. On suppose que k est fixé.

On a :

$$S_k^{(N)} = S_N - \sum_{i=1}^{k} a_{k,i}^{(N)} \cdot g_i(N)$$

où

$a_{k,1}^{(N)}, \ldots, a_{k,k}^{(N)}$ sont solutions du systèmes (I.3). Nous utilisons alors la méthode (A) du chapitre (I) pour résoudre (I.3) à l'étape k. Dans ce cas, le nombre d'opérations nécessaires pour le calcul de $S_k^{(N)}$ est de l'ordre de $\frac{2}{3}k^3+2kp$. Et il faudra ajouter le nombre d'opérations nécessaires pour construire la matrice du système (I.3).

III Etude du cas : $g_i(n) = \Delta S_{n+i-1}$, $m=0$

Nous commencerons ce paragraphe par quelques résultats théoriques et nous proposerons ensuite un algorithme de calcul de $S_k^{(n)}$, pour k et n variables, qui sera appelé le (S-β)-algorithme et dont nous donnerons les propriétés algébriques de convergence et d'accélération de la convergence. Nous l'appliquerons ensuite à certaines classes de suites de vecteurs et proposerons une méthode de calcul des éléments propres d'une matrice. Nous l'utiliserons enfin pour la résolution des systèmes linéaires et non linéaires.

Si $g_i(n) = \Delta S_{n+i-1}$ et $m=0$, alors :

$$S_k^{(n)} = \frac{\begin{vmatrix} S_n & \Delta S_n & \cdots & \Delta S_{n+k-1} \\ \langle \phi_1, \Delta S_n \rangle & \langle \phi_1, \Delta^2 S_n \rangle & \cdots & \langle \phi_1, \Delta^2 S_{n+k-1} \rangle \\ \cdots & \cdots & \cdots & \cdots \\ \langle \phi_k, \Delta S_n \rangle & \langle \phi_k, \Delta^2 S_n \rangle & \cdots & \langle \phi_k, \Delta^2 S_{n+k-1} \rangle \end{vmatrix}}{\begin{vmatrix} \langle \phi_1, \Delta^2 S_n \rangle & \cdots & \langle \phi_1, \Delta^2 S_{n+k-1} \rangle \\ \cdots & \cdots & \cdots \\ \langle \phi_k, \Delta^2 S_n \rangle & \cdots & \langle \phi_k, \Delta^2 S_{n+k-1} \rangle \end{vmatrix}} \quad ; n \geq 0, k \geq 1.$$

Remarque 4 :

Si $\phi_j = e_j$ $j=1,\dots,p$ alors $S_p^{(n)}$ n'est autre que la transformation d'Henrici [20].

1. Propriétés :

Proposition 4 :

Si $S_n = S + \sum_{i=1}^{k} a_i \, \Delta S_{n+i-1}$

alors :

$$S_k^{(n)} = S$$

dem : prop 2 avec $g_i(n) = \Delta S_{n+i-1}$; $i = 1, \dots, k$ □

Proposition 5 :

Si $\sum_{i=0}^{k} a_i (S_{n+i} - S) = 0$ où $a_i \in \mathbb{R}$; $i = 0, \dots, k$

avec :

$\sum_{i=0}^{k} a_i \neq 0$ et $a_k \neq 0$ alors :

$$S_k^{(n)} = S.$$

dem :

Posons : $r_n = S_n - S$, alors :

$$r_{n+i} = r_n + \Delta r_n + \dots + \Delta r_{n+i-1}.$$

et :

$$\sum_{i=0}^{k} a_i r_{n+i} = a_0 r_n + \sum_{i=1}^{k} a_i \left(r_n + \Delta r_n + \dots + \Delta r_{n+i-1} \right) = 0$$

donc :

$$a_0 r_n + \sum_{i=1}^{k} a_i \left(r_n + \Delta S_n + \dots + \Delta S_{n+i-1} \right) = 0$$

Soit: $$\left(\sum_{i=0}^{k} a_i\right) r_n = - \sum_{i=1}^{k} a_i \left(\Delta S_n + \cdots + \Delta S_{n+i-1}\right)$$

$$= - \left(a_1 \Delta S_n + a_2 \left(\Delta S_n + \Delta S_{n+1}\right) + \cdots + a_k \left(\Delta S_n + \cdots + \Delta S_{n+k-1}\right)\right)$$

d'où :

$$- \left(\sum_{i=0}^{k} a_i\right) r_n = \Delta S_n \left(a_1 + \cdots + a_k\right) + \Delta S_{n+1} \left(a_2 + \cdots + a_k\right) + \cdots + a_k \Delta S_{n+k-1}$$

or par hypothèse:

$$\sum_{i=0}^{k} a_i \neq 0 \quad , \text{ donc:}$$

$$r_n = b_1 \Delta S_n + b_2 \Delta S_{n+1} + \cdots + b_k \Delta S_{n+k-1}$$

avec:

$$b_i = - \frac{a_i + a_{i+1} + \cdots + a_k}{\sum_{i=0}^{k} a_i} \quad ; i = 1, \ldots, k$$

donc:

$$S_n = S + \sum_{i=1}^{k} b_i \Delta S_{n+i-1}$$

Ce qui montre d'après la proposition 4 que $S_k^{(n)} = S$ □

2 - Le S-β algorithme :

Nous proposons maintenant un algorithme qui permet de calculer récursivement $S_k^{(n)}$ pour tout $(k,n) \in \mathbb{N}^2$.

On définit sur $\mathbb{R}^{2p}$ les formes z_j par:

$$\langle z_j, V \rangle_{2p} = \langle \phi_j, V_2 - V_1 \rangle_p \quad ; j = 1, \ldots, k . \text{ où } V = \begin{pmatrix} V_1 \\ V_2 \end{pmatrix} .$$

On pose:

$$X_{n+i} = \begin{pmatrix} S_{n+i} \\ S_{n+i+1} \end{pmatrix} \quad ; i = 0, \ldots, k .$$

donc: $\langle z_j, x_{n+i}\rangle = \langle \phi_j, \Delta S_{n+i}\rangle$; $i=0,\dots,k$; $j=1,\dots,k$.

Soit $\tilde{e}_k^{(n)}$ le vecteur de $\mathbb{R}^{2p}$ obtenue par application de la seconde variante du CRPA [5] à $x_n, \dots, x_{n+k}$:

$$[.1)\quad \left\| \begin{array}{l} \tilde{e}_0^{(n)} = \begin{pmatrix} S_n \\ S_{n+1} \end{pmatrix} \\[2ex] \tilde{e}_k^{(n)} = \dfrac{\langle z_k, \tilde{e}_{k-1}^{(n+1)}\rangle \tilde{e}_{k-1}^{(n)} - \langle z_k, \tilde{e}_{k-1}^{(n)}\rangle \tilde{e}_{k-1}^{(n+1)}}{\langle z_k, \tilde{e}_{k-1}^{(n+1)}\rangle - \langle z_k, \tilde{e}_{k-1}^{(n)}\rangle} \end{array} \right.$$

Et on a:

$$\tilde{e}_k^{(n)} = \begin{pmatrix} S_k^{(n)} \\ \bar{S}_k^{(n)} \end{pmatrix}$$

où

$\bar{S}_k^{(n)}$ est la quantité obtenue à partir de $S_k^{(n)}$ en remplaçant dans l'expression de cette dernière la première ligne du numérateur par: $(S_{n+1}, \dots, S_{n+k+1})$.

Posons: $\beta_k^{(n)} = \bar{S}_k^{(n)} - S_k^{(n)}$

alors:

$$\beta_k^{(n)} = \begin{vmatrix} \Delta S_n & \Delta S_{n+1} & \cdots & \Delta S_{n+k} \\ \langle \phi_1, \Delta S_n\rangle & \langle \phi_1, \Delta S_{n+1}\rangle & \cdots & \langle \phi_1, \Delta S_{n+k}\rangle \\ \cdots & \cdots & \cdots & \cdots \\ \langle \phi_k, \Delta S_n\rangle & \langle \phi_k, \Delta S_{n+1}\rangle & \cdots & \langle \phi_k, \Delta S_{n+k}\rangle \end{vmatrix} \Bigg/ \begin{vmatrix} 1 & 1 & \cdots & 1 \\ \langle \phi_1, \Delta S_n\rangle & \langle \phi_1, \Delta S_{n+1}\rangle & \cdots & \langle \phi_1, \Delta S_{n+k}\rangle \\ \cdots & \cdots & \cdots & \cdots \\ \langle \phi_k, \Delta S_n\rangle & \langle \phi_k, \Delta S_{n+1}\rangle & \cdots & \langle \phi_k, \Delta S_{n+k}\rangle \end{vmatrix}$$

Si on applique une deuxième fois la seconde variante du CRPA à $\Delta S_n, \ldots, \Delta S_{n+k-1}$ alors $\beta_k^{(n)}$ se calcule récursivement par:

$$(\beta)\quad \begin{cases} \beta_0^{(n)} = \Delta S_n \quad ; \; n \geq 0 \\ \beta_k^{(n)} = \dfrac{\beta_{k-1}^{(n)} - a_k^{(n)} \cdot \beta_{k-1}^{(n+1)}}{1 - a_k^{(n)}} \quad \text{où} \quad a_k^{(n)} = \dfrac{\langle \phi_k, \beta_{k-1}^{(n)} \rangle}{\langle \phi_k, \beta_{k-1}^{(n+1)} \rangle} \\ \qquad\qquad\qquad\qquad\qquad\qquad k \geq 1 \end{cases}$$

Reprenons l'algorithme (III.1). On a:

$$\langle z_k, \tilde{e}_{k-1}^{(n)} \rangle = \langle \phi_k, \bar{S}_{k-1}^{(n)} - S_{k-1}^{(n)} \rangle = \langle \phi_k, \beta_{k-1}^{(n)} \rangle .$$

Si on ne considère que les p premières composantes dans (III.1) nous obtenons l'algorithme :

$$(S\text{-}\beta)\quad \begin{cases} S_0^{(n)} = S_n \quad ; \; \beta_0^{(n)} = \Delta S_n ; \; n \geq 0 \\ S_k^{(n)} = \dfrac{S_{k-1}^{(n)} - a_k^{(n)} S_{k-1}^{(n+1)}}{1 - a_k^{(n)}} \\ \beta_k^{(n)} = \dfrac{\beta_{k-1}^{(n)} - a_k^{(n)} \beta_{k-1}^{(n+1)}}{1 - a_k^{(n)}} \quad \text{où} \quad a_k^{(n)} = \dfrac{\langle \phi_k, \beta_{k-1}^{(n)} \rangle}{\langle \phi_k, \beta_{k-1}^{(n+1)} \rangle} \\ \qquad\qquad\qquad\qquad\qquad\qquad k \geq 1 \end{cases}$$

Cet algorithme sera appelé le (S-β)-algorithme.

Remarque 5 :

Le calcul de $S_k^{(n)}$ nécessite la connaissance de $S_n, \dots, S_{n+k+1}$.

Nous allons maintenant écrire le (S-β) de manière plus simple en utilisant des vecteurs doubles.

On pose :

$$\Gamma_k^{(n)} = \begin{pmatrix} S_k^{(n)} \\ \beta_k^{(n)} \end{pmatrix} \quad \text{et} \quad \Phi'_j = (0, \dots, 0, \Phi_j)^T \in \mathbb{R}^{2p}$$

et on a l'algorithme, qui n'est autre qu'une façon d'écrire le (S-β) :

$$(\Gamma) \quad \left\| \begin{array}{l} \Gamma_0^{(n)} = \begin{pmatrix} S_n \\ \Delta S_n \end{pmatrix} \; ; \; n \geq 0 \\[2ex] \Gamma_k^{(n)} = \dfrac{\Gamma_{k-1}^{(n)} - a_k^{(n)} \Gamma_{k-1}^{(n+1)}}{1 - a_k^{(n)}} \quad \text{avec } a_k^{(n)} = \dfrac{\langle \Phi'_k, \Gamma_{k-1}^{(n)} \rangle}{\langle \Phi'_k, \Gamma_{k-1}^{(n+1)} \rangle} \end{array} \right.$$

Si on place $\Gamma_k^{(n)}$ dans un tableau à double entrée, on a alors le schéma de calcul suivant : k numéro de colonne et n celui d'une diagonale.

$$\begin{tikzcd}
\Gamma_0^{(0)} \arrow[dr] & & & \\
\Gamma_0^{(1)} \arrow[r] \arrow[dr] & \Gamma_1^{(0)} \arrow[dr] & & \\
\Gamma_0^{(2)} \arrow[r] \arrow[dr] & \Gamma_1^{(1)} \arrow[r] \arrow[dr] & \Gamma_2^{(0)} \arrow[dr] & \\
\Gamma_0^{(3)} \arrow[r] & \Gamma_1^{(2)} \arrow[r] & \Gamma_2^{(1)} \arrow[r] & \Gamma_3^{(0)} \\
\vdots & \vdots & \vdots & \vdots
\end{tikzcd}$$

Pour calculer $S_k^{(n)}$ avec le (S-β), $0 \leq k \leq n \leq K$, il faudra stocker en mémoire $K+1$ vecteurs de $\mathbb{R}^{2p}$.

Il faudra $p(K^2+K)$ multiplications et autant d'additions. On doit rajouter

le nombre d'opérations nécessaires pour le calcul de $a_k^{(n)}$; ce nombre peut être négligeable si les ϕ_j sont bien choisis : c'est le cas par exemple lorsque $\phi_j = e_j$, base canonique de $\mathbb{R}^p$ et dans ce cas :

$$a_k^{(n)} = \frac{\left(\Gamma_{k-1}^{(n)}\right)_{k+p}}{\left(\Gamma_{k-1}^{(n+1)}\right)_{k+p}} = \frac{\left(\beta_{k-1}^{(n)}\right)_p}{\left(\beta_{k-1}^{(n+1)}\right)_p} \quad ; k = 1, \dots, p.$$

Ceci nous permettra l'implémentation de la méthode d'Henrici par le (S-β), comme nous le verrons dans le dernier paragraphe. Dans ce cas, le calcul de $S_p^{(0)}$ par le (S-β) nécessite $2p^3$ opérations arithmétiques, alors que le H-algorithme nécessiterait $\frac{5}{2}p^3$ opérations.

Avant de voir quelques propriétés de convergence et d'accélération de la convergence du (S-β)-algorithme, nous donnons tout d'abord des règles particulières de cet-algorithme qui nous permetteraient d'éviter les singularités qu'on peut rencontrer lors du calcul de $\beta_k^{(n)}$ et $S_k^{(n)}$ par le (S-β)-algorithme.

III.3. Règles particulières :

proposition 6 :

pour $m \geq 0$, on a :

$$\beta_{k+m}^{(n)} = \begin{vmatrix} S_k^{(n)} & \Delta S_k^{(n)} & \cdots & \Delta S_k^{(n+m-1)} \\ \langle \phi_{k+1}, \beta_k^{(n)} \rangle & \langle \phi_{k+1}, \Delta\beta_k^{(n)} \rangle & \cdots & \langle \phi_{k+1}, \Delta\beta_k^{(n+m-1)} \rangle \\ \cdots & \cdots & \cdots & \cdots \\ \langle \phi_{k+m}, \beta_k^{(n)} \rangle & \langle \phi_{k+1}, \Delta\beta_k^{(n)} \rangle & \cdots & \langle \phi_{k+m}, \Delta\beta_k^{(n+m-1)} \rangle \end{vmatrix} \Bigg/ \begin{vmatrix} \langle \phi_{k+1}, \Delta\beta_k^{(n)} \rangle & \cdots & \langle \phi_{k+1}, \Delta\beta_k^{(n+m-1)} \rangle \\ \cdots & \cdots & \cdots \\ \langle \phi_{k+m}, \Delta\beta_k^{(n)} \rangle & \cdots & \langle \phi_{k+m}, \Delta\beta_k^{(n+m-1)} \rangle \end{vmatrix}$$

et une expression similaire pour $\beta_{k+m}^{(n)}$ en remplaçant la première ligne du numérateur par $\beta_k^{(n)}, \ldots, \beta_k^{(n+m)}$.

Dem : Analogue à celle faite dans [6] pour le ε-algorithme $\square$.

Si $k=0$, on a l'expression de $S_k^{(n)}$ comme rapport de déterminants.

Si $m=1$, nous obtenons les règles du $(s-\beta)$-algorithme.

En appliquant la formule de Schur, on a :

$$S_{k+m}^{(n)} = S_k^{(n)} - \left(\Delta S_k^{(n)}, \ldots, \Delta S_k^{(n+m-1)}\right) \cdot \begin{bmatrix} \langle \phi_{k+1}, \Delta\beta_k^{(n)} \rangle & \cdots & \langle \phi_{k+1}, \Delta\beta_k^{(n+m-1)} \rangle \\ \cdots & \cdots & \cdots \\ \langle \phi_{k+m}, \Delta\beta_k^{(n)} \rangle & \cdots & \langle \phi_{k+1}, \Delta\beta_k^{(n+m-1)} \rangle \end{bmatrix}^{-1} \cdot \begin{bmatrix} \langle \phi_{k+1}, \beta_k^{(n)} \rangle \\ \vdots \\ \langle \phi_{k+m}, \beta_k^{(n)} \rangle \end{bmatrix}$$

1.4 - Propriétés de convergence et d'accélération de la convergence :

proposition 7 :

Si $\lim_{n\to\infty} S_{k-1}^{(n)} = S$ et si il existe α et β tels que :
$0<\alpha<1<\beta$ et $\dfrac{\langle \phi_k, \beta_{k-1}^{(n+1)}\rangle}{\langle \phi_k, \beta_{k-1}^{(n)}\rangle} \notin [\alpha,\beta]$ $\forall n>N$, alors :

$$\lim_{n\to\infty} S_k^{(n)} = S$$

dem : La règle principale du (S-β) nous permet d'écrire :

$$S_k^{(n)} = S_{k-1}^{(n)} - \frac{1}{\dfrac{\langle \phi_k, \beta_{k-1}^{(n+1)}\rangle}{\langle \phi_k, \beta_{k-1}^{(n)}\rangle} - 1} \cdot \Delta S_{k-1}^{(n)}$$

et comme $\Delta S_{k-1}^{(n)}$ converge vers 0 et $\dfrac{\langle \phi_k, \beta_{k-1}^{(n+1)}\rangle}{\langle \phi_k, \beta_{k-1}^{(n)}\rangle}$ ne converge pas vers 1, nous avons : $\lim S_k^{(n)} = S$. □

proposition 8 :

Si $\lim_{n\to\infty} S_n = S$ et si $\lim_{n\to\infty} \dfrac{\langle \phi_k, \beta_{k-1}^{(n+1)}\rangle}{\langle \phi_k, \beta_{k-1}^{(n)}\rangle} = b_k \neq 1$, $\forall k$, alors :

$$\lim_{n\to\infty} S_k^{(n)} = S \quad , \forall k$$

dem :

Par récurrence sur k et application successive de la proposition 7. □

<u>proposition 9</u> :

Si les conditions de la proposition 8 sont satisfaites et si en plus : $\lim\limits_{n\to\infty} \dfrac{\langle \phi, S_{k-1}^{(n+1)} - S\rangle}{\langle \phi, S_{k-1}^{(n)} - S\rangle} = b_k$, alors :

$$\forall k, \quad \langle \phi, S_k^{(n)} - S\rangle = o\left(\langle \phi, S_{k-1}^{(n)} - S\rangle\right) \quad (n \to \infty)$$

<u>dem</u> :

$$S_k^{(n)} - S = S_{k-1}^{(n)} - S - \frac{1}{\dfrac{\langle \phi_k, \beta_{k-1}^{(n+1)}\rangle}{\langle \phi_k, \beta_{k-1}^{(n)}\rangle} - 1} \cdot \left(S_{k-1}^{(n+1)} - S - S_{k-1}^{(n)} + S \right)$$

donc pour $\phi \in \mathbb{R}^p$:

$$\frac{\langle \phi, S_k^{(n)} - S\rangle}{\langle \phi, S_{k-1}^{(n)} - S\rangle} = 1 - \frac{1}{\dfrac{\langle \phi_k, \beta_{k-1}^{(n+1)}\rangle}{\langle \phi_k, \beta_{k-1}^{(n)}\rangle} - 1} \cdot \left(\frac{\langle \phi, S_{k-1}^{(n+1)} - S\rangle}{\langle \phi, S_{k-1}^{(n)} - S\rangle} - 1 \right)$$

d'où

$$\lim_{n\to\infty} \frac{\langle \phi, S_k^{(n)} - S\rangle}{\langle \phi, S_{k-1}^{(n)} - S\rangle} = 1 - \frac{b_k - 1}{b_k - 1} = 0 . \quad \square$$

Les conditions des propositions précédentes semblent difficiles à vérifier car elles font intervenir la limite S, mais nous verrons qu'elles sont satisfaites facilement pour des suites de vecteurs telles que l'on connaisse l'expression de $S_n - S$.

IV Applications:

IV.1 — Considérons dans $\mathbb{R}^p$, les suites (S_n) qui sont de la forme:

$$(IV.1) \qquad S_n = S + \sum_{i \geq 1} \lambda_i^n y_i$$

où

$y_i \in \mathbb{R}^p$ et $\lambda_i \in \mathbb{R}$.

On notera (H) l'hypothèse suivante:

$$(H) \quad \begin{cases} \lambda_i \neq 1 \quad ; i \geq 1 \\ |\lambda_1| > |\lambda_2| > \dots > |\lambda_k| > |\lambda_{k+1}| > \dots \end{cases}$$

- Si $|\lambda_1| < 1$ alors (S_n) converge vers S.
- Si $|\lambda_1| > 1$ alors (S_n) n'est pas convergente et S sera appelée dans ce cas anti-limite de (S_n).

Nous considérerons par la suite uniquement le cas où $g_i(n) = \Delta S_{n+i-1}$; $i = 1, \dots, k$.

Dans un premier temps, nous allons appliquer l'algorithme (B) pour le calcul des (y_i) et des (λ_i).

Rappelons que L est un ensemble de p fonctionnelles linéaires, bornées et indépendants.

Theorème 1:

Si on applique le β-algorithme à une suite (S_n) de la forme (IV.1), vérifiant (H) et si $\langle \phi_i, y_j \rangle \neq 0$; $i \geq 1$ et $j = i, i+1$ alors $\forall k \geq 0$:

$$\text{i)} \quad \beta_k^{(n)} \sim \sum_{i \geq k+1} \lambda_i^{n+k} \cdot (\lambda_i - 1) y_i \qquad (n \to \infty)$$

et $\forall k \geq 1$:

$$\text{ii)} \quad \frac{\langle \phi_k, \beta_{k-1}^{(n+1)} \rangle}{\langle \phi_k, \beta_{k-1}^{(n)} \rangle} = \lambda_k + O\left(\left(\frac{\lambda_{k+1}}{\lambda_k} \right)^{n+k} \right) \qquad (n \to \infty)$$

$$\text{iii)} \quad \lim_{n \to \infty} \frac{\beta_{k-1}^{(n)}}{\langle \phi_k, \beta_{k-1}^{(n)} \rangle} = \frac{y_k}{\langle \phi_k, y_k \rangle}$$

Dem.

i) par récurrence sur k.

- $k=0$ $\qquad \beta_0^{(n)} = \Delta S_n = \sum_{i \geq 1} \lambda_i^n (\lambda_i - 1) y_i$
- supposons que i) est vraie jusqu'à l'ordre $k-1$, alors:

$$\langle \phi, \beta_{k-1}^{(n)} \rangle \sim \sum_{i \geq k} \lambda_i^{n+k-1} \cdot (\lambda_i - 1) \cdot \langle \phi, y_i \rangle \qquad (n \to \infty).$$

donc:

$$\langle \phi, \beta_{k-1}^{(n+1)} \rangle \sim \sum_{i \geq k} \lambda_i^{n+k} \cdot (\lambda_i - 1) \cdot \langle \phi, y_i \rangle \qquad (n \to \infty)$$

et,

$$\langle \phi, \Delta \beta_{k-1}^{(n)} \rangle \sim \sum_{i \geq k} \lambda_i^{n+k-1} \cdot (\lambda_i - 1)^2 \cdot \langle \phi, y_i \rangle \qquad (n \to \infty).$$

En développant, nous obtenons:

$$\langle \Phi, \beta_{k-1}^{(n+1)} \rangle \sim \lambda_k^{n+k} \cdot (\lambda_k - 1) \langle \Phi, y_k \rangle \times \left[1 + \sum_{i \geq k+1} \left(\frac{\lambda_i}{\lambda_k} \right)^{n+k} \cdot \frac{\lambda_i - 1}{\lambda_k - 1} \cdot \frac{\langle \Phi, y_i \rangle}{\langle \Phi, y_k \rangle} \right]$$

et (H) $\Rightarrow$ $\left| \frac{\lambda_i}{\lambda_k} \right| < 1$ pour $i > k$

donc:

$$\langle \Phi, \beta_{k-1}^{(n+1)} \rangle \sim \lambda_k^{n+k} (\lambda_k - 1) \langle \Phi, y_k \rangle \qquad (n \longrightarrow \infty)$$

et de même :

$$\langle \Phi, \Delta \beta_{k-1}^{(n)} \rangle \sim \lambda_k^{n+k-1} \cdot (\lambda_k - 1)^2 \langle \Phi, y_k \rangle \qquad (n \longrightarrow \infty)$$

La règle du β-algorithme s'écrit sous la forme:

$$\beta_k^{(n)} = \beta_{k-1}^{(n+1)} - \frac{\langle \Phi_k, \beta_{k-1}^{(n+1)} \rangle}{\langle \Phi_k, \Delta \beta_{k-1}^{(n)} \rangle} \cdot \Delta \beta_{k-1}^{(n)}$$

donc: $\forall \Phi \in L$

$$\langle \Phi, \beta_k^{(n)} \rangle \sim \left(\sum_{i \geq k} \lambda_i^{n+k} \cdot (\lambda_i - 1) \langle \Phi, y_i \rangle \right) - \frac{\left(\lambda_k^{n+k} \cdot (\lambda_k - 1) \langle \Phi_k, y_k \rangle \right) \cdot \left(\lambda_k^{n+k-1} \cdot (\lambda_k - 1)^2 \cdot \langle \Phi, y_k \rangle \right)}{\lambda_k^{n+k-1} \cdot (\lambda_k - 1)^2 \cdot \langle \Phi_k, y_k \rangle}$$

et après simplification, nous obtenons:

$$\langle \Phi, \beta_k^{(n)} \rangle \sim \sum_{i \geq k} \lambda_i^{n+k} \cdot (\lambda_i - 1) \langle \Phi, y_i \rangle - \lambda_k^{n+k} (\lambda_k - 1) \langle \Phi, y_k \rangle \quad (n \to \infty)$$

Soit:

$$\langle \Phi, \beta_k^{(n)} \rangle \sim \sum_{i \geq k+1} \lambda_i^{n+k} \cdot (\lambda_i - 1) \langle \Phi, y_i \rangle$$

Ce qui montre i).

ii) d'après ce qui précède, on a:

$$\frac{\langle \phi_k, \beta_{k-1}^{(n+1)} \rangle}{\langle \phi_k, \beta_{k-1}^{(n)} \rangle} \sim \lambda_k \times \left[1 + \left(\frac{\lambda_{k+1}}{\lambda_k} \right)^{n+k} \cdot \frac{\lambda_{k+1}-1}{\lambda_k - 1} \cdot \frac{\langle \phi_k, y_{k+1} \rangle}{\langle \phi_k, y_k \rangle} \right] \quad (n \to \infty)$$

Soit encore:

$$\frac{\langle \phi_k, \beta_{k-1}^{(n+1)} \rangle}{\langle \phi_k, \beta_{k-1}^{(n)} \rangle} = \lambda_k + O\left(\left(\frac{\lambda_{k+1}}{\lambda_k} \right)^{n+k} \right) \quad (n \to \infty).$$

iii) On a pour tout $\phi \in L$:

$$\frac{\langle \phi, \beta_{k-1}^{(n)} \rangle}{\langle \phi_k, \beta_{k-1}^{(n)} \rangle} \sim \frac{\lambda_k^{n+k-1} (\lambda_k - 1) \cdot \langle \phi, y_k \rangle}{\lambda_k^{n+k-1} (\lambda_k - 1) \cdot \langle \phi_k, y_k \rangle}$$

donc:

$$\lim_{n \to \infty} \frac{\langle \phi, \beta_{k-1}^{(n)} \rangle}{\langle \phi_k, \beta_{k-1}^{(n)} \rangle} = \frac{\langle \phi, y_k \rangle}{\langle \phi_k, y_k \rangle}$$

Et en prenant respectivement pour ϕ, les éléments d'une base de $\mathbb{R}^p$.

On obtient:

$$\forall k \geq 1, \quad \lim_{n \to \infty} \frac{\beta_{k-1}^{(n)}}{\langle \phi_k, \beta_{k-1}^{(n)} \rangle} = \frac{y_k}{\langle \phi_k, y_k \rangle} \qquad \square$$

V.2 - Calcul des valeurs et vecteurs propres :

On considère la suite (S_n) de $\mathbb{R}^p$ produite par :

(V.2) $S_{n+1} = A S_n$; S_0 donnée.

où A est une matrice carrée d'ordre p, vérifiant l'hypothèse (H) et tel que $|\lambda_1| < 1$. Dans ce cas l'itération (V.2) converge : $\lim_{n \to \infty} S_n = 0$.

Soient $v_1, \dots, v_p$ les vecteurs propres associés aux valeurs propres $\lambda_1, \dots, \lambda_p$, alors S_0 s'écrit dans la base $\{v_1, \dots, v_p\}$:

$S_0 = a_1 v_1 + \dots + a_p v_p$.

On pose : $y_i = a_i v_i$; $i = 1, \dots, p$ et on suppose que :

$\langle \phi_j, y_i \rangle \neq 0$; $i = j, j+1$.

Avec toutes ces hypothèses, nous avons le résultat suivant :

Théorème 2 : Si on applique le β-algorithme à la suite (S_n) définie par (V.2) et si les conditions précédentes sont satisfaites, alors : $\forall k \in \{1, \dots, p\}$:

$$i) \quad \frac{\langle \phi_k, \beta_{k-1}^{(n+1)} \rangle}{\langle \phi_k, \beta_{k-1}^{(n)} \rangle} = \lambda_k + O\left(\left(\frac{\lambda_{k+1}}{\lambda_k} \right)^{n+k} \right) \quad (n \to \infty)$$

$$ii) \quad \lim_{n \to \infty} \frac{\beta_{k-1}^{(n)}}{\langle \phi_k, \beta_{k-1}^{(n)} \rangle} = \frac{v_k}{\langle \phi_k, v_k \rangle}$$

dém :

$$S_n = A S_{n-1} = \dots = A^n S_0$$

or : $S_0 = a_1 v_1 + \dots + a_p v_p$

donc $S_n = \sum_{i=1}^{p} a_i \lambda_i^n v_i = \sum_{i=1}^{p} \lambda_i^n y_i$

(S_n) est de la forme (IV.1), avec $s=0$ et $\lambda_i = 0$ pour $i > p$. On applique alors le théorème 1. □

Remarque 6 :

1°) Cette méthode est surtout intéressante lorsqu'on ne calcule que les valeurs propres de plus grands modules.

2°) Le calcul de λ_k par le β-algorithme nécessite la connaissance de $S_n, \dots, S_{n+k+2}$, alors que l'ε-algorithme vectoriel et avec une erreur de même ordre, nécessite la connaissance de $S_n, \dots, S_{n+2k}$.

3°) Si on pose $b_k^{(n)} = \dfrac{\langle \Phi_k, \beta_{k-1}^{(n+1)} \rangle}{\langle \Phi_k, \beta_{k-1}^{(n)} \rangle}$, alors d'après le résultat du théorème 2 et le théorème (2, chap II), on peut appliquer l'ε-algorithme scalaire à $(b_k^{(n)})_n$, k fixé, pour accélérer la convergence de $(b_k^{(n)})_n$ vers λ_k.

Exemples numeriques:

Exemple 1:

$$A = \begin{pmatrix} -1 & 0 & 1 & 2 & 3 \\ & 2 & -5 & -5 & -5 \\ & & 5 & -14 & -15 \\ & 0 & & 3 & -27 \\ & & & & -7 \end{pmatrix}$$

Si on applique le β-algorithme avec $\phi_j = e_j$; $j=1,\dots,7$, alors l'approximation de λ_k est donnée par. $\dfrac{(\beta_{k-1}^{(n+1)})_k}{(\beta_{k-1}^{(n)})_k}$; $k=1,\dots,7$

et nous obtenons les résultats suivants :

n	App (λ_1)
[illegible]	[illegible]

n	App (λ_2)
[illegible]	[illegible]

n	App (λ_3)
[illegible]	[illegible]

n	App (λ_4)
[illegible]	[illegible]

Pour λ_1, λ_2 et λ_3 les résultats s'améliorent quand N devient grand, pour λ_4 il y a dégradation pour $N > 17$.

Exemple 2 :

$$A = \begin{pmatrix} -2 & 1 & -3 & 2 \\ 1 & 1 & 1 & 0 \\ 4 & -2 & 5 & -3 \\ 5 & -1 & 5 & 6 \end{pmatrix}$$

Les valeurs propres de A sont :

$\lambda_1 = 4$; $\lambda_2 = 3$; $\lambda_3 = 2$ et $\lambda_4 = 1$.

On applique l'algorithme (β) à la suite $S_{n+1} = AS_n$ en prenant :

$\Phi_j = e_j$ base canonique de $\mathbb{R}^4$. Et nous obtenons les résultats suivants :

n	App(λ_1)
[illegible]	[illegible]

n	App(λ_2)
[illegible]	[illegible]

n	App(λ_3)
[illegible]	[illegible]

IV.3 – Accélération de la convergence des suites (IV.1) :

On considère les suites (S_n) de la forme (IV.1) et on suppose que $|\lambda_1| < 1$, alors :

Théorème 3 :

Si on applique le $(S-\beta)$-algorithme à une suite (S_n) de la forme (IV.1), vérifiant (H) et tel que : $\langle \Phi_j, y_j \rangle \neq 0, \forall j \geq 1$ alors :

$\forall k \geq 0$ i) $S_k^{(n)} - S \sim \sum_{i \geq k+1} \lambda_i^{n+k} y_i \quad (n \to \infty)$

$\forall k \geq 1$ ii) $\dfrac{\|S_k^{(n)} - S\|}{\|S_{k-1}^{(n)} - S\|} \sim \left| \dfrac{\lambda_{k+1}}{\lambda_k} \right|^{n+k-1} \cdot |\lambda_{k+1}| \cdot \dfrac{\|y_{k+1}\|}{\|y_k\|} \quad (n \to \infty)$

iii) $\|S_k^{(n)} - S\| = o\left(\|S_{k-1}^{(n)} - S\|\right) \quad (n \to \infty)$.

dem :

i) Par récurrence sur k en faisant une démonstration analogue à celle du théorème 1.

ii) A partir de i) et en utilisant l'hypothèse (H).

iii) Se déduit directement de ii) et le fait que $|\lambda_{k+1}| < |\lambda_k|$.

I - Résolution des systèmes linéaires :

Théorème 4

Si on applique le $(S-\beta)$ algorithme à une suite (S_n) générée par : $S_{n+1} = AS_n + b$ où A est une matrice carrée, réelle d'ordre p et b un vecteur de $\mathbb{R}^p$ et tel que $(I-A)$ soit inversible, alors :

$$S_m^{(n)} = S$$

où m est le degré du polynôme minimal de A par rapport à $S_0 - S$ avec $S = (I-A)^{-1}.b$

Dem :

Soit $p(t) = \sum_{i=0}^{m} a_i t^i$ le polynôme minimal de A par rapport à $S_0 - S$. On suppose que 0 n'est pas racine de ce polynôme donc $a_0 \neq 0$. Comme 1 n'est pas valeur propre de A on a aussi :

$$\sum_{i=0}^{m} a_i \neq 0$$

Nous avons : $S_n = AS_{n-1} + b$

$$S = AS + b$$

donc : $S_n - S = A(S_{n-1} - S) = A^n(S_0 - S)$

et pour $i = 0, \dots, m$ on a : $S_{n+i} - S = A^{n+i}(S_0 - S)$

d'où :

$$\sum_{i=0}^{m} a_i (S_{n+i} - S) = A^n . \sum_{i=0}^{m} a_i A^i (S_0 - S)$$

et comme :

$$\sum_{i=0}^{m} a_i A^i(S_0-S)=0 \quad \text{alors} \quad \sum_{i=0}^{m} a_i (S_{n+i}-S)=0$$

avec :

$\sum_{i=0}^{m} a_i \neq 0$, ce qui montre d'après la propriété (III.4) que :

$$S_m^{(n)} = S \quad \square.$$

On va considérer maintenant le cas où 0 est racine du polynôme minimal de A par rapport à S_0-S.

Propriété 6 :

Si on applique le (S-β) algorithme à une suite (S_n) vérifiant les conditions du théorème 4 et si en plus zéro est racine du polynôme minimal de A par rapport à S_0-S et r sa multiplicité, alors : $\forall n \geq 0$: $S_{m-r}^{(n+r)} = S$

dem :

Le polynôme minimal de A par rapport à S_0-S s'écrit :

$$\phi(t) = \sum_{i=r}^{m} a_i t^i \quad \text{avec} \quad \phi(1) = \sum_{i=r}^{m} a_i \neq 0 \text{ et } a_r \neq 0.$$

or :

$$S_i - S = A^i(S_0-S) \Rightarrow A^n \sum_{i=r}^{m} a_i (S_i-S) = A^n . \sum_{i=r}^{m} a_i A^i (S_0-S) = 0$$

d'où :

$$\sum_{i=r}^{m} a_i (S_{n+i}-S) = 0 \Rightarrow \sum_{i=0}^{m-r} a_{i+r} (S_{n+i+r}-S) = 0$$

ou encore :

$$\sum_{i=0}^{m-r} b_i (S_{n+i+r}-S) = 0 \quad \text{avec} \quad \sum_{i=0}^{m-r} b_i = \sum_{i=0}^{m-r} a_{i+r} = \sum_{i=r}^{m} a_i \neq 0$$

En utilisant la propriété (III.5) on a : $S_{m-r}^{(n+r)} = S \quad \square$

-Considerons maintenant le cas où 1 est valeur propre de A. On supposera que l'equation $S = AS + b$ admet une infinité de solution c'est à dire que $b \in Im(I-A)$. On a alors :

proposition 7:

Si on applique le (S-β) algorithme à la suite (S_n) definie par: $S_{n+1} = AS_n + b$ où A est une matrice carrée reelle d'ordre p tel que 1 soit valeur propre de A d'ordre 1 et $b \in Im(I-A)$, alors:

$\forall q \qquad S_{m-1}^{(q)}$ est solution de: $S = AS + b$.

dem:

m est le degré du polynôme minimal de A par rapport à $S_0 - S$ où S est solution de $S = AS + b$.

$p(t) = \sum_{i=0}^{m} a_i t^i$ et verifie: $p(1) = 0$ et $p(A).(S_0 - S) = 0$

donc:

$$p(t) = (1-t) \sum_{i=0}^{m-1} b_i t^i \quad \text{avec} \quad \sum_{i=0}^{m-1} b_i \neq 0 \text{ puisque } 1$$

est racine simple de $p(t)$.

Soit $q \in \mathbb{N}$, alors:

$$S_{q+i} - S = A^{q+i}.(S_0 - S) \quad ; i = 0, \dots, m-1$$

donc: $$\sum_{i=0}^{m-1} b_i S_{q+i} = S.\sum_{i=0}^{m-1} b_i + \left(\sum_{i=0}^{m-1} b_i A^{q+i}\right).(S_0 - S)$$

et:

$$(I-A)\sum_{i=0}^{m-1} b_i S_{q+i} = (I-A)S\sum_{i=0}^{m-1} b_i + (I-A)\left(\sum_{i=0}^{m-1} b_i A^{q+i}\right).(S_0 - S)$$

donc:

$$(I-A)\left(\sum_{i=0}^{m-1} b_i A^{q+i}\right)(S_0 - S) = A^q (I-A)\left(\sum_{i=0}^{m} b_i A^i\right)(S_0 - S)$$
$$= A^q . p(A)(S_0 - S) = 0$$

donc :

$$(I-A)\sum_{i=0}^{m-1} b_i S_{q+i} = (I-A) S . \sum_{i=0}^{m-1} b_i$$

Soit :

$$\sum_{i=0}^{m-1} b_i S_{q+i} - \sum_{i=0}^{m-1} b_i \left(S_{q+i+1} - b\right) = b . \sum_{i=0}^{m-1} b_i$$

d'où :

$$\sum_{i=0}^{m-1} b_i S_{q+i} = \sum_{i=0}^{m-1} b_i S_{q+i+1} \Longrightarrow$$

$\forall q$: $\sum_{i=0}^{m-1} b_i S_{q+i} = M$: constante indépendante de q

posons : $M = H . \sum_{i=0}^{m-1} b_i$

alors :

$$\sum_{i=0}^{m-1} b_i (S_{q+i} - H) = 0 \Longrightarrow \text{(d'après la prop III.4)} \quad S_{m-1}^{(q)} = H$$

Montrons que H est solution de : $S = AS + b$:

D'après ce qui précède on a :

$$(I-A) . H . \sum_{i=0}^{m-1} b_i = (I-A) S \sum_{i=0}^{m-1} b_i$$

et comme $\sum_{i=0}^{m-1} b_i \neq 0$, alors :

$$(I-A) H = (I-A) S = b \Longrightarrow H = AH + b$$

Soit :

$$\forall q \quad : \quad S_{m-1}^{(q)} = A . S_{m-1}^{(q)} + b \quad \square$$

$$\begin{array}{c} \begin{array}{cccc} 2 & 1 & 3 & 4 \end{array} \\ \begin{bmatrix} 1 & -3 & 1 & 5 \\ 3 & 1 & 6 & -2 \\ 4 & 5 & -2 & -1 \end{bmatrix} \end{array} \times X = \begin{bmatrix} 10 \\ 4 \\ 8 \\ 6 \end{bmatrix} \quad ; \; m = p = 4 \quad ; \; r = 0$$

dont la solution est $x^* = (1,1,1,1)^T$. En partant du point $(0,0,0,0)^T$ on obtient :

	n	colonne 1	colonne 2	colonne 3	colonne 4.
$X_n(1)$	1 2 3 4				
$X_n(2)$	1 2 3 4				
$X_n(3)$	1 2 3 4				

II- Résolution des systèmes non linéaires :

Soit $F : \mathbb{R}^p \longrightarrow \mathbb{R}^p$ supposé Frechet differentiable dans un voisinage de x : $x = F(x)$, tel que $I - F'(x)$ soit inversible On notera $J = F'(x)$.

Nous considerons l'algorithme suivant :

$$
(4) \quad \left\| \begin{array}{ll}
x_0 \text{ donnée} & \\
u_0 = x_k & ; k \geq 1 \\
u_{i+1} = F(u_i) & ; i = 0, \ldots, m \\
x_{k+1} = S_m^{(0)} &
\end{array} \right.
$$

où m est le degré du polynôme minimal de J par rapport à $u_0 - x$. Donc :

$$
x_{k+1} = \frac{\begin{vmatrix} u_0 & u_1 & \cdots & u_m \\ \langle \phi_1, \Delta u_0 \rangle & \langle \phi_1, \Delta u_1 \rangle & \cdots & \langle \phi_1, \Delta u_m \rangle \\ \vdots & & & \\ \langle \phi_m, \Delta u_0 \rangle & \langle \phi_m, \Delta u_1 \rangle & \cdots & \langle \phi_m, \Delta u_m \rangle \end{vmatrix}}{\begin{vmatrix} 1 & 1 & \cdots & 1 \\ \langle \phi_1, \Delta u_0 \rangle & \langle \phi_1, \Delta u_1 \rangle & \cdots & \langle \phi_1, \Delta u_m \rangle \\ \vdots & & & \\ \langle \phi_m, \Delta u_0 \rangle & \langle \phi_m, \Delta u_1 \rangle & \cdots & \langle \phi_m, \Delta u_m \rangle \end{vmatrix}}
$$

où

$\{\phi_j\}_{j=1,\ldots,m}$ est un système libre de $\mathbb{R}^p$.

sons :

$$\Delta U_0 = [\Delta u_0, ---, \Delta u_{m-1}] \in \mathcal{M}(p,m)$$

$$\Delta^2 U_0 = [\Delta^2 u_0, ---, \Delta^2 u_{m-1}] \in \mathcal{M}(p,m)$$

$$\Phi = [\phi_1, ---, \phi_m] \in \mathcal{M}(p,m)$$

Avec ces notations X_{k+1} se met sous la forme :

$$\boxed{X_{k+1} = u_0 - \Delta U_0 \left(\Phi^T . \Delta^2 U_0\right)^{-1} \Phi^T \Delta u_0}$$

<u>Remarque</u> :

Si $m = p$, les matrices considérées sont d'ordre p et comme Φ est inversible : $X_{k+1} = u_0 - \Delta U_0 (\Delta^2 U_0)^{-1} \Delta u_0$

C'est la méthode d'Henrici [20]. On notera (H1) l'hypothèse suivante :

(H1) les vecteurs $\Delta u_0, --, \Delta u_m$ sont linéairement uniformément indépendants [24].

<u>Théorème 5</u> :

Soit $F : \mathbb{R}^p \longrightarrow \mathbb{R}^p$ tel qu'il existe x vérifiant $x = F(x)$. On suppose que F est deux fois différentiable dans un voisinage de x, que $I - J$ est inversible et (H1) vérifiée. Alors il existe un voisinage V de x tel que si F'' est bornée dans ce voisinage et si $x_0 \in V$ on a :

$$\| X_{k+1} - X \| = O\left(\| X_k - X \|^2\right).$$

dem:

Rappelons l'algorithme:

x_0 donné

$u_0 = x_K$

$u_{i+1} = F(u_i) \quad ; i = 0, ---, m$

où m est le degré du polynôme minimal de $J = F'(x)$ par rapport à $u_0 - x$.

Lemme 1:

Il existe une matrice $A_K \in M(p,p)$ inversible tel que:

$$\Delta^2 U_0 = - A_K . \Delta U_0$$

dem

Voir [24]

Posons:

$$B_K = I - A_K \quad \text{et} \quad \Delta U'_0 = [\Delta u_1, ---, \Delta u_m]$$

alors:

$$\Delta U'_0 - \Delta U_0 = - A_K . \Delta U_0$$

soit:

$$\Delta U'_0 = B_K . \Delta U_0 \qquad \text{où } B_K = I - A_K$$

donc:

$$u_{i+1} - u_i = B_K \left(u_i - u_{i-1} \right) \quad ; i = 1, ---, m .$$

$\Rightarrow$

$$u_{i+1} - B_k u_i = u_i - B_k u_{i-1} = --- = u_1 - B_k u_0 = b_k.$$

d'où :

$$(*) \quad u_{i+1} = B_k u_i + b_k \qquad ; i=0,---,m$$

Soit X'_{k+1} le point fixe de cette itération $\left(X'_{k+1} = (I-B_k)^{-1} \cdot b_k\right)$ et montrons que $X'_{k+1} = X_{k+1}$: vecteur donné par l'algorithme (II.1)

Or,

d'après la proposition (III.5), le point fixe de (*) est donné par: $X'_{k+1} = S^{(0)}_{m'}$, où m' est le degré du polynôme minimal de B_k par rapport à $u_0 - X'_{k+1}$ et on a:

$$\sum_{i=0}^{m'} a_i \left(u_i - X'_{k+1}\right) = 0 \quad \text{avec} \quad \sum_{i=0}^{m'} a_i \neq 0$$

Lemme 2 :

m est le degre du polynôme minimal de B_k par rapport à $u_0 - X'_{k+1}$.

Dem:

Dans [2] on montre que m est le degré du polynôme minimal de B_k par rapport à $u_1 - u_0$. Or:

$$\sum_{i=0}^{m} a_i B_k^i \left(u_0 - X'_{k+1}\right) = \sum_{i=0}^{m} a_i B_k^i \left(u_0 - u_1\right) + \sum_{i=0}^{m} a_i B_k^i \left(u_1 - X'_{k+1}\right) = 0$$

et comme:

$$u_1 - X'_{k+1} = B_k \left(u_0 - X'_{k+1}\right) \quad , \text{alors:}$$

$$(I - B_k) \sum_{i=0}^{m} a_i B_k^i (u_0 - x'_{k+1}) = 0$$

$$\Rightarrow \quad \sum_{i=0}^{m} a_i B_k^i (u_0 - x'_{k+1}) = 0 \quad \text{car } I - B_k \text{ est inversible.}$$

d'où :

$$m' \leq m.$$

Si $m' < m$ alors $\sum_{i=0}^{m'} a_i B_k^i (u_0 - x'_{k+1}) = 0 \Longrightarrow$

$$\sum_{i=0}^{m'} a_i B_k^i (u_0 - u_1) = \sum_{i=0}^{m'} a_i B_k^i (u_0 - x'_{k+1}) - \sum_{i=0}^{m'} a_i B_k^i (u_1 - x'_{k+1})$$

$$= (I - B_k) \sum_{i=0}^{m'} a_i B_k^i (u_0 - x'_{k+1}) = 0$$

Ce qui est en contradiction avec le fait que m est le degré du polynôme minimal de $B_k / u_1 - u_0$.

Donc $m = m' \Rightarrow x'_{k+1} = x_{k+1}$

D'autre part :

$$(\text{V}.2) \quad u_{i+1} - x = J.(u_i - x) + a(u_i) \cdot \|u_i - x\|^2.$$

et puisque F'' est majorée dans un voisinage de x alors $\|a(u_i)\| \leq M$ tant que u_i est dans ce voisinage.

Et d'après (*) on a :

$$(\text{V}.3) \quad x_{k+1} - u_{i+1} = B_k (x_{k+1} - u_i)$$

En ajoutant (V.2) à (V.3), nous obtenons :

$$(\text{II}.4) \qquad X_{k+1} - X = (J - B_k)\cdot(u_i - X) + B_k(X_{k+1} - X) + a(u_i)\,\|u_i - X\|^2.$$

soit :

$$(**) \qquad (I - B_k)\cdot(X_{k+1} - X) = (J - B_k)\cdot(u_i - X) + a(u_i)\cdot\|u_i - X\|^2.$$

et d'après (II.2), nous avons :

$$\|u_{i+1} - X\| \leq \left(\|J\| + M\,\|u_i - X\|\right)\cdot\|u_i - X\|$$

Si x_0 est bien choisi dans un voisinage de X, alors :

$$\|u_i - X\| \leq \rho$$

d'où : $\|u_{i+1} - X\| \leq (\|J\| + M\rho)\cdot\|u_i - X\| \leq (\|J\| + M\rho)^i\cdot\|u_0 - X\|.$

$i = 0, \ldots, m.$

$(**)$ s'écrit alors :

$$(I - B_k)\cdot(X_{k+1} - X) = (J - B_k)\cdot(u_i - X) + \underset{\substack{\downarrow \\ \text{indépendant de } k}}{O}\left(\|u_0 - X\|^2\right).$$

multiplions les deux membres par a_i et sommons sur i, $i = 0, \ldots, m$; nous obtenons :

$$(\text{II}.5) \qquad (I - B_k)(X_{k+1} - X)\cdot\sum_{i=0}^{m} a_i = (J - B_k)\sum_{i=0}^{m} a_i(u_i - X) + \left(\sum_{i=0}^{m} a_i\right)\cdot O\left(\|u_0 - X\|^2\right)$$

Or d'après $(*)$ on a :

$$\sum_{i=0}^{m} a_i\,(u_i - X) = (X_{k+1} - X)\cdot\sum_{i=0}^{m} a_i$$

en simplifiant par $\sum_{i=0}^{m} a_i \neq 0$ dans (II.5), nous avons :

$$(I - J)\cdot(X_{k+1} - X) = O\left(\|u_0 - X\|^2\right)$$

et comme $I - J$ est inversible, alors :

$$\|X_{k+1} - X\| = O\left(\|X_k - X\|^2\right). \quad \square$$

<u>Remarque:</u>

Si 0 est racine du polynôme minimal de J par rapport à $u_0 - x$ et si r est sa multiplicité ($r \geq 1$), l'algorithme (V.1) est remplacé par:

$$(V.6) \quad \left\| \begin{array}{ll} u_r = x_k & ; k \geq 1. \\ u_{i+1} = F(u_i) & ; i = r, \ldots, m \\ x_{k+1} = S_{m-r}^{(r)} & \end{array} \right.$$

et on montre de manière analogue que: $\| x_{k+1} - x \| = O(\| x_k - x \|^2)$.

Exemple 1 :

$$\left.\begin{cases} x = -(2x+y) + 0.127 \\ y = -0{,}5 \cdot (x+3y)^5 + 0.10512 \end{cases}\right) m = p = 2$$

La solution est $\begin{cases} x^* = 0.1 \\ y^* = 0.1 \end{cases}$

En partant du point : $\begin{cases} x_0 = 0 \\ y_0 = 0 \end{cases}$

On obtient les résultats suivants :

k	x_k	y_k

Exemple 2 :

$$\left.\begin{cases} x = 2xy - x^2 \\ y = x^2 - 2.9x\,(y-1) \end{cases}\right) m = p = 2$$

La solution est $x^* = 1$; $y^* = 1$

En partant du point $(1.05,\ 0.9)$ on obtient les résultats :

k	xk	yk

Exemple 3 :

$$\begin{cases} x = -\dfrac{y^4}{4} - \dfrac{3}{4} \\ y = -0.405 \cdot e^{1-x^2} + 1.405 \end{cases} \qquad p = m = 2$$

Solution est $x^* = -1$; $y^* = 1$.

On post de point : $x_0 = 0$; $y_0 = 0$; on obtient :

k	X_k	Y_k

- Chapitre V -

ACCELERATION DES METHODES DE PROJECTION-MINIMISATION ET METHODES DE PROJECTION OBLIQUE

INTRODUCTION :

L'objet de ce chapitre est "d'accélérer" les méthodes de projection-minimisation pour la résolution des systèmes linéaires. A chaque méthode de projection, nous associerons par un procédé que nous definirons, sa transformée. Nous montrerons que si la première converge, la seconde converge au moins aussi vite. Nous verrons que certaines transformées contiennent des méthodes connues comme le gradient conjugué ou le résidu conjugué. On se limitera pour les tests numériques aux méthodes associés à une décomposition d'une norme.

Dans le dernier paragraphe, nous montrerons que la plupart des méthodes d'extrapolation, sont des méthodes de projection oblique lorsqu'elles sont appliquées à des suites produites par des itérations linéaires.

I. Méthodes de projection-minimisation:

I.1 - Notations et definitions:

Soit à résoudre le système régulier d'ordre N :

$$(I.1) \quad A.x = b$$

Soit G une matrice d'ordre N, symétrique et définie positive. Le produit scalaire par rapport à G sera noté $(\ ,\ )_G$ et est définit par:

Si x et y sont deux vecteurs de $\mathbb{R}^N$ alors:

$$(x, y)_G = (x, Gy)$$

où $(\ ,\ )$ désigne le produit scalaire usuel sur $\mathbb{R}^N$. La norme par rapport à G est définit par : $\|x\|_G = \sqrt{(x, Gx)}$

Nous considérerons des méthodes itératives de résolution de (I.1) et nous adopterons les notations suivantes:

$$\begin{cases} x_k : & \text{solution approchée à l'étape } k \\ x^* : & \text{la solution de (I.1)} \\ \varepsilon_k = x_k - x^* : & \text{l'erreur à l'étape } k. \\ r_k = Ax_k - b : & \text{le résidu à l'étape } k. \end{cases}$$

* Si $\{p_1, \dots, p_j\}$ est un système libre de $\mathbb{R}^N$, nous noterons $\mathrm{span}\{p_1, \dots, p_j\}$ le sous-espace vectoriel de $\mathbb{R}^N$ engendré par $\{p_1, \dots, p_j\}$.

* On appelera aussi la partie symetrique de A, la matrice notée A^s et définie par:

$$A^s = \frac{1}{2}\left(A + A^T\right)$$

où A^T designe la matrice transposée de A.

Les méthodes que nous considérerons consistent à minimiser une certaine fonctionnelle positive $g(x)$ qui prend son minimum à la solution $x = x^*$ de (I.1) et tel que: $0 \leq g(x_{k+1}) \leq g(x_k)$; $k \geq 0$.

Nous utiliserons le critère d'accélération suivant:

<u>Definition:</u>

Soit (F) un algorithme qui genère la suite $(x_k)_{k \geq 1}$ à partir de x_0 donné et tel que:

$$\frac{g(x_{k+1})}{g(x_k)} \leq \rho < 1 \quad , \forall k \qquad \text{où } \rho \in \mathbb{R}_+$$

Soit (T) l'algorithme obtenue par une transformation de (F) et $(y_k)_{k \geq 0}$ la suite générée par (T) à partir de y_0 donné.

Nous dirons que (T) converge plus vite que (F) par rapport à g si: il existe $\rho' < \rho$ tel que:

$$\frac{g(y_{k+1})}{g(y_k)} \leq \rho' \quad , \forall k.$$

I.2 - Présentation :

Une méthode de projection-minimisation pour résoudre (I.1), est une méthode itérative qui consiste, à chaque étape k, à définir une matrice de direction $V_k = [v_1^{(k)}, \dots, v_q^{(k)}]$ de type (N,q) où q est fixé, $(1 \leq q \leq N)$ et à calculer ensuite x_{k+1} tel que :

$$x_{k+1} \in L_k \cap K_k$$

où :

$$L_k = \{ x \in \mathbb{R}^N / \ x = x_k + V_k \Lambda \quad \text{où } \Lambda \in \mathbb{R}^N \}$$

$$K_k = \{ x \in \mathbb{R}^N / \ V_k^T G . (x - x^*) = 0 \}$$

Il se pose alors le problème du choix de V_k pour assurer la convergence de la méthode. Des études ont été faites dans ce sens, notamment dans [1], [13], [16], [18] et [21]. Notre but ici sera de transformer ces méthodes pour en obtenir d'autres qui convergent plus vite au sens de la définition précédente.

$$x_{k+1} \in L_k \cap K_k \Rightarrow$$

$$(V_k^T G V_k) \Lambda = - V_k^T G s_k .$$

Pour que Λ existe et soit unique il faut que : $V_k^T G V_k$, matrice de type (q,q), soit régulière. Nous supposerons par la suite que cette condition est vérifiée.

Donc :

$$\Lambda = \Lambda_k = -\left(V_k^T G V_k\right)^{-1} V_k^T G S_k$$

d'où

$$\boxed{x_{k+1} = x_k - V_k \left(V_k^T G V_k\right)^{-1} V_k^T G S_k.}$$

3 - <u>Algorithme de calcul: RPA.</u>

On pose : $P_q^{(k)} = V_k \left(V_k^T G V_k\right)^{-1} V_k^T G S_k.$

alors : $x_{k+1} = x_k - P_q^{(k)}.$

Nous avons :

$$V_k^T G P_q^{(k)} = \left(V_k^T G V_k\right) . \left(V_k^T G V_k\right)^{-1} V_k^T G S_k$$
$$= V_k^T G S_k$$

En posant : $P_q^{(k)} = \alpha_1 v_1^{(k)} + \dots + \alpha_q v_q^{(k)}$, le calcul de $P_q^{(k)}$ revient alors à résoudre le problème d'interpolation généralisé suivant :

$$\begin{cases} P_q^{(k)} = \alpha_1 v_1^{(k)} + \dots + \alpha_q v_q^{(k)} \\ \left(v_i^{(k)}, P_q^{(k)}\right)_G = \left(v_i^{(k)}, S_k\right)_G \quad ; i = 1, \dots, q. \end{cases}$$

A chaque étape k, le calcul de $P_q^{(k)}$ se fait récursivement par le RIA en posant dans celui-ci :

$$x_i = v_i^{(k)} \quad ; \quad z_i = v_i^{(k)}$$

$$\omega_i = \left(v_i^{(k)}, S_k\right)_G \quad ; i = 1, \dots, q.$$

D'où l'algorithme :

$$P_0^{(k)} = 0 \quad ; \quad g_{qi}^{(k)} = v_i^{(k)}$$

$$P_j^{(k)} = P_{j-1}^{(k)} + \frac{(v_j^{(k)}, s_k)_G - (v_j^{(k)}, P_{j-1}^{(k)})_G}{(v_j^{(k)}, g_{j-1,j}^{(k)})_G} \cdot g_{j-1,j}^{(k)} \quad ; \; j = 1, \dots, q.$$

$$g_{j,i}^{(k)} = g_{j-1,i}^{(k)} - \frac{(v_j^{(k)}, g_{j-1,i}^{(k)})_G}{(v_j^{(k)}, g_{j-1,j}^{(k)})_G} \times g_{j-1,j}^{(k)} \quad ; \; i > j > 0$$

Exemple.

prenons $G = I$ et $V_k = A^T W_k$

alors:

$$(v_j^{(k)}, s_k)_G = (A^T w_j^{(k)}, s_k) = (w_j^{(k)}, r_k) \quad ; \; j = 1, \dots, q.$$

Dans ce cas le calcul de $P_q^{(k)}$ nécessite :

* $N(q^2 + 4q)$ multiplications
* $N(q^2 + 6q)$ additions et soustractions
* le calcul des vecteurs $A^T w_j^{(k)}$; $j = 1, \dots, q$

Remarque 1 :

q est pris en général très petit devant la taille N du système à résoudre.

I.4 - Propriétés:

propriété 1 :

$$\boxed{\begin{aligned} &i)\quad V_k^T G s_{k+1} = 0 \\ &ii)\quad \|s_{k+1}\|_G = \min_{x \in L_k} \|x - x^*\|_G \end{aligned}}$$

dem:

i) $x_{k+1} \in K_k \Rightarrow V_k^T G (x_{k+1} - x^*) = V_k^T G s_{k+1} = 0.$

ii) $x \in L_k \Rightarrow \quad x = x_k + V_k \Lambda$

donc: $x - x^* = s_k + V_k \Lambda$

d'où: $\|x - x^*\|_G^2 = (s_k + V_k \Lambda, G s_k + G V_k \Lambda)$

$$= \|s_k\|_G^2 + 2 \Lambda^T V_k^T G s_k + \Lambda^T V_k^T G V_k \Lambda$$

Le minimum de cette dernière expression est atteint pour,

$$\Lambda = \Lambda_k = - (V_k^T G V_k)^{-1} V_k^T G s_k .$$

soit pour $x = x_{k+1}$ ■

Remarques:

1°) La fonctionnelle positive $g(x)$ sera dans toute la suite: $\|x\|_G$.

2°) $\|s_{k+1}\|_G^2 = \|s_k\|_G^2 - \|V_k \Lambda_k\|_G^2 \Rightarrow 0 \leq \|s_{k+1}\|_G \leq \|s_k\|_G$.

3°) x_{k+1} est la projection orthogonale, par rapport à G, de x_k sur K_k.

4°) Comme S_k n'est pas connue, G sera choisit de telle sorte que GS_k soit calculable.

Voici maintenant trois classes de méthodes de projection, qui ont été étudiées séparément dans [21], correspondants à trois choix de la matrice G.

a) <u>classe A</u> :

$$G = A^T A \Rightarrow GS_k = A^T r_k$$

donc : $$x_{k+1} = x_k - V_k^T \left(V_k^T A^T A V_k \right)^{-1} V_k^T A^T r_k .$$

b) <u>classe B</u> :

$$G = I \quad \text{et} \quad V_k = A^T W_k$$

donc : $$x_{k+1} = x_k - A^T W_k \left(W_k^T A A^T W_k \right)^{-1} W_k^T r_k .$$

c) <u>classe C</u> :

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

$$x_{k+1} = x_k - V_k \left(V_k^T A V_k \right)^{-1} V_k^T r_k .$$

_Nous nous proposons par la suite de transformer ces méthodes et nous montrerons que ces méthodes transformées convergent au moins aussi vite que les premières.

II - TRANSFORMATION DES MÉTHODES DE PROJECTION :

Definition du procédé:

Considerons l'algorithme de projection-minimisation pour la résolution de (I.1) : q étant fixé

$$(\mathrm{II}.1)\quad \left\| \begin{array}{l} x_0 \text{ donné et à l'étape } k: \\ Z_k \text{ de type } (N,q)\text{, choisi} \\ x_{k+1} = x_k + Z_k \Lambda_k \quad \text{où } \Lambda_k = -\left(Z_k^T G Z_k\right)^{-1} Z_k^T G S_k \\ Z_k = Z(k, r_k) \quad \text{où } Z : \mathbb{N} \times \mathbb{R}^N \longrightarrow \mathcal{M}(N,q). \end{array} \right.$$

Nous appelerons cette méthode : " méthode originale". Z_k ne dépend que de k et de r_k: c'est le cas par exemple des méthodes associées à une sur-décomposition d'une norme [16] et d'autres méthodes étudiées dans [13], [18] et [21].

Nous associons à cette méthode (II.1) la "méthode transformée" definie de la manière suivante :

$$x_{k+1} = x_k + V_k \Lambda'_k \quad \text{où } \Lambda'_k = -\left(V_k^T G V_k\right)^{-1} V_k^T G S_k$$

où V_k ne dépends pas uniquement de k et de r_k, comme pour (II.1), mais aussi de $V_{k-1}, \dots, V_0$:

$$V_k = Z_k + \sum_{i=0}^{k-1} V_i \cdot \Gamma_k^{(i)}$$

où $\Gamma_k^{(i)}$ est une matrice (q,q) que l'on déterminera en

imposant aux V_i $(i=0,\dots,k)$ de vérifier la relation de G-orthogonalité suivante :

$$(\text{II}.2) \quad V_i^T G V_j = 0 \in M(q,q) \quad \text{pour } i \neq j.$$

donc pour $j \in \{0,\dots,k-1\}$, on a :

$$V_j^T G V_k = V_j^T G Z_k + \sum_{i=0}^{k-1} \left(V_j^T G V_i\right) . \Gamma_k^{(i)} = 0$$

Or $(\text{II}.2) \Rightarrow V_i^T G V_j = 0$ pour $i \neq 0$, donc :

$$V_j^T G V_k = V_j^T G Z_k + \left(V_j^T G V_j\right) . \Gamma_k^{(j)}$$

on supposera que $\det\left(V_j^T G V_j\right) \neq 0$

d'où :

$$(\text{II}.3) \quad \underline{\underline{\Gamma_k^{(j)} = -\left(V_j^T G V_j\right)^{-1} . V_j^T G Z_k}} \quad ; j=0,\dots,k-1.$$

On obtient finalement l'algorithme "transformé" suivant :

$$\boxed{\begin{array}{l} x_0 \text{ donné} \; ; \; V_0 = Z_0 \text{ et à l'étape } k : \\ x_{k+1} = x_k + V_k . \Lambda'_k \quad \text{où } \Lambda'_k = -\left(V_k^T G V_k\right)^{-1} V_k^T G S_k \\ \Gamma_{k+1}^{(i)} = -\left(V_i^T G V_i\right)^{-1} . V_i^T G Z_{k+1} \quad ; i=0,\dots,k \\ V_{k+1} = Z_{k+1} + \sum_{i=0}^{k} V_i . \Gamma_{k+1}^{(i)} \end{array}}$$

(II.4)

Le problème qui se pose est que l'on est obligé stocker en mémoire $V_0,\dots,V_k$ pour le calcul de V_{k+1}. Alors une autre façon de

procéder serait de ne prendre dans l'expression de V_{k+1} que les m dernières matrices de direction V_j où m est un entier fixé choisi selon la place mémoire dont on dispose. Dans le cas :

$$(\text{II}.5) \quad \boxed{V_{k+1} = Z_{k+1} + \sum_{i=\max(0,k-m+1)}^{k} V_i \cdot \Gamma_{k+1}^{(i)}}$$

où les $\Gamma_{k+1}^{(i)}$ restent ceux donnés par l'expression (II.3). Nous verrons que les propriétés que nous donnerons pour la méthode (II.4) resteront vraies pour (II.5). Nous traiterons particulièrement la méthode (II.5) dans le cas où $q=1$.

Donnons tout d'abord quelques résultats qui nous serviront par la suite.

Propriétés :

Propriétés :

$$\boxed{\begin{aligned} &i)\ V_i^T G S_k = 0 \quad ; \ i=0,\dots,k-1 \\ &ii)\ V_k^T G S_k = Z_k^T G S_k \\ &iii)\ V_k^T G V_k = Z_k^T G Z_k - \sum_{i=0}^{k-1} \Gamma_k^{(i)T} \left(V_i^T G V_i\right) \Gamma_k^{(i)} \end{aligned}}$$

dem :

$$i) \quad S_k = S_{k-1} - V_{k-1}\left(V_{k-1}^T G V_{k-1}\right)^{-1} \cdot V_{k-1}^T G S_{k-1}$$

Soit $j \in \{0,\dots,k-1\}$, alors :

$$S_k = S_j - \sum_{i=j}^{k-1} V_i \cdot \left(V_i^T G V_i\right)^{-1} \cdot V_i^T G S_i$$

donc :

$$V_j^T G S_k = V_j^T G S_j - \sum_{i=j}^{k-1} \left(V_j^T G V_i\right) \cdot \left(V_i^T G V_i\right)^{-1} V_i^T G S_i$$

Or :

$$V_j^T G V_i = 0 \quad \text{pour } i \neq j \quad , \text{ d'où}$$

$$V_j^T G S_k = V_j^T G S_j - V_j^T G S_j$$

$$= 0 \qquad , j = 0, \dots, k-1$$

ii)

$$V_k = Z_k + \sum_{i=0}^{k-1} V_i \Gamma_k^{(i)} \implies$$

$$V_k^T G S_k = Z_k^T G S_k + \sum_{i=0}^{k-1} \Gamma_k^{(i)T} V_i^T G S_k$$

Or d'après i) on a :

$$V_i^T G S_k = 0 \quad \text{pour } i = 0, \dots, k-1$$

donc :

$$V_k^T G S_k = Z_k^T G S_k .$$

iii)

$$V_k^T G V_k = V_k^T G Z_k + \sum_{i=0}^{k-1} \left(V_k^T G V_i\right) \Gamma_k^{(i)}$$

Or : $V_k^T G V_i = 0$ pour $i = 0, \dots, k-1$, d'après (I.2), donc :

$$V_k^T G V_k = V_k^T G Z_k$$

$$= \left(Z_k^T + \sum_{i=0}^{k-1} \Gamma_k^{(i)T} V_i^T\right) G Z_k$$

$$= Z_k^T G Z_k + \sum_{i=0}^{k-1} \Gamma_k^{(i)T} \underbrace{V_i^T G Z_k} \qquad (*)$$

or $\Gamma_k^{(i)}$ est donné par :

$$V_i^T G Z_k = -\left(V_i^T G V_i\right) \Gamma_k^{(i)} \quad ; i = 0, \dots, k-1 .$$

en remplaçant dans (*), nous obtenons :

$$V_k^T G V_k = Z_k^T G Z_k - \sum_{i=0}^{k-1} \Gamma_k^{(i)T} \left(V_i^T G V_i\right) \Gamma_k^{(i)} . \blacksquare$$

propriété 3

Soit $(x_k)_k$ la suite produite par l'algorithme (II.4) et $s_k = x_k - x^*$, alors:

$$\frac{\|s_{k+1}\|_G}{\|s_k\|_G} \leq \tau'_k = \sqrt{1 - \frac{\|V_k^T G s_k\|^2}{\|s_k\|_G^2 \cdot \rho(V_k^T G V_k)}} .$$

dem.

$$s_{k+1} = s_k - V_k (V_k^T G V_k)^{-1} V_k^T G s_k$$

donc:

$$\|s_{k+1}\|_G^2 = \|s_k\|_G^2 - s_k^T G V_k (V_k^T G V_k)^{-1} V_k^T G s_k$$

et

$$\frac{\|s_{k+1}\|_G^2}{\|s_k\|_G^2} = 1 - \frac{s_k^T G V_k (V_k^T G V_k)^{-1} V_k^T G s_k}{\|s_k\|_G^2}$$

$$\leq 1 - \frac{\|V_k^T G s_k\|^2}{\|s_k\|_G^2 \cdot \rho(V_k^T G V_k)}$$

Ce qui montre la propriété. ■.

De la même manière, on montre que si s_k est cette fois-ci la suite produite par l'algorithme originale (II.1), on a:

$$\frac{\|s_{k+1}\|_G}{\|s_k\|_G} \leq \tau_k = \sqrt{1 - \frac{\|Z_k^T G s_k\|^2}{\|s_k\|_G^2 \cdot \rho(Z_k^T G Z_k)}}$$

τ_k et τ'_k sont dits "taux de convergence" des méthodes (II.1) et de sa transformée (II.4).

Si $\tau_k < 1$, $\forall k$ alors la méthode originale (II.1) converge et on va montrer maintenant que la méthode transformée (II.4) converge au moins aussi vite.

§ Accélération de la convergence:

Théorème 1:

i) Si τ_k et τ'_k sont les taux de convergence définis précédemment alors $\tau'_k \leq \tau_k$.

ii) Si la méthode (II.1) converge, la méthode transformée converge au moins aussi vite.

dem:

A l'étape k on part de la même solution approché x_k pour (II.1) et (II.4).

On a vu que:

$$\tau_k = \sqrt{1 - \frac{\|Z_k^T G S_k\|^2}{\|S_k\|_G^2 \cdot \rho(Z_k^T G Z_k)}} \quad ; \quad \tau'_k = \sqrt{1 - \frac{\|V_k^T G S_k\|^2}{\|S_k\|_G^2 \cdot \rho(V_k^T G V_k)}}.$$

or d'après la propriété 2, on a:

$$V_k^T G S_k = Z_k^T G S_k$$

Montrer que $\tau'_k \leq \tau_k$ revient donc à montrer que:

$$\rho(V_k^T G V_k) \leq \rho(Z_k^T G Z_k).$$

Et d'après le (iii) prop 2), on a:

$$V_k^T G V_k = Z_k^T G Z_k - \sum_{i=0}^{k-1} \Gamma_k^{(i)T} \cdot (V_i^T G V_i) \Gamma_k^{(i)}$$

donc:

$$(*) \quad \sup_{\|x\|=1} x^T V_k^T G V_k x \leq \sup_{\|x\|=1} x^T Z_k^T G Z_k x - \sum_{i=0}^{k-1} \inf_{\|x\|=1} x^T \Gamma_k^{(i)T} (V_i^T G V_i) \Gamma_k^{(i)} x$$

G symetrique et definie positive $\Longrightarrow \Gamma_k^{(i)T}(V_i^T G V_i)\Gamma_k^{(i)}$ est symetrique semi-definie positive, $i=0,\dots,k-1$. D'où :

$$\mu_k^{(i)} = \inf_{\|x\|=1} x^T \Gamma_k^{(i)T} (V_i G V_i) \Gamma_k^{(i)} x \geq 0 \quad ; i=0,\dots,k-1.$$

et $$\mu_k = \sum_{i=0}^{k-1} \mu_k^{(i)} \geq 0$$

$$(*) \Longrightarrow \rho(V_k^T G V_k) \leq \rho(Z_k^T G Z_k) - \mu_k$$

Comme $\mu_k \geq 0 \quad \Longrightarrow \rho(V_k^T G V_k) \leq \rho(Z_k^T G Z_k)$.

donc : $\tau'_k \leq \tau_k$.

Si $\tau_k < 1$ alors la méthode transformée (II.4) converge au moins aussi vite que la méthode originale (II.1) au sens de la definition donnée précédement ■

Remarque 3 :

On peut donner des résultats similaires à ceux des propriétés 2 et 3 et du theorème 1 pour la méthode (II.5).

Dans ce qui suit, nous allons nous interesser au cas $q=1$.

III. Etude du cas $q=1$.

Méthode générale

Les matrices de direction V_k et Z_k deviennent dans ce cas des vecteurs de direction que l'on notera : p_k et z_k.

L'algorithme transformé (II.4) devient dans ce cas :

(III.1)

$$\boxed{\begin{array}{l} x_0 \text{ donné}, \ p_0 = z_0 \\ x_{k+1} = x_k - \alpha_k p_k \quad \text{où} \quad \alpha_k = \dfrac{(p_k, G s_k)}{(p_k, G p_k)} \\ p_{k+1} = z_{k+1} + \displaystyle\sum_{i=0}^{k} \gamma_{k+1}^{(i)} \cdot p_i \quad \text{où} \\ \qquad \gamma_{k+1}^{(i)} = -\dfrac{(p_i, G z_{k+1})}{(p_i, G p_i)} \ ; \ i = 0, \dots, k. \end{array}}$$

Et nous avons le résultat suivant :

Propriété 4 :

$$\boxed{\begin{array}{ll} i) & (p_i, G p_j) = 0 \ ; \ \text{pour } i \neq j \\ ii) & (p_i, G s_j) = 0 \ ; \ \text{si } i < j \\ iii) & (p_i, G s_i) = (z_i, G s_i) \ ; \ \forall i. \end{array}}$$

dém

dérive de la prop 2 □

L'inconvénient de la méthode (III.1) est qu'elle nécessite le stockage de $p_0, \dots, p_k$ nécessaires au calcul de p_{k+1}.

On modifie alors (III.1) de la manière suivante :

on ne prends dans l'expression de p_{k+1} que les m dernières directions p_i, où m est fixé choisi. On obtient donc l'algorithme :

$$(\text{III}.2)\quad \left\| \begin{array}{l} x_0 \text{ donné} \; ; \; p_0 = z_0 \\ x_{k+1} = x_k - \alpha_k p_k \\ p_{k+1} = z_{k+1} + \displaystyle\sum_{i=\max(0,k-m+1)}^{k} \gamma_{k+1}^{(i)} \cdot p_i \end{array} \right.$$

où $\gamma_{k+1}^{(i)}$ est donnée dans (III.1) ainsi que α_k.

Le choix de la matrice G et des directions z_k nous donnera soit des algorithmes comme le gradient conjugué [19] ou le résidu conjugué généralisé [23] soit de nouveaux algorithmes qui convergent plus vite que ceux auxquels ils sont associés.

Pour les tests numériques nous allons nous intéresser aux transformées des méthodes de projection associées à des décompositions de normes [16].

II.2 Méthodes basées sur une décomposition d'une norme.

Les normes φ_1 et φ_2 sont definies par:

$$\varphi_1(x) = \sum_{i=1}^{N} |x_i| \quad ; \quad \varphi_2(x) = \left(\sum_{i=1}^{N} x_i^2\right)^{1/2}$$

On considère le choix suivant $G = I$ et on pose $z_k = A^T \zeta_k, \forall k$

La méthode basée sur une décomposition d'une norme φ est définie par:

Méthode I:

x_0 donné et à l'iteration k:

$$x_{k+1} = x_k - \alpha_k A^T \zeta_k \quad \text{avec} \quad \alpha_k = \frac{(\zeta_k, r_k)}{\|A^T \zeta_k\|^2}$$

où:

$$\varphi(r_k) = \zeta_k^T \cdot r_k \quad \text{donc} \quad \alpha_k = \frac{\varphi(r_k)}{\|A^T \zeta_k\|^2}.$$

La méthode transformée est donc:

Methode II:

x_0 donné ; $p_0 = A^T \zeta_0$

$$x_{k+1} = x_k - \alpha'_k p_k \quad \text{où} \quad \alpha'_k = \frac{(\zeta_k, r_k)}{\|p_k\|^2} = \frac{\varphi(r_k)}{\|p_k\|^2}$$

$$p_{k+1} = A^T \zeta_{k+1} + \sum_{i=0}^{k} \gamma_{k+1}^{(i)} p_i \quad \text{où}$$

$$\gamma_{k+1}^{(i)} = - \frac{(p_i, A^T \zeta_{k+1})}{\|p_i\|^2}.$$

Exemple 1 :

A matrice d'ordre 25 non symétrique :

$$A = \begin{pmatrix} B & I & & & \\ I & B & & 0 & \\ & & \ddots & & \\ & 0 & & & I \\ & & & I & B \end{pmatrix}$$

avec :

$$B = \begin{pmatrix} 4 & -1-\delta & & & 0 \\ -1+\delta & 4 & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & -1-\delta \\ 0 & & & -1+\delta & 4 \end{pmatrix}$$

d'ordre 5

$\delta = 0.01$

0.5

k	(I) Méthode associée à φ_1	(II) Méthode transformée
	$\varphi_1(x_k - x^*)$	$\varphi_1(x_k - x^*)$
1	.2428000844D+00	.2339964000D+00
2	.2397702325D+00	.2245[illegible]38816D+00
3	.2321028366D+00	.2181378483D+00
4	.2301465622D+00	.2120962143D+00
5	.2288318784D+00	.1928748312D+00
6	.2277588775D+00	.1418457346D+00
7	.2248894887D+00	.1004779674D+00
8	.2246534887D+00	.6302541836D-01
9	.2224834203D+00	.3360706654D-01
10	.2224834203D+00	.2706898031 7D-01
11	.2207450519D+00	.1783758585D-01
12	.2181589381 1D+00	.7848278267D-02
13	.2181589381 1D+00	.1318309463D-02
14	.2162718184D+00	.6277700739D-03
15	.2149213228D+00	.5239937952D-03
16	.2137898700D+00	.4557330210D-03
17	.2125241528D+00	.1097963815 4D-03
18	.2115674888D+00	.6299852790D-04
19	.2129135388D+00	.2241232843D-05
20	.2107898039D+00	.1692182290 7D-05
21	.2094389497D+00	.5458484028D-06
22	.2053379474D+00	.1557752463 7D-06
23	.2053379474D+00	.3778334860D-07
24	.2045042215D+00	.2470828339 1D-08
25	.2035424882D+00	.1809823387D-13
58	.1779820720D+00	0
59	.1770869325D+00	
60	.1768517987D+00	

Exemple 2 :

Reprenons la même fame pour A que pour l'exemple 1 avec $\delta = 0$. A est donc symetrique d'ordre 25. Nous obtenons pour la transformée de la méthode associée à une decomposition de φ_1 les résultats suivants :

$(x_0)_i = 0.5$; $i = 1, \ldots, 25$

k	(II) $\varphi_1(x_k - x^*)$
1	.18000000000D+00
2	.17448973582D+00
3	.16520388350D+00
4	.14161347036D+00
5	.12332916881D+00
6	.10786050452D+00
7	.87369888883D-01
8	.30588876172D-01
9	.19358933781D-01
10	.12306830807D-01
11	.44512850703D-02
12	.10848102870D-02
13	.39968028887D-14

Exemple 3 :

A matrice de Hilbert d'ordre 20 : $A(i,j) = 1/(i+j-1)$.

$(x_0)_i = 0.5$; $i = 1, \ldots, 20$

	(I) Méthode associée à φ_1	(II) Méthode transformée
k	$\varphi_1(x_k - x_*)$	$\varphi_1(x_k - x_*)$
1	.82413687932D-01	.10270523358D-01
2	.48332747767D-01	.42758935513D-02
3	.39048819852D-01	.14724757733D-02
4	.33112570511D-01	.47230407374D-03
5	.32712118352D-01	.13856012238D-03
6	.32856748538D-01	.36449775080D-04
7	.29323915908D-01	.88735983473D-05
8	.29737441807D-01	.20048773343D-05
9	.26483703750D-01	.41878682644D-06
10	.26580551838D-01	.46272983580D-06

Exemple 4 :

$$A = \begin{pmatrix} 2 & -1-\delta & & & 0 \\ -1+\delta & 2 & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & -1-\delta \\ 0 & & & -1+\delta & 2 \end{pmatrix}$$

avec $\delta = 0.01$. A d'ordre 25. $(x_0)_i = 0.5$; $i = 1, \dots, 25$

(I) Méthode associée à φ_1	
k	$\varphi_1(x_k - x_*)$
1	.[illegible]
2	.[illegible]
11	.[illegible]
17	.[illegible]
18	.[illegible]
19	.[illegible]
20	.[illegible]
21	.[illegible]
22	.[illegible]
23	.[illegible]
24	.[illegible]
25	.[illegible]

(II) Méthode Transformée	
k	$\varphi_1(x_k - x_*)$
1	.[illegible]
2	.[illegible]
11	.[illegible]
17	.[illegible]
18	.[illegible]
19	.[illegible]
20	.[illegible]
21	.[illegible]
22	.[illegible]
23	.[illegible]

1.3 Etude unifiée des méthodes de gradient conjugué et de résidu conjugué:

Soit H une matrice carrée d'ordre N, symétrique et définie positive. On pose:

$$G = {}^tA\,H\,A \quad \text{et } z_k = r_k \quad ; \ \forall k.$$

L'algorithme (III.1) devient alors:

$$(\text{III}.3)\qquad \boxed{\begin{array}{l} x_0 \text{ donné}, \ p_0 = r_0 \quad \text{et à l'itération } k : \\ x_{k+1} = x_k - \alpha_k p_k \quad \text{où} \quad \alpha_k = \dfrac{(Ap_k, Hr_k)}{(p_k, Gp_k)} \\ r_{k+1} = r_k - \alpha_k A p_k \\ p_{k+1} = r_{k+1} + \displaystyle\sum_{i=0}^{k} \gamma_{k+1}^{(i)} \cdot p_i \quad \text{où} \quad \gamma_{k+1}^{(i)} = -\dfrac{(p_i, Gr_{k+1})}{(p_i, Gp_i)} \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad i = 0, \dots, k. \end{array}}$$

* Si $H = I$ l'algorithme (III.3) n'est autre que le résidu conjugué généralisé [23].

* Si $H = A^{-1}$ et A symétrique et définie positive, on obtient le gradient conjugué [19]:

En effet si $H = A^{-1}$ et A symetrique et définie positive:

$G = A$.

montrons que $\gamma_{k+1}^{(i)} = 0$ pour $i = 0, \dots, k-1$.

$\gamma_{k+1}^{(i)} = -(p_i, Ar_{k+1}) / (p_i, Ap_i)$

Nous allons montrer que $(p_i, Ar_{k+1}) = 0$ pour $i = 0, \dots, k-1$ et pour cela nous allons utiliser le 6° de la propriétés c'est à dire que:

$$(*) \quad (r_i, r_j) = 0 \quad \text{si } i < j .$$

d'après l'algorithme (III.3), nous avons:

$$r_{i+1} - r_i = -\alpha_i A p_i$$

donc pour $i \in \{0, \dots, k-1\}$ on a:

$$(p_i, Ar_{k+1}) = (r_{k+1}, Ap_i) \quad \text{puisque } A \text{ est symétrique}$$

$$= \left(r_{k+1}, -\frac{1}{\alpha_i}(r_{i+1} - r_i)\right)$$

$$= -\frac{1}{\alpha_i}\left[(r_{k+1}, r_{i+1}) - (r_{k+1}, r_i)\right]$$

et d'après (*):

$$(r_{k+1}, r_{i+1}) = 0 \text{ et } (r_{k+1}, r_i) = 0 \quad \text{pour } i = 0, \dots, k-1$$

donc $(p_i, Ar_{k+1}) = 0 \Rightarrow \gamma_{k+1}^{(i)} = 0$; $i = 0, \dots, k-1$.

Et dans ce cas (II.3) n'est autre que le gradient conjugué.

Dans toute la suite, nous supposerons que la partie symétrique de HA est définie positive. Nous avons les résultats suivants :

propriétés :

1) $(p_i, G p_j) = 0 \quad ; i \neq j$

2°) $(Ap_i, Hr_j) = 0 \quad ; i < j$

3°) $(Ap_i, Hr_i) = (Ar_i, Hr_i) \quad , \forall i$

4°) $(Hr_i, Ap_j) = (Hr_0, Ap_j) \quad ; \quad i \leq j$

5°) $(r_i, G p_i) = (p_i, G p_i) \quad ; \quad \forall i$

6°) $(Ar_i, Hr_j) = 0 \qquad \text{si } i < j$

dem :

1°), 2°) et 3°) se déduisent directement de la prop 4.

4°) Soit $j \geq i$. On a :

$$r_i = r_0 - \sum_{l=0}^{i-1} \alpha_l A p_l$$

$\Rightarrow$

$$Hr_i = Hr_0 - \sum_{l=0}^{i-1} \alpha_l HAp_l$$

donc

$$(Hr_i, Ap_j) = (Hr_0, Ap_j) - \sum_{l=0}^{i-1} \alpha_l (HAp_l, Ap_j)$$

Or :

$$(HAp_l, Ap_j) = ({}^tAHAp_l, p_j) = (Gp_l, p_j)$$
$$= 0 \quad \text{car } l < j$$

d'où : $(Hr_i, Ap_j) = (Hr_0, Ap_j)$ pour $l \leq j$.

5°) $p_i = r_i + \sum_{l=0}^{i-1} \gamma_i^{(l)} \cdot p_l$

$\Rightarrow$

$$r_i = p_i - \sum_{l=0}^{i-1} \gamma_i^{(l)} \cdot p_l$$

donc :

$$(r_i, G p_i) = (p_i, G p_i) - \sum_{l=0}^{i-1} \gamma_i^{(l)} (p_l, G p_i)$$

et comme $l < i$ alors :

$$(p_l, G p_i) = 0 \quad \text{pour } l = 0, \dots, i-1$$

d'où

$$(r_i, G p_i) = (p_i, G p_i).$$

6°) on a : $Ar_i = Ap_i - \sum_{l=0}^{i-1} \gamma_i^{(l)} Ap_l$

Soit $j > i$, alors :

$$(Ar_i, Hr_j) = (Ap_i, Hr_j) - \sum_{l=0}^{i-1} \gamma_i^{(l)} (Ap_l, Hr_j)$$

Or d'après 2°) :

$$(Ap_l, Hr_j) = 0 \quad \text{car } l < j$$

et $(Ap_i, Hr_j) = 0$ car $i < j$

donc :

$$(Ar_i, Hr_j) = 0 \quad ; \quad i < j. \ \square$$

propriétés :

1°) $(p_i, Gp_i) \leq (r_i, Gr_i)$, $\forall i$

2°) si $r_i \neq 0$ alors $p_i \neq 0$.

dem :

1°) $Gp_i = Gr_i + \sum_{l=0}^{i-1} \gamma_i^{(l)} Gp_l$, $\forall i$

donc :

$$(r_i, Gp_i) = (r_i, Gr_i) + \sum_{l=0}^{i-1} \gamma_i^{(l)} (r_i, Gp_l)$$

or :

$$(r_i, Gp_i) = (p_i, Gp_i) \text{ , } \forall i \text{ d'après la prop 5.}$$

et

$$\gamma_i^{(l)} = -\frac{(p_l, Gr_i)}{(p_l, Gp_l)} \text{ ; } l = 0, \dots, i-1$$

donc :

$$(r_i, Gr_i) = (p_i, Gp_i) + \sum_{l=0}^{i-1} \frac{(r_i, Gp_l)^2}{(p_l, Gp_l)}$$

Comme G est symetrique et definie positive, alors :

$$\forall i : \quad (r_i, Gr_i) \geq (p_i, Gp_i).$$

2°) Supposons que $p_i = 0$, alors $(Ap_i, Hr_i) = 0$

or d'après la prop 4 : $(Ap_i, Hr_i) = (Ar_i, Hr_i)$, donc

$(Ar_i, Hr_i) = 0$, $\forall i$.

et comme H est symetrique, alors $(HAr_i, r_i) = 0$. Mais la partie symetrique de HA est definie positive. D'où

$$(HAr_i, r_i) = 0 \Longrightarrow r_i = 0 \quad \square$$

On definit la fonctionnelle e par:

$$e(x) = (Ax-b, Ax-b)_H = \|Ax-b\|_H.$$

On a:

$e(x^*)=0$, x^* est l'unique point qui réalise le minimum de $e(x)$ sur $\mathbb{R}^N$. Nous avons le résultat suivant:

Theorème 2:

x_{k+1} réalise le minimum de $e(x)$ sur l'espace affine $x_0 + \text{sev}\{p_0, \dots, p_k\}$.

dem:

Supposons que $r_i \neq 0$ pour $i \in \{0, \dots, k\}$, sinon le théorème est demontré. $r_i \neq 0 \Longrightarrow p_i \neq 0$ (prop 6). Et puisque le système $\{p_0, \dots, p_k\}$ est G-orthogonal alors il est libre.

Soit x un élément de l'espace affine: $x_0 + \text{sev}\{p_0, \dots, p_k\}$, alors $x = x_0 + \sum_{i=0}^{k} \alpha_i p_i$

et:

$$e(x) = (Ax-b, H(Ax-b))$$
$$= \left(r_0 + \sum_{i=0}^{k} \alpha_i Ap_i \,,\, Hr_0 + \sum_{i=0}^{k} \alpha_i HAp_i\right)$$

En developpant cette expression, on obtient:

$$e(x) = \|r_0\|_H^2 + 2\sum_{i=0}^{k} \alpha_i (r_0, HAp_i) + \sum_{i=0}^{k}\sum_{j=0}^{k} \alpha_i \alpha_j (Ap_i, HAp_j)$$

Or :

$$(Ap_i, HAp_j) = (p_i, Gp_j)$$
$$= 0 \quad \text{si } i \neq j$$

donc : $e(x) = \|r_0\|_H^2 + 2\sum_{i=0}^{k} \alpha_i (r_0, HAp_i) + \sum_{i=0}^{k} \alpha_i^2 (p_i, Gp_i)$

Le minimum de $e(x)$ est donc atteint pour :

$$\alpha_i = -\frac{(r_0, HAp_i)}{(p_i, Gp_i)} \quad ; i = 0, \dots, k.$$

Or :

$$(r_0, HAp_i) = (Hr_0, Ap_i)$$
$$= (Hr_i, Ap_i) \quad \text{d'après la prop4.}$$

d'où :

$$\alpha_i = -\frac{(Hr_i, Ap_i)}{(p_i, Gp_i)} \quad ; i = 0, \dots, k$$

En reportant dans l'expression de x on a :

$$x = x_{k+1} = x_0 + \sum_{i=0}^{k} \alpha_i p_i$$

x_{k+1} réalise donc le minimum de $e(x)$ sur l'espace affine :

$x_0 + \text{Sev}\{p_0, \dots, p_k\}$ $\square$

Nous obtenons enfin le résultat suivant :

Théorème 3 :

> L'algorithme (III.3) converge en N itérations au plus vers la solution x^* du système : $Ax = b$.

dem :

* Si $r_i = 0$ pour $i \in \{0, \dots, N-1\}$ alors $x_i = x^*$ et l'algorithme converge en $i < N$, itérations.

* Sinon, $\forall i \in \{0, \dots, N-1\}$, $r_i \neq 0 \Longrightarrow p_i \neq 0$ (prop 6), donc $\{p_0, \dots, p_{N-1}\}$ est un système libre (sur G-orthogonal) donc $x_0 + \mathrm{Sev}\{p_0, \dots, p_{N-1}\} = \mathbb{R}^N$.

Or d'après le théorème 2, x_N réalise le minimum de $e(x)$ sur : $x_0 + \mathrm{Sev}\{p_0, \dots, p_{N-1}\} = \mathbb{R}^N$

$$\Longrightarrow \quad x_N = x^*.$$

L'algorithme (III.3) converge donc en N itérations au plus. $\square$

1.4 Modification de la méthode (III.3) :

Soit m un entier fixé choisi. Considérons l'algorithme (II.5) avec $q=1$ et :

$$G = {}^tAHA \quad ; \quad z_k = r_k \quad , \forall k.$$

où H est symétrique définie positive d'ordre N. Nous obtenons alors l'algorithme :

$$(\text{III}.4) \quad \left\| \begin{array}{l} x_0 \text{ donné} \quad ; \; p_0 = r_0 \\ x_{k+1} = x_k - \alpha_k \cdot p_k \qquad ; \quad \alpha_k = \dfrac{(Ap_k, Hr_k)}{(p_k, Gp_k)} \\ r_{k+1} = r_k - \alpha_k A p_k \\ p_{k+1} = r_{k+1} + \displaystyle\sum_{i=\max(0,k-m+1)}^{k} \gamma_{k+1}^{(i)} \cdot p_i \quad \text{où} \\ \qquad\qquad \gamma_{k+1}^{(i)} = - \dfrac{(p_i, Gr_{k+1})}{(p_i, Gp_i)} \end{array} \right.$$

Remarque :

Si $H = I$, l'algorithme (III.4) n'est autre que la méthode Orthomin (m) [23].

Nous allons étudier la convergence de l'algorithme (II.4) et pour cela on supposera dans toute la suite que la partie symétrique de HA est définie positive

propriété 7 :

1°) $(p_i, G p_j) = 0$

2°) $(A p_i, H r_j) = 0$, $i = j-m, \dots, j-1$.

3°) $(A p_i, H r_i) = (A r_i, H r_i)$

4°) $(H r_i, A p_j) = (H r_{i-m}, A p_j)$; $i = j-m, \dots, j$; $j \geq m$

5°) $(r_i, G p_i) = (p_i, G p_i)$.

6°) $(H r_j, A r_i) = 0$; $i = j-m, \dots, j-1$

7°) $r_i \neq 0 \Longrightarrow p_i \neq 0 \quad \forall i$.

dem : Analogue à celles des prop 5 et 6 □

propriété 8 :

soit (x_k) la suite générée par (III.4), alors x_{k+1} réalise le minimum de $e(x)$ sur l'espace affine :

$$x_{k-m} + \mathrm{sev}\{p_{k-m}, \dots, p_k\}$$

dem :

Analogue à celle du théorème 2.

Nous avons enfin le résultat de convergence suivant :

Theorème 4 :

L'algorithme (III.4) est convergent :

$$\lim_{k\to\infty} r_k = 0$$

dem :

On pose $e_k = e(x_k)$

on a : $e_{k+1} = (r_{k+1}, Hr_{k+1})$

$$= (r_k - \alpha_k Ap_k, Hr_k - \alpha_k HAp_k)$$

donc : $e_{k+1} = (r_k, Hr_k) - 2\alpha_k (Ap_k, Hr_k) + \alpha_k^2 (Ap_k, HAp_k)$

avec :

$$\alpha_k = \frac{(Hr_k, Ap_k)}{(p_k, Gp_k)} \quad \text{et} \quad (Ap_k, HAp_k) = (p_k, Gp_k)$$

en remplaçant dans l'expression de e_{k+1}, on obtient :

$$e_{k+1} = e_k - \frac{(Ap_k, Hr_k)^2}{(p_k, Gp_k)}$$

* s'il existe k tel que $e_{k+1} = e_k$, alors :

$$(Ap_k, Hr_k) = 0$$

or $(Ap_k, Hr_k) = (Ar_k, Hr_k) = (r_k, HAr_k)$

donc si $e_{k+1} = e_k$ alors $(r_k, HAr_k) = 0$

$\Rightarrow r_k = 0$ puisque la partie symétrique de HA est définie positive.

* s'il n'existe pas de $k \in \mathbb{N}$ tel que $e_{k+1} = e_k$, alors la suite (e_k) est décroissante minorée par zéro, donc converge vers une limite finie, d'où :

$$\lim_{k \to \infty} (e_{k+1} - e_k) = \lim_{k \to \infty} \frac{(Ap_k, Hr_k)^2}{(p_k, Gp_k)} = 0$$

donc :

$\forall \varepsilon > 0$, $\exists k_0 \in \mathbb{N}$ tel que :

$$k > k_0 \Rightarrow (Ap_k, Hr_k)^2 \leq \varepsilon . (p_k, Gp_k)$$

or :

$$(Ap_k, Hr_k) = (r_k, HAr_k) \geq \lambda_{min}(HA) . \|r_k\|^2$$

et $(p_k, Gp_k) \leq (r_k, Gr_k) \leq \lambda_{max}(G) . \|r_k\|^2$

d'où :

$$\|r_k\|^2 \leq \varepsilon . \frac{\lambda_{max}(G)}{\lambda^2_{min}(HA)}$$

donc :

$$\lim_{k \to \infty} r_k = 0 . \quad \square$$

IV. Méthodes de projection oblique :

V.1 Présentation :

Soit à résoudre le système régulier d'ordre N.

$$(IV.1) \quad A.x = b$$

Soit x_k la solution approchée à l'étape k et $r_k = Ax_k - b$ le résidu correspondant.

On choisit :

* $q_k \in \mathbb{N}$
* L_{q_k} et K_{q_k} deux sous-espaces de $\mathbb{R}^N$ de dimension q_k.

Soit : $V_k = \{v_1^{(k)}, \dots, v_{q_k}^{(k)}\}$ une base de K_{q_k}

$W_k = \{w_1^{(k)}, \dots, w_{q_k}^{(k)}\}$ une base de L_{q_k}.

nous noterons R_{q_k} l'image par l'opérateur A de K_{q_k}.

alors on définit r_{k+1} par :

r_{k+1} est la projection oblique de r_k sur $L_{q_k}^{\perp}$, parallèlement à R_{q_k}.

($L_{q_k}^{\perp}$ désigne l'orthogonal de L_{q_k}).

donc : $r_{k+1} - r_k \in R_{q_k}$

$r_{k+1} \in L_{q_k}^{\perp}$

Soit: $\quad r_{k+1} - r_k = A.z_k \quad$ avec $z_k \in K_{q_k}$

et

$$r_{k+1} = r_k + Az_k \in L_{q_k}^{\perp}$$

d'où :

$$\text{avec:} \quad \begin{cases} x_{k+1} = x_k + z_k \\ z_k \in K_{q_k} \quad \text{et} \quad r_k + Az_k \in L_{q_k}^{\perp} \end{cases}$$

Si on note V_k et W_k les matrices de type (N, q_k) dont les colonnes sont formées respectivement par: $\{v_1^{(k)}, \dots, v_{q_k}^{(k)}\}$ et $\{w_1^{(k)}, \dots, w_{q_k}^{(k)}\}$, alors les relations précédentes se traduisent par:

$$z_k \in K_{q_k} \Rightarrow z_k = V_k . y_k \quad \text{avec } y_k \in \mathbb{R}^N$$

$$r_k + Az_k \in L_{q_k}^{\perp} \Rightarrow$$

$$W_k^T (Az_k + r_k) = 0$$

donc:

$$W_k^T (A V_k y_k + r_k) = 0 \Rightarrow (W_k^T A V_k) y_k = - W_k^T r_k .$$

pour que y_k existe et soit unique, nous supposerons que $\det(W_k^T A V_k) \neq 0$. D'où

$$y_k = - (W_k^T A V_k)^{-1} W_k^T r_k$$

nous obtenons alors l'algorithme :

$$\left\| \begin{array}{l} x_0 \text{ donné ; } q_k, V_k \text{ et } W_k \text{ choisis} \\ x_{k+1} = x_k - V_k \left(W_k^T A V_k\right)^{-1} W_k^T r_k. \end{array} \right.$$

Nous distinguerons après, deux classes de méthodes suivant le choix de q_k. Donnons tout d'abord un algorithme de calcul basé sur le RPA, pour chaque étape k.

<u>Algorithme de calcul : RPA</u>

soit :
$$P_{q_k} = V_k \left(W_k^T A V_k\right)^{-1} W_k^T r_k$$
$$= \sum_{i=1}^{q_k} \alpha_i v_i^{(k)}$$

alors :
$$W_k^T A P_{q_k} = \left(W_k^T A V_k\right) \cdot \left(W_k^T A V_k\right)^{-1} W_k^T r_k$$
$$= W_k^T r_k$$

Le calcul de P_{q_k} revient donc à la résolution du problème d'interpolation généralisée suivant :

$$\begin{cases} P_{q_k} = \sum_{i=1}^{q_k} \alpha_i \cdot v_i^{(k)} \\ \left(A^T w_i^{(k)}, P_{q_k}\right) = \left(w_i^{(k)}, r_k\right) \quad ; i = 1, \dots, q_k. \end{cases}$$

En utilisant le RIA, nous obtenons l'algorithme récursif :

$$
\begin{cases}
P_0 = 0 \quad ; \quad g_{0,i}^{(k)} = v_i^{(k)} \quad ; i = 1, \dots, q_k \\[2ex]
P_j = P_{j-1} + \dfrac{(w_j^{(k)}, r_k) - (A^T w_j^{(k)}, P_{j-2})}{(A^T w_j^{(k)}, g_{j-2,j}^{(k)})} \cdot g_{j-2,j}^{(k)} \quad ; j = 1, \dots, q_k \\[2ex]
g_{i,j}^{(k)} = g_{i-1,j}^{(k)} - \dfrac{(A^T w_j^{(k)}, g_{j-2,i}^{(k)})}{(A^T w_j^{(k)}, g_{j-2,j}^{(k)})} \times g_{j-1,j}^{(k)} \quad ; i > j > 1 .
\end{cases}
\tag{3}
$$

Le calcul de P_{q_k} nécessite : Nq_k^2 multiplications et Nq_k^2 additions plus le calcul des vecteurs $A^T w_j^{(k)}$; $j = 1, \dots, q_k$.

Lorsque la solution exacte du système (IV.1) est obtenue en un seul itéré ($x_{k+1} = x^*$), la méthode est dite à convergence finie. Sinon la méthode est dite incomplète.

Si on choisit K_{q_k} de telle sorte que $r_k \in R_{q_k}$ alors $r_{k+1} = 0$.

Pour obtenir une méthode à convergence finie, il suffit de prendre $q_k = k$. Dans ce cas, lorsque $k = N$ alors $L_N = \mathbb{R}^N$ et $L_N^{\perp} = \{0\}$, et puisque r_{N+1} est la projection oblique de r_N sur $L_N^{\perp}$ alors $r_{N+1} = 0$.

Pour cette classe de méthodes, Saad a donné dans [23] des algorithmes basés sur l'algorithme de biorthogonalisation de Lanczos.

Nous nous interesserons par la suite uniquement aux méthodes à convergence finie.

IV.3 _ Méthodes à convergence finie:

Nous prenons $q_k = k$. Alors :

$$x_{k+1} = x_k - V_k \left(W_k^T A V_k \right)^{-1} W_k^T r_k .$$

Et d'après la formule de Schur [8], on obtient :

$$x_{k+1} = \frac{\begin{vmatrix} x_k & \overset{(k)}{v_1} & \cdots & \overset{(k)}{v_k} \\ W_k^T r_k & & W_k^T A V_k & \end{vmatrix}}{\left| W_k^T A V_k \right|}$$

Soit encore :

$$x_{k+1} = \frac{\begin{vmatrix} x_k & \overset{(k)}{v_1} & \cdots & \overset{(k)}{v_k} \\ (\overset{(k)}{w_1}, r_k) & (A^T \overset{(k)}{w_1}, \overset{(k)}{v_1}) & \cdots & (A^T \overset{(k)}{w_1}, \overset{(k)}{v_k}) \\ \cdots & \cdots & \cdots & \cdots \\ (\overset{(k)}{w_k}, r_k) & (A^T \overset{(k)}{w_k}, \overset{(k)}{v_1}) & \cdots & (A^T \overset{(k)}{w_k}, \overset{(k)}{v_k}) \end{vmatrix}}{\begin{vmatrix} (A^T \overset{(k)}{w_1}, \overset{(k)}{v_1}) & \cdots & (A^T \overset{(k)}{w_1}, \overset{(k)}{v_k}) \\ \cdots & \cdots & \cdots \\ (A^T \overset{(k)}{w_k}, \overset{(k)}{v_1}) & \cdots & (A^T \overset{(k)}{w_k}, \overset{(k)}{v_k}) \end{vmatrix}}$$

Cette forme de x_{k+1}, nous permettra de montrer que la MMPE [26], la MPE [27], la RRE [27], l'ε-algorithme topologique [3], la seconde transformation composite vectorielle [11] et la transformation de GERMAIN-BONNE [17] sont des méthodes de projection obliques sur des sous-espaces de Krylov, pour la résolution du système (IV.1).

Rques:

- Cette idée est dûe à BREZINSKI qui a montré ce résultat pour la MPE, la RRE et l'ε-algorithme topologique, [1].
- Cette étude a été reprise dans un récent article [27] par Sidi.

Nous utiliserons des démonstrations différentes des précédentes.

Théorème 5:

La MMPE, la MPE et la RRE sont des méthodes de projection obliques sur des sous-espaces de Krylov.

dem:

Soit $(S_n)_n$ la suite de $\mathbb{R}^p$ définie par:

$$S_{n+1} = BS_n + b \qquad \text{où} \quad B = I - A.$$

Soit $n \geq 1$ fixé. Posons:

$$x_k = S_n \quad \text{et} \quad v_i = \Delta S_{n+i-1} \quad ; i = 1, \dots, k$$

On supposera que les systèmes: $\{v_1, \dots, v_k\}$ et $\{w_1, \dots, w_k\}$ sont libres.

Calculons les quantités :

$$(A^T w_i, v_j) \text{ et } (w_i, r_k) \quad ; i = 1, ---, k$$

qui figurent dans l'expression (IV.4).

$$* \ (A^T w_i, v_j) = (w_i, A v_j) \quad ; i, j = 1, ---, k$$

or :

$$A v_j = A . \Delta S_{n+j-2}$$

et :

$$\Delta S_{n+j-1} = -A \Delta S_{n+j-1} + b$$

donc : $A . \Delta S_{n+j-1} = - \Delta^2 S_{n+j-2}$.

d'où :

$$(A^T w_i, v_j) = -(w_i, \Delta^2 S_{n+j-1}) \quad ; i, j = 1, ---, k.$$

D'autre part :

$$x_k = S_n \Rightarrow r_k = A S_n - b$$

$$= (I - B) S_n - b = -\Delta S_n.$$

donc : $(w_i, r_k) = -(w_i, \Delta S_n) \quad ; i = 1, ---, k.$

En reportant dans (IV.4), nous obtenons :

$$x_{k+1} = \begin{vmatrix} S_n & \Delta S_n & --- & \Delta S_{n+k-2} \\ (w_1, \Delta S_n) & (w_1, \Delta^2 S_n) & --- & (w_1, \Delta^2 S_{n+k-1}) \\ --- & --- & --- & --- \\ (w_k, \Delta S_n) & (w_k, \Delta^2 S_n) & --- & (w_k, \Delta^2 S_{n+k-1}) \end{vmatrix} \Bigg/ \begin{vmatrix} (w_1, \Delta^2 S_n) & --- & (w_1, \Delta^2 S_{n+k-1}) \\ --- & --- & --- \\ (w_k, \Delta^2 S_n) & --- & (w_k, \Delta^2 S_{n+k-1}) \end{vmatrix}$$

Ce n'est autre que l'expression obtenue par application de la MMPE à la suite (S_n).

De plus :

$$v_1 = \Delta S_n$$

et $\Delta S_{n+i} = B\,\Delta S_{n+i-1} = \cdots = B^i.\Delta S_n$

donc : $v_i = B^{i-1}.v_1 \Longrightarrow K_k = \mathrm{Sev}\left\{v_1, Bv_1, \ldots, B^{k-1}v_1\right\}$

En prenant :

$$W_k = V_k = \left\{v_1, Bv_1, \ldots, B^{k-1}v_1\right\}, \text{ on obtient la MPE.}$$

pour obtenir la RRB, on prends toujours : $x_k = S_n$; $v_i = \Delta S_{n+i-1}$ et $w_i = \Delta S_{n+i}$; $i = 1, \ldots, k$.

dans ce cas :

$$V_k = \left\{v_1, Bv_1, \ldots, B^{k-1}v_1\right\}$$

$$W_k = \left\{w_1, Bw_1, \ldots, B^{k-1}w_1\right\}$$

Dans les trois cas précédents L_k et K_k sont des sous-espaces de Krylov.

<u>Propriété 9</u> :

Si B est inversible, alors $\left\{\Delta S_{n+i-1}\right\}_{i=1,\ldots,d}$ est un système libre où d est le degré du polynôme minimal de B par rapport à ΔS_1.

dem:

Soit d le degré du polynôme minimal de B par rapport à ΔS_1. Supposons que $\{\Delta S_{n+i-1}\}_{i=1,\dots,d}$ est lié. Alors il existent d scalaires non tous nuls tel que:

$$\sum_{i=1}^{d} b_i \, \Delta S_{n+i-1} = 0 \Rightarrow \sum_{i=0}^{d-1} b_{i+1} \cdot \Delta S_{n+i} = 0$$

Or:

$$\Delta S_{n+i} = B^{n+i-1} \cdot \Delta S_1$$

donc: $B^{n-1} \cdot \sum_{i=0}^{d-1} b_{i+1} B^i \Delta S_1 = 0$

et comme B est inversible, on a:

$$\sum_{i=0}^{d-1} b_{i+1} B^i \Delta S_1 = 0$$

Les $\{b_{im}\}_{i=0,\dots,d-1}$ ne sont pas tous nuls, ce qui est en contradiction avec le faite que d est le degré du polynôme minimal de B par rapport à ΔS_1 □

On montre de la même manière que $\{\Delta S_{n+i}\}_{i=1,\dots,d}$ est libre.

Théorème 6:

L'ε-algorithme topologique, la seconde transformation composite vectorielle et la transformation de GERMAIN-BONNE sont des méthodes de projection oblique.

dem:

soit $n \geq 1$ fixé.

) posons $x_k = S_n$ où (S_n) est definie par $S_{n+1} = BS_n + b$

prenons: $v_i = \Delta S_{n+i-1}$; $i = 1, \dots, k$

$w_i = (B^i)^T . y$ avec $y \in \mathbb{R}^N$; $i = 1, \dots, k$.

On a:

$$* \quad (A^T w_i, v_j) = (w_i, A v_j)$$

Or nous avons déjà vu (theorème 5) que:

$$A \Delta S_{n+i-1} = - \Delta^2 S_{n+i-1} \quad \text{et } r_k = - \Delta S_n .$$

donc:

$$(w_i, A v_j) = - (y, B^j \Delta^2 S_{n+i-1})$$

et comme:

$$B^j \Delta^2 S_{n+i-1} = \Delta^2 S_{n+i+j-1} \quad , \text{ alors:}$$

$$(A^T w_j, v_i) = - (y, \Delta^2 S_{n+i+j-1}) \quad ; i, j = 1, \dots, k.$$

d'autre part:

$$* \quad (w_j, r_k) = - (w_j, \Delta S_n)$$

$$= - (y, \Delta S_{n+j-1}) \qquad j = 1, \dots, k.$$

En reportant dans l'expression (IV.4) et en simplifiant par $(-1)^k$ on obtient : $x_{k+1} = \mathcal{E}_{2k}^{(n)}$.

2°) pour retrouver la famille de transformations proposée par GERMAIN-BONNE dans [17, page 68], on prends :

$$v_i = \Delta^q S_{n+i-1} \quad ; q \in \mathbb{N}^* \quad ; i = 1, \dots, k$$

$$w_j = \Delta S_{n+j-1} \quad ; j = 1, \dots, k$$

et on montre comme précédement que si A est symetrique :

$$(A^T w_j, v_i) = -\left(\Delta^2 S_{n+j-1}, \Delta^q S_{n+i-1}\right)$$

et

$$(w_j, z_k) = -\left(\Delta S_{n+j-1}, \Delta S_n\right)$$

dans ce cas : $V_k = \{v_1, Bv_1, \dots, B^{k-1} v_1\}$

$W_k = \{w_1, Bw_1, \dots, B^{k-1} w_1\}$.

3°) posons :

$$z_k = t_1^{(n)} \quad \text{et} \quad v_i = t_2^{(n+i)} - t_1^{(n+i)} \quad ; i = 1, \dots, k$$

où t_1 et t_2 sont deux transformations de suites vectorielles tels que :

$$t_1^{(n+i)} = B t_1^{(n+i-1)} + b$$
$$t_2^{(n+i)} = B t_2^{(n+i-1)} + b \qquad i = 1, \dots, k$$

donc :

$$t_2^{(n+i)} - t_1^{(n+i)} = B.\left(t_2^{(n+i-1)} - t_1^{(n+i-1)}\right)$$

alors :

$$A v_i = A.\left(t_2^{(n+i-1)} - t_1^{(n+i-1)}\right) \quad \text{où } A = I - B$$
$$= \left(t_2^{(n+i-1)} - B t_2^{(n+i-1)} - b\right) - \left(t_1^{(n+i-1)} - B t_1^{(n+i-1)} - b\right)$$
$$= -\left(\Delta t_2^{(n+i-1)} - \Delta t_1^{(n+i-1)}\right) \quad ; i = 1, \dots, k.$$

d'où : $(A^T w_j, v_i) = (w_j, A v_i)$

$$= -\left(w_j, \Delta t_2^{(n+i-1)} - \Delta t_1^{(n+i-1)}\right) \; ; \; i,j = 1, \dots, k$$

d'autre part :

$$r_k = A t_1^{(n)} - b$$

$$= - \Delta t_1^{(n)}$$

$$\Longrightarrow \quad (w_j, r_k) = -\left(w_j, \Delta t_1^{(n)}\right) \quad , j = 1, \dots, k.$$

si $k = N$ et si $\{w_j\}_{j=1,\dots,N}$ est la base canononique de $\mathbb{R}^N$ alors (V.4) n'est autre que l'expression donnée dans [11] par la seconde transformation composite vectorielle. De plus :

$$V_N = \left\{ v_1, B v_1, \dots, B^{N-1} v_1 \right\} . \square$$

- REFERENCES -

[1] J. BEUNEU :

Méthodes de projection à convergence finie, remarques sur leur forme incomplète, ANO 80 (1982), Université de Lille I.

[2] J. BEUNEU :

Some projection methods for non linear problems. ANO 134 (1984), Université de Lille I.

[3] C. BREZINSKI :

Accélération de la convergence en analyse numérique, LNM 584, Springer Verlag, Heidelberg, 1977.

[4] C. BREZINSKI :

A general extrapolation algorithm, Num. Math., 35 (1980), 175-187.

[5] C. BREZINSKI :

Recursive interpolation, extrapolation and projection, J. Comp. Appl. Math., 9 (1983), 369-376.

[6] C. BREZINSKI :

Algebric properties of the E-transformation, in "Numerical methods", Banach center publications, à paraître.

[7] C. BREZINSKI :

Convergence acceleration methods: the past decade. J. Comp. Appl. Math., 12 et 13 (1985), pp. 19-36

[8] C. BREZINSKI:

Other manifestations of the schur Complement. A paraître.

[9] C. BREZINSKI :

Bordering methods and progressive forms for sequence transformations. Zastos. Math., à paraître.

[10] C. BREZINSKI:

Some determinantal identities in a vector space, with applications. In "Padé approximation and its applications", H. Werner and H.J. Bünger eds., LNM 1071, Springer-Verlag, Heidelberg, 1984.

[11] C. BREZINSKI, H. SADOK :

Vector sequence transformations and fixed point methods. In "Numerical methods in laminar and turbulent flows", C. Taylor et al. eds, Prineridge Press, Swansea, 1987.

[12] S. CABY, L.W. JACKSON :

A polynomial extrapolation method for finding limits and antilimits of vector sequences. SIAM J. Num. Anal. 13 (1976), pp 734-752

[13] C. ESPINOZA :

Contribution à la résolution numérique de certains systèmes, Thèse de 3e cycle (1977), Université de Grenoble

[14] D.K. FADDEEV, V.N. FADDEEVA :

Computational methods of linear algebra, W.H. Freeman, San Francisco, 1963.

[15] W.F. FORD, A. SIDI :

Recursive Algorithms for vector extrapolation methods, à paraître.

[16] N. GASTINEL :

Sondécomposition de normes générales et procédés itératifs. Num. Math., 5 (1963), pp. 142-147.

[17] B. GERMAIN-BONNE :

Estimation de la limite de suites et formalismes de procédés d'accélération de convergence, Thèse Université de Lille I, 1978.

[18] A.H. HOUSEHLDER, F.L. BAUER :

On certain iterative methods for solving linear systems, Num. Math., 2 (1960), pp. 55-59.

[19] M. HESTENES, E. STIEFEL :

Methods of conjugate gradients for solving linear systems, J. Rest. Nat. Bur. Standards, 49 (1952), pp. 409-436

[20] P. HENRICI :

Elements of numerical analysis.

John wiely, New-York, 1964.

[21] A. LEMBARKI :

Méthodes de projection et extentions : étude théorique et pratique, Thèse de 3ème cycle (1984), Université de Lille I.

[22] B.P. PUGACHEV :

Acceleration of the convergence of iterative processes and a method of solving systems of non-linear équations, U.S.S.R. Comput. Maths. phys., 17 (1978), pp. 199-207.

[23] Y. SAAD :

The Lanczos biorthogonalisation algorithm and other oblique projection methods for solving large unsymetric systems, SIAM. J. Num. Math., 19 (1982), pp. 485-506.

[24] H. SADOK :

Accélération de la convergence de suites vectorielles et méthodes de point fixe, Thèse, université de Lille I, à paraître.

[25] F. SLOBODA :

A parallel projection method for linear algebrique systems, aplikace matematiky, 23 (1978), pp. 185-198.

[26] D.A. SMITH, W.F. FORD, A. SIDI:

Extrapolation methods for vector sequences, SIAM Review, 29 (1987), pp 199-233.

[27] A. SIDI, D.A. SMITH, W.F. FORD:

Acceleration of convergence of vector sequences, SIAM J. Num. Anal., 23 (1986), pp 197-209.

[27] A. SIDI:

Extrapolation vs projection methods for linear systems of equations, à paraître.

[28] P. WYNN:

On the convergence and stability of the epsilon algorithm, SIAM J. Num. Anal., 3 (1966), pp. 91-120.

[29] R.P. EDDY:

Extrapolating to the limit of a vector sequence, in Information linkage between applied mathematics and industry, P.C.C. Wang, eds., Academic Press, New-York, 1979, pp. 387-396