

THESIS / THÈSE

MASTER EN SCIENCES MATHÉMATIQUES

Résolution approchée d'équations elliptiques au moyen de la méthode multigrille

LAMBOTTE, Marie-Agnès

Award date:
1984

Awarding institution:
Universite de Namur

[Link to publication](#)

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal ?

Take down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

**RESOLUTION APPROCHÉE D'ÉQUATIONS
ELLIPTIQUES AU MOYEN DE
LA MÉTHODE MULTIGRILLE.**

MARIE-AGNES LAMBOTTE.

Il m'est agréable de présenter mes sincères remerciements à toutes les personnes qui m'ont permis de mener à bien la réalisation de ce mémoire.

Je remercie tout spécialement :

Monsieur VERGISON, chargé du centre d'informatique scientifique de la S.A. SOLVAY, qui m'a apporté son expérience, son aide et ses encouragements au cours de ce travail;

Monsieur le Professeur NGUYEN VAN HIEN pour la confiance qu'il m'a toujours accordée;

Monsieur le Professeur WEXLER pour ses conseils judicieux.

M. Lambotte

T A B L E D E S M A T I E R E S

	Page
1. INTRODUCTION	1
2. POSITION DU PROBLEME	5
2.1. Formulation fonctionnelle du problème	5
2.2. Formulation discrète du problème	11
2.3. Méthodes de résolution numériques d'un système linéaire associé à un problème elliptique	19
2.3.1. Méthodes directes	20
2.3.2. Méthodes itératives	29
3. METHODE MULTIGRILLE	37
3.1. Généralités	37
3.2. Méthode multigrille unidimensionnelle	58
3.3. Méthode multigrille bidimensionnelle	80
3.4. Applications numériques	94
3.4.1. Résolution de l'équation de Poisson	94
3.4.2. Résolution d'une équation elliptique non linéaire	100
4. CONCLUSIONS	
ANNEXES	
REFERENCES	

1. INTRODUCTION

Etant donné l'importance des modèles mathématiques gouvernés par des équations aux dérivées partielles en physique et en chimie (transferts thermiques, transferts de masse, problèmes de potentiel, mécanique des fluides,...), l'étude des techniques de résolution de tels systèmes s'est beaucoup développée au cours de ces dernières années.

L'approche numérique couplée à la puissance des ordinateurs est la plus couramment utilisée. Elle conduit, dans le cas de problèmes linéaires ou linéarisés, à la résolution de grands systèmes creux c'est-à-dire comportant un grand nombre de composantes nulles.

La méthode multigrille à laquelle nous nous intéressons dans ce travail est un outil récent de résolution de tels systèmes qui connaît un vif succès aujourd'hui.

Le chapitre 2 présente cette méthode en la situant dans le contexte des équations aux dérivées partielles sur un problème modèle : l'équation de Poisson.

Ce chapitre est divisé en trois parties :

- Dans la première, nous commençons par formuler analytiquement le problème à résoudre. En réalité, nous le présentons de trois manières différentes équivalentes donnant chacune un éclairage particulier sur la question : la formulation variationnelle, la formulation de Galerkin et la formulation différentielle classique. Cette dernière :

$$\left\{ \begin{array}{l} \text{trouver } u \text{ suffisamment régulière solution} \\ \text{de :} \\ \left\{ \begin{array}{l} - \Delta u(x,y) = f(x,y) , \forall (x,y) \in \Omega = (0,1)^2 \\ u(x,y) = g(x,y) , \forall (x,y) \in \Gamma = \partial\Omega \end{array} \right. \\ f \text{ et } g \text{ étant connues,} \end{array} \right.$$

a conduit à la méthode des différences finies. C'est de celle-ci dont il sera question dans la suite.

- Le paragraphe 2.2 est consacré à une présentation théorique de cette méthode numérique. Nous avons tenu à faire le lien entre le problème analytique et sa formulation discrète.

- La dernière partie de ce chapitre est consacrée à une analyse comparée de méthodes de résolution de systèmes linéaires en portant plus particulièrement notre attention sur les grands systèmes creux. Des programmes de calcul reprenant divers algorithmes de calcul sont repris en annexe.

Le chapitre 3 est entièrement consacré à la méthode multigrille :

- Dans le premier paragraphe, nous en décrivons les caractéristiques essentielles. Nous y abordons en particulier les problèmes de convergence et de l'effort de calcul que la méthode implique. Nous montrons ainsi qu'asymptotiquement, le nombre d'opérations requises est proportionnel au nombre de points de la grille de discrétisation (N) alors que les méthodes classiques en demandent N^2 ou $N \log N$.
- Dans la deuxième partie, nous analysons plus en détail les propriétés de la méthode multigrille sur un exemple unidimensionnel. Nous y quantifions en particulier les différents paramètres qui la caractérisent tels le rayon spectral de la matrice d'itération et le facteur de proportionnalité donnant le nombre de calculs à effectuer.
- Dans la troisième partie, nous étendons de façon naturelle les résultats au cas bidimensionnel.

- Enfin, nous consacrons le dernier paragraphe aux applications numériques. Nous reprenons d'abord un résultat classique [1],[11] pour l'équation de Poisson pour l'étendre ensuite au cas de la résolution d'un problème non linéaire [12] .

Dans ce travail, nous avons surtout voulu mettre l'accent sur le fait que la méthode multigrille repose sur des bases saines et est bien adaptée pour résoudre des systèmes linéaires dérivant d'équations aux dérivées partielles.

Des essais comparatifs entre méthodes et sur des cas d'applications plus complexes devront le compléter. Mais dès à présent, en nous appuyant sur les résultats des essais effectués par Trottenberg et Stüben [10],[11], on peut se montrer optimistes quant à l'avenir de la méthode multigrille.

** **

2. POSITION DU PROBLEME

2.1 FORMULATION FONCTIONNELLE DU PROBLEME

Beaucoup de problèmes de physique mathématique peuvent être décrits mathématiquement par une ou plusieurs équations fonctionnelles que nous écrivons sous la forme simplifiée :

$$Lu = f$$

où L opère d'un espace X dans un espace Y

f est donné dans Y

u est cherché dans X

Dans ce travail, nous étudierons des modèles gouvernés par des équations aux dérivées partielles de type elliptique avec conditions aux limites de Dirichlet :

$$\begin{cases} L^{\Omega} u(x) = f^{\Omega} & x \in \Omega \\ L^{\Gamma} u(x) = f^{\Gamma} & x \in \Gamma = \partial \Omega \end{cases} \quad (2.1.)$$

où $x = (x_1, \dots, x_d) \in \mathbb{R}^d$

$\Omega \subset \mathbb{R}^d$ est un domaine ouvert borné de frontière Γ

L^{Ω} est un opérateur différentiel elliptique linéaire

L^{Γ} est un opérateur de frontière linéaire

De tels problèmes peuvent parfois être formulés sous forme variationnelle :

$$\begin{cases} J(u) = \inf_{v \in V} J(v) \\ u \in V \end{cases} \quad (2.2)$$

où $J(v) := \frac{1}{2} a(v,v) - f(v)$

et où $a(\cdot, \cdot)$ est une forme bilinéaire symétrique sur $V \times V$

$f(\cdot)$ est une forme linéaire sur V

V est une classe de fonctions représentant l'ensemble des états admissibles.

Afin de rendre l'exposé plus clair, nous analyserons le cas particulier de l'équation de Poisson sur le carré unité.

L'espace V associé au problème est l'espace de SOBOLEV des fonctions de carré sommable dont les dérivées généralisées sont de carré sommable et nulles sur le bord du domaine Ω :

$$V = H_0^1(\Omega) = \left\{ v \in L^2(\Omega), \frac{\partial v}{\partial x_i} \in L^2(\Omega), i=1,2; v|_{\Gamma} = 0 \right\}$$

Le problème (2.1) se réécrit alors : Trouver $u \in H_0^1(\Omega)$ tel que

$$\begin{cases} L^{\Omega} u(x) := -\Delta u(x) = f^{\Omega}(x) & x \in \Omega = (0,1)^2 \\ L^{\Gamma} u(x) := u(x) = 0 & x \in \Gamma \end{cases} \quad (2.3)$$

c'est-à-dire où L^{Ω} est l'opérateur de Laplace

et L^{Γ} est la restriction de la fonction u sur la frontière.

La formulation variationnelle correspondante s'écrit

$$\begin{cases} J(u) = \inf_{v \in H_0^1(\Omega)} J(v) \\ u \in H_0^1(\Omega) \end{cases} \quad (2.4)$$

où $J(v) := \frac{1}{2} \iint_{\Omega} |\nabla v|^2 dx - \iint_{\Omega} f v dx$

On démontre alors la proposition suivante

Proposition 2.1.

u est solution du problème (2.3)
ssi.
 u est solution du problème d'optimisation (2.4)

La démonstration de cette proposition importante a été reprise en annexe.

Nous nous sommes limités à l'étude de l'équation de Poisson en montrant qu'elle dérivait d'un problème variationnel. Nous avons formalisé ce problème dans un cadre plus général (espaces fonctionnels). Des résultats utilisant ce formalisme ont été obtenus pour des systèmes d'équations elliptiques plus généraux [6]

Pour un même problème nous avons obtenu trois formulations équivalentes :

$$\begin{array}{c}
 \text{minimiser } J(v) \text{ sur } V \\
 \Updownarrow \\
 \text{résoudre } a(u,v) = L(v) \quad \forall v \in V \\
 \Updownarrow \\
 \text{résoudre } -\Delta u = f
 \end{array}$$

Nous allons reprendre ces différentes formulations et donner une idée des méthodes de résolution auxquelles elles conduisent.

2.1.2 Résoudre $a(u,v) = L(v) \quad \forall v \in V$

Comme dans le cas précédent, le problème est discrétisé sur un espace V_n . Nous devons résoudre

$$a(u,v) = L(v) \quad \forall v \in V_n$$

ou encore si $\{v_1, \dots, v_n\}$ est une base de l'espace V_n

$$a(u, v_j) = L(v_j) \quad \forall j = 1, \dots, n$$

en remplaçant u par $\sum_{i=1}^n \alpha_i v_i$ il vient

$$a\left(\sum_{i=1}^n \alpha_i v_i, v_j\right) = L(v_j) \quad \forall j = 1, \dots, n.$$

de plus, a étant bilinéaire, nous obtenons

$$\sum_{i=1}^n \alpha_i a(v_i, v_j) = L(v_j) \quad \forall j = 1, \dots, n$$

comme $a(\dots)$ et $L(\dots)$ sont connus, il s'agit de résoudre un système linéaire où les α_i , $i = 1, \dots, n$ sont les inconnues.

Pour appliquer un tel schéma de résolution, il faudra résoudre des problèmes de :

- choix de l'espace V_n
- existence et unicité d'une solution u_n du problème approché
- convergence de u_n solution du problème approché vers la solution du problème initial.

2.1.3 Résoudre $Lu = f$

C'est par le biais de cette troisième formulation que nous allons aborder la résolution du problème.

Nous allons discrétiser l'équation $Lu = f$

en $L_h u_h = f_h$. Cette discrétisation ainsi que les problèmes qu'elle soulève sont étudiés en détail dans la suite.

Nous sommes maintenant en mesure de dresser un tableau mettant en parallèle formulations continues et formulations discrètes du problème :

Problème continu

- Min $J(v)$, $v \in V$

- Résoudre

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

- Résoudre

$$L u = f$$

Problème discrétisé

- Min $J^*(\alpha_1, \dots, \alpha_n)$, $\alpha_j \in \mathbb{R}$

- Résoudre le système linéaire

$$\sum_{i=1}^n \alpha_i (v_i, v_j) = L(v_j), \quad j = 1, \dots, n$$

- Résoudre

$$L_h u_h = f_h$$

Afin de présenter un travail complet, nous allons décrire le lien qui existe entre la formulation fonctionnelle de Galerkin $a(u, v) = L(v)$ et la formulation classique des différences finies. Cela fera l'objet du paragraphe suivant.

2.2 FORMULATION DISCRETE DU PROBLEME

On étudie l'équation de Poisson :

$$\begin{cases} -\Delta u = f & x \in \Omega = (0,1) \times (0,1) \subset \mathbb{R}^2 \\ u = 0 & x \in \partial \Omega = \Gamma \end{cases} \quad (2.5.)$$

Posons Q_h = le réseau de points qui maillent le carré fermé.

Soit σ_M un parallélotope centré sur le point M et de côtés (h_1, h_2)

Soit $\sigma(M;1)$ la croix centrée sur M et de centre σ (voir figure ci-dessous)

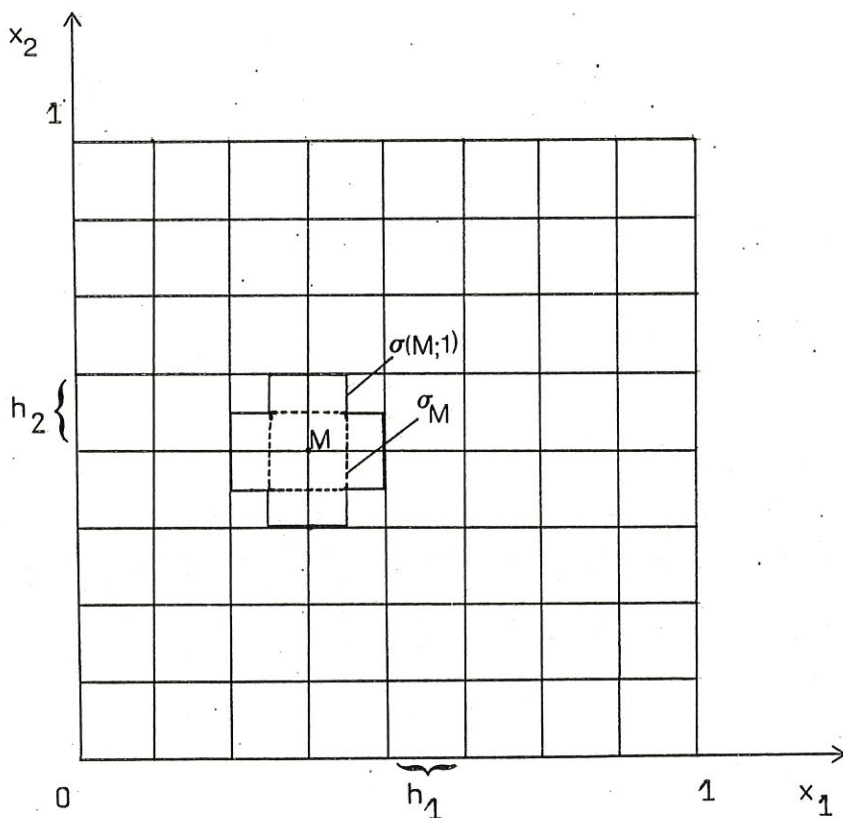


figure (2.1).

L'objectif de cette section est de répondre aux questions posées dans le paragraphe 2.1. c'est-à-dire définir concrètement l'espace V_h , montrer que la solution u_h du problème approché existe et est unique et enfin, s'assurer de la convergence de la solution u_h vers la solution u du problème initial.

Nous avons à résoudre le problème analytique : trouver une fonction $u \in H_0^1((0,1) \times (0,1))$ satisfaisant à :

$$\begin{aligned} a(u, v) &= \sum_{i=1}^2 \iint_{0,0}^{1,1} \frac{\partial u}{\partial x_i} \frac{\partial v}{\partial x_i} dx_1 dx_2 \\ &= \iint_{0,0}^{1,1} f \cdot v dx_1 dx_2 = L(v) \end{aligned} \quad (2.6)$$

pour tout $v \in H_0^1((0,1) \times (0,1))$

Discrétiser le problème revient à faire choix d'un espace finidimensionnel V_h et d'une approximation $a_h(u, v)$ de la forme bilinéaire et $L_h(v)$ de la forme linéaire.

Définition de l'espace V_h

Soit $\Omega_h = \{ M \in Q_h \mid \sigma(M; 1) \subset \Omega \}$

Soit w_M , la fonction caractéristique de σ_M (remarquons toutefois que $w_M \notin H_0^1(\Omega)$)

Définissons V_h comme l'espace engendré par les fonctions w_M .

L'espace V_h est une approximation externe de $H_0^1(\Omega)$: $V_h \not\subset H_0^1(\Omega)$

et ce, contrairement à la méthode des éléments finis où on choisit

$V_h \subset H_0^1(\Omega)$

Choix de la forme approchée de a : a_h

On approche la dérivée partielle $\frac{\partial}{\partial x_j}$ par un quotient différentiel δ_j :

$$\delta_j \varphi(x) = \frac{\varphi(x + h_j e_j / 2) - \varphi(x - h_j e_j / 2)}{h_j}$$

On définit alors a_h comme

$$a_h(u, v) = \sum_{i=1}^2 \int_0^1 \int_0^1 \delta_i u \delta_j v \, dx_1 \, dx_2$$

Le problème discret équivalent au problème (2.6) s'écrit :

$$\left\{ \begin{array}{l} \text{Trouver une fonction } u_h \in V_h \text{ telle que :} \\ a_h(u_h, v_h) = \int_0^1 \int_0^1 f_h v_h \, dx_1 \, dx_2 \\ \text{pour tout } v_h \in V_h \end{array} \right.$$

Remarque : f_h est une approximation de f , supposée continue.

Par exemple, en posant

$$\begin{aligned} f_h(M) &= \frac{1}{h_1 h_2} \int_{M - \frac{h_1}{2}}^{M + \frac{h_1}{2}} \int_{M - \frac{h_2}{2}}^{M + \frac{h_2}{2}} f(x_1, x_2) \, dx_1 \, dx_2 \\ &= \frac{1}{h_1 h_2} \iint_{\sigma_M} f(x_1, x_2) \, dx_1 \, dx_2 \end{aligned}$$

il vient :

$$f_h(x) = \sum_M f_h(M) w_M(x)$$

En vertu des hypothèses que nous avons faites, en particulier sur Ω_h (on a choisi des croix donc on est loin du bord), on peut montrer aisément que la formule suivante d'intégration par partie est valable :

$$\int_0^1 \int_0^1 \sum_{i=1}^2 \delta_i u_h \delta_i w_M \, dx_1 \, dx_2 = - \int_0^1 \int_0^1 \sum_{i=1}^2 \delta_i^2 u_h w_M \, dx_1 \, dx_2$$

$$\begin{aligned}
\text{En effet : } & \int_0^1 \int_0^1 \sum_{i=1}^2 \delta_i u_h \delta_i v_h \, dx_1 \, dx_2 \\
= & \int_0^1 \int_0^1 \frac{u_h(x_1 + \frac{h}{2}, x_2) - u_h(x_1 - \frac{h}{2}, x_2)}{h} \cdot \frac{v_h(x_1 + \frac{h}{2}, x_2) - v_h(x_1 - \frac{h}{2}, x_2)}{h} \, dx_1 \, dx_2 \\
= & \frac{1}{h^2} \int_0^1 \int_0^1 \frac{u_h(x_1, x_2) - u_h(x_1 - h, x_2)}{h} v_h(x_1, x_2) \, dx_1 \, dx_2 \\
- & \frac{1}{h^2} \int_0^1 \int_0^1 \frac{u_h(x_1 + h, x_2) - u_h(x_1, x_2)}{h} v_h(x_1, x_2) \, dx_1 \, dx_2 \\
= & - \int_0^1 \int_0^1 (\delta_1^2 u_h) v_h \, dx_1 \, dx_2
\end{aligned}$$

En procédant de façon analogue pour les dérivées en x_2 , il vient :

$$\int_0^1 \int_0^1 \left(- \sum_{i=1}^2 \delta_i^2 u_h \right) v_h \, dx_1 \, dx_2 = \int_0^1 \int_0^1 f_h v_h \, dx_1 \, dx_2$$

pour tout $v_h \in V_h$

D'où en particulier pour tout $w_M \in V_h$:

$$\int_0^1 \int_0^1 \left(- \sum_{i=1}^2 \delta_i^2 u_h \right) w_M \, dx_1 \, dx_2 = \int_0^1 \int_0^1 f_h w_M \, dx_1 \, dx_2$$

ou encore :

$$- \left(\sum_{i=1}^2 \delta_i^2 u_h(M) \right) h_1 h_2 = f_h(M) h_1 h_2$$

et l'on retrouve la formulation classique :

$$- \sum_{i=1}^2 \delta_i^2 u_h(M) = f_h(M) \quad M \in \Omega_h$$

ou encore si l'on réarrange les termes :

$$L_h u_h = f_h \quad M \in \Omega_h$$

Revenons au problème approché écrit sous la forme :

$$a_h(\tilde{u}_h, w_h) = \iint_{0^1 0^1} f_h w_h dx_1 dx_2$$

pour tout $w_M \in V_h$

La solution peut s'écrire :

$$u_h = \sum_{M \in \Omega_h} \alpha_M w_M$$

d'où

$$\begin{aligned} a_h(u_h, w_M) &= a_h\left(\sum_{M' \in \Omega_h} \alpha_{M'} w_{M'}, w_M\right) \\ &= \sum_{M' \in \Omega_h} \alpha_{M'} a_h(w_{M'}, w_M) \\ &= f_h(M) \end{aligned}$$

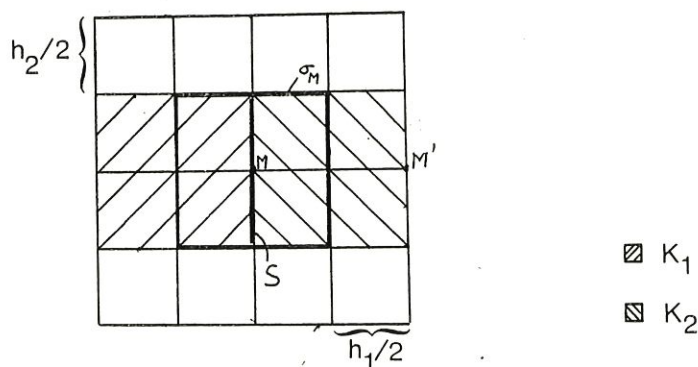
Calculons les éléments de la matrice de ce système linéaire :

Si $M = M'$:

$$a_h(w_M, w_M) = \sum_{i=1}^2 \iint_{0^1 0^1} (\delta_i w_M)^2 dx_1 dx_2$$

Par définition de δ_j (cfr figure)

$$\delta_j w_M(x) = \begin{cases} 1/h_j & \text{sur } K_1 \\ 0 & \text{sur } S \\ -1/h_j & \text{sur } K_2 \\ 0 & \text{partout ailleurs} \end{cases}$$



figure(2.2)

D'où, pour $i = 1, 2$ et $j \neq i$

$$\iint_{0 \cdot 0}^{1 \cdot 1} (\delta_i w_M)^2 dx_1 dx_2 = \frac{1}{h_i^2} \iint_{K_1 \cup K_2} dx_1 dx_2 = 2 \frac{h_j}{h_i}$$

et si $h_i = h_j$ on obtient :

$$a_h(w_M, w_M) = 4$$

Si $M \neq M'$ et si M' est le centre d'un rectangle adjaçant à celui

qui est centré sur M (voir par exemple le cas de la figure (2.2))

$$\begin{aligned} \iint_{0 \cdot 0}^{1 \cdot 1} \delta_i w_M \delta_i w_{M'} dx_1 dx_2 &= -\frac{1}{h_i^2} \iint_{K_2} dx_1 dx_2 \\ &= -\frac{h_1 h_2}{h_i^2} \end{aligned}$$

D'où, si $h_1 = h_2$

$$a_h(w_M, w_{M'}) = -1$$

Dans tous les autres cas :

$$\iint_{0 \cdot 0}^{1 \cdot 1} \delta_i w_M \delta_i w_{M'} dx_1 dx_2 = 0$$

Finalement, nous retrouvons encore la forme :

$$L_h u_h = f_h$$

En outre, on a la propriété de convergence :

Théorème 2.3. :

Quand $h \rightarrow 0$: la suite $\{u_h\}$ converge vers

u dans $L^2(\Omega)$

$$u_h \xrightarrow{L^2(\Omega)} u$$

$$\delta_i u_h \xrightarrow{L^2(\Omega)} \frac{\partial u}{\partial x_i}$$

Nous ne démontrerons pas ce théorème qui fait appel à des éléments non triviaux d'analyse fonctionnelle. [6]

2.3. METHODES DE RESOLUTION NUMERIQUES D'UN SYSTEME LINEAIRE ASSOCIE A UN PROBLEME ELLIPTIQUE

La discrétisation de l'équation de Poisson nous a conduit à résoudre un système linéaire :

$$L_h u_h = f_h$$

de grande taille, à structure creuse c'est-à-dire possédant un grand nombre de zéros.

Dans l'arsenal des méthodes de résolution de systèmes linéaires, on distingue deux grandes familles :

- les méthodes directes et
- les méthodes itératives.

Ces dernières sont depuis longtemps très utilisées pour résoudre des systèmes dérivant de modèles mathématiques gouvernés par des équations aux dérivées partielles. Elles ont le mérite de prendre en compte la structure creuse des matrices et de lisser les erreurs d'arrondi.

Les premières qui étaient évitées à cause des problèmes de stabilité numérique qu'elles impliquent ainsi que du problème du temps calcul (la méthode de Gauss conduit à $\frac{n^3}{3}$ opérations, n désignant la dimension de la matrice) connaissent un regain d'intérêt depuis que des stratégies adaptées à ce type de systèmes linéaires ont été développées. Dans le paragraphe suivant, nous exposons certaines de ces méthodes, la suite du travail étant consacré à une méthode itérative particulière, la méthode multigrille.

2.3.1 METHODES DIRECTES

Méthode de Gauss

Soit à résoudre le système linéaire :

$$A x = b \quad \text{où} \quad A \in \mathbb{R}^{n \times n}$$

$$x, b \in \mathbb{R}^{n \times 1}$$

L'idée de cette méthode est de décomposer la matrice A en un produit de deux matrices l'une triangulaire inférieure (L), l'autre triangulaire supérieure (U) et de ramener ainsi la résolution du système :

$$A x = L U x = b$$

à celle de deux systèmes linéaires simples :

- 1) élimination gaussienne : on pose $U x = y$ et on résout le système triangulaire

$$L y = b$$

- 2) substitution inverse : ayant obtenu y on résout ensuite le système :

$$U x = y$$

Ce qui revient à calculer la composante x_i du vecteur x à partir des composantes x_{i+1} , ..., x_n .

Nous allons analyser plus en détail la procédure d'élimination gaussienne et montrer qu'elle se ramène simplement à prémultiplier les deux membres du système $A x = b$ par des matrices convenablement choisies.

A partir de l'élément a_{11} de la matrice A , appelé pivot, on effectue les transformations élémentaires suivantes :

- remplacer la $k^{\text{ième}}$ ligne de A (notée A^k) par

$$A^k - a_{k1}(a_{11})^{-1}A^1 \quad \text{pour } k \neq 1$$

c'est-à-dire effectuer l'opération

$$a_{k1}^k = a_{k1}^{k-1} - \frac{a_{k1}^{k-1}}{a_{11}^{k-1}} a_{11}^{k-1} \quad \text{pour } k \neq 1$$

ce qui revient à prémultiplier la matrice A par une matrice de transformation triangulaire M_1 .

Cette opération a pour effet de construire une matrice M_1A dont les éléments sous diagonaux de la première colonne sont nuls

avec

$$M_1 = \begin{bmatrix} 1 & & & & \\ -m_{21} & \dots & & & 0 \\ \vdots & & \ddots & & \\ & & & 1 & \\ -m_{n1} & \dots & & & 1 \end{bmatrix}$$

$$\text{et } m_{j1} = a_{j1}/a_{11} \quad \text{pour } 1 < j \leq n$$

- dans le même temps, on multiplie le second membre (b) par la matrice M_1 .

On continue ainsi de proche en proche, ligne par ligne.

Ainsi, à la $i^{\text{ième}}$ ligne, à partir de l'élément a_{ii} choisi comme pivot, on effectue les transformations élémentaires suivantes :

- remplacer A^k , la $k^{\text{ième}}$ ligne de A par

$$A^k - (a_{ii})^{-1} a_{ki} A^i \quad \text{pour } k < i$$

ce qui revient à prémultiplier la matrice des coefficients

$$M_{i-1} \dots M_1 A \quad \text{par une matrice de transformation } M_i.$$

Les éléments sous-diagonaux de la $i^{\text{ième}}$ colonne de la matrice produit $M_i M_{i-1} \dots M_1 A$ sont nuls

avec

$$M_i = \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & -m_{i,i} & \ddots & \\ & & \vdots & \ddots & \\ & & -m_{ni} & & 1 \end{bmatrix} \quad \text{et } m_{ji} = a_{ji}/a_{ii} \text{ pour } i < j \leq n$$

- On effectue la même opération de prémultiplication sur le second membre $(M_{i-1} \dots M_1 b)$ du système.

Nous obtenons ainsi après la $n-1^{\text{ième}}$ étape :

$$M_{n-1} M_{n-2} \dots M_2 M_1 A x = M_{n-1} M_{n-2} \dots M_2 M_1 b$$

ou, en posant $M_{n-1} \dots M_2 M_1 = U$ et $M_{n-1} \dots M_2 M_1 b = y$,

$$U x = y$$

où U est une matrice triangulaire supérieure.

La matrice $L^{-1} = M_{n-1} \dots M_2 M_1$ a la forme :

$$L^{-1} = \begin{bmatrix} 1 & & & & \\ -m_{21} & \ddots & & & \\ \vdots & & 1 & & \\ -m_{i,i} & \ddots & \vdots & \ddots & \\ -m_{n1} & -m_{ni} & & & 1 \end{bmatrix}$$

En outre, dans ce qui précède, nous avons choisi comme pivots les éléments diagonaux des matrices obtenues dans la séquence d'élimination à commencer par l'élément a_{11} de la matrice A pour continuer par l'élément a_{22} de la matrice $M_1 A$, etc... Cette méthode de sélection de pivots présente toutefois de sérieux inconvénients. Considérons par exemple le système :

$$\begin{cases} x + 499 y = 333 \\ x + 1 y = 1 \end{cases}$$

Il a pour solution exacte :

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 1/3 \\ 2/3 \end{pmatrix}$$

Si nous effectuons un calcul approché à trois chiffres exacts, nous obtenons avec la méthode de Gauss par élimination simple:

$a_{11} = 1$, premier pivot

$$\left[\begin{array}{cc|c} 1 & 499 & 333 \\ 1 & 1 & 1 \end{array} \right]$$

avec

$$M_1 = \begin{pmatrix} 1 & 0 \\ -1 & 1 \end{pmatrix}$$

nous obtenons

$$\left[\begin{array}{cc|c} 1 & 499 & 333 \\ 1 & -498 & -332 \end{array} \right]$$

ce qui donne

$$y = -332/-498 \approx 0,67$$

$$x = 333 - 499 \cdot 0,667 \approx 0,167 !$$

Il est donc indispensable si on veut tirer parti des méthodes directes pour résoudre des systèmes linéaires issus d'équations aux dérivées partielles :

- 1- de disposer d'une stratégie de choix de pivot robuste
- 2- d'éviter au maximum d'effectuer des opérations que la présence de termes nuls rend inutiles.
- 3- d'éviter de stocker de l'information inutile

Nous allons passer rapidement en revue les éléments principaux d'une stratégie possible (KEY : [5]) pour résoudre un grand système creux (non nécessairement symétrique) par une méthode directe.

CHOIX DU PIVOT

1- Gauss par pivotage complet.

Cette technique permet d'éviter les problèmes de précision de la méthode de Gauss par élimination simple. Elle consiste à choisir pour pivots des éléments suffisamment grands. Pratiquement, on cherche, parmi tous les éléments de la matrice, le plus grand élément n'ayant jamais appartenu à une ligne ou à une colonne pivot.

Reprenons l'exemple précédent mais en choisissant comme pivot le plus grand élément de la matrice A c'est-à-dire 499

$$\left[\begin{array}{cc|c} 1 & 499 & 333 \\ 1 & 1 & 1 \end{array} \right]$$

nous obtenons après pivotage

$$\left[\begin{array}{cc|c} 1 & 499 & 333 \\ 0.998 & 0 & 0.333 \end{array} \right]$$

d'où

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 0.333 \\ 0.667 \end{pmatrix}$$

Cette technique plus précise que la précédente implique toutefois un coût relativement élevé.

2- Gauss par pivotage partiel

Cette technique à mi-chemin entre les méthodes de Gauss par élimination simple et Gauss par pivotage complet permet d'éviter les erreurs dues à un manque de précision sans atteindre un coût trop élevé.

Dans ce procédé, la colonne de pivotage est choisie comme dans la méthode de Gauss par élimination simple. Le pivot étant choisi comme le plus grand coefficient n'ayant jamais appartenu à une ligne pivot.

Il existe également des techniques de choix de pivot permettant de tirer un profit maximum des systèmes creux (KEY:[6])

En voici quelques unes.

3- Minimum ligne - Minimum colonne

Dans cette technique de sélection du pivot, la ligne pivot est celle comprenant le plus petit nombre d'éléments non nuls et n'ayant jamais été sélectionnée comme ligne pivot (dans le cas où plusieurs lignes satisfont cette condition, on choisit celle dont l'index de ligne est minimum).

La colonne pivot est déterminée en examinant toutes les colonnes correspondant aux éléments non nuls de la ligne pivot et en choisissant celle qui comprend un nombre minimum d'entrées (si plusieurs colonnes satisfont cette condition, on choisit celle dont l'index colonne est minimum).

4- Minimum colonne - Minimum ligne.

Cette technique est similaire à la précédente. On cherche d'abord la colonne comprenant le minimum d'entrées non nulles et qui n'a pas encore été sélectionnée. Ce sera la colonne pivot. Ensuite, les lignes contenant les éléments non nuls de la colonne pivot sont passées en revue pour déterminer celle comprenant le nombre minimum d'entrées.

5- Maximum colonne - Minimum ligne.

La colonne pivot est la colonne comprenant le maximum d'éléments non nuls et qui n'a pas encore été sélectionnée. La ligne pivot est déterminée de manière similaire au cas précédent.

6- Minimum du produit entre entrées colonne et entrées lignes.

Pour tout élément non nul de la matrice, on effectue le produit nombre d'entrées colonne non nulles \times nombre d'entrées ligne non nulles. L'élément correspondant au produit minimum est choisi comme pivot. Si plusieurs éléments ont le même produit, l'élément correspondant à l'indice ligne minimum est sélectionné. Si ces éléments sont dans la même ligne, celui correspondant à l'indice colonne minimum est sélectionné.

A ces différentes possibilités de choix de pivot, on associe une stratégie de stockage très simple :

Soit par exemple la matrice

$$B = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 1 \\ 0 & 2 & 4 & 0 & 0 \\ 1 & 0 & 0 & 2 & 0 \\ 0 & 0 & 2 & 0 & 3 \end{bmatrix}$$

On lui associe deux matrices, respectivement A et ICOL contenant l'une les éléments non nuls de B et l'autre l'indice de la colonne correspondant aux différents éléments.

Il vient :

$$A = \begin{bmatrix} 3 & 0 \\ 2 & 1 \\ 2 & 4 \\ 1 & 2 \\ 2 & 3 \end{bmatrix} \quad \text{et} \quad \text{ICOL} = \begin{bmatrix} 1 & 0 \\ 2 & 5 \\ 2 & 3 \\ 1 & 4 \\ 3 & 5 \end{bmatrix}$$

Ces différentes stratégies (choix du pivot, stockage des éléments non nuls de la matrice du système) sont combinées dans un algorithme de Gauss programmé de manière à éviter les opérations inutiles.

Le code du programme de calcul est repris en annexe.

2.3.2. METHODES ITERATIVES

Ces méthodes de résolution d'un système d'équations linéaires ont été souvent associées à des systèmes provenant de la discrétisation d'équations aux dérivées partielles. C'est dans cette optique que nous les envisagerons ici.

Ces méthodes ont l'avantage de tenir compte de manière naturelle de la structure creuse du système et de lisser naturellement les erreurs d'arrondi bien que demandant en principe une infinité d'itérations pour obtenir la solution cherchée.

Parmi les plus connues et les plus simples, on peut citer celles de Jacobi et de Gauss-Seidel.

Soit u_h^j , une approximation de la solution u_h de $L_h u_h = f_h$ sur Ω_h .

Toutes les méthodes considérées dans ce chapitre peuvent s'exprimer sous la forme :

$$u_h^{j+1} = G u_h^j + c, \quad j = 0, 1, 2, \dots \quad (2.7)$$

où G est la matrice d'itération $n \times n$ de la méthode et c est un vecteur associé connu.

La méthode est dite du premier degré dans la mesure où u_h^{j+1} dépend explicitement seulement de u_h^j et pas de u_h^{j-1}, \dots, u_h^0 .

Elle est linéaire puisque ni G , ni c ne dépendent de u_h^j et stationnaire puisque ni G , ni c ne dépendent de n .

Nous notons par :

$$v_h^j := u_h - u_h^j$$

l'erreur de l'approximation u_h^j (aussi appelée correction de u_h^j),

et par

$$d_h^j := f_h - L_h u_h^j$$

le défaut (ou résidu) de u_h^j .

Puisque l'opérateur L_h est linéaire, il est équivalent d'étudier le système :

$$L_h u_h = f_h$$

ou le système

$$L_h v_h^j = d_h^j \quad (2.8)$$

Si dans (2.8) on remplace l'opérateur L_h par un opérateur "plus simple" :

\hat{L}_h tel que \hat{L}_h^{-1} existe,

la solution v_h^j de :

$$\hat{L}_h v_h^j = d_h^j$$

fournit une nouvelle approximation u_h^{j+1} de la solution u_h par :

$$u_h^{j+1} = u_h^j + v_h^j$$

Si l'on se donne une valeur arbitraire u_h^0 de départ, ce procédé définit une procédure itérative dont

$$(I_h - \hat{L}_h^{-1} L_h) : \mathbb{G}(\Omega_h) \rightarrow \mathbb{G}(\Omega_h)$$

est la matrice d'itération (I_h représente l'identité sur Ω_h).

En effet :

$$\begin{aligned} u_h^{j+1} &= u_h^j + v_h^j \\ &= u_h^j + \hat{L}_h^{-1} d_h^j \\ &= u_h^j + \hat{L}_h^{-1} (f_h - L_h u_h^j) \end{aligned}$$

$$\text{d'où } u_h^{j+1} = (I_h - \hat{L}_h^{-1} L_h) u_h^j + \hat{L}_h^{-1} f_h \quad j=0,1,\dots \quad (2.9)$$

et l'on retrouve la forme (2.7) avec $\mathbb{G} = (I_h - \hat{L}_h^{-1} L_h)$ et $c = \hat{L}_h^{-1} f_h$

Il nous reste à effectuer un choix judicieux de l'opérateur \hat{L}_h .

Nous allons tout d'abord rappeler trois méthodes connues qui s'appliquent à des matrices générales : Jacobi, Jacobi relaxée et Gauss - Seidel.

1- Méthode de Jacobi

Dans la méthode de Jacobi, on remplace la matrice de l'opérateur L_h par sa partie diagonale D (On décompose L_h en $L_h = L + D + U$ où D, L, U sont des matrices composées respectivement des parties diagonale, triangulaire inférieure et triangulaire supérieure de la matrice de l'opérateur L_h , tout le reste étant égal à zéro).

La relation (2.9) devient dans ce cas :

$$u_h^{j+1} = (I_h - D^{-1} L_h) u_h^j + D^{-1} f_h$$

$$j = 0, 1, 2, \dots$$

ou encore plus simplement en fonction de la variable v_h^j :

$$v_h^{j+1} = (I_h - D^{-1} L_h) v_h^j$$

$$j = 0, 1, 2, \dots$$

Soulignons dès à présent les limitations de cette méthode.

Le rayon spectral de la matrice $(I_h - D^{-1} L_h)$ qui mesure la vitesse de convergence est égal à

$$\rho(I_h - D^{-1} L_h) = 1 - \alpha(h^2)$$

Il est donc voisin de l'unité traduisant ainsi une convergence lente.

2- Méthode de Jacobi relaxée

Si nous introduisons un facteur de relaxation ω dans la méthode de Jacobi, les formules deviennent :

$$u_h^{j+1} = (I_h - \omega D^{-1} L_h) u_h^j + \omega D^{-1} f_h$$

$$j = 0, 1, 2, \dots$$

et

$$v_h^{j+1} = (I_h - \omega D^{-1} L_h) v_h^j$$

$$j = 0, 1, 2, \dots$$

Mêmes remarques que pour la méthode de Jacobi bien que la convergence soit plus rapide, elle reste lente.

Lorsque $0 < \omega \leq 1$, il y a convergence :

$$\rho(I_h - \omega D^{-1} L_h) = 1 - O(h^2)$$

Si par contre $\omega \leq 0$ ou $\omega > 1$:

$$\rho(I_h - \omega D^{-1} L_h) \quad (\text{si } h \text{ est suffisamment petit),}$$

la procédure diverge.

3- Méthode de Gauss-Seidel

La matrice de l'opérateur L_h est remplacée ici par ses parties triangulaire inférieure et diagonale.

La formule (2.9) nous donne cette fois :

$$u_h^{j+1} = (I_h - (L+D)^{-1} L_h) u_h^j + (L+D)^{-1} f_h$$

$$j = 0, 1, 2, \dots$$

et

$$v_h^{j+1} = (I_h - (L+D)^{-1} L_h) v_h^j$$

$$j = 0, 1, 2, \dots$$

Le rayon spectral de la matrice d'itération est ici de l'ordre de $1 - O(h)$.

La convergence bien que plus rapide que dans la procédure de Jacobi n'est pas exceptionnelle.

Dans la suite, ces trois premières méthodes seront appelées méthodes de relaxation. Remarquons que bien qu'elles aient été appliquées à un système dérivant d'une équation aux dérivées partielles elles sont en réalité très générales. La méthode suivante est par contre plus liée au problème discret sous-jacent.

4- Méthode de correction par passage à une grille grossière.

On peut aussi utiliser pour \hat{L}_h , une approximation L_H de L_h sur une grille plus grossière Ω_H , ceci dans le but de réduire le nombre d'opérations à effectuer.

Ce qui signifie que la résolution du système (2.8) :

$$L_h v_h^j = d_h^j, \quad j=0,1,2,\dots$$

sera remplacée par la résolution de :

$$L_H v_H^j = d_H^j, \quad j=0,1,2,\dots \quad (2.10)$$

où $L_H : \mathbb{G}(\Omega_H) \rightarrow \mathbb{G}(\Omega_H)$ avec $\dim \mathbb{G}(\Omega_h) < \dim \mathbb{G}(\Omega_H)$ et où l'on suppose que L_H^{-1} existe.

Comme v_H^j et d_H^j sont des fonctions de grille sur Ω_H , deux opérateurs de transfert sont nécessaires :

1) Un opérateur de restriction : I_h^H

$$I_h^H : \mathbb{G}(\Omega_h) \rightarrow \mathbb{G}(\Omega_H)$$

envoie le défaut d_h^j (donné sur Ω_h) sur sa restriction d_H^j (sur Ω_H)

$$d_H^j := I_h^H d_h^j$$

2) Un opérateur d'interpolation : I_H^h

$$I_H^h : \mathbb{G}(\Omega_H) \rightarrow \mathbb{G}(\Omega_h)$$

envoie l'erreur v_H^j (calculée sur Ω_H) sur sa prolongée, v_h^j (sur Ω_h)

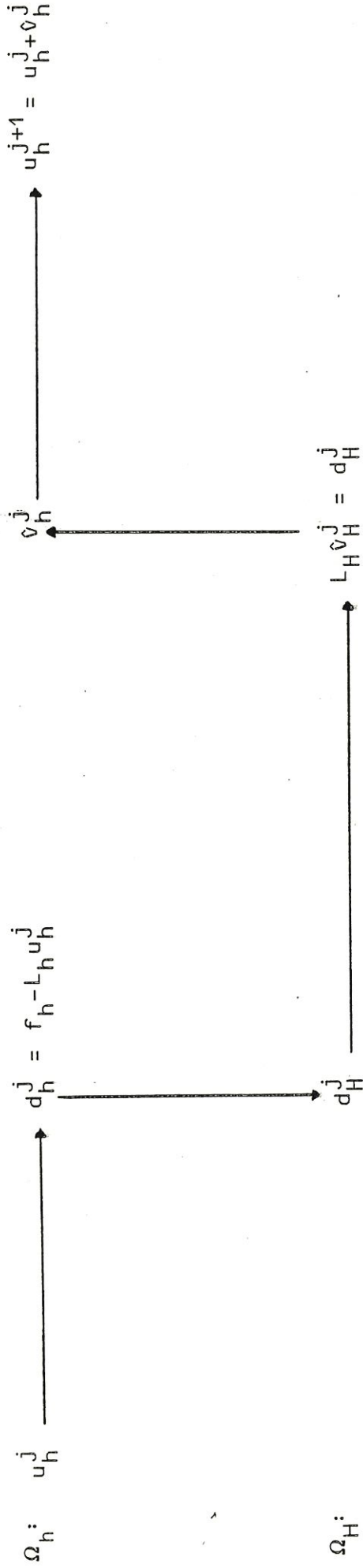
$$v_h^j := I_H^h v_H^j$$

Une itération de cette méthode se décompose comme suit :

- | | |
|--|------------------------------|
| 1) Calcul du défaut | $d_h^j := f_h - L_h u_h^j$ |
| 2) Restriction du défaut | $d_H^j := I_h^H d_h^j$ |
| 3) Résolution exacte sur Ω_H de : | $L_H v_H^j = d_H^j$ |
| qui donne la correction | v_H^j |
| 4) Interpolation de la correction | $v_h^j := I_H^h v_H^j$ |
| 5) Calcul de la nouvelle approximation | $u_h^{j+1} := u_h^j + v_h^j$ |

La figure suivante schématise ces différentes opérations

Structure du procédé de correction par grille grossière.



La matrice d'itération associée est donnée par :

$$(I_h - I_H^h L_h^{-1} I_H^h L_h)$$

Par la formule (2.9) on a :

$$u_h^{j+1} = (I_h - I_{H^h} L_H^{-1} I_h^H L_h) u_h^j + I_{H^h} L_H^{-1} I_h^H f_h$$

$$j = 0, 1, 2, \dots$$

et

$$v_h^{j+1} = (I_h - I_{H^h} L_H^{-1} I_h^H L_h) v_h^j$$

$$j = 0, 1, 2, \dots$$

Soulignons que cette procédure utilisée seule (I_h^H , I_H^h opérateurs linéaires) n'est pas convergente c'est-à-dire l'équation des résidus (2.10) sur la grille grossière Ω_H n'est pas une bonne approximation de l'équation originale des résidus (2.8) sur Ω_h . Le procédé est même divergent. On montre en effet aisément que le rayon spectral de la matrice d'itération associée à la méthode de correction par grille grossière est plus grand ou égal à un

$$\rho(I_h - I_{H^h} L_H^{-1} I_h^H L_h) \geq 1 \quad (2.11)$$

Il suffit de prendre un vecteur w_h non nul dans $\mathbb{G}(\Omega_h)$ tel que :

$L_h w_h$ appartienne au noyau de I_h^H .

Un tel w_h sera exactement reproduit par la matrice d'itération.

D'où (2.11).

Les méthodes qui viennent d'être décrites sont simples dans leur conception mais présentent de nombreux inconvénients : convergence lente voire divergence. Nous allons montrer dans le chapitre suivant qu'en les combinant astucieusement on peut obtenir une méthode de résolution très performante.

3. METHODE MULTIGRILLE

3.1 GENERALITES

L'idée de base de cette méthode est de combiner les méthodes de relaxation et de correction par grille grossière (décrites précédemment) en utilisant les avantages de chacune d'elles.

Notons $\bar{w}_h = \text{RELAX}^{\nu} (w_h, L_h, f_h)$

où \bar{w}_h est le résultat de ν pas de relaxation appliqués à

l'équation $L_h u_h = f_h$ avec w_h pris comme première approximation.

Description d'une itération deux-grilles

Partant de l'approximation u_h^j , on obtiendra u_h^{j+1} en effectuant les opérations suivantes :

(1) Phase de lissage

- Calcul de \bar{u}_h^j par l'application à u_h^j de ν_1 (≥ 0) pas d'une méthode de relaxation donnée.

$$\bar{u}_h^j := \text{RELAX}^{\nu_1}(u_h^j, L_h, f_h)$$

(2) Phase de correction par passage à une grille grossière (C G C)

- Calcul du défaut : $\bar{d}_h^j = f_h - L_h \bar{u}_h^j$
- Restriction du défaut : $\bar{d}_H^j = I_h^H \bar{d}_h^j$
- Résolution exacte sur Ω_H de : $L_H \hat{v}_H^j = \bar{d}_H^j$
qui donne la correction : \hat{v}_H^j
- Interpolation de la correction : $\hat{v}_h^j = I_H^h \hat{v}_H^j$
- Calcul de l'approximation corrigée : $\bar{u}_h^j + \hat{v}_h^j$

(3) Phase de lissage

-
- Calcul de \bar{u}_h^{j+1} , nouvelle approximation, par l'application à $\bar{u}_h^j + \hat{v}_h^j$ de $\nu_2 (\geq 0)$ pas de la méthode de relaxation donnée.

$$u_h^{j+1} := \text{RELAX}^{\nu_2}(\bar{u}_h^j + \hat{v}_h^j, L_h, f_h)$$

La matrice d'itération de la méthode deux-grilles (h,H)

décrite ci-dessus est donnée par :

$$M_h^H = S_h^{\nu_2} K_h^H S_h^{\nu_1}$$

avec

$$K_h^H = I_h - I_H^h L_H^{-1} I_h^H L_h$$

où - S_h représente la matrice d'itération de la procédure de relaxation choisie

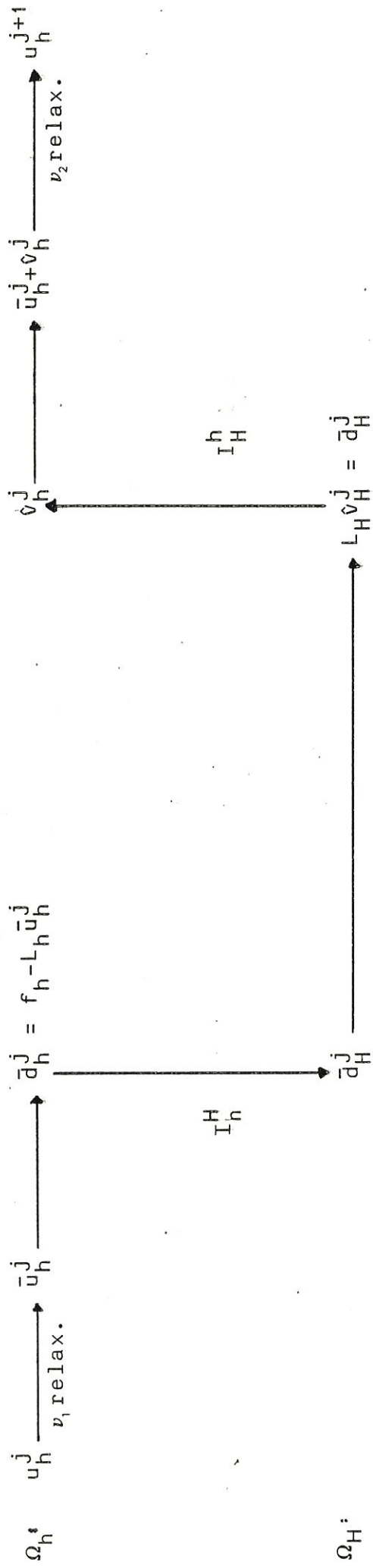
- K_h^H représente la matrice d'itération associée à la méthode C.G.C.

Différentes composantes de la méthode deux-grilles seront à spécifier :

- le lissage (caractérisé par S_h)
- les nombres ν_1 et ν_2
- la grille grossière Ω_H
- l'opérateur de restriction I_h^H
- l'opérateur d'interpolation I_H^h
- l'opérateur L_H sur la grille grossière

La figure ci-dessous schématise les différentes opérations

STRUCTURE D'UNE ITERATION DEUX-GRILLES



LA MATRICE D'ITERATION ASSOCIEE EST DONNEE PAR :

$$(S_h^{1/2} (I_h - I_H L_H^{-1} I_H L_h^{-1}) S_h^{1/2})$$

Description d'une itération multigrille

Dans la section précédente, nous avons décrit le principe multigrille seulement dans sa version deux-grilles. Nous allons décrire à présent la méthode multigrille générale.

Rappelons les différentes étapes d'une itération deux-grilles:

- lissage
- restriction du défaut sur la grille grossière
- résolution exacte sur la grille grossière de l'équation du résidu : $L_H \hat{v}_H^j = \bar{d}_H^j$
- interpolation de la correction sur la grille fine
- lissage

Observons qu'il n'est pas nécessaire de résoudre exactement l'équation du défaut sur la grille grossière. En considérant Ω_H comme la grille fine et en introduisant une grille plus grossière que Ω_H , nous pouvons remplacer la résolution exacte sur la grille Ω_H par une seconde application de la méthode deux-grilles et ainsi de suite.

Cette idée peut être appliquée récursivement en utilisant des grilles de plus en plus grossières. Une méthode de résolution exacte est alors appliquée sur la grille la plus grossière.

Voici quelques schémas illustrant la structure d'un pas d'itération d'une méthode multigrille. Les symboles \circ , \square , \backslash et $/$ signifient respectivement lissage, résolution exacte, restriction et interpolation. Le paramètre γ représente le nombre d'itérations à un niveau donné.

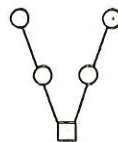
Structure d'une itération multigrille pour différents nombres
de grilles et différentes valeurs de ν

* méthode deux-grilles

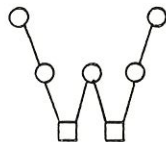


$\nu = 1$

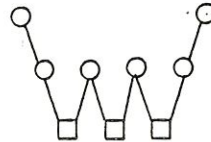
* méthode trois-grilles



$\nu = 1$

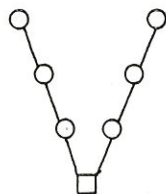


$\nu = 2$

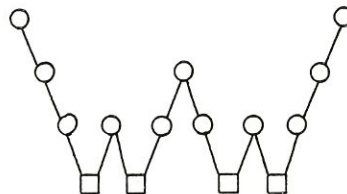


$\nu = 3$

* méthode quatre-grilles

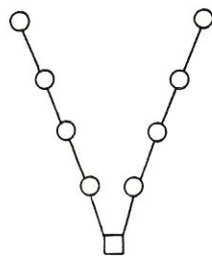


$\nu = 1$

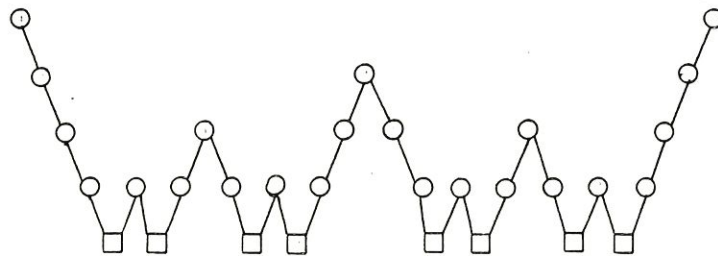


$\nu = 2$

* méthode cinq-grilles



$\nu = 1$



$\nu = 2$

Afin de décrire formellement la méthode multigrille, nous aurons besoin d'une suite de grilles de plus en plus fines Ω_{h_l} caractérisée par leur pas de discrétisation $h_l (l=0,1,2,\dots)$ respectifs.

Pour simplifier les notations, nous remplacerons dans la suite l'index h_l par l (pour les grilles, fonctions de grilles et opérateurs de grilles).

Pour chaque grille Ω_h , nous supposons que les opérateurs linéaires :

$$L_1 : \mathbb{G}(\Omega_1) \rightarrow \mathbb{G}(\Omega_1)$$

$$S_1 : \mathbb{G}(\Omega_1) \rightarrow \mathbb{G}(\Omega_1)$$

$$I_1^{l-1} : \mathbb{G}(\Omega_1) \rightarrow \mathbb{G}(\Omega_{l-1})$$

$$I_{l-1}^1 : \mathbb{G}(\Omega_{l-1}) \rightarrow \mathbb{G}(\Omega_1)$$

et les équations

$$L_1 u_1 = f_1 \quad \text{sur } \Omega_1$$

sont donnés.

Où L_1 est un opérateur inversible, $\mathbb{G}(\Omega_1)$ représente l'espace des fonctions de grille sur Ω_1 et S_1 est l'opérateur linéaire d'itération correspondant à la méthode de relaxation donnée.

Le résultat de ν pas de relaxation appliqués à $L_1 u_1 = f_1$ avec w_1 pour première approximation est noté :

$$\bar{w}_1 = \text{RELAX}^\nu(w_1, L_1, f_1)$$

Voici la description d'un pas d'une itération multigrille - plus précisément d'une itération $(l+1)$ -grilles- pour résoudre l'équation :

$$L_l u_l = f_l \quad \text{sur } \Omega_l$$

avec l fixé ≥ 1

Nous utilisons les grilles Ω_k et les opérateurs L_k , S_k , I_k^{k-1} , I_{k-1}^k , ($k=1, l-1, l-2, \dots, 0$). Nous supposons les paramètres ν_1 , ν_2 et γ fixés.

Partant de l'approximation u_l^j de u_l , nous obtenons la nouvelle approximation u_l^{j+1} en effectuant les opérations suivantes :

Si $l = 1$: procéder comme lors de la méthode deux-grilles en notant Ω_h et Ω_H respectivement Ω_1 et Ω_0

Si $l > 1$:

(1) Phase de lissage

- Calcul de \bar{u}_1^j en appliquant à u_1^j ν_1 (≥ 0) pas d'une méthode de relaxation donnée

$$\bar{u}_1^j := \text{RELAX}^{\nu_1}(u_1^j, L_1, f_1)$$

(2) Phase de correction par passage à une grille grossière

- Calcul du défaut : $\bar{d}_1^j := f_1 - L_1 u_1^j$
- Restriction du défaut : $\bar{d}_{1-1}^j := I_1^{1-1} \bar{d}_1^j$
- Calcul d'une approximation \hat{v}_{1-1}^j de l'équation du défaut sur Ω_{1-1}

$$L_{1-1} \hat{v}_{1-1}^j = \bar{d}_{1-1}^j$$

en appliquant $\nu \geq 1$ itérations d'une méthode 1-grilles (utilisant les grilles Ω_{1-1} , Ω_{1-2} , ..., Ω_0 et les opérateurs de grille correspondants) et avec la fonction de grille nulle pour première approximation.

- Interpolation de la correction : $\tilde{v}_1^j := I_{1-1}^1 \hat{v}_{1-1}^j$
- Calcul de l'approximation corrigée sur Ω_1 : $\bar{u}_1^j + \tilde{v}_1^j$

(3) Phase de lissage

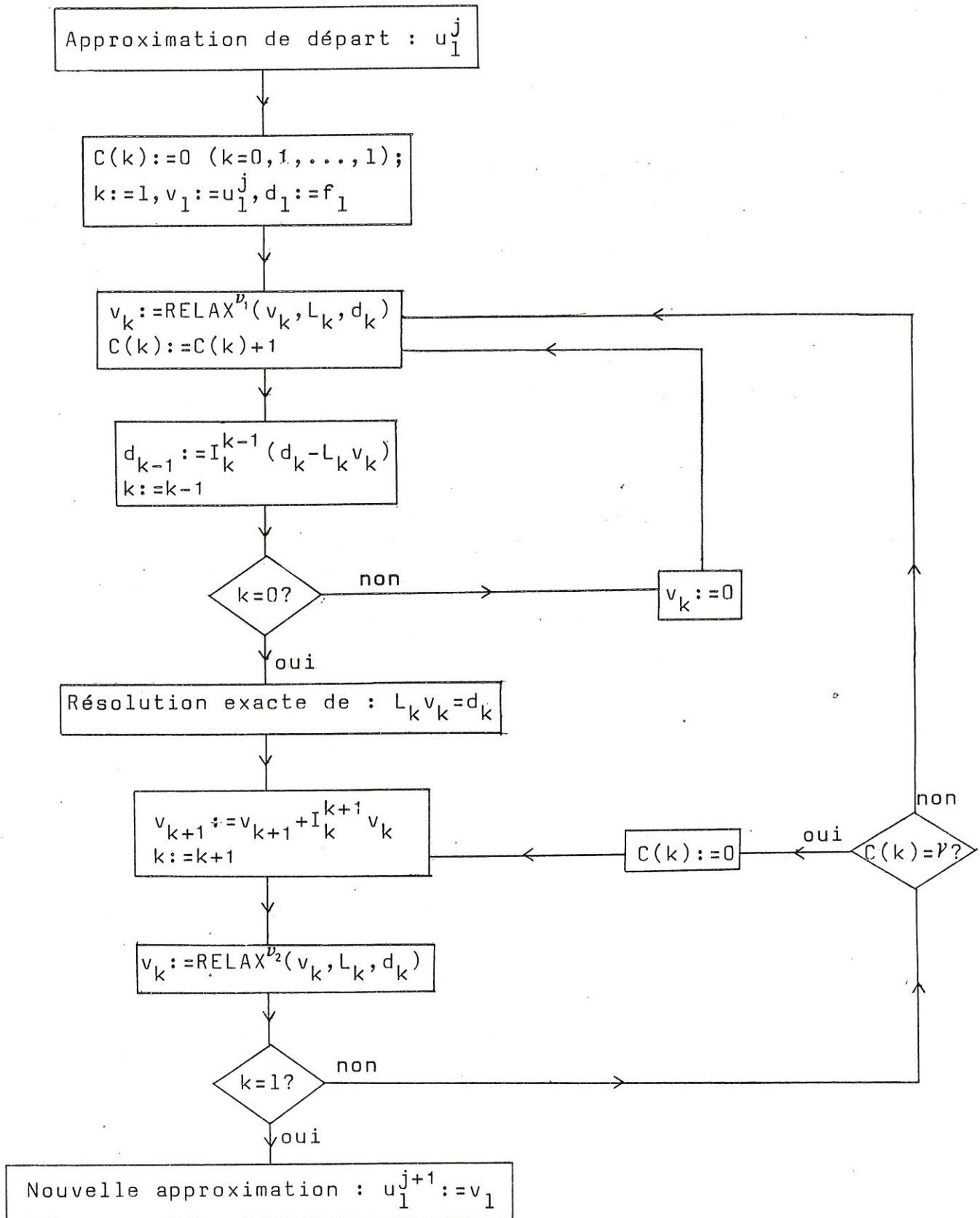
- Calcul de u_1^{j+1} en appliquant $\nu_2 (\geq 0)$ pas d'une méthode de relaxation donnée à $\bar{u}_1^j + \tilde{v}_1^j$:

$$u_1^{j+1} := \text{RELAX}^{\nu_2}(\bar{u}_1^j + \tilde{v}_1^j, L_1, f_1)$$

La même procédure est décrite dans l'organigramme suivant.

Remarquons qu'un paramètre de contrôle $C(k)$ où $0 \leq C(k) \leq \gamma$ est introduit afin de décider quand il faut passer à une grille plus grossière où au contraire quand il faut retourner à une grille fine.

ORGANIGRAMME D'UNE ITERATION MULTIGRILLE POUR LA RESOLUTION
DE $L_h u_h = f_h \quad (1 \geq 1)$



Matrice d'itération d'un cycle multigrille complet

Proposition 3.1

La matrice d'itération M_1 de la méthode multigrille décrite précédemment est donnée par la récurrence

$$(a) \quad M_1 = S_1^{\nu_2} (I_1 - I_0^1 L_0^{-1} I_1^0 L_1) S_1^{\nu_1} \quad (3.1)$$

$$(b) \quad M_{k+1} = S_{k+1}^{\nu_2} (I_{k+1} - I_k^{k+1} (I_k - M_k^{\nu_1}) L_k^{-1} I_{k+1}^k L_{k+1}) S_{k+1}^{\nu_1} \quad (3.2)$$

pour $k = 1, 2, \dots, l-1$

Démonstration

(a) Nous avons vu au début de ce chapitre que la matrice d'itération de la méthode deux-grilles était donnée par :

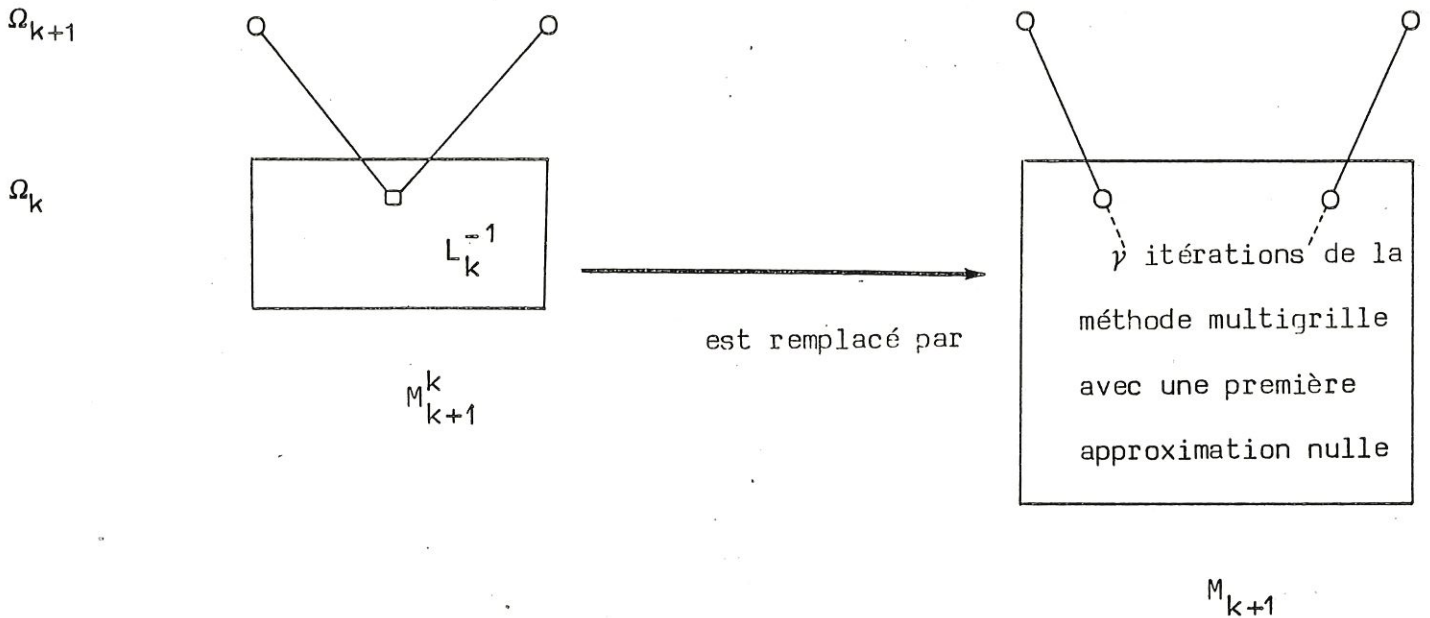
$$M_h^H = S_h^{\nu_2} (I_h - I_H^h L_H^{-1} I_h^H L_h) S_h^{\nu_1}$$

Réécrivons cette formule en utilisant les notations introduites précédemment : h_1 au lieu de h et h_0 au lieu de H

Nous obtenons bien la forme (3.1) :

$$M_1 = M_1^0 = S_1^{\nu_2} (I_1 - I_0^1 L_0^{-1} I_1^0 L_1) S_1^{\nu_1}$$

(b) Nous avons vu lors de la description de la méthode multi grille qu'une telle méthode consistait à remplacer la résolution directe de l'équation du défaut sur la grille grossière par γ itérations de la méthode multi grille elle-même c'est-à-dire :



Il s'agit donc, dans la formule connue :

$$M_{k+1}^k = S_{k+1}^{\nu_2} (I_{k+1} - I_k^{k+1} L_k^{-1} I_{k+1}^k L_{k+1}) S_{k+1}^{\nu_1} \quad (3.3)$$

de remplacer l'opérateur L_k^{-1} par un opérateur représentant γ itérations de la méthode multigrille avec une première approximation nulle.

Or, si un système non singulier d'équations linéaires $L_k \hat{v}_k = \bar{d}_k$ est résolu approximativement par γ pas d'une méthode itérative :

$\hat{v}_k^{j+1} = M \hat{v}_k^j + c$ avec $\hat{v}_k^0 = 0$, alors le γ ème itéré peut être représenté comme :

$$\hat{v}_k^\gamma = (I - M^\gamma) L_k^{-1} \bar{d}_k^j$$

$$\begin{aligned}
 \text{En effet : } \quad \hat{v}_k^1 &= M\hat{v}_k^0 + c = c \\
 \hat{v}_k^2 &= Mc + c \\
 \hat{v}_k^3 &= M(Mc + c) + c = M^2c + Mc + c \\
 &\vdots \\
 \hat{v}_k^p &= M^{p-1}c + \dots + Mc + c \\
 &= (M^{p-1} + \dots + M + I)c \\
 &= (I - M^p)(I - M)^{-1}c \\
 &= (I - M^p)(I - M)^{-1}(I - M)L_k^{-1}\bar{a}_k \\
 &= (I - M^p)L_k^{-1}\bar{a}_k
 \end{aligned}$$

En remplaçant dans la formule (3.3) L_k^{-1} par la valeur ci-dessus, nous obtenons :

$$M_{k+1} = S_{k+1}^{v_2} (I_{k+1} - I_k^{k+1} (I - M_k^p) L_k^{-1} I_{k+1}^k L_{k+1}) S_{k+1}^{v_1}$$

Ce qui démontre la proposition.

Nous allons à présent montrer qu'il est possible d'exprimer l'opérateur M_k comme une perturbation de l'opérateur deux-grilles M_k^{k-1} .

En effet, grâce à la proposition(3.3), nous avons le

Corollaire 3.1.

Pour $k = 1, \dots, l-1$, on a :

$$M_{k+1} = M_{k+1}^k + A_k^{k+1} M_k^y A_{k+1}^k$$

où

$$A_k^{k+1} := S_{k+1}^{y_2} I_k^{k+1} : \mathbb{G}(\Omega_k) \rightarrow \mathbb{G}(\Omega_{k+1})$$

$$A_{k+1}^k := L_k^{-1} I_{k+1}^k L_{k+1} S_{k+1}^{y_1} : \mathbb{G}(\Omega_{k+1}) \rightarrow \mathbb{G}(\Omega_k)$$

Preuve

De la proposition (3.3), il résulte que

$$M_{k+1} = S_{k+1}^{y_2} (I_{k+1} - I_k^{k+1} (I_k - M_k^y) L_k^{-1} I_{k+1}^k L_{k+1}) S_{k+1}^{y_1}$$

pour $k = 1, \dots, l-1$

$$= S_{k+1}^{y_2} (I_{k+1} - I_k^{k+1} L_k^{-1} I_{k+1}^k L_{k+1}) S_{k+1}^{y_1} \\ + S_{k+1}^{y_2} I_k^{k+1} M_k L_k^{-1} I_{k+1}^k L_{k+1} S_{k+1}^{y_1}$$

où l'on reconnaît la forme de l'opérateur deux-grilles vue précédemment, soit

$$M_{k+1}^k = S_{k+1}^{y_2} (I_{k+1} - I_k^{k+1} L_k^{-1} I_{k+1}^k L_{k+1}) S_{k+1}^{y_1}$$

Posons :

$$A_k^{k+1} = S_{k+1}^{y_2} I_k^{k+1}$$

$$A_{k+1}^k = L_k^{-1} I_{k+1}^k L_{k+1} S_{k+1}^{y_1}$$

Il vient :

$$M_{k+1} = M_{k+1}^k + A_k^{k+1} M_k^y A_{k+1}^k$$

ce qui termine la preuve de ce corollaire.

Cette représentation de l'opérateur M_k comme une perturbation de l'opérateur deux-grilles M_{k+1}^k nous permet de donner une estimation de la norme de M_1 lorsque une estimation de la norme des opérateurs M_{k+1}^k , A_k^{k+1} , A_{k+1}^k ($k \leq l-1$) est connue à priori.

En effet on a la proposition :

Proposition 3.2

Si $\|M_{k+1}^k\| \leq \sigma^*$, $\|A_k^{k+1}\| \|A_{k+1}^k\| \leq C$ pour tout $k \leq l-1$

Alors on a :

$$\|M_1\| \leq \eta_1$$

où η_1 est défini par :

$$\eta_1 := \sigma^*$$

$$\eta_{k+1} := \sigma^* + C \eta_k^p \quad k=1, \dots, l-1$$

La preuve de cette proposition se fait par récurrence :

Nous avons vu (démonstration de la proposition (3.1)) que :

$$M_1 = M_1^0$$

d'où l'on tire :

$$\|M_1\| = \|M_1^0\| \leq \sigma^* = \eta_1$$

Reprenons l'expression de M_k comme une perturbation de M_{k+1}^k

$$M_{k+1} = M_{k+1}^k + A_k^{k+1} M_k^\gamma A_{k+1}^k$$

ce qui nous donne :

$$\begin{aligned} \|M_{k+1}\| &\leq \|M_{k+1}^k\| + \|A_k^{k+1}\| \|A_{k+1}^k\| \|M_k\|^\gamma \\ &\leq \sigma^* + C \eta_k^\gamma \\ &= \eta_{k+1} \end{aligned}$$

En particulier, il vient pour $k = 1-1$:

$$\|M_1\| \leq \eta_1$$

avec η_k , $k = 1, \dots, 1-1$ définis comme ci-dessus.

Ce qui termine la démonstration de la proposition (3.2).

Cette proposition nous permet d'obtenir pour le cas $\gamma = 2$, l'estimation suivante de la norme de l'opérateur M_1 :

Corollaire 3.2

Dans le cas où $\gamma = 2$ et si $4C\sigma^* \leq 1$, nous obtenons l'estimation :

$$\|M_1\| \leq \eta := (1 - \sqrt{1 - 4C\sigma^*}) / 2C \leq 2\sigma^* \quad 1 \geq 1 \quad (3.4)$$

où la borne η est indépendante de l .

En effet, supposons que la récurrence

$$\eta_{k+1} = \sigma^* + C \eta_k^2$$

$$\eta_1 = \sigma^*$$

converge, il vient :

$$\eta = \sigma^* + C \eta^2 \Leftrightarrow C \eta^2 - \eta + \sigma^* = 0$$

d'où les solutions $\eta_{1,2} = \frac{1 \pm \sqrt{1 - 4C\sigma^*}}{2C}$

Comme $4C\sigma^* < 1$, les solutions sont réelles.

En outre on peut montrer aisément, en utilisant le principe de contraction, que la récurrence converge sur l'intervalle $[0, \frac{1}{2C}]$ ce qui conduit à considérer la racine

$$\eta = \frac{1 - \sqrt{1 - 4C\sigma^*}}{2C}$$

d'où la thèse.

Remarquons que pour σ^* suffisamment petit, nous obtenons $\eta \approx \sigma^*$ pour la borne η donné dans (3.4)

Par exemple, si $C = 1$, nous tirons de (3.4) :

$$\eta \leq 0.113 \text{ si } \sigma^* \leq 0.1$$

(la constante C est toujours ≥ 1 et proche de 1. Par exemple, pour la méthode traitée dans la section 3.3, nous avons $C = 1$)

En conséquence, si la méthode deux-grilles converge suffisamment bien (σ^* petit), alors la méthode multigrille correspondante avec $\nu = 2$ possède les mêmes propriétés de convergence.

Ainsi pour le problème considéré il suffit, pour ce qui touche la convergence, de ne regarder que l'étape deux-grilles.

Pour terminer, nous allons montrer que la méthode multigrille est asymptotiquement optimale dans la mesure où le nombre d'opérations requises pour résoudre le système $L_h u_h = f_h$ est proportionnel au nombre de points de grilles, la constante de proportionnalité étant petite.

Efficacité de la méthode et volume de calcul nécessaire

(Par volume de calcul, nous entendons une mesure du nombre des opérations arithmétiques)

Le travail effectué au cours d'un cycle multigrille peut s'écrire, compte tenu du corollaire (3.1) :

$$W_1 = W_1^0 + W_0 \quad (3.5)$$

$$W_{k+1} = W_{k+1}^k + \gamma_k W_k \quad (3.6)$$

où W_0 désigne le volume des opérations nécessaires pour calculer la solution exacte sur la grille grossière.

W_{k+1}^k désigne le volume des opérations nécessaires pour résoudre un cycle deux-grilles (h_{k+1} , h_k)

γ_k est le nombre d'itérations du cycle deux-grilles (h_{k+1} , h_k).

De (3.5) et (3.6) on tire en supposant $\gamma_k = \gamma$:

$$\begin{aligned} W_1 &= \gamma^{1-1} W_1^0 + \gamma^{1-2} W_2^1 + \dots + \gamma^0 W_1^{1-1} + \gamma^{1-1} W_0 \\ &= \sum_{k=1}^1 \gamma^{1-k} W_k^{k-1} + \gamma^{1-1} W_0 \end{aligned} \quad (3.7)$$

Estimation du nombre d'opérations arithmétiques nécessaires pour effectuer un cycle deux-grilles (h_{k+1} , h_k).

Si nous désignons par C le nombre des opérations réclamées en chaque point par les composantes du cycle :

- relaxation
- calcul des défauts
- passage de la grille fine à la grille grossière
- et inversement,

il vient dans le cas où toutes les composantes multigrilles sont construites de la même façon sur toutes les grilles (ce qui est le cas dans le problème que nous étudions (cfr § 3.4.1))

$$w_k^{k+1} \doteq C \mathcal{N}_k$$

où \mathcal{N}_k désigne le nombre de points de la grille \mathcal{N}_k ($\mathcal{N}_k = \# \Omega_k$)

\doteq est une notation signifiant l'égalité à des effets d'ordre inférieur près (comportement aux frontières)

Nous pouvons estimer la constante C de manière plus précise en prenant pour conventions :

- $\nu = \nu_1 + \nu_2$ représente le nombre total de relaxations au cours d'un cycle
- w_0 , w_1 , w_2 sont des mesures du nombre d'opérations nécessaires par point de la grille Ω_k pour les différents éléments qui composent un cycle deux-grilles à savoir respectivement :
 - w_0 : une relaxation sur Ω_k
 - w_1 : calcul du défaut et transfert de Ω_k vers Ω_{k-1}
 - w_2 : interpolation de la correction de Ω_{k-1} vers Ω_k et son ajout à l'approximation précédente.

Il vient

$$W_k^{k-1} \doteq (\nu w_0 + w_1 + w_2) \mathcal{N}_k$$

Dans notre cas, w_0 , w_1 et w_2 ne dépendent pas de k et peuvent être estimés ([11]) à :

	w_0	w_1	w_2	W_k^{k-1}
+/-	5	25/4	7/4	$(5\nu + 8) \mathcal{N}_k$
*	1	5/4	3/4	$(\nu + 2) \mathcal{N}_k$

En comptabilisant de la même manière les additions et les multiplications, il vient :

$$C = (6\nu + 10)$$

$$\text{d'où } W_k^{k-1} \doteq (6\nu + 10) \mathcal{N}_k$$

Le travail nécessaire pour effectuer un cycle complet est alors donné par :

$$\begin{aligned} W_1 &= C (\nu^{1-1} \mathcal{N}_1 + \nu^{1-2} \mathcal{N}_{1-1} + \dots + \nu^0 \mathcal{N}_1) + \nu^{1-1} W_0 \\ &= C \mathcal{N}_1 (\nu^{1-1} + \frac{\nu^{1-2}}{2^2} + \frac{\nu^{1-3}}{2^4} + \dots + \frac{\nu^0}{2^{2(1-1)}}) + \nu^{1-1} W_0 \end{aligned}$$

Si $\nu = 1$

$$\begin{aligned} W_1 &= C \mathcal{N}_1 \left(1 + \left(\frac{1}{2^2}\right)^1 + \left(\frac{1}{2^2}\right)^2 + \left(\frac{1}{2^2}\right)^3 + \dots \right) + W_0 \\ &= C \mathcal{N}_1 \left(\frac{\left(\frac{1}{2^2}\right)^1 - 1}{\left(\frac{1}{2^2}\right) - 1} \right) + W_0 \\ &\doteq C \mathcal{N}_1 \frac{4}{3} + W_0 \end{aligned}$$

d'où $W_1 = \frac{4}{3} C \mathcal{N}_1 + W_0$

En particulier, dans le programme que nous avons travaillé :

$W_0 \approx 0$ (cfr § 3.4.1.)

D'où $W_1 = \frac{4}{3} C \mathcal{N}_1$

Nous obtenons des résultats analogues pour $\nu = 2$ et 3 ce qui nous permet de dresser le tableau récapitulatif ci-dessous :

$$W_1 \doteq \begin{cases} \frac{4}{3} C \mathcal{N}_1 & (\text{pour } \nu = 1) \\ 2 C \mathcal{N}_1 & (\text{pour } \nu = 2) \\ 4 C \mathcal{N}_1 & (\text{pour } \nu = 3) \end{cases}$$

3.2. METHODE MULTIGRILLE UNIDIMENSIONNELLE

3.2.1 Problème modèle

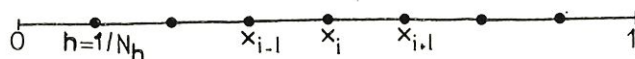
Nous allons exposer la méthode en prenant comme modèle l'équation de Poisson avec les conditions aux limites de Dirichlet :

$$\begin{cases} -\frac{d^2u}{dx^2} = f & \text{sur } \Omega = (0,1) \\ u = g & \text{sur } \Gamma = \partial\Omega = \{0,1\} \end{cases} \quad (3.8)$$

3.2.2 Formulation discrète du problème

$$\text{Soit } \Omega_h = \left\{ \frac{1}{N_h}, \frac{2}{N_h}, \dots, \frac{N_h-1}{N_h} \right\}$$

l'équivalent discret de pas h du domaine Ω où $N_h = 2^m$ est fixé.



$$\text{et soit } \Omega_H = \Omega_{2h} = \left\{ \frac{2}{N_h}, \frac{4}{N_h}, \dots, \frac{2(N_h-1)}{N_h} \right\} \subset \Omega_h$$



En approchant la dérivée seconde $\left(\frac{d^2u}{dx^2}\right)_i$ au point i par l'expression

$$\frac{u_{i+1} - 2u_i + u_{i-1}}{h^2} \quad \text{où } u(x_i) := u_i$$

encore notée

$$\frac{1}{h^2} [1 \quad -2 \quad 1] u_h$$

on obtient la forme discrète de l'équation de Poisson

$$\frac{1}{h^2} [1 \quad -2 \quad 1] u_h^j = f_h \quad (3.9)$$

avec les conditions aux limites

$$\begin{cases} u_h(0) = g(0) \\ u_h(1) = g(1) \end{cases} \quad (3.10)$$

en incluant les relations (3.10) dans le système (3.9), on obtient

un système matriciel qui, dans le cas particulier où $N_h = 8$

peut s'écrire :

$$\frac{1}{h^2} \begin{bmatrix} 2 & -4 & 0 & 0 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 2 \end{bmatrix} \begin{bmatrix} u(h) \\ u(2h) \\ u(3h) \\ u(4h) \\ u(5h) \\ u(6h) \\ u(7h) \end{bmatrix} = \begin{bmatrix} f(h) + \frac{1}{h^2} g(0) \\ f(2h) \\ f(3h) \\ f(4h) \\ f(5h) \\ f(6h) \\ f(7h) + \frac{1}{h^2} g(1) \end{bmatrix}$$

Ce système sera noté sous forme condensée :

$$L_h u_h = f_h \quad (3.11)$$

On observe tout d'abord que L_h est une matrice symétrique dont les vecteurs propres s'écrivent

$$\begin{aligned} \varphi^n(x) &= \sin(n\pi x) & x \in \Omega_h, \\ & & n = 1, \dots, N_h - 1 \end{aligned}$$

où

$$\varphi^n(x) = \begin{bmatrix} \varphi^n(1/N_h) \\ \varphi^n(2/N_h) \\ \varphi^n(3/N_h) \\ \varphi^n(4/N_h) \\ \varphi^n(5/N_h) \\ \varphi^n(6/N_h) \\ \varphi^n(7/N_h) \end{bmatrix}$$

Les valeurs propres sont positives et égales à :

$$\frac{2}{h^2} (1 - \cos n\pi h) \quad n = 1, \dots, N_h - 1$$

En effet :

$$\begin{aligned} L_h \varphi^n &= \frac{2}{\omega h^2} (-1 + (-1 - \omega + \omega \cos n\pi h)) \varphi^n \\ &= \frac{2}{\omega h^2} \omega (1 - \cos n\pi h) \varphi^n \\ &= \frac{2}{h^2} (1 - \cos n\pi h) \varphi^n \quad n = 1, \dots, N_h - 1 \end{aligned}$$

On en déduit que la solution u_h de $L_h u_h = f_h$ peut s'écrire sous la forme :

$$u_h = \sum_{i=1}^{N_h-1} \alpha_i \varphi^i(x)$$

Nous aurons besoin de ce résultat dans la suite.

■ Nous allons maintenant détailler la méthode multigrille décrite dans l'introduction ce qui revient tout d'abord à analyser chaque composante de la méthode deux-grilles.

3.2.3. Méthode deux-grilles

a. Définition de l'opérateur de lissage

Appliquons la procédure de Jacobi au système

$$L_h u_h := \frac{1}{h^2} \begin{bmatrix} -1 & 2 & -1 \end{bmatrix} u_h = f_h$$

nous obtenons en remplaçant L_h par sa composante diagonale

$$\frac{2}{h^2} u_h^{j+1} + \frac{1}{h^2} \begin{bmatrix} -1 & 0 & -1 \end{bmatrix} u_h^j = f_h$$

ce qui donne

$$u_h^{j+1} = \frac{1}{2} \begin{bmatrix} 1 & 0 & 1 \end{bmatrix} u_h^j + \frac{h^2}{2} \frac{1}{h^2} \begin{bmatrix} -1 & 2 & -1 \end{bmatrix} u_h^j$$

en posant $v_h^j = u_h - u_h^j$, il vient

$$v_h^{j+1} = \frac{1}{2} \begin{bmatrix} 1 & 0 & 1 \end{bmatrix} v_h^j$$

Et en introduisant un paramètre de relaxation ω ($0 \leq \omega \leq 1$), nous obtenons

$$v_h^{j+1} = \frac{\omega}{2} \begin{bmatrix} 1 & 2(\frac{1}{\omega}-1) & 1 \end{bmatrix} v_h^j$$

où $\frac{\omega}{2} \begin{bmatrix} 1 & 2(\frac{1}{\omega}-1) & 1 \end{bmatrix} := S_h(\omega)$ est, par définition, l'opérateur de lissage.

Cet opérateur peut encore s'exprimer sous la forme :

$$\begin{aligned} S_h(\omega) &= \frac{\omega}{2} \begin{bmatrix} 1 & \frac{2}{\omega} - 2 & 1 \end{bmatrix} \\ &= \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} + \frac{\omega}{2} \begin{bmatrix} 1 & -2 & 1 \end{bmatrix} \\ &= I_h - \frac{\omega}{2} h^2 L_h \end{aligned}$$

Il admet les mêmes vecteurs propres que L_h :

$$\begin{aligned} \varphi^n(x) &= \sin(n\pi x) & n &= 1, \dots, N_h - 1 \\ & & x &\in \Omega_h \end{aligned}$$

Ses valeurs propres sont égales à

$$K_n(\omega) = 1 - \omega + \omega \cos n\pi h \quad n=1, \dots, N_h - 1$$

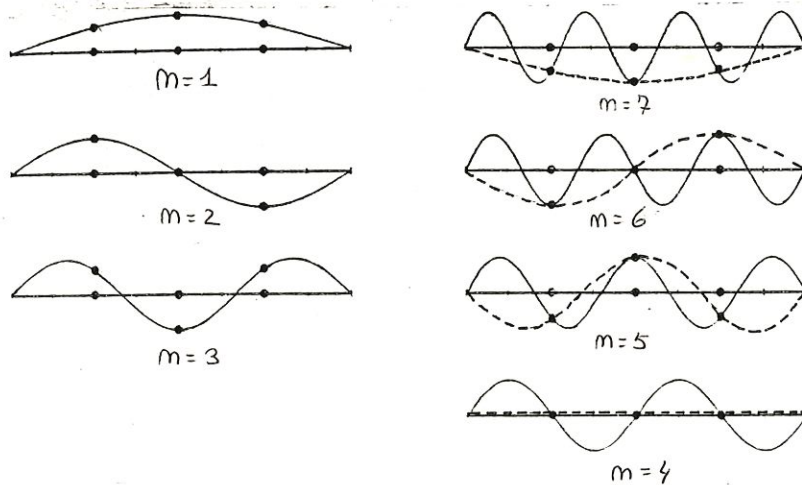
En effet

$$\begin{aligned}
 S_h(\omega) \varphi^n &= (I_h - \frac{\omega}{2} h^2 L_h) \varphi^n \\
 &= (1 - \frac{\omega}{2} h^2 \frac{2}{h^2} (1 - \cos n\pi h)) \varphi^n \\
 &= (1 - \omega + \omega \cos n\pi h) \varphi^n \\
 n &= 1, \dots, N_h - 1
 \end{aligned}$$

Remarquons dès à présent comment sont affectés les vecteurs propres $\varphi^n(x) = \sin(n\pi x)$ $x \in \Omega_h, n=1, \dots, N_h-1$ par le passage de la grille fine Ω_h à la grille grossière Ω_{2h} .

Examinons le cas où $N_h = 8$

$$\begin{aligned}
 \varphi^n(x) &= \sin(n\pi x) & x \in \left\{ \frac{1}{8}, \frac{2}{8}, \frac{3}{8}, \dots, \frac{7}{8} \right\} \\
 n &= 1, 2, \dots, 7.
 \end{aligned}$$



Les vecteurs propres φ^n pour $n = 1, 2, 3$ sont aussi représentables sur la grille grossière Ω_{2h} .
Nous les appellerons vecteurs propres "basse fréquence".

Les vecteurs propres φ^n pour $n = 4, 5, 6, 7$ ne sont pas "représentables" sur la grille grossière Ω_{2h} .
En particulier la représentation de φ^4 sur la grille grossière Ω_{2h} est la fonction nulle.
Ces vecteurs propres sont dits "haute fréquence".

Nous avons constaté que toute solution u_h de $L_h u_h = f_h$ peut s'exprimer comme une combinaison linéaire des vecteurs propres de l'opérateur L_h

$$u_h = \sum_{n=1}^{N_h-1} \alpha_n \varphi^n$$

ou en remplaçant les vecteurs propres φ^n par leurs valeurs :

$$u_h = \sum_{n=1}^{N_h-1} \alpha_n \sin(n\pi x) \quad x \in \Omega_h$$

Puisque les opérateurs L_h et $S_h(\omega)$ possèdent les mêmes vecteurs propres, nous pouvons exprimer $S_h u_h$ comme :

$$\begin{aligned} S_h u_h &= \sum_{n=1}^{N_h-1} \alpha_n S_h \sin(n\pi x) \\ &= \sum_{n=1}^{N_h-1} \alpha_n (1 - \omega + \omega \cos(n\pi h)) \sin(n\pi x) \end{aligned}$$

Nous remarquons que les composantes haute fréquence de la solution u_h sont réduites.

Si nous prenons $\omega = \frac{2}{3}$ par exemple, ces composantes seront au moins réduites d'un facteur

$$\begin{aligned} \mu(h, S_h(\frac{2}{3})) &= \max_{\frac{N_h}{2} \leq n \leq N_h-1} \left| \frac{1}{3} + \frac{2}{3} \cos(n\pi h) \right| \\ &= \frac{1}{3} \end{aligned}$$

à chaque itération de la méthode de Jacobi.

Cet effet de lissage est mesuré grâce au paramètre

$$\mu(h, S_h(\omega)) = \max_{\frac{N_h}{2} \leq n \leq N_h - 1} |1 - \omega + \omega \cos(n\pi h)|$$

appelé facteur de lissage.

A titre d'exemple, estimons ce facteur pour différentes valeurs du paramètre de relaxation ω :

$$\underline{\omega = 1} : K_n = \cos(n\pi h)$$

$$\begin{aligned} \mu(h, S_h(\omega)) &= \max_{\frac{N_h}{2} \leq n \leq N_h - 1} |\cos(n\pi h)| = |\cos((N_h - 1)\pi h)| \\ &= |\cos(\pi h)| \\ &= 1 - O(h^2) \end{aligned}$$

La méthode de Jacobi utilisée avec un facteur de relaxation $\omega = 1$ possède de mauvaises propriétés de lissage.

$$\underline{\omega = 1/2} : K_n = \frac{1}{2} + \frac{1}{2} \cos(n\pi h)$$

$$\mu(h, S_h(\omega)) = \max_{\frac{N_h}{2} \leq n \leq N_h - 1} \left| \frac{1}{2} + \frac{1}{2} \cos(n\pi h) \right| = \frac{1}{2}$$

La méthode de Jacobi utilisée avec un facteur de relaxation $\omega = \frac{1}{2}$ possède de bonnes propriétés de lissage : les composantes haute fréquence de la solution seront au moins réduites d'un facteur $\frac{1}{2}$ par itération de Jacobi.

Lemme 3.1

Si h est suffisamment petit, nous avons :

$$\mu(h, S_h(\omega)) = \begin{cases} 1 - \omega & \text{si } \omega \leq 2/3 \\ \omega - 1 - \omega \cos(\pi h) & \text{si } \omega > 2/3 \end{cases}$$

En outre, la valeur $\omega = 2/3$ est optimale et conduit donc au plus petit facteur de lissage :

$$\mu(h, S_h(2/3)) = 1/3.$$

Preuve :

Rappelons que le facteur de lissage μ est donné par l'expression :

$$\mu = \frac{N_h}{2} \max_{n \leq N_h - 1} |1 - \omega + \omega \cos(n\pi h)|$$

Afin de tracer le graphe de cette fonction dépendant de ω , on analyse séparément :

$$\mu^1 = |1 - \omega + \omega \cos(N_h - 1)\pi h|$$

et
$$\mu^2 = |1 - \omega + \omega \cos \frac{N_h}{2} \pi h|$$

en observant que $\mu = \max\{\mu^1, \mu^2\}$

Etude de $\mu^1 = |1 - \omega + \omega \cos(N_h - 1)\pi h|$

Il vient :

$$\cos(N_h - 1)\pi h = -\cos\pi h = -1 + o(h^2)$$

d'où

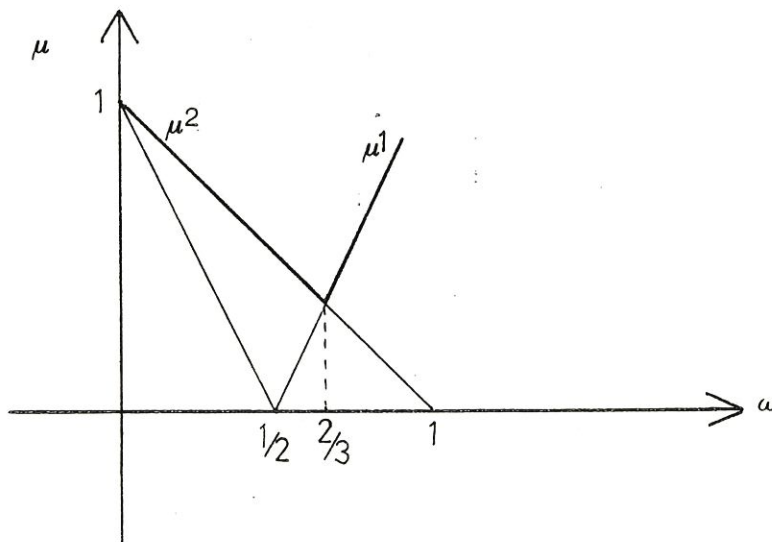
$$\mu^1 = |1 - 2\omega + o(h^2)|$$

et aux termes du second ordre près :

$$\mu^1 = \begin{cases} 2\omega - 1 & \text{si } \omega \geq 1/2 \\ 1 - 2\omega & \text{si } \omega < 1/2 \end{cases}$$

Etude de $\mu^2 = |1 - \omega| = 1 - \omega$

La représentation graphique de μ^1 et de μ^2 met en évidence le point critique $\omega = 2/3$



b. Définition de l'opérateur de correction par grille grossière (CGC)

Nous avons vu que cette méthode remplace la résolution du système :

$$L_h v_h^j = d_h^j, \quad j = 0, 1, 2, \dots$$

sur la grille fine Ω_h

par la résolution de :

$$L_H v_H^j = d_H^j, \quad j = 0, 1, 2, \dots$$

sur la grille grossière Ω_H

En plus d'une méthode de résolution pour le système $L_H v_H^j = d_H^j$,

nous aurons donc besoin de deux opérateurs de transfert :

un opérateur de restriction I_h^{2h} et un opérateur de

prolongement I_{2h}^h

1- L'opérateur de restriction : I_h^{2h}

envoie le résidu donné dans Ω_h soit d_h^j sur sa restriction

dans Ω_{2h} : d_{2h}^j

Cet opérateur doit être simple à calculer tout en perdant le moins d'information possible lors du transfert de grille à grille.

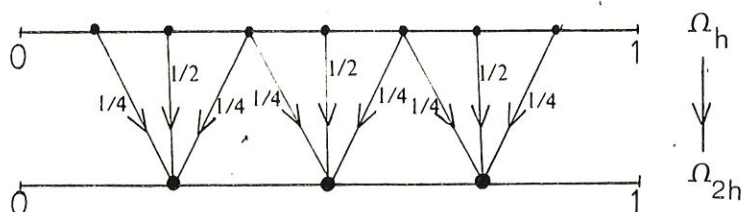
Nous choisissons

$$I_h^{2h} = \frac{1}{4} \begin{bmatrix} 1 & 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 2 & 1 \end{bmatrix}$$

qui correspond à la molécule :

$$\frac{1}{4} [1 \quad 2 \quad 1]$$

ou graphiquement :



2- A l'inverse, l'opérateur de prolongement : I_{2h}^h —
 envoie l'erreur v_{2h}^j calculée dans Ω_{2h} (qui est solution
 du système $L_{2h} v_{2h}^j = d_{2h}^j$) sur sa prolongée dans Ω_h : v_h^j
 De nouveau nous désirons un opérateur simple à calculer et perdant
 un minimum d'informations.

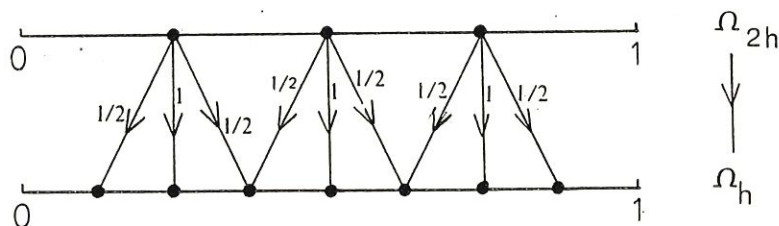
Nous choisissons

$$I_{2h}^h = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 2 \\ 0 & 0 & 1 \end{bmatrix}$$

qui correspond à la molécule

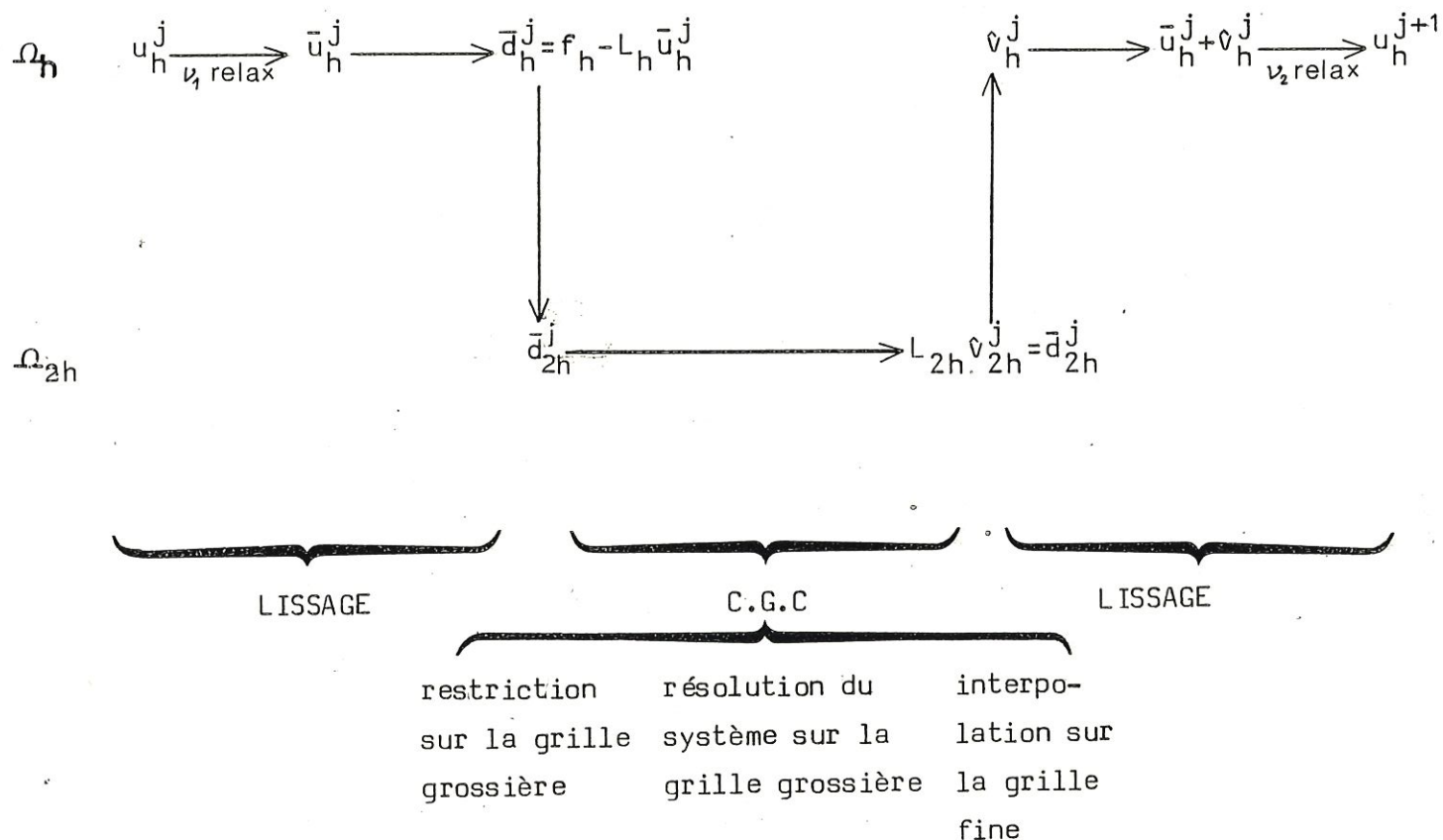
$$\frac{1}{2}] 1 \quad 2 \quad 1 [$$

ou graphiquement



c. Définition de l'opérateur d'itération deux-grilles

Combinons les opérations de lissage, de transfert et de résolution vues dans le chapitre précédent suivant le schéma "deux-grilles" exposé au chapitre 2.1



Une itération "deux-grilles" est alors caractérisée par l'opérateur matriciel :

$$M_h^{2h} = S_h^{\nu_2} (I_h - I_{2h}^h L_{2h}^{-1} I_h^{2h} L_h) S_h^{\nu_1}$$

En posant $K_h^{2h} = I_h - I_{2h}^h L_{2h}^{-1} I_h^{2h} L_h$, il vient :

$$M_h^{2h} = S_h^{\nu_2} K_h^{2h} S_h^{\nu_1}$$

Pour caractériser les propriétés de convergence de l'opérateur M_h^{2h} , nous devons calculer son rayon spectral. Comme il est trop compliqué de calculer directement celui-ci, nous allons construire \hat{M}_h^{2h} , un équivalent orthogonal de la matrice M_h^{2h} simple à manipuler. Afin de simplifier les calculs nous prendrons $N_h = 8$.

Observons tout d'abord que :

$$\begin{aligned} \phi^n(x) &:= \varphi^n(x) = -\varphi^{N_h-n}(x) & x \in \Omega_{2h} \\ \phi^{N_h/2}(x) &:= \varphi^{N_h/2}(x) = 0 & x \in \Omega_{2h} \end{aligned}$$

En effet :

$$\begin{aligned} \varphi^{N_h-n}(x) &= -\sin((N_h-n)\pi x) \\ &= -\sin(N_h\pi x) \cos(n\pi x) + \sin(n\pi x) \cos(N_h\pi x) \\ &= \sin(n\pi x) \\ &= \varphi^n(x) \end{aligned} \quad \text{où } x = lh \text{ avec } l = 2, 4, 6.$$

$$\begin{aligned} \varphi^{N_h/2}(x) &= \sin\left(\frac{N_h}{2} \pi x\right) \\ &= \sin\left(l \frac{\pi}{2}\right) \\ &= 0 \end{aligned} \quad \text{où } x = lh \text{ avec } l = 2, 4, 6.$$

Posons $E_{h,n} := \text{span} [\varphi^n, \varphi^{N_h-n}]$ c'est-à-dire l'espace engendré par les vecteurs $\varphi^n, \varphi^{N_h-n}$ et $E_{h, \frac{N_h}{2}} = \text{span} [\varphi^{\frac{N_h}{2}}]$

Nous allons d'abord montrer que ces sous-espaces de $\mathbb{G}(\Omega_h)$ sont laissés invariants par l'opérateur K_h^{2h} , c'est-à-dire

$$K_h^{2h} : E_{h,n} \rightarrow E_{h,n} \quad \left(n \leq \frac{N_h}{2}\right)$$

En effet,

$L_h : E_{h,n} \rightarrow E_{h,n}$: car φ^n et φ^{N_h-n} sont des vecteurs propres de L_h

$$\underline{I_h^{2h} : E_{h,n} \rightarrow \text{span} [\Phi^n]}$$

En effet, si nous appliquons l'opérateur I_h^{2h} au vecteur propre φ^n où $n \leq \frac{N_h}{2}$ il vient,

$$\frac{1}{2} \begin{bmatrix} 1/2 & 1 & 1/2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 1 & 1/2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1 & 1/2 \end{bmatrix} \begin{bmatrix} \sin(n\pi h) \\ \sin(2n\pi h) \\ \sin(3n\pi h) \\ \sin(4n\pi h) \\ \sin(5n\pi h) \\ \sin(6n\pi h) \\ \sin(7n\pi h) \end{bmatrix}$$

$$= \frac{1}{2} \begin{bmatrix} \frac{1}{2} (\sin(n\pi h) + \sin(3n\pi h)) + \sin(2n\pi h) \\ \frac{1}{2} (\sin(3n\pi h) + \sin(5n\pi h)) + \sin(4n\pi h) \\ \frac{1}{2} (\sin(5n\pi h) + \sin(7n\pi h)) + \sin(6n\pi h) \end{bmatrix}$$

$$= \frac{1}{2} \begin{bmatrix} \sin(2n\pi h) (\cos(n\pi h) + 1) \\ \sin(4n\pi h) (\cos(n\pi h) + 1) \\ \sin(6n\pi h) (\cos(n\pi h) + 1) \end{bmatrix}$$

$$= \frac{1}{2} (\cos(n\pi h) + 1) \begin{bmatrix} \sin(2n\pi h) \\ \sin(4n\pi h) \\ \sin(6n\pi h) \end{bmatrix}$$

Appliquons à présent ce même opérateur I_h^{2h} au vecteur propre φ^{N_h-n}
 où $n < \frac{N_h}{2}$ il vient

$$\frac{1}{2} \begin{bmatrix} \frac{1}{2} & 1 & \frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 1 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & 1 & \frac{1}{2} \end{bmatrix} \begin{bmatrix} \sin((N_h-n)\pi h) \\ \sin((N_h-n)2\pi h) \\ \sin((N_h-n)3\pi h) \\ \sin((N_h-n)4\pi h) \\ \sin((N_h-n)5\pi h) \\ \sin((N_h-n)6\pi h) \\ \sin((N_h-n)7\pi h) \end{bmatrix}$$

$$= \frac{1}{2} \begin{bmatrix} \frac{1}{2} & 1 & \frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 1 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & 1 & \frac{1}{2} \end{bmatrix} \begin{bmatrix} \sin(n\pi h) \\ -\sin(2n\pi h) \\ \sin(3n\pi h) \\ -\sin(4n\pi h) \\ \sin(5n\pi h) \\ -\sin(6n\pi h) \\ \sin(7n\pi h) \end{bmatrix}$$

$$= \frac{1}{2} \begin{bmatrix} \frac{1}{2} (\sin(n\pi h) + \sin(3n\pi h)) - \sin(2n\pi h) \\ \frac{1}{2} (\sin(3n\pi h) + \sin(5n\pi h)) - \sin(4n\pi h) \\ \frac{1}{2} (\sin(5n\pi h) + \sin(7n\pi h)) - \sin(6n\pi h) \end{bmatrix}$$

$$= \frac{1}{2} \begin{bmatrix} \sin(2n\pi h)(\cos(n\pi h) - 1) \\ \sin(4n\pi h)(\cos(n\pi h) - 1) \\ \sin(6n\pi h)(\cos(n\pi h) - 1) \end{bmatrix}$$

$$= \frac{1}{2} (\cos n\pi h - 1) \begin{bmatrix} \sin(2n\pi h) \\ \sin(4n\pi h) \\ \sin(6n\pi h) \end{bmatrix}$$

$L_{2h} : \text{span}[\phi^n] \rightarrow \text{span}[\phi^n]$: car ϕ^n sont des vecteurs propres de L_{2h}

$I_{2h}^h : \text{span}[\phi^n] \rightarrow E_{h,n}$

En effet :

$$\begin{bmatrix} \frac{1}{2} & 0 & 0 \\ 1 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 1 & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 1 \\ 0 & 0 & \frac{1}{2} \end{bmatrix} \begin{bmatrix} \sin 2n\pi h \\ \sin 4n\pi h \\ \sin 6n\pi h \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{2} \sin 2n\pi h \\ \sin 2n\pi h \\ \frac{1}{2} \sin 2n\pi h + \frac{1}{2} \sin 4n\pi h \\ \sin 4n\pi h \\ \frac{1}{2} \sin 4n\pi h + \frac{1}{2} \sin 6n\pi h \\ \sin 6n\pi h \\ \frac{1}{2} \sin 6n\pi h \end{bmatrix}$$

$$= \alpha \begin{bmatrix} \sin n\pi h \\ \sin 2n\pi h \\ \sin 3n\pi h \\ \sin 4n\pi h \\ \sin 5n\pi h \\ \sin 6n\pi h \\ \sin 7n\pi h \end{bmatrix} + \beta \begin{bmatrix} \sin n\pi h \\ -\sin 2n\pi h \\ \sin 3n\pi h \\ -\sin 4n\pi h \\ \sin 5n\pi h \\ -\sin 6n\pi h \\ \sin 7n\pi h \end{bmatrix}$$

Pour les composantes paires ($l = 2, 4, 6$), il vient :

$$\alpha \sin l n \pi h - \beta \sin l n \pi h = \sin l n \pi h$$

$$\text{d'où } (\alpha - \beta) = 1$$

Pour les composantes impaires ($l = 1, 3, 5, 7$), il vient :

$$\alpha \sin l n \pi h + \beta \sin l n \pi h = \frac{1}{2} \sin (l-1) n \pi h + \frac{1}{2} \sin (l+1) n \pi h$$

$$\text{d'où } (\alpha + \beta) \sin l n \pi h = \cos n \pi h \sin l n \pi h$$

$$\text{et } \alpha + \beta = \cos n \pi h$$

On en déduit les valeurs de α et β :

$$\alpha = \frac{1}{2} (\cos n \pi h + 1)$$

$$\beta = \frac{1}{2} (\cos n \pi h - 1)$$

Nous pouvons maintenant construire \hat{K}_h^{2h} l'équivalent orthogonal de K_h^{2h} , formé des images par K_h^{2h} des vecteurs de base de $\mathbb{G}(\Omega_h)$ pris dans l'ordre : φ^1, φ^7 ; φ^2, φ^6 ; φ^3, φ^5 et φ^4 .

Rappelons que $K_h^{2h} = I_h - I_{2h}^h L_{2h}^{-1} I_h^{2h} L_h$

et appliquons cet opérateur aux vecteurs φ^n où $n \leq \frac{N_h}{2}$

$$\begin{aligned} & (I_h - I_{2h}^h L_{2h}^{-1} I_h^{2h} L_h) \varphi^n \\ &= \varphi^n - 2N_h^2 (1 - \cos n \pi h) I_{2h}^h L_{2h}^{-1} I_h^{2h} L_h \varphi^n \\ &= \varphi^n - 2N_h^2 (1 - \cos n \pi h) \frac{1}{2} (\cos n \pi h + 1) I_{2h}^h L_{2h}^{-1} \varphi^n \\ &= \varphi^n - N_h^2 (1 - \cos n \pi h) (1 + \cos n \pi h) \frac{2}{N_h^2 (1 - \cos 2n \pi h)} I_{2h}^h \varphi^n \\ &= \varphi^n - \frac{2 \sin^2 n \pi h}{(1 - \cos 2n \pi h)} \left(\frac{1}{2} (\cos n \pi h + 1) \varphi^n + \frac{1}{2} (\cos n \pi h - 1) \varphi^{N_h - n} \right) \\ &= \varphi^n - \cos^2 \left(\frac{n \pi h}{2} \right) \varphi^n + \sin^2 \left(\frac{n \pi h}{2} \right) \varphi^{N_h - n} \end{aligned}$$

Posons $\sin^2 \frac{n\pi h}{2} = \xi$ d'où $\cos^2 \frac{n\pi h}{2} = 1 - \xi$

Il vient,

$$K_h^{2h} \varphi^n = \varphi^n - ((1 - \xi) \varphi^n - \xi \varphi^{N_h - n})$$

En appliquant la même procédure à $\varphi^{N_h - n}$ où $n < \frac{N_h}{2}$ et à $\varphi^{N_h/2}$, nous obtenons

$$K_h^{2h} \varphi^{N_h - n} = \varphi^{N_h - n} - (-(1 - \xi) \varphi^n + \xi \varphi^{N_h - n})$$

$$K_h^{2h} \varphi^{N_h/2} = \varphi^{N_h/2}$$

D'où l'on tire que l'équivalent orthogonal \hat{K}_{2h}^h de K_{2h}^h est diagonal par blocs $(\hat{K}_{2h,n}^h)$ où

$$\hat{K}_{h,n}^{2h} = \begin{cases} \begin{bmatrix} \xi & \xi - 1 \\ -\xi & 1 - \xi \end{bmatrix} & \text{si } n < \frac{N_h}{2} \\ \text{la matrice d'identité} & \text{si } n = \frac{N_h}{2} \\ \text{de dimension 1} & \end{cases}$$

D'autre part, comme les vecteurs φ^n ($n=1, \dots, N_h-1$) sont aussi des vecteurs propres de S_h , on en déduit que les espaces $E_{h,n}$ ($n \leq \frac{N_h}{2}$) sont aussi laissés invariants par l'opérateur S_h :

$$S_h: E_{h,n} \longrightarrow E_{h,n} \quad n \leq \frac{N_h}{2}$$

L'équivalent diagonal \hat{S}_h de S_h se réduit à une diagonale constituée des valeurs propres $K_n = (1 - \omega + \omega \cos n\pi h)$, $n=1, \dots, N_h-1$ de S_h prises

dans l'ordre $K_1, K_{N_h-1}; K_2, K_{N_h-2}; \dots; K_{\frac{N_h}{2}-1}, K_{\frac{N_h}{2}+1}; K_{\frac{N_h}{2}}$. Les espaces $E_{h,n}$ ($n \leq N_h/2$) étant laissés invariants par K_h^{2h} et S_h , ils le sont aussi par M_h^{2h} . Et nous obtenons la représentation

$$\hat{M}_{h,n}^{2h} \text{ de } M_{h,n}^{2h} :$$

$$\hat{M}_{h,n}^{2h} = \hat{M}_{h,n}^{2h}(v_1, v_2; \omega) = \hat{S}_{h,n}(\omega) \hat{K}_{h,n}^{2h} \hat{S}_{h,n}(\omega)$$

Nous avons donc réduit le calcul du rayon spectral $\rho(M_h^{2h})$ au calcul des rayons spectraux de matrices (au plus) de dimension 2.

$$\rho(M_h^{2h}) = \max \left\{ \rho(\hat{M}_{h,n}^{2h}) : n \leq N_h/2 \right\}$$

Cette quantité dépend en particulier des paramètres ω , ν_1 et ν_2 ou plus exactement de ω et $\nu := \nu_1 + \nu_2$.

En effet, comme $\rho(AB) = \rho(BA)$ où A et B sont des opérateurs linéaires,

$\rho(M_h^{2h})$ ne dépend pas de ν_1 et ν_2 individuellement mais de leur somme (ν).

Pour la méthode développée dans ce chapitre, c'est-à-dire quand la technique de lissage utilisée est celle de Jacobi relaxée, on obtient pour ω fixé égal à $1/2$ les valeurs suivantes du rayon spectral de l'opérateur M_h^{2h} :

	$\nu = 1$	$\nu = 2$	$\nu = 3$	$\nu = 4$
$\rho(M_h^{2h})$	1/2	1/4	1/8	0.083 ...

Dans le paragraphe suivant, nous allons étendre la méthode multigrille au cas bidimensionnel en résolvant l'équation de Poisson avec les conditions aux limites de Dirichlet sur le carré unité.

3.3 METHODE MULTIGRILLE BIDIMENSIONNELLE.

3.1. Problème modèle

Soit à résoudre l'équation de Poisson

$$-\Delta u = -\frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial y^2} = f \quad \text{sur } \Omega = (0,1) \times (0,1) \quad (3.12)$$

avec les conditions aux limites de Dirichlet

$$u = g \quad \text{sur } \Gamma = \partial \Omega$$

3.2. Formulation discrète du problème

Soit $\Omega_h = \{x = (ih, jh) : i, j = 1, \dots, N_h - 1\}$ où $N_h = \frac{1}{h}$
 et soit $\Omega_H = \Omega_{2h} = \{x = (2ih, 2jh) : i, j = 1, \dots, \frac{N_h}{2} - 1\}$

Si l'on discrétise le problème (3.12) par la méthode des différences finies sur Ω_h , on obtient en utilisant la notation moléculaire l'approximation du second ordre classique :

$$L_h^{\Omega} = -\Delta_h = \frac{1}{h^2} \begin{bmatrix} & -1 & \\ -1 & 4 & -1 \\ & -1 & \end{bmatrix}$$

L'élimination des conditions aux limites discrètes conduit à une équation de grille de la forme :

$$L_h u_h := \frac{1}{h^2} \begin{bmatrix} & -1 & \\ -1 & 4 & -1 \\ & -1 & \end{bmatrix} u_h = f_h \quad (\Omega_h) \quad (3.13)$$

où f_h contient une composante de la frontière Γ en tous les points $x = (ih, jh)$ avec i ou $j = 1$ ou $N_h - 1$

Observons que la représentation matricielle du système (3.13) dépend de l'ordre de parcours des points de la grille Ω_h . En effet, si nous choisissons l'ordre lexicographique c'est-à-dire si les points de la grille sont indicés comme l'indique la figure 1,

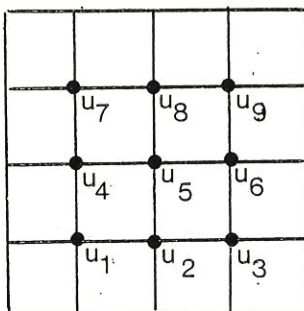
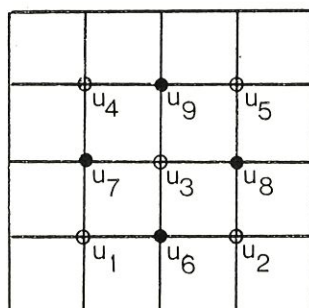


fig.1

Nous obtenons la représentation matricielle de L_h suivante :

$$L_h = \frac{1}{h^2} \begin{bmatrix} 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 4 & 0 & 0 & -1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 4 & -1 & 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & -1 & 4 & -1 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & -1 & 4 & 0 & 0 & -1 \\ 0 & 0 & 0 & -1 & 0 & 0 & 4 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 \end{bmatrix}$$

Par contre, en choisissant l'ordre "red-black" c'est-à-dire en indiquant les points comme sur la figure ci-dessous



○ point rouge
● point noir

fig.2

Nous obtenons pour L_h la représentation matricielle :

$$L_h = \frac{1}{h^2} \begin{bmatrix} 4 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 & -1 & 0 & -1 & 0 \\ 0 & 0 & 4 & 0 & 0 & -1 & -1 & -1 & -1 \\ 0 & 0 & 0 & 4 & 0 & 0 & -1 & 0 & -1 \\ 0 & 0 & 0 & 0 & 4 & 0 & 0 & -1 & -1 \\ \hline -1 & -1 & -1 & 0 & 0 & 4 & 0 & 0 & 0 \\ -1 & 0 & -1 & -1 & 0 & 0 & 4 & 0 & 0 \\ 0 & -1 & -1 & 0 & -1 & 0 & 0 & 4 & 0 \\ 0 & 0 & -1 & -1 & -1 & 0 & 0 & 0 & 4 \end{bmatrix}$$

Dans ce paragraphe, nous allons traiter le cas où L_h est la matrice associée à l'ordre lexicographique, tous les développements ultérieurs pouvant être traités de manière similaire pour le cas de l'ordre "red-black". Observons tout d'abord que L_h est une matrice symétrique dont les vecteurs propres sont donnés par :

$$\varphi^n(x) = 2 \sin(n_1 \pi x_1) \sin(n_2 \pi x_2) \quad x \in \Omega_h; |n| \leq N_h - 1$$

$$\text{où } n = (n_1, n_2) \in \mathbb{N}^2$$

$$\text{et } |n| = \max(n_1, n_2)$$

Ce qui donne dans le cas particulier où $N_h = 4$:

$$\varphi^n(x) = \begin{pmatrix} \sin(n_1 \pi h) \sin(n_2 \pi h) ; \sin(2n_1 \pi h) \sin(n_2 \pi h) ; \sin(3n_1 \pi h) \sin(n_2 \pi h) \\ \sin(n_1 \pi h) \sin(2n_2 \pi h) ; \sin(2n_1 \pi h) \sin(2n_2 \pi h) ; \sin(3n_1 \pi h) \sin(2n_2 \pi h) \\ \sin(n_1 \pi h) \sin(3n_2 \pi h) ; \sin(2n_1 \pi h) \sin(3n_2 \pi h) ; \sin(3n_1 \pi h) \sin(3n_2 \pi h) \end{pmatrix}$$

Les valeurs propres correspondantes de L_h sont toutes positives et données par l'expression :

$$K_n = \frac{1}{h^2} (2 - \cos(n_1 \pi h) - \cos(n_2 \pi h))$$

3.3.3. Méthode deux-grilles

Comme dans le cas unidimensionnel, nous allons considérer deux phases : le lissage et la correction par grille grossière.

1- Le lissage

Un pas de la méthode de Jacobi relaxée appliquée au problème (3.12) avec w_h comme première approximation sera noté :

$$\bar{w}_h = \text{RELAX} (w_h, L_h, f_h; \omega)$$

Où la matrice d'itération associée est donnée par l'expression

$$S_h = S_h(\omega) = I_h - \frac{\omega h^2}{4} L_h$$

Par cette formulation, on constate que les vecteurs propres de L_h et ceux de $S_h(\omega)$ sont les mêmes :

$$\varphi^n(x) = 2 \sin(\eta_1 \pi x_1) \sin(\eta_2 \pi x_2) \quad x \in \Omega_h, |n| \leq N_h - 1$$

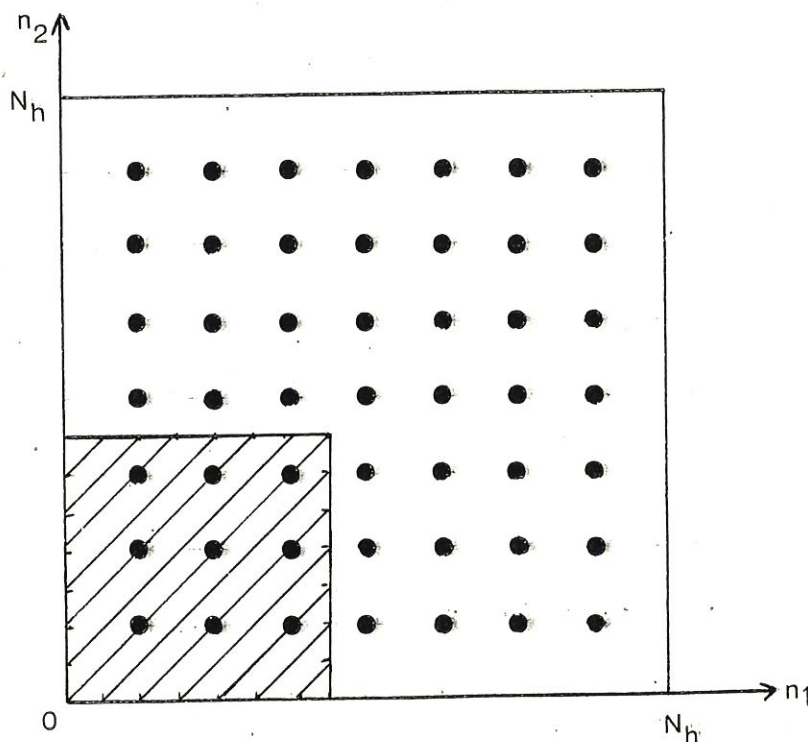
Les valeurs propres de $S_h(\omega)$ sont données par :

$$\chi_n = 1 - \frac{\omega}{2} (2 - \cos(\eta_1 \pi h) - \cos(\eta_2 \pi h)) \quad (3.14)$$

Comme pour le cas unidimensionnel, les propriétés de lissage de sont mesurées en distinguant basses et hautes fréquences (par rapport à la grille grossière Ω_H choisie).

Rappelons que les basses fréquences sont les vecteurs propres représentables sur la grille grossière Ω_H tandis que les vecteurs haute fréquence ne sont "pas visibles" sur Ω_H .

On définira en conséquence les vecteurs propres basse fréquence comme étant les φ^n pour lesquels $|n| < N_h/2$ (dans la figure ci-dessous, ils occupent la partie hachurée). Les vecteurs propres haute fréquence seront les φ^n pour lesquels $N_h/2 \leq |n| \leq N_h - 1$ (ils se situent dans la surface non hachurée)



Comme pour le cas unidimensionnel, nous définissons le facteur de lissage $\mu_h(\omega)$ de l'opérateur $S_h(\omega)$ par l'expression :

$$\mu_h(\omega) := \max \left\{ |x_n| : N_h \leq |n| \leq N_h - 1 \right\} \quad (3.15)$$

Ou encore, le facteur de lissage est le plus mauvais (ie le plus grand) facteur de lissage des composantes de l'erreur par pas d'itération de la méthode de relaxation employée.

En appliquant la définition(3.15) pour les valeurs trouvées en(3.14), il vient :

$$\mu(h, \omega) := \max \left\{ \left| 1 - \omega(2 - \cos\pi h) \right|, \left| 1 - \omega(1 + \cos\pi h) \right| \right\} \quad (3.16)$$

et nous obtenons :

$$\begin{cases} \mu(h, \omega) \geq 1 & \text{pour } \omega \leq 0 \text{ ou } \omega > 1 \\ \mu(h, \omega) < 1 & \text{pour } 0 < \omega < 1 \end{cases} \quad (\text{avec } h \text{ petit})$$

En particulier, l'expression(3.16) donne :

$$\mu(h, \omega) = \begin{cases} \cos\pi h & \text{si } \omega = 1, \\ (2 + \cos\pi h)/4 & \text{si } \omega = 1/2, \\ (1 + 2\cos\pi h)/5 & \text{si } \omega = 4/5. \end{cases}$$

Le suprémum de $\mu(h, \omega)$ par rapport à h sera noté $\mu^*(\omega)$ avec :

$$\mu^*(\omega) := \sup \left\{ \mu(h, \omega) : h < 1/4 \right\} \quad (3.17)$$

La forme(3.16) donne :

$$\mu^*(\omega) = \max \left\{ \begin{array}{l} |1 - \omega/2|, |1 - 2\omega| \\ \left. \begin{array}{ll} 1 - \omega/2 & \text{si } \omega \leq 4/5 \\ 1 - 2\omega & \text{si } \omega > 4/5 \end{array} \right\} \end{array} \right\}$$

Cette fonction rest représentée dans la figure ci-dessous :

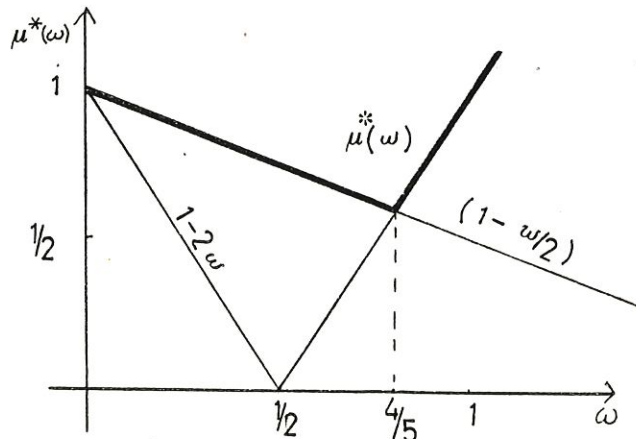


fig. 3

Nous obtenons en particulier

$$\mu^*(\omega) = \begin{cases} 1 & \text{si } \omega = 1 \\ 3/4 & \text{si } \omega = 1/2 \\ 3/5 & \text{si } \omega = 4/5 \end{cases}$$

La valeur optimale de $\mu^*(\omega)$ est donnée par $\omega = 4/5$ (cfr figure 3)

2- Correction par grille grossière

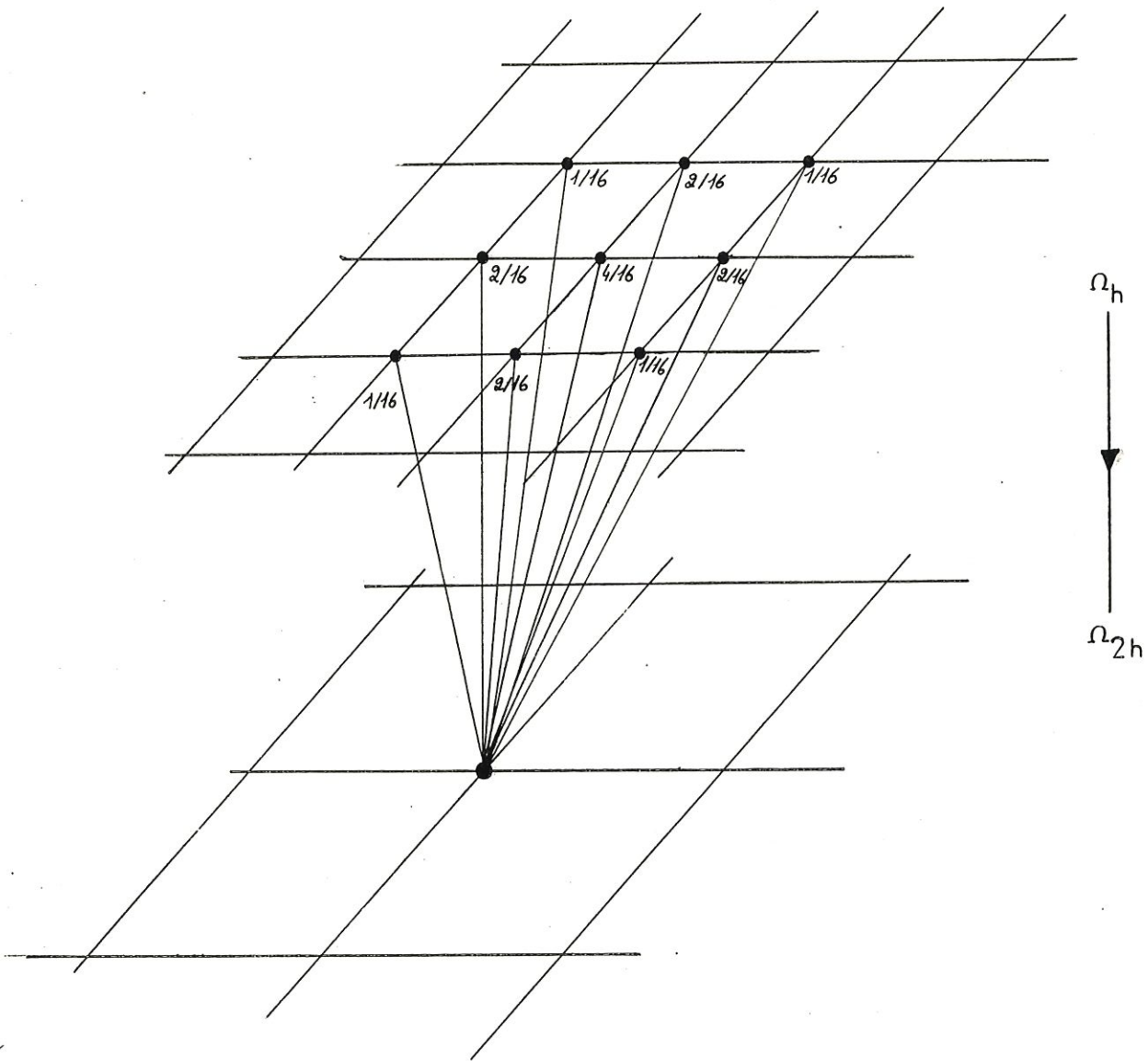
Rappelons que le passage de l'équation du défaut $L_h v_h^j = \bar{d}_h^j$ sur Ω_h à l'équation $L_H v_H^j = \bar{d}_H^j$ sur Ω_H nécessite deux opérateurs de transferts : un opérateur de restriction I_h^H qui envoie le défaut \bar{d}_h^j sur sa restriction \bar{d}_H^j et un opérateur d'interpolation I_H^h qui envoie la correction v_H^j sur sa prolongée v_h^j

L'opérateur de restriction : I_h^H

Nous utiliserons pour opérateur de restriction l'opérateur d'interpolation pondérée, en notation moléculaire :

$$I_h^{2h} := \frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Ce qui est encore illustré dans la figure suivante :



L'opérateur d'interpolation : I_H^h

Nous choisissons comme opérateur de prolongement l'opérateur d'interpolation bilinéaire, c'est-à-dire en notation moléculaire :

$$I_{2h}^h := \frac{1}{4} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 1 \\ 1 & 2 & 1 \end{bmatrix}$$

3- Définition de l'opérateur deux-grilles

Description d'une itération deux-grilles :

Pour résoudre l'équation de Poisson discrétisée sur le carré unité, nous considérons l'algorithme décrit page 45. La $j^{\text{ième}}$ itération calculant u_h^{j+1} à partir de u_h^j se décompose comme suit :

lissage : calcul de \bar{u}_h^j en appliquant ν_1 pas de la méthode de Jacobi avec un facteur de relaxation ω à u_h^j :

$$\bar{u}_h^j = \text{RELAX}^{\nu_1} (u_h^j, L_h, f_h; \omega)$$

correction par grille grossière :

- calcul du défaut

$$\bar{d}_h^j := f_h - L_h \bar{u}_h^j$$

- restriction du défaut

$$\bar{d}_{2h}^j := I_h^{2h} \bar{d}_h^j$$

en utilisant pour I_h^{2h} l'opérateur d'interpolation pondérée

$$I_h^{2h} = \frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

- calcul de \hat{v}_{2h}^j solution exacte de l'équation du défaut sur Ω_{2h} (ici L_{2h} est défini de manière analogue à L_h)

$$L_{2h} \hat{v}_{2h}^j = \bar{d}_{2h}^j$$

- interpolation de la correction \hat{v}_{2h}^j en utilisant l'opérateur d'interpolation bilinéaire I_{2h}^h ,

$$I_{2h}^h := \frac{1}{4} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

- calcul de l'approximation corrigée sur Ω_h :

$$\bar{u}_h^j + \hat{v}_h^j$$

lissage : calcul de u_h^{j+1} en appliquant ν_2 pas de la méthode de Jacobi avec un facteur de relaxation ω à $\bar{u}_h^j + \hat{v}_h^j$:

$$\bar{u}_h^{j+1} := \text{RELAX}^{\nu_2} (\bar{u}_h^j + \hat{v}_h^j, L_h, f_h; \omega)$$

Rayon spectral de l'opérateur deux-grilles

Nous considérons l'opérateur deux-grilles

$$\begin{aligned} M_h^{2h} &= S^{\nu_2}(\omega) K_h^{2h} S^{\nu_1}(\omega) \\ &= S^{\nu_2}(\omega) (I_h - I_{2h}^h L_{2h}^{-1} I_h^{2h} L_h) S^{\nu_1}(\omega) \end{aligned}$$

dont les composantes ont été définies ci-dessus.

Comme dans le cas unidimensionnel, nous allons mettre en évidence \hat{K}_h^{2h} équivalent orthogonal de K_h^{2h} . Où \hat{K}_h^{2h} est diagonale par bloc, composée de blocs $\hat{K}_{h,n}^{2h}$ de dimension 4×4 (au plus).

Pour cela, nous allons montrer que les espaces $E_{h,n}$ définis comme :

$$\begin{aligned} E_{h,n} &:= \text{span} \left\{ \varphi^{n_1, n_2}, \varphi^{N_h - n_1, N_h - n_2}, \varphi^{n_1, n_2 - \varphi^{n_1, N_h - n_2}} \right\} \\ & \quad (|n_1| \leq N_h/2) \end{aligned}$$

sont laissés invariants par l'opérateur K_h^{2h} .

Pour la description des matrices $\hat{K}_{h,n}^{2h}$ nous utiliserons aussi une base de vecteurs propres de L_{2h} donnée par :

$$\begin{aligned} \phi^n(x) &:= 2 \sin(n_1 \pi x_1) \sin(n_2 \pi x_2) \\ \text{pour } x \in \Omega_{2h} ; |n| \leq N_h/2 - 1 \end{aligned}$$

Sur la grille grossière Ω_{2h} , nous avons les égalités :

$$\begin{aligned} \phi^{n_1, n_2}(x) = \varphi^{n_1, n_2}(x) = \varphi^{N_h - n_1, N_h - n_2}(x) = -\varphi^{N_h - n_1, n_2}(x) = -\varphi^{n_1, N_h - n_2}(x) \\ \text{pour } x \in \Omega_{2h} ; |n| \leq N_h/2 - 1 \end{aligned}$$

Pour $n_1 = N_h/2$ et/ou $n_2 = N_h/2$, les espaces $E_{h,n}$ sont respectivement de dimension 1 ou 2 et leurs vecteurs propres de base coïncident sur Ω_{2h} avec la fonction de grille nulle.

Les opérateurs composant K_h^{2h} possèdent les propriétés :

$$\begin{aligned} \underline{L_h} : E_{h,n} \longrightarrow E_{h,n} & \quad \text{: car les vecteurs générant} \\ & \quad E_{h,n} \text{ sont aussi vecteurs propres} \\ & \quad \text{de } L_h . \end{aligned}$$

$$\underline{I_h^{2h}} : E_{h,n} \longrightarrow \text{span} \{ \phi^n \} \text{ pour } n \leq N_h/2 - 1$$

$$I_h^{2h} \begin{pmatrix} \varphi^{n_1, n_2} \\ \varphi^{N_h - n_1, N_h - n_2} \\ -\varphi^{N_h - n_1, n_2} \\ -\varphi^{n_1, N_h - n_2} \end{pmatrix} = \begin{pmatrix} (1-\xi)(1-\eta) \\ \xi \eta \\ \xi(1-\eta) \\ (1-\xi)\eta \end{pmatrix} \phi^{n_1, n_2}$$

$$\text{pour } |n| \leq N_h/2 - 1 \text{ et où } \begin{cases} \xi = \sin^2(n_1 \pi / 2N_h), \\ \eta = \sin^2(n_2 \pi / 2N_h) \end{cases}$$

$$I_h^{2h} \phi^{n_1, n_2} = 0 \quad \text{pour } n_1 = N_h/2 \text{ et/ou } n_2 = N_h/2$$

$$\underline{L_{2h}} : \text{span}\{ \phi^n \} \longrightarrow \text{span}\{ \phi^n \} \quad \text{: car les vecteurs } \phi^n \text{ sont} \\ \text{vecteurs propres de } L_{2h} .$$

$$\underline{I_{2h}^h} : \text{span}\{\phi^n\} \longrightarrow E_{h,n} \text{ pour } |n| \leq N_h/2 - 1$$

$$\begin{aligned} I_{2h}^h \phi^{n_1, n_2} &= (1-\xi)(1-\eta) \phi^{n_1, n_2} + \xi\eta \phi^{N_h-n_1, N_h-n_2} \\ &= \xi(1-\eta) \phi^{N_h-n_1, n_2} + (1-\xi)\eta \phi^{n_1, N_h-n_2} \end{aligned}$$

Nous pouvons ainsi décrire les blocs composant la matrice \hat{K}_h^{2h} :

$$\hat{K}_{h,n}^{2h} = \begin{cases} \cdot I - \frac{1}{\Lambda} (b_i c_j)_{4,4} & |n| < N_h/2 \\ \cdot \text{matrice d'identité de} \\ \text{dimension 2} & \text{si } n_1 \text{ ou } n_2 = N_h/2 \\ \cdot \text{matrice d'identité de} \\ \text{dimension 1} & \text{si } n_1 = n_2 = N_h/2 \end{cases}$$

où $\Lambda = \xi(1-\xi) + \eta(1-\eta)$ et

$$\begin{aligned} b_1 &= (1-\xi)(1-\eta), & b_2 &= \xi\eta, & b_3 &= \xi(1-\eta), & b_4 &= (1-\xi)\eta \\ c_1 &= b_1(\xi+\eta), & c_2 &= b_2(2-\xi-\eta), & c_3 &= b_3(1-\xi+\eta), & c_4 &= b_4(1+\xi-\eta) \end{aligned}$$

Nous avons aussi :

$$\underline{S_h(\omega)} : E_{h,n} \longrightarrow E_{h,n} \text{ car nous avons vu que les vecteurs } \phi^n \text{ sont aussi vecteurs propres de } S_h(\omega).$$

Par (3.14), nous pouvons décrire les blocs composant la matrice diagonale

$\hat{S}_h(\omega)$, équivalent orthogonal de $S_h(\omega)$ comme :

$$\hat{S}_{h,n}(\omega) = \begin{cases} \begin{bmatrix} (1-\omega(\xi+\eta)) & & & \\ & (1-\omega(2-\xi-\eta)) & & \\ & & (1-\omega(1-\xi+\eta)) & \\ & & & (1-\omega(1+\xi-\eta)) \end{bmatrix}_{4,4} & \text{si } |n| < N_h/2 \\ \begin{bmatrix} 1-\omega(\xi+\eta) & \\ & 1-\omega(2-\xi-\eta) \end{bmatrix}_{2,2} & \text{si } n_1 \text{ ou } n_2 = N_h/2 \\ \begin{bmatrix} 1-\omega(\xi+\eta) \end{bmatrix}_{1,1} & \text{si } n_1 = n_2 = N_h/2 \end{cases}$$

Les espaces $E_{h,n}$ ($|n| \leq N_h/2$) étant laissés invariant par les opérateurs K_h^{2h} et $S_h(\omega)$, ils le sont aussi par M_h^{2h} .

Nous obtenons la matrice diagonale par bloc \hat{M}_h^{2h} , équivalent orthogonal de M_h^{2h} . Avec :

$$\hat{M}_{h,n}^{2h} = \hat{M}_{h,n}^{2h}(\nu_1, \nu_2; \omega) = \hat{S}_{h,n}^{\nu_2}(\omega) \hat{K}_{h,n}^{2h} \hat{S}_{h,n}^{\nu_1}(\omega)$$

Nous avons donc réduit le calcul du rayon spectral de la matrice d'itération de la méthode deux-grilles ($\rho(M_h^{2h})$) au calcul des rayons spectraux de matrices au plus de dimension 4 :

$$\rho(M_h^{2h}) = \max \left\{ \rho(\hat{M}_{h,n}^{2h}) : |n| \leq N_h/2 \right\}$$

Comme dans le cas unidimensionnel, ce rayon spectral dépend de

$\nu = \nu_1 + \nu_2$, du paramètre ω et du pas de discrétisation h .

Il sera encore noté :

$$\rho(h, \nu; \omega) := \rho(M_h^{2h}(\nu_1, \nu_2; \omega))$$

Nous introduisons encore le facteur $\rho^*(\nu, \omega)$ défini comme :

$$\rho^*(\nu, \omega) = \sup \left\{ \rho(h, \nu; \omega) : h \leq 1/4 \right\}$$

Dans le tableau (I) proposé par Trottenberg [11], nous trouvons quelques valeurs de ρ^* donné comme fonction de ν et de deux paramètres de relaxation fixés $\omega = 0,5$ et $\omega = 0,8$.

ν	$\omega = 0,5$	$\omega = 0,8$
1	0.750	0.600
2	0.563	0.360
3	0.422	0.216
4	0.316	0.137
5	0.237	0.113

Tableau (I) : comparaison des facteurs de convergence ρ^* pour différentes valeurs de ν et de ω .

Nous constatons que les facteurs de convergence ρ^* diminuent avec l'augmentation des valeurs de ν . Observons aussi que le facteur de relaxation $\omega = 0.8$ donne de meilleurs facteurs de convergence que la valeur $\omega = 0.5$ (ce qui avait déjà été suggéré par la valeur des facteurs de lissage respectifs).

Donnons, pour terminer ce chapitre, quelques valeurs de $\rho(h, \nu; \omega)$ comme une fonction de h dans le tableau (II) (Trottenberg [11])

h	$\omega = 0.5$				$\omega = 0.8$			
	$\nu = 1$	$\nu = 2$	$\nu = 3$	$\nu = 4$	$\nu = 1$	$\nu = 2$	$\nu = 3$	$\nu = 4$
1/4	0.667	0.458	0.310	0.217	0.483	0.233	0.171	0.130
1/8	0.731	0.534	0.391	0.285	0.570	0.324	0.185	0.130
1/16	0.745	0.555	0.414	0.308	0.592	0.351	0.208	0.135
1/32	0.749	0.561	0.420	0.314	0.598	0.358	0.214	0.137
1/64	0.750	0.562	0.421	0.316	0.600	0.359	0.215	0.137
1/128	0.750	0.562	0.422	0.316	0.600	0.360	0.216	0.137
$\rho^*(\nu; \omega)$	0.750	0.563	0.422	0.316	0.600	0.360	0.216	0.137

Tableau (II) : $(h, \nu; \omega)$ facteur de convergence de la méthode deux-grilles.

Ce tableau montre que le facteur ρ tend vers ρ^* assez rapidement. Donc, dans les cas considérés, ρ^* contient l'information essentielle sur la convergence de la méthode deux-grilles.

3.4. APPLICATIONS NUMERIQUES.

3.4.1. Résolution de l'équation de Poisson

La méthode multigrille appliquée à la résolution de l'équation de Poisson sur le carré unité a été programmée par FOERSTER et WHITSCH [1] et [11] conformément à l'organigramme de la page 45

Nous allons décrire les principales options du programme à savoir :

- le choix d'une première approximation
- la méthode de lissage
- le choix des grilles grossières et leur nombre
- la méthode de résolution directe utilisée sur la grille la plus fine
- les opérateurs de transfert (restriction et prolongement)

- le choix d'une première approximation

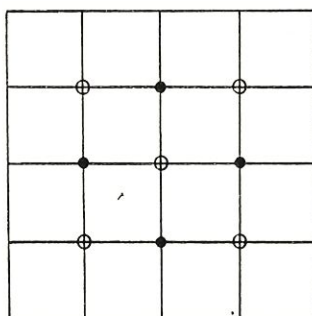
On adopte sur ce point une stratégie très générale :

$$u_h = 0 \text{ en tout point de } \Omega_h$$

$$u_h = f_h \text{ en tout point de } \Gamma_h$$

- méthode de lissage

Notre choix se porte sur un opérateur de lissage "Red-Black" c'est-à-dire où un lissage complet se décompose en deux étapes correspondant respectivement à des opérations sur le sous ensemble des points rouges puis sur les points noirs de la grille Ω_h (voir figure)



○ point rouge

● point noir

Dans la première étape, tous les points rouges de la grille sont traités par la méthode de Jacobi. Les valeurs calculées en ces points n'interfèrent pas et ne font intervenir que de l'information provenant des points noirs.

Cette étape terminée, on balaye alors de manière analogue tous les points noirs de la grille. Cette méthode de lissage est décrite par deux opérateurs partiels notés S_h^{red} et S_h^{black} respectivement et définis comme :

lissage partiel sur les points rouges

$$(S_h^{\text{red}} \zeta)(x) = \begin{cases} (I_h - \frac{h^2}{4} L_h) \zeta(x) & \text{si } x \text{ est un point rouge} \\ \zeta(x) & \text{si } x \text{ est un point noir} \end{cases}$$

lissage partiel sur les points noirs

$$(S_h^{\text{black}} \zeta)(x) = \begin{cases} \zeta(x) & \text{si } x \text{ est un point rouge} \\ (I_h - \frac{h^2}{4} L_h) \zeta(x) & \text{si } x \text{ est un point noir} \end{cases}$$

- choix des grilles grossières

Le choix d'une grille Ω_H , moins fine que la grille initiale Ω_h , est déterminé par H . Ici, nous choisirons H de façon standard en posant $H = 2h$

Dans le programme, la grille la plus fine correspond à une valeur de $h = 1/64$, le nombre de grilles possibles est ainsi limité à 6

$$(h = \frac{1}{64}, \frac{1}{32}, \frac{1}{16}, \frac{1}{8}, \frac{1}{4}, \frac{1}{2})$$

- méthode de résolution directe utilisée sur la grille la plus fine

Observons tout d'abord les dimensions des différentes grilles :

$$\# \Omega_5 : (63)^2 = 3969$$

$$\# \Omega_4 : (31)^2 = 961$$

$$\# \Omega_3 : (15)^2 = 225$$

$$\# \Omega_2 : (7)^2 = 49$$

$$\# \Omega : (3)^2 = 9$$

$$\# \Omega_0 : (1)^2 = 1$$

On constate en particulier que la grille la plus grossière ne contient qu'un seul point intérieur, rendant la méthode de résolution d'un système linéaire évidente.

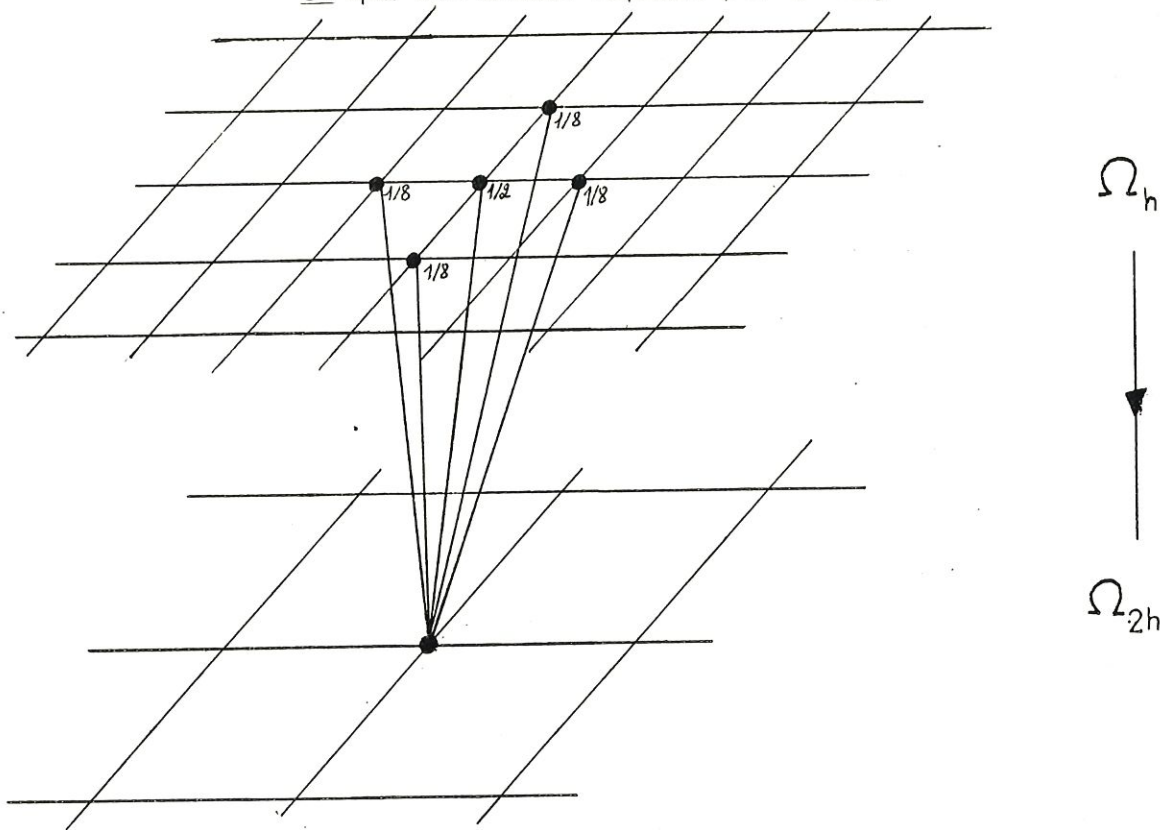
- les opérateurs de transfert

. l'opérateur de restriction : I_h^{2h}

Nous choisissons l'opérateur de restriction défini comme :

$$I_h^{2h} := \frac{1}{8} \begin{bmatrix} & & 1 & & \\ & & & & \\ 1 & & 4 & & 1 \\ & & & & \\ & & 1 & & \end{bmatrix}$$

ce qui est encore exprimé par la figure ci-dessous



Remarquons qu'après un pas de lissage, le défaut est nul sur tous les points noirs. Cela résulte immédiatement de la définition de l'opérateur de Laplace discret et de la stratégie "Red-Black".

La procédure de transfert de la grille fine vers la grille grossière devient alors extrêmement simple en particulier, pour le choix que nous avons fait de l'opérateur de restriction. Le transfert s'effectue en multipliant les points appartenant à l'intersection de la grille fine et de la grille grossière par un facteur 0.5 : Cet opérateur est appelé "demi-injection".

. l'opérateur de prolongement : I_{2h}^h

Nous choisissons un opérateur d'interpolation bilinéaire que nous avons déjà évoqué dans ce travail :

$$I_{2h}^h = \frac{1}{4} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Nous pourrions cependant économiser des calculs si l'on observe la chose suivante. Après avoir effectué le transfert grille grossière vers grille fine, nous allons lisser la nouvelle approximation à l'aide de l'opérateur "Red-Black".

Le lissage sur les points rouges donne :

$$\begin{aligned} S_h^{\text{red}} \zeta(x) &= \left(I_h - \frac{h^2}{4} L_h \right) \zeta(x) \\ &= \left(\begin{bmatrix} 0 & & \\ 0 & 1 & 0 \\ & & 0 \end{bmatrix} - \begin{bmatrix} & -\frac{1}{4} & \\ -\frac{1}{4} & 1 & -\frac{1}{4} \\ & & -\frac{1}{4} \end{bmatrix} \right) \zeta(x) \\ &= \begin{bmatrix} & \frac{1}{4} & \\ \frac{1}{4} & 0 & \frac{1}{4} \\ & \frac{1}{4} & \end{bmatrix} \zeta(x) \end{aligned}$$

Ce qui signifie que le calcul des nouvelles valeurs de l'approximation sur les points rouges ne dépend que des valeurs sur les points noirs adjacents. Il sera donc inutile de procéder à l'interpolation des valeurs de la correction sur les points rouges de la grille fine.

Le code du programme FORTRAN écrit par FOERSTER et WITSCH [11] a été modifié pour permettre la résolution d'équations elliptiques plus générales. Nous allons traiter cette question dans le paragraphe suivant.

3.4.2. Résolution d'une équation elliptique non linéaire

Tout au long de ce travail, nous avons étudié des problèmes linéaires. Nous allons montrer qu'il est possible d'étendre le champ d'application de la méthode multigrille à des problèmes non linéaires.

Soit le problème :

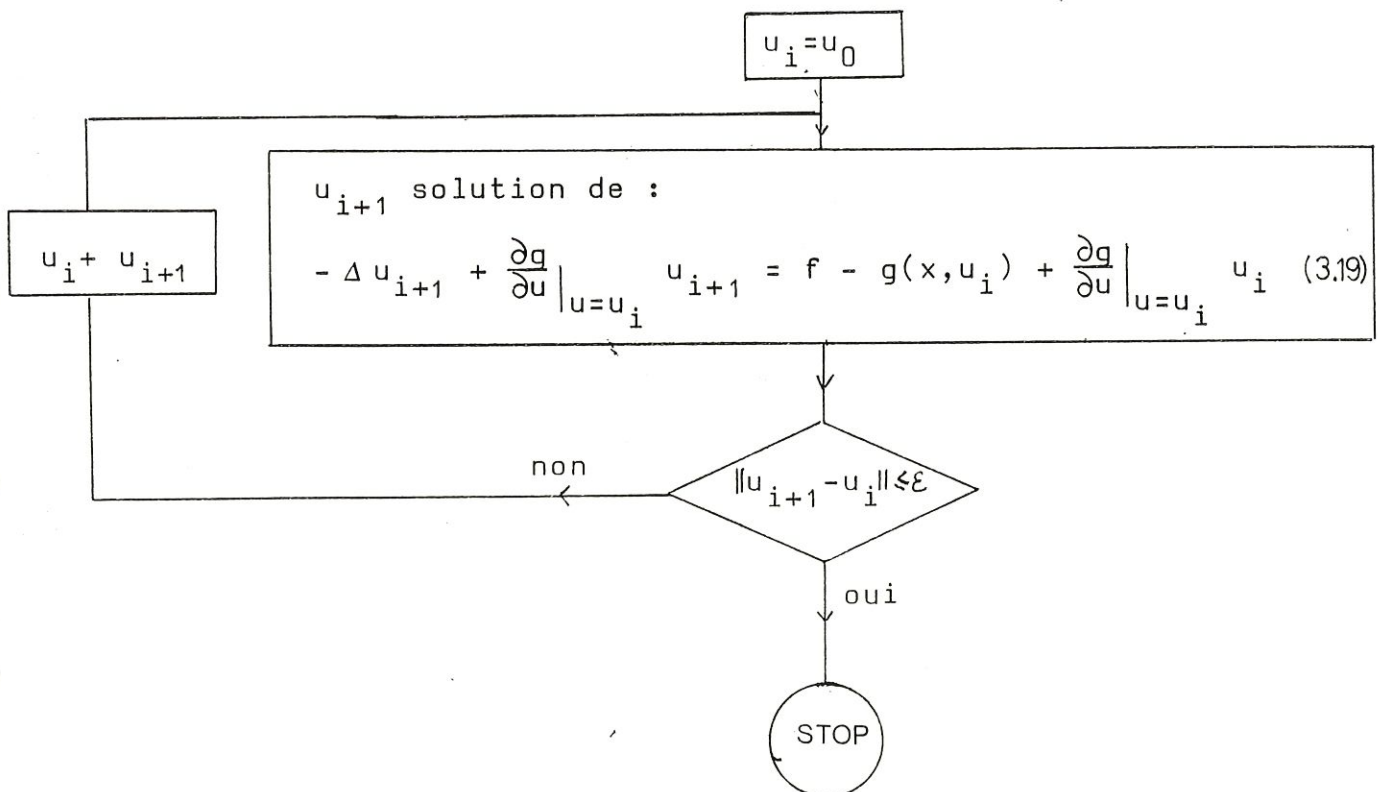
$$\begin{cases} \mathbb{L}^{\Omega} u(x) = f^{\Omega}(x) & x \in \Omega \\ \mathbb{L}^{\Gamma} u(x) = f^{\Gamma}(x) & x \in \Gamma := \partial\Omega \end{cases} \quad (3.18)$$

où

\mathbb{L} est cette fois un opérateur de la forme $-\Delta u + g^{\Omega}(x, u)$;

g étant une fonction suffisamment régulière de x et de u .

Nous allons résoudre cette équation par la méthode de Newton appliquée au système linéarisé (cfr schéma ci-dessous)



A chaque étape (3.19), nous pouvons utiliser la méthode multigrille pour résoudre le système linéaire correspondant.

Remarquons que le système linéarisé est de la forme

$$-\Delta u + c u = f$$

donc un peu plus général que l'équation de Poisson traitée au § 3.4.1. L'extension de la méthode à ce type d'équation ne pose aucun problème.

Nous avons étendu le programme de calcul linéaire [1], [11] à la résolution du système (3.18). Le code FORTRAN de ce programme est repris dans l'annexe 3.

Pour illustrer cette extension de la méthode au cas non linéaire nous allons résoudre le système :

$$\begin{cases} -\Delta u + \lambda u + u^3 = f & \text{dans } \Omega \\ u = 0 & \text{sur } \Gamma = \partial\Omega \end{cases} \quad (3.20)$$

L'équation (3.20) a été étudiée par R. TEMAN [12] qui a démontré l'existence de l'unicité de la solution dans des conditions très générales.

Nous nous sommes placés dans les conditions suivantes :

$$\Omega = (0,1) \times (0,1)$$

$$f = -200 x_1 (x_1 - 1) - 200 x_2 (x_2 - 1) + x_1 (x_1 - 1) x_2 (x_2 - 1) + 10^6 x_1^3 (x_1 - 1)^3 x_2^3 (x_2 - 1)^3$$

$$\lambda = 100$$

On peut montrer que la solution exacte est :

$$u = 100 x_1 x_2 (x_1 - 1)(x_2 - 1) .$$

Quatre itérations de la méthode de Newton suffisent pour atteindre une précision de l'ordre du millionième (voir annexe 3)

Aussi, en couplant les avantages de la méthode multigrille et ceux de la méthode de Newton, on peut s'attendre à d'excellents résultats dans la résolution de problèmes non linéaires.

4. C O N C L U S I O N S

Dans ce travail, nous avons présenté la méthode multigrille, une méthode de résolution de systèmes linéaires provenant d'équations aux dérivées partielles elliptiques linéaires.

Ces systèmes présentent les caractéristiques suivantes :

- une structure régulière
- une grande taille
- un grand nombre de composantes nulles.

Cette méthode est en fait la composée de méthodes classiques (Jacobi, Gauss-Seidel relaxées ou non, correction par grille grossière (C.G.C.), ...) dont elle a su tirer parti des propriétés les plus adéquates (lissage pour Jacobi ou Gauss-Seidel, écrêtage des composantes hautes fréquences de la solution pour C.G.C.).

Nous avons montré que la méthode multigrille était simple dans sa conception et que les paramètres qui la caractérisaient étaient faciles à quantifier.

Elle est en outre très flexible dans la mesure où ses composantes peuvent être choisies au mieux du problème considéré.

Sa programmation ne pose aucun problème et s'est avérée facile à modifier.

Enfin, les résultats comparatifs avec d'autres méthodes sont des plus encourageant et confirment les résultats théoriques. En particulier, le nombre de calculs qu'elle nécessite est proportionnel au nombre de points de la grille avec une constante de proportionnalité très raisonnable. Cette dernière propriété ajoutée au fait que la méthode multigrille peut, du moins formellement, être adaptée à des problèmes plus complexes en fait un outil particulièrement prometteur. C'est pourquoi les applications qui en font usage se multiplient aujourd'hui, tant dans le domaine des équations elliptiques que dans celui des équations paraboliques, linéaires ou non.

*
* *
*

ANNEXE 1.

Démonstration de la proposition 1 établissant l'équivalence entre le problème de Poisson de (2.3) et la formulation variationnelle (2.4)

Démontrons la condition nécessaire :

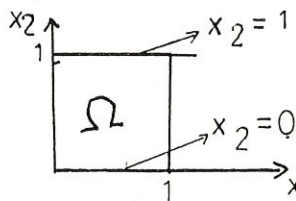
L'espace des fonctions utilisées sera :

$$V = H_0^1(\Omega) = \left\{ v \in L^2(\Omega), \frac{\partial v}{\partial x_i} \in L^2(\Omega), i=1, \dots, n; v|_{\Gamma} = 0 \right\}$$

Prenons $h \in H_0^1(\Omega)$, arbitraire, il vient

$$\begin{aligned} - \int \int_{\Omega} h(x) \cdot \Delta u(x) \, dx &= \int \int_{\Omega} h(x) \cdot f(x) \, dx \\ - \int \int_{\Omega} h(x_1, x_2) \left(\frac{\partial^2 u}{\partial x_1^2}(x_1, x_2) + \frac{\partial^2 u}{\partial x_2^2}(x_1, x_2) \right) dx_1 \, dx_2 \\ &= \int \int_{\Omega} h(x_1, x_2) f(x_1, x_2) \, dx_1 \, dx_2 \quad (*) \end{aligned}$$

Comme Ω a la forme particulière :



nous obtenons

$$\int \int_{\Omega} h(x_1, x_2) \frac{\partial^2 u}{\partial x_2^2}(x_1, x_2) \, dx_1 \, dx_2 = \int_0^1 \left\{ \int_0^1 h(x_1, x_2) \frac{\partial^2 u}{\partial x_2^2}(x_1, x_2) \, dx_2 \right\} dx_1$$

Après intégration par parties, il vient

$$\int_0^1 \left\{ h(x_1, x_2) \frac{\partial u}{\partial x_2}(x_1, x_2) \Big|_{x_2=0}^{x_2=1} - \int_0^1 \frac{\partial h}{\partial x_2}(x_1, x_2) \frac{\partial u}{\partial x_2}(x_1, x_2) \, dx_2 \right\} dx_1$$

Le premier terme s'annule puisque $h \in H_0^1(\Omega)$

$$= - \int_0^1 \left\{ \int_0^1 \frac{\partial h}{\partial x_2}(x_1, x_2) \frac{\partial u}{\partial x_2}(x_1, x_2) dx_2 \right\} dx_1$$

$$= - \iint_{\Omega} \frac{\partial h}{\partial x_2}(x_1, x_2) \frac{\partial u}{\partial x_2}(x_1, x_2) dx_1 dx_2$$

Ainsi (*) devient

$$\iint_{\Omega} \frac{\partial h}{\partial x_1}(x_1, x_2) \frac{\partial u}{\partial x_1}(x_1, x_2) + \frac{\partial h}{\partial x_2}(x_1, x_2) \frac{\partial u}{\partial x_2}(x_1, x_2) dx =$$

$$= \iint_{\Omega} f \cdot h dx$$

D'où $\iint_{\Omega} \nabla h \cdot \nabla u dx - \iint_{\Omega} f \cdot h dx = 0 \quad \forall h \in H_0^1(\Omega)$

Ajoutons le terme positif $\frac{1}{2} \iint_{\Omega} |\nabla h|^2 dx$, nous obtenons

$$\iint_{\Omega} \nabla h \cdot \nabla u dx - \iint_{\Omega} f \cdot h dx + \frac{1}{2} \iint_{\Omega} |\nabla h|^2 dx \geq 0 \quad \forall h \in H_0^1(\Omega)$$

ou encore $\iint_{\Omega} \frac{1}{2} \left[|\nabla u|^2 + |\nabla h|^2 + 2 \nabla u \cdot \nabla h \right] dx$ (**)

$$- \iint_{\Omega} f \cdot (h + u) dx - \frac{1}{2} \iint_{\Omega} |\nabla u|^2 dx + \iint_{\Omega} f \cdot v dx \geq 0 \quad \forall h \in H_0^1(\Omega)$$

Posons $J(v) = \frac{1}{2} \iint_{\Omega} |\nabla v|^2 dx - \iint_{\Omega} f \cdot v dx$

et $a(v, w) = \iint_{\Omega} \nabla v \cdot \nabla w dx$

La relation (**) devient

$$\frac{1}{2} \iint_{\Omega} |\nabla(u+h)|^2 dx - \iint_{\Omega} f \cdot (u+h) dx - J(u) \geq 0 \quad \forall h \in H_0^1(\Omega)$$

ou encore

$$J(u+h) - J(u) \geq 0 \quad \forall h \in H_0^1(\Omega)$$

D'où

$$J(u) \leq J(u+h) \quad \forall h \in H_0^1(\Omega)$$

et

$$J(u) = \inf_{v \in H_0^1(\Omega)} J(v)$$

Ce qui démontre la condition nécessaire de la proposition et nous permet d'établir entre le problème de Dirichlet et le cas général la correspondance :

Problème de Dirichlet

$$V = H_0^1(\Omega)$$

$$a(u, v) = \iint_{\Omega} \nabla u \cdot \nabla v \, dx$$

$$L(v) = \iint_{\Omega} f \cdot v \, dx$$

$$J(v) = \frac{1}{2} \iint_{\Omega} |\nabla v|^2 \, dx - \iint_{\Omega} f \cdot v \, dx$$

Cas général

V espace de Hilbert

a(u, v) forme bilinéaire continue
sur V

L(v) forme linéaire continue
sur V

$$J(v) = \frac{1}{2} a(v, v) - L(v)$$

Démontrons à présent la condition suffisante :

si u est solution du problème d'optimisation (2.4)

alors u est solution du problème (2.3)

$$\text{Partons de } J(v) = \frac{1}{2} \iint_{\Omega} |\nabla v|^2 \, dx - \iint_{\Omega} f \cdot v \, dx$$

$$\text{avec } a(v, w) = \iint_{\Omega} \nabla v \cdot \nabla w \, dx$$

$$\text{et } L(v) = \iint_{\Omega} f \cdot v \, dx$$

Il vient

$$J(u + \lambda v) = \frac{1}{2} (a(u, u) + \lambda^2 a(v, v) + 2\lambda a(u, v) - 2L(u) - 2\lambda L(v))$$

$$\text{où } u, v \in H_0^1(\Omega) \text{ et } \lambda \in \mathbb{R}$$

Et

$$\left. \frac{d}{d\lambda} J(u + \lambda v) \right|_{\lambda=0} = 0$$

Calculons

$$\begin{aligned} \left. \frac{d}{d\lambda} J(u+\lambda v) \right|_{\lambda=0} &= 2 a(v, v) + 2a(u, v) - 2L(v) \Big|_{\lambda=0} \\ &= 2a(u, v) - 2L(v) \end{aligned}$$

d'où

$$a(u, v) = L(v) \quad \forall v \in H_0^1(\Omega)$$

ou encore

$$\iint_{\Omega} \nabla u \cdot \nabla v \, dx = \iint_{\Omega} f \cdot v \, dx \quad \forall v \in H_0^1(\Omega)$$

Et par la formule de Green :

$$- \iint_{\Omega} \Delta u \cdot v \, dx + \iint_{\Gamma} v \cdot \frac{\partial u}{\partial n} \, dx = \iint_{\Omega} f \cdot v \, dx$$

le second terme s'annule puisque $v \in H_0^1(\Omega)$, et nous obtenons

$$\begin{aligned} - \iint_{\Omega} \Delta u \cdot v \, dx &= \iint_{\Omega} f \cdot v \, dx & \forall v \in H_0^1(\Omega) \\ \iint_{\Omega} (\nabla u + f) v \, dx &= 0 & \forall v \in H_0^1(\Omega) \end{aligned}$$

il vient

$$\nabla u + f = 0 \quad : \text{équation de Poisson}$$

ce qui démontre la proposition et nous donne la correspondance

Cas général

$$\begin{aligned} a(u, v) &= L(v) \\ \forall v \in V \end{aligned}$$

Problème de Dirichlet

$$\iint_{\Omega} \Delta u \cdot v \, dx = - \iint_{\Omega} f \cdot v \, dx \quad \forall v \in H_0^1(\Omega)$$

ANNEXE 2.

Programme de résolution d'un grand système linéaire, creux et non symétrique.

 LARGE SPARSE, UNSYMMETRIC SYSTEM OF LINEAR EQUATIONS SOLVER.

AUTHOR : JOHN E. KEY (ALABAMA UNIVERSITY-HUNTSVILLE ALABAMA)
 AUGUST 1972
 REVISED BY E. VERGISON (SOLVAY S.A.) SEPTEMBER 1981.

REFERENCE : COMPUTER PROGRAM FOR SOLUTION OF LARGE SPARSE
 UNSYMMETRIC SYSTEMS OF LINEAR EQUATIONS.
 JOHN E. KEY
 NTIS - AD-755 878 - AUGUST 1972

SUBROUTINES CALLED : SASMX, SBSMX, ... TO SISMX, CPUTIME (*)
 (*) CPUTIME IS A VAX 11/780 DEPENDENT
 ROUTINE THAT TELLS THE USER THE CPU TIME
 USED BY THE ALGORITHM; NEED TO BE ADAPTED
 ON ANOTHER COMPUTER.

CALLING SEQUENCE

 CALL SUSMX (NU,MR,MC,A,P,ICOL,ZTEST,ICC,IRC,JELE,JELIM,JCOL
 NPIV,TIME)

- NU = NUMBER OF ROWS IN SYSTEM OF EQUATIONS TO BE SOLVED
- MR = MAXIMUM NUMBER OF ROWS IN EQUATION SOLVER AND IN SYSTEM OF
 EQUATIONS TO BE SOLVED.
- MC = MAXIMUM NUMBER OF COLUMNS IN EQUATION SOLVER
- A = COEFFICIENT MATRIX IN COMPRESSED FORM.
- P = VECTOR TO BE SOLVED.
- ICOL = RECORDS POSITION OF NON-ZERO COEFF. IN SYSTEM OF EQUATIONS
 TO BE SOLVED
- ZTEST = TEST TO DECIDE WHEN ELEMENTS OTHER THEN PIVOTAL MUST BE SET
 EQUAL TO ZERO.
- ICC = NUMBER OF NON-ZERO COEFF. IN ANY GIVEN ROW OF A
- IRC = NUMBER OF NON-ZERO COEFFICIENTS IN ANY GIVEN COLUMN OF A
- JELE = NUMBER OF NON-ZERO COEFFICIENTS IN THE COEFFICIENT ARRAY A
- JELIM = NUMBER OF TERMS ELIMINATED BY THE ELIMINATION PROCESS
- JCOL = MAXIMUM NUMBER OF COLUMNS USED IN THE SOLUTION PROCESS
- NPIV = PARAMETER DETERMINING THE PIVOT ALGORITHM
- NPIV = 1 : GAUSS - JORDAN ELIMINATION ALGORITHM
- NPIV = 2 : GAUSS - JORDAN PARTIAL PIVOTING ALGORITHM

```

PAGE 2
0043 NPIV = 3 : GAUSS - JORDAN FULL PIVOTING ALGORITHM
0044 NPIV = 4 : MINIMUM ROW - MINIMUM COLUMN ALGORITHM
0045 NPIV = 5 : MINIMUM COLUMN - MINIMUM ROW ALGORITHM
0046 NPIV = 6 : MAXIMUM COLUMN - MINIMUM ROW ALGORITHM
0047 NPIV = 7 : MINIMUM ROW ENTRIES TIMES COLUMN ENTRIES ALGORITHM
0048 TIME = CPU TIME USED BY THE ALGORITHM
0049 A(I,1)= SOLUTION VECTORS RETURNED IN FIRST COLUMN OF A
0050
0051
0052
0053
0054
0055
0056
0057
0058
0059
0060
0061
0062
0063
0064
0065
0066
0067
0068
0069
0070
0071
0072
0073
0074
0075
0076
0077
0078
0079
0080
0081
0082
0083
0084

SUBROUTINE SUSMX (NU,MR,MC,A,P,ICOL,ZTEST,ICC,IRC,JELE,JELIM,JCOL
1, NPIV,TIME)
REAL *8 A,C,P,X,ZTEST,ATEST
INTEGER*4 PIVROW,PIVCOL,OPROW
COMMON /SINTG1/ PIVROW,PIVCOL,OPROW
COMMON /SINTG2/ IY
COMMON /SINTG3/ LKJ
DIMENSION IRC(MR),ICC(MR),A(MR,MC),ICOL(MR,MC),P(MR)
IMP = 0
TELL THE USER WHICH PIVOTAL STRATEGY HE IS USING.
GOTO (610,620,630,640,650,660,670), NPIV
WRITE(IMP,2100)
GOTO 680
WRITE(IMP,2200)
GOTO 680
WRITE(IMP,2300)
GOTO 680
WRITE(IMP,2400)
GOTO 680
WRITE(IMP,2500)
GOTO 680
WRITE(IMP,2600)
GOTO 680
WRITE(IMP,2700)
CONTINUE
TIME1 = 0
CALL CPU TIME (TIME1)
INITIALIZE ROW ENTRY COUNTER

```

```

AGE 3      0085      C
           0086      C
           0087      99
           0088      C
           0089      C
           0090      C
           0091
           0092
           0093
           0094
           0095
           0096
           0097
           0098
           0099
           0100
           0101
           0102
           0103
           0104
           0105
           0106
           0107
           0108
           0109
           0110
           0111
           0112
           0113
           0114
           0115
           0116
           0117
           0118
           0119
           0120
           0121
           0122
           0123
           0124
           0125
           0126

           DO 99 I=1,NU
           IRC(I) = 0

           COUNT ROW AND COLUMN ENTRIES

           JELE = 0
           DO 101 I=1,NU
           DO 102 J=1,MC
           IC = ICOL(I,J)
           IF(IC.EQ.0) GOTO 101
           JELE = JELE+1
           ICC(I) = J
           IRC(IC) = IRC(IC)+1
           CONTINUE
           CONTINUE
           JCOL = 0
           JELIM = 0
           DO 800 I=1,NU
           IF(ICC(I).GT.JCOL) JCOL=ICC(I)
           800

           MAIN LOOP

           DO 66 LKJ = 1,NU

           CHOICE OF THE PIVOT ALGORITHM.

           GOTO (710,720,730,740,750,760,770), NPIV

           CALL SCSTMX(IRC,ICC,ICOL,A,MR,MC,NU)
           GOTO 780
           CALL SDSMX(IRC,ICC,ICOL,A,MR,MC,NU)
           GOTO 780
           CALL SESMX(IRC,ICC,ICOL,A,MR,MC,NU)
           GOTO 780
           CALL SFSMX(IRC,ICC,ICOL,A,MR,MC,NU)
           GOTO 780
           CALL SGSMX(IRC,ICC,ICOL,A,MR,MC,NU)
           GOTO 780
           CALL SHSMX(IRC,ICC,ICOL,A,MR,MC,NU)
           GOTO 780
           CALL SISMX(IRC,ICC,ICOL,A,MR,MC,NU)
           GOTO 780

```



```

PAGE 4      0127      780      CONTINUE
            0128      C
            0129      C      NORMALIZE PIVROW
            0130      C
            0131      C      X = A(PIVROW,IY)
            0132      C      IC = ICC(PIVROW)
            0133      C      DO 5 J=1,IC
            0134      C      A(PIVROW,J) = A(PIVROW,J)/X
            0135      C      A(PIVROW,IY) = 1.D+00
            0136      C      P(PIVROW) = P(PIVROW)/X
            0137      C
            0138      C      SELECT ROWS THAT CAN BE OPERATED ON
            0139      C
            0140      C      SECOND LOOP
            0141      C
            0142      C      DO 106 I=1,NU
            0143      C      IF(IRC(PIVCOL).EQ.1) GOTO 107
            0144      C      IF(I.EQ.PIVROW) GOTO 106
            0145      C      IC = IABS(ICC(I))
            0146      C
            0147      C      INNER LOOP
            0148      C
            0149      C      DO 105 J=1,IC
            0150      C      IF(ICOL(I,J) - PIVCOL) 105,77,106
            0151      C
            0152      C      IF YOU CAN GET TO THIS POINT OPROW CONTAINS PIVOTAL ELEMENT
            0153      C
            0154      C      OPROW = I
            0155      C      JKOP = 1
            0156      C      JKPI = 1
            0157      C      C = -A(OPROW,J)
            0158      C      P(OPROW) = P(PIVROW)*C+P(OPROW)
            0159      C      CONTINUE
            0160      C      IF(ICOL(PIVROW,JKPI) .EQ.0) GOTO 106
            0161      C      IF(ICOL(OPROW,JKOP) .EQ.0) GOTO 8U
            0162      C      IF(ICOL(PIVROW,JKPI)-ICOL(OPROW,JKOP)) 80,81,82
            0163      C
            0164      C      OPROW DOES NOT CONTAIN THIS ELEMENT,ADD ELEMENT TO OPROW
            0165      C
            0166      C      ICC(I) = ICC(I)+1
            0167      C      IF(ICC(I).LE.0) ICC(I) = ICC(I)-2
            0168      C      II = IABS(ICC(I))

```

```

PAGE 5      0169      IF(II.GT.JCOL) JCOL=II
            0170      IF(II.LE.MC) GOTO 85
            0171      WRITE(IMP,2000)
            0172      RETURN
            0173      JKL = JKOP+1
            0174      IX = II-1
            0175      A(OPROW,II) = A(OPROW,IX)
            0176      ICOL(OPROW,II) = ICOL(OPROW,IX)
            0177      II = IX
            0178      IF(II.GE.JKL) GOTO 90
            0179      A(OPROW,JKOP) = A(PIVROW,JKPI)*C
            0180      ICOL(OPROW,JKOP) = ICOL(PIVROW,JKPI)
            0181      IX = ICOL(OPROW,JKOP)
            0182      IRC(IX) = IRC(IX)+1
            0183      GOTO 83
            0184
            0185      C
            0186      C
            0187      C
            0188      C
            0189      C
            0190      C
            0191      C
            0192      C
            0193      C
            0194      C
            0195      C
            0196      C
            0197      C
            0198      C
            0199      C
            0200      C
            0201      C
            0202      C
            0203      C
            0204      C
            0205      C
            0206      C
            0207      C
            0208      C
            0209      C
            0210      C

            85      JKL = JKOP+1
            90      IX = II-1
            81      IX = ICOL(OPROW,JKOP)
            11      IF(IX.EQ.PIVCOL) GOTO 11
            11      X = A(PIVROW,JKPI)*C+A(OPROW,JKOP)
            11      A(OPROW,JKOP) = X
            11      TEST OPROW TO SEE IF ANY ELEMENTS WERE ELIMINATED OTHER THAN
            11      THOSE IN THE PIVOTAL COLUMN
            11      ATEST = DABS(X)-ZTEST
            11      IF(ATEST.GT.0.) GOTO 83
            11      IRC(IX) = IRC(IX)-1
            11      JELIM = JELIM+1
            11      ICC(OPROW) = ICC(OPROW)-1
            11      IF(ICC(OPROW)) 140,141,142
            141      CALL SASMX(IRC,ICC,ICOL,A,MR,MC,NU)
            140      CONTINUE
            142      ICC(OPROW) = ICC(OPROW)+2
            142      IX = IABS(ICC(OPROW))
            142      DO 131 NK = JKOP,IX
            142      A(I,NK) = A(I,NK+1)
            131      ICOL(I,NK) = ICOL(I,NK+1)
            131      IX = IX+1
            131      ICOL(I,IX) = 0
            131      JKPI = JKPI+1

```



```

PAGE 1
0001 SUBROUTINE SBSMX(IDUM,NR,NC,MR,MC,A,ICOL)
0002 PRINTING OF THE COEFFICIENT MATRIX (A) AND THE COLUMN INDEX
0003 MATRIX (ICOL) IN CASE OF SINGULARITIES.
0004
0005 REAL*8 A
0006 DIMENSION A(MR,MC),ICOL(MR,MC)
0007 IPRINT = 12
0008 IF(IDUM.NE.1) IPRINT = 25
0009 IPR = IPRINT-1
0010 DO 35 K=1,NC,IPRINT
0011 MAX = K+IPR
0012 IF(MAX.GT.NC) MAX=NC
0013 IF(K.NE.1) WRITE(6,103)
0014 IF(IDUM.EQ.1) GOTO 45
0015 WRITE(6,101) (I,I=K,MAX)
0016 DO 20 J=1,NR
0017 WRITE(6,104) J,(ICOL(J,I),I=K,MAX)
0018 GOTO 35
0019 WRITE(6,102) (I,I=K,MAX)
0020 DO 40 J=1,NR
0021 WRITE(6,105) J,(A(J,I),I=K,MAX)
0022 CONTINUE
0023 RETURN
0024
0025 PRINTING FORMATS
0026
0027 FORMAT(6X,25I4)
0028 FORMAT(6X,12I10)
0029 FORMAT('1')
0030 FORMAT(' ',15,25I4)
0031 FORMAT(' ',15,12G10.3)
0032 END

```

0001

0002

0003

0004

0005

0006

0007

0008

0009

0010

0011

0012

0013

0014

0015

0016

0017

0018

0019

0020

0021

0022

0023

0024

0025

0026

0027

0028

0029

0030

0031

0032

20

45

40

35

C

C

C

101

102

103

104

105

```

0001 SUBROUTINE SCSTMX(IRC,ICC,ICOL,A,MR,MC,NU)
0002 GAUSS ELIMINATION ALGORITHM
0003 REAL*8 A
0004 INTEGER*4 PIVROW,PIVCOL,OPROW
0005 COMMON /SINTG1/ PIVROW,PIVCOL,OPROW
0006 COMMON /SINTG2/ IY
0007 COMMON /SINTG3/ LKJ
0008 DIMENSION IRC(MR),ICC(MR),ICOL(MR,MC),A(MR,MC)
0009 PIVROW = LKJ
0010 PIVCOL = LKJ
0011 IY = 1
0012
0013
0014
0015
0016

```

C

C

C

C

```

PAGE 1
0001 SUBROUTINE SDSMX(IRC,ICC,ICOL,A,MR,MC,NU)
0002 C
0003 C
0004 GAUSS JORDAN - PARTIAL PIVOTING ALGORITHM
0005 REAL*8 A,AA,ATEST
0006 INTEGER*4 PIVROW,PIVCOL,OPROW
0007 COMMON /SINTG1/ PIVROW,PIVCOL,OPROW
0008 COMMON /SINTG2/ IY
0009 COMMON /SINTG3/ LKJ
0010 DIMENSION IRC(MR),ICC(MR),ICOL(MR,MC),A(MR,MC)
0011 PIVCOL = LKJ
0012 ATEST = 0.D+00
0013 DO 712 I=1,NU
0014 IC = ICC(I)
0015 IF(IC.LE.0) GOTO 712
0016 II = ICOL(I,1)
0017 IF(II.GT.PIVCOL) GOTO 712
0018 AA = DABS(A(I,1))
0019 IF(AA.LE.ATEST) GOTO 712
0020 ATEST = AA
0021 PIVROW = I
0022 CONTINUE
0023 IY = 1
0024 C
0025 C
0026 END GAUSS JORDAN-PARTIAL PIVOTING
RETURN
END
712
C
C

```

PAGE 1 SUBROUTINE SESMX(IRC,ICC,ICOL,AMR,MC,NU)

C GAUSS JORDAN FULL PIVOTING ALGORITHM

REAL*8 A,AA,ATEST

INTEGER*4 PIVROW,PIVROW,PIVROW,PIVROW

COMMON /SINTG1/ PIVROW,PIVROW,PIVROW

COMMON /SINTG2/ IY

DIMENSION IRC(MR),ICC(MR),ICOL(MR,MC),A(MR,MC)

ATEST = 0.D+00

DO 710 I=1,NU

IC = ICC(I)

IF(IC.LE.0) GOTO 710

DO 711 J=1,IC

II = ICOL(I,J)

IR = IRC(II)

IF(IR.LE.0) GOTO 711

AA = DABS(A(I,J))

IF(AA.LE.ATEST) GOTO 711

ATEST = AA

PIVROW = I

PIVROW = II

IY = J

CONTINUE

CONTINUE

END GAUSS JORDAN FULL PIVOTING

RETURN

END

0001

0002

0003

0004

0005

0006

0007

0008

0009

0010

0011

0012

0013

0014

0015

0016

0017

0018

0019

0020

0021

0022

0023

0024

0025

0026

0027

0028

0029

C

C

711

710

C

C

C


```

PAGE 1      0001 SUBROUTINE SF5MX(IRC,ICC,ICOL,A,MR,MC,NU)
              0002 C
              0003 C   MINIMUM ROW - MINIMUM COLUMN ALGORITHM
              0004 C
              0005 C   INTEGER*4 PIVROW,PIVCOL,OPROW
              0006 C   REAL*8 A
              0007 C   COMMON /SINTG1/ PIVROW,PIVCOL,OPROW
              0008 C   COMMON /SINTG2/ IY
              0009 C   DIMENSION IRC(MR),ICC(MR),ICOL(MR,MC),A(MR,MC)
              0010 C
              0011 C   SELECT FIRST MIN ROW THEN FIRST MIN COL
              0012 C   SELECT ROW WITH MINIMUM ENTRIES
              0013 C   IK = 100000
              0014 C   DO 700 I=1,NU
              0015 C   IC = ICC(I)
              0016 C   IF(IC.GE.IK.OR.IC.LE.0) GOTO 700
              0017 C   PIVROW = I
              0018 C   IK = IC
              0019 C   CONTINUE
              0020 C
              0021 C   SELECT SMALLEST AVIABLE COLUMN FROM PIVROW
              0022 C   IK = 100000
              0023 C   IC = ICC(PIVROW)
              0024 C   DO 701 I=1,IC
              0025 C   II = ICOL(PIVROW,I)
              0026 C   IR = IRC(II)
              0027 C   IF(IR.GE.IK.OR.IR.LE.0) GOTO 701
              0028 C   PIVCOL = II
              0029 C   IK = IR
              0030 C   IY = I
              0031 C   CONTINUE
              0032 C
              0033 C   END FIRST MIN ROW THEN FIRST MIN COL
              0034 C   RETURN
              0035 C   END

```

0001
0002
0003
0004
0005
0006
0007
0008
0009
0010
0011
0012
0013
0014
0015
0016
0017
0018
0019
0020
0021
0022
0023
0024
0025
0026
0027
0028
0029
0030
0031
0032
0033
0034

PAGE 1

```

SUBROUTINE SGSMX(IRC,ICC,ICOL,A,MR,MC,NU)
MINIMUM COLUMN - MINIMUM ROW ALGORITHM
SELECT FIRST MIN COL THEN FIRST MIN ROW

INTEGER*4 PIVROW,PIVCOL,OPROW
REAL*8 A
COMMON /SINTG1/ PIVROW,PIVCOL,OPROW
COMMON /SINTG2/ IY
DIMENSION IRC(MR),ICC(MR),ICOL(MR,MC),A(MR,MC)
IK = 100000
DO 706 I=1,NU
IR = IRC(I)
IF(IR.GE.IK.OR.IR.LE.0) GOTO 706
PIVCOL = I
IK = IR
CONTINUE
IK = 100000
DO 707 I=1,NU
IC = ICC(I)
IF(IC.LE.0) GOTO 707
DO 708 J=1,IC
IF(ICOL(I,J).LT.PIVCOL) GOTO 708
IF(ICOL(I,J).GT.PIVCOL.OR.IC.GE.IK) GOTO 707
IK = IC
PIVROW = I
IY = J
CONTINUE
CONTINUE
END SELECT FIRST MIN COL THEN FIRST MIN ROW

RETURN
END

```

C

C

C

C

706

708

707

C

C

C

```

PAGE 1      0001      SUBROUTINE SHSMX(IRC,ICC,ICOL,A,MR,MC,NU)
              0002      MAXIMUM COLUMN = MINIMUM ROW ALGORITHM
              0003      SELECT FIRST MAX COL THEN FIRST MIN ROW
              0004      INTEGER*4 PIVROW,PIVCOL,OPROW
              0005      REAL*8 A
              0006      COMMON /SINTG1/ PIVROW,PIVCOL,OPROW
              0007      COMMON /SINTG2/ IY
              0008      DIMENSION IRC(MR),ICC(MR),ICOL(MR,MC),A(MR,MC)
              0009      IK = -1
              0010      DO 703 I=1,NU
              0011      IR = IRC(I)
              0012      IF(IR.LE.IK.OR.IR.LE.0) GOTO 703
              0013      PIVCOL = I
              0014      IK = IR
              0015      CONTINUE
              0016      IK = 100000
              0017      DO 704 I=1,NU
              0018      IC = ICC(I)
              0019      IF(IC.LE.0) GOTO 704
              0020      DO 705 J=1,IC
              0021      IF(ICOL(I,J).LT.PIVCOL) GOTO 705
              0022      IF(ICOL(I,J).GT.PIVCOL.OR.IC.GE.IK) GOTO 704
              0023      IK = IC
              0024      PIVROW = I
              0025      IY = J
              0026      CONTINUE
              0027      CONTINUE
              0028      C
              0029      C      END SELECT FIRST MAX COL THEN FIRST MIN ROW
              0030      C
              0031      RETURN
              0032      END

```

C

C

703

705

704

C

C

C

```

0001 SUBROUTINE SISMX(IRC,ICC,ICOL,A,MR,MC,NU)
0002 MINIMUM ROW ENTRIES TIMES COLUMN ENTRIES ALGORITHM
0003 SELECT FIRST MIN (ROW.COL)
0004 INTEGER*4 PIVROW,PIVCOL,OPROW
0005 REAL*8 A
0006 COMMON /SINTG1/ PIVROW,PIVCOL,OPROW
0007 COMMON /SINTG2/ IY
0008 DIMENSION IRC(MR),ICC(MR),ICOL(MR,MC),A(MR,MC)
0009 IK = 100000
0010 DO 709 I=1,NU
0011 IC = ICC(I)
0012 IF(IC.LE.0) GOTO 709
0013 DO 702 J=1,IC
0014 II = ICOL(I,J)
0015 IR = IRC(II)
0016 IF(IR.LE.0) GOTO 702
0017 III = IC*IR
0018 IF(III.GE.IK) GOTO 702
0019 PIVROW = I
0020 PIVCOL = II
0021 IK = III
0022 IY = J
0023 CONTINUE
0024 702 CONTINUE
0025 C
0026 C
0027 END SELECT FIRST MIN (ROW.COL)
0028 END

```

C

C

702

709

C

C

```

*****
TEST DU PROGRAMME DE RESOLUTION D'UN SYSTEME LINEAIRE DONT LA
MATRICE EST CREUSE.
*****

```

```

AUTEUR : E. VERGISON (SOLVAY COMPUTER CENTER)
        NOVEMBRE 1983.

```

```

REAL *8 A,AT,P
DIMENSION A(50,50),P(50),AT(50,50)
DIMENSION IRC(50),ICC(50),ICOL(50,50),IXA(50)
EQUIVALENCE (N,NU)
DATA LEC/5/,IMP/6/
MR = 50
MC = 50
NU = 9
ZTEST = 1.E-10

```

```

BOUCLE PRINCIPALE PARCOURANT LES SEPT ALGORITHMES DE RESOLUTION
DE SYSTEMES LINEAIRES.

```

```

DO 10 NPIV = 1,7

```

```

INITIALISATIONS - MISES A ZERO.

```

```

DO 20 II = 1,NU
  IXA(II) = 0
  P(II) = 0.D+00
CONTINUE
DO 30 J = 1,NU
DO 30 I = 1,NU
  A(I,J) = 0.D+00
  AT(I,J) = 0.D+00
  ICOL(I,J) = 0
CONTINUE

```

```

DEFINITION DU SECOND MEMBRE DU SYSTEME LINEAIRE.

```

```

P(1) = -2.D+00
P(2) = -1.D+00
P(3) = 4.D+00
P(4) = 3.D+00

```

0001

AGE 1

0002

0003

0004

0005

0006

0007

0008

0009

0010

0011

0012

0013

0014

0015

0016

0017

0018

0019

0020

0021

0022

0023

0024

0025

0026

0027

0028

0029

0030

0031

0032

0033

0034

0035

0036

0037

0038

0039

0040

0041

0042

C

C

C

C

C

C

C

20

30

C

C

C

```

0043 P(6) = 7.D+00
0044 P(7) = 16.D+00
0045 P(8) = 11.D+00
0046 P(9) = 22.D+00
0047
0048 C
0049 C
0050 C
0051 MATRICE EXEMPLATIVE.
0052
0053 DO 40 I = 1,9
0054 AT(I,I) = 4.D+00
0055 CONTINUE
0056 DO 50 I = 1,6
0057 AT(I,I+3) = -1.D+00
0058 AT(I+3,I) = -1.D+00
0059 CONTINUE
0060 AT(1,2) = -1.D+00
0061 AT(2,3) = -1.D+00
0062 AT(4,5) = -1.D+00
0063 AT(5,6) = -1.D+00
0064 AT(7,8) = -1.D+00
0065 AT(8,9) = -1.D+00
0066 AT(2,1) = -1.D+00
0067 AT(3,2) = -1.D+00
0068 AT(5,4) = -1.D+00
0069 AT(6,5) = -1.D+00
0070 AT(8,7) = -1.D+00
0071 AT(9,8) = -1.D+00
0072
0073 C
0074 C
0075 C
0076 ALGORITHME DE COMPACTAGE.
0077
0078 DO 60 J = 1,NU
0079 DO 60 I = 1,NU
0080 IF(AT(I,J) .EQ. 0.D+00) GO TO 60
0081 IXA(I) = IXA(I) + 1
0082 ICOL(I,IXA(I)) = J
0083 A(I,IXA(I)) = AT(I,J)
0084 CONTINUE
0085
0086 C
0087 C
0088 C
0089 C
0090 C
0091 C
0092 C
0093 C
0094 C
0095 C
0096 C
0097 C
0098 C
0099 C
0100 C
0101 C
0102 C
0103 C
0104 C
0105 C
0106 C
0107 C
0108 C
0109 C
0110 C
0111 C
0112 C
0113 C
0114 C
0115 C
0116 C
0117 C
0118 C
0119 C
0120 C
0121 C
0122 C
0123 C
0124 C
0125 C
0126 C
0127 C
0128 C
0129 C
0130 C
0131 C
0132 C
0133 C
0134 C
0135 C
0136 C
0137 C
0138 C
0139 C
0140 C
0141 C
0142 C
0143 C
0144 C
0145 C
0146 C
0147 C
0148 C
0149 C
0150 C
0151 C
0152 C
0153 C
0154 C
0155 C
0156 C
0157 C
0158 C
0159 C
0160 C
0161 C
0162 C
0163 C
0164 C
0165 C
0166 C
0167 C
0168 C
0169 C
0170 C
0171 C
0172 C
0173 C
0174 C
0175 C
0176 C
0177 C
0178 C
0179 C
0180 C
0181 C
0182 C
0183 C
0184 C
0185 C
0186 C
0187 C
0188 C
0189 C
0190 C
0191 C
0192 C
0193 C
0194 C
0195 C
0196 C
0197 C
0198 C
0199 C
0200 C
0201 C
0202 C
0203 C
0204 C
0205 C
0206 C
0207 C
0208 C
0209 C
0210 C
0211 C
0212 C
0213 C
0214 C
0215 C
0216 C
0217 C
0218 C
0219 C
0220 C
0221 C
0222 C
0223 C
0224 C
0225 C
0226 C
0227 C
0228 C
0229 C
0230 C
0231 C
0232 C
0233 C
0234 C
0235 C
0236 C
0237 C
0238 C
0239 C
0240 C
0241 C
0242 C
0243 C
0244 C
0245 C
0246 C
0247 C
0248 C
0249 C
0250 C
0251 C
0252 C
0253 C
0254 C
0255 C
0256 C
0257 C
0258 C
0259 C
0260 C
0261 C
0262 C
0263 C
0264 C
0265 C
0266 C
0267 C
0268 C
0269 C
0270 C
0271 C
0272 C
0273 C
0274 C
0275 C
0276 C
0277 C
0278 C
0279 C
0280 C
0281 C
0282 C
0283 C
0284 C
0285 C
0286 C
0287 C
0288 C
0289 C
0290 C
0291 C
0292 C
0293 C
0294 C
0295 C
0296 C
0297 C
0298 C
0299 C
0300 C
0301 C
0302 C
0303 C
0304 C
0305 C
0306 C
0307 C
0308 C
0309 C
0310 C
0311 C
0312 C
0313 C
0314 C
0315 C
0316 C
0317 C
0318 C
0319 C
0320 C
0321 C
0322 C
0323 C
0324 C
0325 C
0326 C
0327 C
0328 C
0329 C
0330 C
0331 C
0332 C
0333 C
0334 C
0335 C
0336 C
0337 C
0338 C
0339 C
0340 C
0341 C
0342 C
0343 C
0344 C
0345 C
0346 C
0347 C
0348 C
0349 C
0350 C
0351 C
0352 C
0353 C
0354 C
0355 C
0356 C
0357 C
0358 C
0359 C
0360 C
0361 C
0362 C
0363 C
0364 C
0365 C
0366 C
0367 C
0368 C
0369 C
0370 C
0371 C
0372 C
0373 C
0374 C
0375 C
0376 C
0377 C
0378 C
0379 C
0380 C
0381 C
0382 C
0383 C
0384 C
0385 C
0386 C
0387 C
0388 C
0389 C
0390 C
0391 C
0392 C
0393 C
0394 C
0395 C
0396 C
0397 C
0398 C
0399 C
0400 C
0401 C
0402 C
0403 C
0404 C
0405 C
0406 C
0407 C
0408 C
0409 C
0410 C
0411 C
0412 C
0413 C
0414 C
0415 C
0416 C
0417 C
0418 C
0419 C
0420 C
0421 C
0422 C
0423 C
0424 C
0425 C
0426 C
0427 C
0428 C
0429 C
0430 C
0431 C
0432 C
0433 C
0434 C
0435 C
0436 C
0437 C
0438 C
0439 C
0440 C
0441 C
0442 C
0443 C
0444 C
0445 C
0446 C
0447 C
0448 C
0449 C
0450 C
0451 C
0452 C
0453 C
0454 C
0455 C
0456 C
0457 C
0458 C
0459 C
0460 C
0461 C
0462 C
0463 C
0464 C
0465 C
0466 C
0467 C
0468 C
0469 C
0470 C
0471 C
0472 C
0473 C
0474 C
0475 C
0476 C
0477 C
0478 C
0479 C
0480 C
0481 C
0482 C
0483 C
0484 C
0485 C
0486 C
0487 C
0488 C
0489 C
0490 C
0491 C
0492 C
0493 C
0494 C
0495 C
0496 C
0497 C
0498 C
0499 C
0500 C
0501 C
0502 C
0503 C
0504 C
0505 C
0506 C
0507 C
0508 C
0509 C
0510 C
0511 C
0512 C
0513 C
0514 C
0515 C
0516 C
0517 C
0518 C
0519 C
0520 C
0521 C
0522 C
0523 C
0524 C
0525 C
0526 C
0527 C
0528 C
0529 C
0530 C
0531 C
0532 C
0533 C
0534 C
0535 C
0536 C
0537 C
0538 C
0539 C
0540 C
0541 C
0542 C
0543 C
0544 C
0545 C
0546 C
0547 C
0548 C
0549 C
0550 C
0551 C
0552 C
0553 C
0554 C
0555 C
0556 C
0557 C
0558 C
0559 C
0560 C
0561 C
0562 C
0563 C
0564 C
0565 C
0566 C
0567 C
0568 C
0569 C
0570 C
0571 C
0572 C
0573 C
0574 C
0575 C
0576 C
0577 C
0578 C
0579 C
0580 C
0581 C
0582 C
0583 C
0584 C
0585 C
0586 C
0587 C
0588 C
0589 C
0590 C
0591 C
0592 C
0593 C
0594 C
0595 C
0596 C
0597 C
0598 C
0599 C
0600 C
0601 C
0602 C
0603 C
0604 C
0605 C
0606 C
0607 C
0608 C
0609 C
0610 C
0611 C
0612 C
0613 C
0614 C
0615 C
0616 C
0617 C
0618 C
0619 C
0620 C
0621 C
0622 C
0623 C
0624 C
0625 C
0626 C
0627 C
0628 C
0629 C
0630 C
0631 C
0632 C
0633 C
0634 C
0635 C
0636 C
0637 C
0638 C
0639 C
0640 C
0641 C
0642 C
0643 C
0644 C
0645 C
0646 C
0647 C
0648 C
0649 C
0650 C
0651 C
0652 C
0653 C
0654 C
0655 C
0656 C
0657 C
0658 C
0659 C
0660 C
0661 C
0662 C
0663 C
0664 C
0665 C
0666 C
0667 C
0668 C
0669 C
0670 C
0671 C
0672 C
0673 C
0674 C
0675 C
0676 C
0677 C
0678 C
0679 C
0680 C
0681 C
0682 C
0683 C
0684 C
0685 C
0686 C
0687 C
0688 C
0689 C
0690 C
0691 C
0692 C
0693 C
0694 C
0695 C
0696 C
0697 C
0698 C
0699 C
0700 C
0701 C
0702 C
0703 C
0704 C
0705 C
0706 C
0707 C
0708 C
0709 C
0710 C
0711 C
0712 C
0713 C
0714 C
0715 C
0716 C
0717 C
0718 C
0719 C
0720 C
0721 C
0722 C
0723 C
0724 C
0725 C
0726 C
0727 C
0728 C
0729 C
0730 C
0731 C
0732 C
0733 C
0734 C
0735 C
0736 C
0737 C
0738 C
0739 C
0740 C
0741 C
0742 C
0743 C
0744 C
0745 C
0746 C
0747 C
0748 C
0749 C
0750 C
0751 C
0752 C
0753 C
0754 C
0755 C
0756 C
0757 C
0758 C
0759 C
0760 C
0761 C
0762 C
0763 C
0764 C
0765 C
0766 C
0767 C
0768 C
0769 C
0770 C
0771 C
0772 C
0773 C
0774 C
0775 C
0776 C
0777 C
0778 C
0779 C
0780 C
0781 C
0782 C
0783 C
0784 C
0785 C
0786 C
0787 C
0788 C
0789 C
0790 C
0791 C
0792 C
0793 C
0794 C
0795 C
0796 C
0797 C
0798 C
0799 C
0800 C
0801 C
0802 C
0803 C
0804 C
0805 C
0806 C
0807 C
0808 C
0809 C
0810 C
0811 C
0812 C
0813 C
0814 C
0815 C
0816 C
0817 C
0818 C
0819 C
0820 C
0821 C
0822 C
0823 C
0824 C
0825 C
0826 C
0827 C
0828 C
0829 C
0830 C
0831 C
0832 C
0833 C
0834 C
0835 C
0836 C
0837 C
0838 C
0839 C
0840 C
0841 C
0842 C
0843 C
0844 C
0845 C
0846 C
0847 C
0848 C
0849 C
0850 C
0851 C
0852 C
0853 C
0854 C
0855 C
0856 C
0857 C
0858 C
0859 C
0860 C
0861 C
0862 C
0863 C
0864 C
0865 C
0866 C
0867 C
0868 C
0869 C
0870 C
0871 C
0872 C
0873 C
0874 C
0875 C
0876 C
0877 C
0878 C
0879 C
0880 C
0881 C
0882 C
0883 C
0884 C
0885 C
0886 C
0887 C
0888 C
0889 C
0890 C
0891 C
0892 C
0893 C
0894 C
0895 C
0896 C
0897 C
0898 C
0899 C
0900 C
0901 C
0902 C
0903 C
0904 C
0905 C
0906 C
0907 C
0908 C
0909 C
0910 C
0911 C
0912 C
0913 C
0914 C
0915 C
0916 C
0917 C
0918 C
0919 C
0920 C
0921 C
0922 C
0923 C
0924 C
0925 C
0926 C
0927 C
0928 C
0929 C
0930 C
0931 C
0932 C
0933 C
0934 C
0935 C
0936 C
0937 C
0938 C
0939 C
0940 C
0941 C
0942 C
0943 C
0944 C
0945 C
0946 C
0947 C
0948 C
0949 C
0950 C
0951 C
0952 C
0953 C
0954 C
0955 C
0956 C
0957 C
0958 C
0959 C
0960 C
0961 C
0962 C
0963 C
0964 C
0965 C
0966 C
0967 C
0968 C
0969 C
0970 C
0971 C
0972 C
0973 C
0974 C
0975 C
0976 C
0977 C
0978 C
0979 C
0980 C
0981 C
0982 C
0983 C
0984 C
0985 C
0986 C
0987 C
0988 C
0989 C
0990 C
0991 C
0992 C
0993 C
0994 C
0995 C
0996 C
0997 C
0998 C
0999 C
1000 C

```

ALGORITHME DE RESOLUTION DE SYSTEMES LINEAIRES CREUX.

CALL SOSMX(NU,MR,MC,A,P,ICOL,ZTEST,ICC,IRC,JELE,JELIM,JCOL,NPIV,
1 TIME)

```

PAGE 3
0085 C
0086 C IMPRESSION DES RESULTATS.
0087 C
0088 C WRITE(IMP,2000) JELE,JELIM,JCOL,TIME
0089 C WRITE(IMP,2100)((I,A(I,1)),I=1,NU)
0090 C
0091 C CONTINUE
0092 C
0093 C FORMATS.
0094 C
0095 C 2000 FORMAT(1H,5X,'JELE = ',I5,5X,'JELIM = ',I5,5X,'JCOL = ',I5,5X,'
0096 C 2100 1 TIME = ',F8.2,'//)
0097 C 2X,I3,2X,F6.2,1X,';',2X,I3,2X,F6.2,1X,';',
0098 C 2X,I3,2X,F6.2,1X,';',2X,I3,2X,F6.2,1X,';',
0099 C CALL EXIT
END

```

A N N E X E 3

Résolution d'un problème non linéaire par la méthode multigrille

- Programme de calcul
- Résultats


```

PAGE 2      0043      C      IGAM      : TYPE OF CYCLING (IGAM > 0)
            0044      C      IGAM .EQ. 1 : V-CYCLE
            0045      C      IGAM .EQ. 2 : W-CYCLE
            0046      C      ITERMAX: MAXIMUM NUMBER OF ITERATION IN THE NEWTON ITERATION
            0047      C      PROCEDURE.
            0048      C
            0049      C      READ(LEC,*) M, NCYCLE, NFMG, IGAM, ITERMAX
            0050      C
            0051      C      CALL MGOOD(M,NY1,NY2,NFMG,NCYCLE,IGAM,F,G,C,CPRIM,W,IDIM,INITF,
            0052      C      IER,ITERMAX)
            0053      C
            0054      C      IF(IER.EQ.0) GOTO 10
            0055      C      WRITE(IMP,2000) IER,IDIM
            0056      C      CALL EXIT
            0057      C      CONTINUE
            10
            0058      C
            0059      C      2000      FORMAT(16H *** ERROR,IER=, I3, 7H, IDIM=, I6, 4H ***)
            0060      C      CALL EXIT
            0061      C      END

```

PAGE 1
 0001
 0002
 0003
 0004
 0005
 0006
 0007
 0008
 0009
 0010

C-----
 C
 C

```

DOUBLE PRECISION FUNCTION F(X,Y)
DOUBLE PRECISION X, Y, DSIN
F = -200.D+00*X*(X-1) - 200.D+00*Y*(Y-1)
  1 + 1.D+04*X*(X-1.D+00)*Y*(Y-1.D+00)
  2 + 1.D+06*(X*(X-1)*Y*(Y-1))**3
RETURN
END

```

PAGE 1
0001
0002
0003
0004
0005
0006
0007

C-----
C
C

DOUBLE PRECISION FUNCTION G(X,Y)
DOUBLE PRECISION X, Y, DSIN
G = 100.D+00*X*(X-1)*Y*(Y-1)
END

0001
0002
0003
0004
0005
0006
0007
0008

PAGE 1

C-----C
C-----C
C-----C

DOUBLE PRECISION FUNCTION C(X,Y,U)
DOUBLE PRECISION X, Y, U
C = U*(100.D+00 + U*U)
RETURN
END

ADDITIONAL MG ITERATIONS AFTERWARDS

TYPE OF CYCLING (IGAM > 0).
 E.G.: IGAM=1 FOR V-CYCLES, IGAM=2 FOR W-CYCLES
 DOUBLE PRECISION WORK ARRAY OF DIMENSION IDIM
 DIMENSION OF W. APPROXIMATELY: IDIM > 2.8**4**M
 MAXIMUM NUMBER OF ITERATIONS IN THE FIXED POINT
 PROCEDURE.

DOUBLE PRECISION FUNCTION F(X,Y), RIGHT HAND SIDE OF
 THE DIFFERENTIAL EQUATION
 DOUBLE PRECISION FUNCTION G(X,Y), BOUNDARY VALUES
 DOUBLE PRECISION FUNCTION C(X,Y,U(X,Y)),
 NON LINEAR HELMHOLTZ TERM SUPPOSED TO BE NON
 NEGATIVE.
 DOUBLE PRECISION FUNCTION CPRIM(X,Y,U(X,Y)),
 PARTIAL DERIVATIVE WITH RESPECT TO "U" OF
 FUNCTION C(X,Y,U)

GRID #1 AND #M ARE THE FINEST AND COARSEST GRID USED,
 RESPECTIVELY

ERROR INDICATOR
 = 0 NO ERRORS
 = 1 INSUFFICIENT MEMORY, I.E. IDIM TOO SMALL.
 IN THIS CASE IDIM IS USED AS OUTPUT PARAMETER
 TO SHOW THE MINIMAL DIMENSION

ONLY IN CASE IER=1 : MINIMAL LENGTH OF W
 CF. DESCRIPTION OF W
 W CONTAINS THE DISCRETE APPROXIMATION TO THE GIVEN
 BOUNDARY VALUE PROBLEM ON THE FINEST GRID. THE GRID
 VALUES ARE STORED ROWWISE FROM LEFT TO RIGHT AND
 FROM BOTTOM TO TOP.
 I.E. THE GRID VALUE CORRESPONDING TO THE GRID POINT
 (XI,YJ) :

$$XI = (I-1)*H, YJ = (J-1)*H \quad (H=1/N, N=2**M)$$

IGAM
 W
 IDIM
 ITERMAX

EXTERNALS:

F
 G
 C
 CPRIM

REMARK:

OUTPUT:

IER

IDIM
 INITF
 W

0043
 0044
 0045
 0046
 0047
 0048
 0049
 0050
 0051
 0052
 0053
 0054
 0055
 0056
 0057
 0058
 0059
 0060
 0061
 0062
 0063
 0064
 0065
 0066
 0067
 0068
 0069
 0070
 0071
 0072
 0073
 0074
 0075
 0076
 0077
 0078
 0079
 0080
 0081
 0082
 0083
 0084


```

0085 IS STORED AT
0086 W((J-1)*(N+1)+I) (0. < I, J < N+2)
0087
0088 THE CORRESPONDING VALUES OF THE RIGHT HAND SIDE ARE
0089 STORED IN THE SAME MANNER AT
0090
0091 W((J-1)*(N+1)+I+INITF).
0092
0093 THE CORRESPONDING VALUES OF THE HELMOLTZ TERM
0094 IS STORED AT
0095 W((J-1)*(N+1)+I+2*INITF)
0096
0097 THE REMAINING STORAGE CONTAINS COARSE GRID VALUES.
0098
0099 INTEGER IDC, IDF, IDR, IDS, IT, ITM, INITF, L, L1, LEV, LIN, NP,
0100 *NY1, NY2, IGAM, IDIM, IER, NCYCLE, NFMG, M, N, ID(11), NPK(10)
0101
0102 INTEGER ITER, ITERMAX, NMID
0103 DOUBLE PRECISION W(1), HK(10), DFLOAT, F, G, C, CPRIM
0104 EXTERNAL F, G, C, CPRIM
0105 DFLOAT(L1) = DBLE(FLOAT(L1))
0106 IER = 0
0107
0108 IF(M.LE.0 .OR. M.GE.11 .OR. NY1.LE.0 .OR. NY2.LE.0 .OR.
0109 * NCYCLE.LE.0 .OR. IGAM.LE.0 ) IER=2
0110 IF(IER.GT.0) GOTO 50
0111
0112 DETERMINATION OF COARSER GRIDS
0113
0114 NP = 2*M + 1
0115 N = NP - 1
0116 NMID = (NP*NP + 1)/2
0117 HK(1) = 1.000/DFLOAT(NP-1)
0118 ID(1) = 1
0119 ID(2) = NP*NP + 1
0120 NPK(1) = NP
0121 DO 10 L = 2, M
0122 NPK(L) = (NPK(L-1) + 1)/2
0123 HK(L) = HK(L-1) + HK(L-1)
0124 ID(L+1) = ID(L) + NPK(L)**2
0125 CONTINUE
0126
0127 CHECK OF DIMENSIONS

```

```

PAGE 4      0127  C      INITF = ID(M+1)
            0128  C      IF(3*INITF.LE.IDIM) GOTO 20
            0129  C      IER = 1
            0130  C      IDIM = 3*INITF
            0131  C      GOTO 60
            0132  20  C      LIN = 1
            0133  C      IF(NFMG.NE.0) LIN = M
            0134  C      DO 50 ITER = 1,ITERMAX
            0135  C
            0136  C      SET UP ALL GRID VALUES NEEDED
            0137  C
            0138  C      CALL INIT1(NP, HK(1), F, G, C, CPRIM,
            0139  C      W(1), W(INITF+1), W(2*INITF+1), ITER)
            0140  C      CALL INIT2(M, LIN, NPK, ID, W(1), W(INITF+1), W(2*INITF+1), INITF)
            0141  1  C
            0142  C      (FULL) MULTIGRID PROCEDURE
            0143  C
            0144  C
            0145  C      DO 40 L = 1,LIN
            0146  C      LEV = LIN - L + 1
            0147  C      ITM = 1
            0148  C      IF(LEV.EQ.1) ITM=NCYCLE
            0149  C      DO 30 IT = 1,ITM
            0150  C      CALL MGI(LEV, M, NY1, NY2, IGAM, NPK, ID, W, W(INITF+1),
            0151  C      W(2*INITF+1), INITF)
            0151  1  C
            0152  30  C      CONTINUE
            0153  C      IF(L.LT.LIN) THEN
            0154  C      IDF = ID(LEV-1)
            0155  C      IDC = ID(LEV)
            0156  C      IDR = IDF + INITF
            0157  C      IDS = IDR + INITF
            0158  C      CALL INT4(NPK(LEV), NPK(LEV-1), W(IDC), W(IDF), W(IDR), W(IDS))
            0159  C      CALL PUTZB(NPK(LEV),W(IDC))
            0160  C      END IF
            0161  40  C      CONTINUE
            0162  C      DIF = DIFMX(W,NP,G)
            0163  C      DEF = DEFMX(W,W(INITF+1),W(2*INITF+1),NP)
            0164  C      WRITE(6,9100) N, DIF, DEF
            0165  C      WRITE(6,9200) W(NMID)
            0166  C
            0167  9100 C      FORMAT(30H NUMBER OF INTERVALS = , I4, /,
            0168  C      30H MAXIMUM NORM OF THE ERROR = , D12.4, /,

```

PAGE 5 0169 2 30H MAXIMUM NORM OF THE DEFECT = , D12.4)
0170 9200 FORMAT(1H0,30X,'VALEUR CENTRALE DE LA SOLUTION = ',D12.5)
0171 C
0172 50 CONTINUE
0173 60 RETURN
0174 END

CONTINUE
END IF
RETURN
END

40

0043
0044
0045
0046

2

PAGE


```

PAGE 1
C-----
0001 C
0002 C
0003 C
0004 C
0005 C
0006 C
0007 C
0008 C
0009 C
0010 C
0011 C
0012 C
0013 C
0014 C
0015 C
0016 C
0017 C
0018 C
0019 C
0020 C
0021 C
0022 C
0023 C
0024 C
0025 C
0026 C
0027 C
0028 C
0029 C
0030 C
0031 C

SUBROUTINE TRANS(NPC, NPF, UC, FC, CC, UF, FF, CF)
TRANSFER OF F, C AND U FROM GRID NPF TO NPC
INTEGER NC, NPC, NPF, I, IF, J, JF
DOUBLE PRECISION FC(NPC,NPC), CC(NPC,NPC), UC(NPC,NPC),
FF(NPF,NPF), CF(NPF,NPF), UF(NPF,NPF)
1
NC = NPC - 1
DO 20 J = 2,NC
JF = J + J - 1
UC(1,J) = UF(1,JF)
DO 10 I=2,NC
IF = I + I - 1
UC(I,J) = UF(IF,JF)
FC(I,J) = 4.D+00*FF(IF,JF)
CC(I,J) = 4.D+00*CF(IF,JF)
CONTINUE
UC(NPC,J) = UF(NPF,JF)
CONTINUE
DO 30 I = 1,NPC
IF = I + I - 1
UC(I,1) = UF(IF,1)
UC(I,NPC) = UF(IF,NPF)
CONTINUE
RETURN
END
10
20
30

```



```

PAGE 1 0001 C-----
0002 C SUBROUTINE MGI( LEV, M, NY1, NY2, IGAM, NPK, ID, U, FR, CR, IDIM)
0003 C
0004 C ONE MULTIGRID ITERATION STEP (ON ACTUAL FINEST GRID LEV)
0005 C
0006 C
0007 C INTEGER IDIM, LEV, M, NY1, NY2, ID(11), NPK(10), IDC, IDF, IZ,
0008 C K, IGAM, ICGAM(10)
0009 C DOUBLE PRECISION FR(IDIM), CR(IDIM), U(IDIM)
0010 C
0011 C IZ = 1
0012 C DO 10 K = LEV, M
0013 C ICGAM(K) = 0
0014 C CONTINUE
0015 C K = LEV
0016 C IF(K.EQ.M) GOTO 30
0017 C IDF = ID(K)
0018 C
0019 C RELAXATIONS BEFORE CGC
0020 C
0021 C IZ = 1
0022 C IF(K.GT.LEV.AND.ICGAM(K).EQ.0) IZ = 0
0023 C CALL RELAX(NY1+NY1, IZ, NPK(K), U(IDF), FR(IDF), CR(IDF), FR(1))
0024 C ICGAM(K) = ICGAM(K) + 1
0025 C
0026 C RESIDUAL TRANSFER TO NEXT COARSER GRID
0027 C
0028 C IDC = ID(K+1)
0029 C CALL RESTR(NPK(K+1), NPK(K), FR(IDC), U(IDF), FR(IDF), CR(IDF))
0030 C K = K + 1
0031 C IF(K.LT.M) GOTO 20
0032 C IZ = 0
0033 C IDC = ID(M)
0034 C
0035 C EXACT SOLUTION ON COARSEST GRID
0036 C
0037 C CALL RELAX(1, IZ, NPK(M), U(IDC), FR(IDC), CR(IDC), FR(1))
0038 C IF(K.EQ.LEV) GOTO 50
0039 C K = K - 1
0040 C IDF = ID(K)
0041 C IDC = ID(K+1)
0042 C

```

```

PAGE 2
C 0043 LINEAR INTERPOLATION TO NEXT FINER GRID
C 0044 CALL INT2A(NPK(K+1), NPK(K), U(IDC), U(IDF))
C 0045 RELAXATION AFTER CGC
C 0046
C 0047 CALL RELAX(NY2+NY2, 1, NPK(K), U(IDF), FR(IDF), CR(IDF), FR(1))
C 0048
C 0049 IF(K.EQ.LEV) GOTO 50
C 0050 IF(ICGAM(K).LT.IGAM) GOTO 20
C 0051 ICGAM(K) = 0
C 0052 GOTO 40
C 0053 RETURN
C 0054 END
C 0055
50
```

```
PAGE 2      0043      DIAG = 1-D+00/(4-D+00 + CF(I,J))  
            0044      UF(I,J) = DIAG*(FF(I,J) + W(I) + W(I+1))  
            0045      CONTINUE  
            0046      IS = 5 - IS  
            0047      CONTINUE  
            0048      CONTINUE  
            0049      CONTINUE  
            0050      RETURN  
            0051      END
```

```

PAGE 1
0001 C
0002 C-----
0003 C
0004 C SUBROUTINE RESTR(NPC, NPF, FC, UF, FF, CF)
0005 C
0006 C COMPUTATION OF THE DEFECT AND FINE-TO-COARSE TRANSFER
0007 C (HALF-INJECTION)
0008 C
0009 C INTEGER NC, NPC, NPF, I, IF, J, JF
0010 C DOUBLE PRECISION FC(NPC,NPC),FF(NPF,NPF),CF(NPF,NPF),UF(NPF,NPF)
0011 C DOUBLE PRECISION H
0012 C
0013 C NC = NPC - 1
0014 C DO 20 J = 2,NC
0015 C JF = J + J - 1
0016 C DO 10 I = 2,NC
0017 C IF = I + I - 1
0018 C H = FF(IF,JF) - (4.*D+00 + CF(IF,JF))*UF(IF,JF)
0019 C + UF(IF,JF-1) + UF(IF-1,JF) + UF(IF+1,JF) + UF(IF,JF+1)
0020 C FC(I,J) = H + H
0021 C CONTINUE
0022 C CONTINUE
0023 C RETURN
0024 C END

```

```

PAGE 1
0001 C-----
0002 C
0003 C
0004 C
0005 C
0006 C
0007 C
0008 C
0009 C
0010 C
0011 C
0012 C
0013 C
0014 C
0015 C
0016 C
0017 C
0018 C
0019 C
0020 C
0021 C
0022 C
0023 C
0024 C
0025 C
0026 C
0027 C

SUBROUTINE INT2A(NPC, NPF, UC, UF)
COARSE-TO-FINE TRANSFER (BILINEAR INTERPOLATION) AND CORRECTION
INTEGER NC, NPC, NPF, I, IF, J, JF
DOUBLE PRECISION UC(NPC,NPC), UF(NPF,NPF)
NC = NPC - 1
DO 20 J = 2,NC
  JF = J + J - 1
  DO 10 I = 1,NC
    IF = I + I
    UF(IF,JF) = UF(IF,JF) + 0.5DU*(UC(I,J)+UC(I+1,J))
  CONTINUE
CONTINUE
DO 40 J = 1,NC
  JF = J + J
  DO 30 I = 2,NC
    IF = I + I - 1
    UF(IF,JF) = UF(IF,JF) + 0.5DU*(UC(I,J) + UC(I,J+1))
  CONTINUE
CONTINUE
RETURN
END

```



```

PAGE 1
0001 C
0002 C-----
0003 C
0004 C
0005 C
0006 C
0007 C
0008 C
0009 C
0010 C
0011 C
0012 C
0013 C
0014 C
0015 C
0016 C
0017 C
0018 C
0019 C
0020 C
0021 C
0022 C
0023 C

      DOUBLE PRECISION FUNCTION DIFMX(UC, NP, SOL)
      COMPUTE THE MAXIMUM NORM OF THE DIFFERENCE BETWEEN
      SOL (=SOLUTION OF THE BVP) AND THE VALUES IN UC
      INTEGER NP, I, J
      DOUBLE PRECISION UC(NP,NP), SOL, DFLOAT, H, X, Y
      DFLOAT(K) = DBLE(FLOAT(K))
      H = 1.000/DFLOAT(NP-1)
      DIFMX = 0.000
      DO 20 J = 1, NP
        Y = DFLOAT(J-1)*H
        DO 10 I = 1, NP
          X = DFLOAT(I-1)*H
          DIFMX = DMAX1(DIFMX, DABS(SOL(X,Y) - UC(I,J)))
        CONTINUE
      CONTINUE
      RETURN
      END

```


- [9] G. W. STEWART : Introduction to Matrix Computations. Academic Press, 1973.

- [10] K. STUBEN and U. TROTTEBERG : Multigrid Methods: Fundamental Algorithms, Model Problem Analysis and Applications. Proceedings, Köln-Porz, 1981. Lecture Notes in Mathematics, 960. Springer-Verlag, Berlin, 1982.

- [11] K. STUBEN : Méthodes Multigrilles. Cours Gamni, Ecole Centrale de Lyon, septembre 1983.

- [12] R. TEMAM : Analyse Numérique. Presses Universitaires de France, 1970.

- [13] U. TROTTEBERG : Méthodes Multigrilles. Cours Gamni, Ecole Centrale de Lyon, septembre 1983.

- [14] D. WEXLER : Equations aux dérivées partielles, cours donné aux FNDP Namur, 1983.

- [15] O. ZIENKIEWICZ : The finite element method in Engineering Science. Mac Graw Hill, 1971.