



TECHNIQUES  
DE L'INGÉNIEUR

Réf. : AF502 V1

# Algorithmes numériques pour la résolution des grands systèmes

Date de publication :  
**10 octobre 2002**

Cet article est issu de : **Sciences fondamentales | Mathématiques**

par **Pierre SPITERI**

**Pour toute question :**  
Service Relation clientèle  
Techniques de l'Ingénieur  
Immeuble Pleyad 1  
39, boulevard Ornano  
93288 Saint-Denis Cedex

**Par mail :**  
[infos.clients@teching.com](mailto:infos.clients@teching.com)  
**Par téléphone :**  
00 33 (0)1 53 35 20 20

Document téléchargé le : **06/04/2019**  
Pour le compte : **7200043660 - centralesupelec // 195.221.160.3**

# Algorithmes numériques pour la résolution des grands systèmes

par **Pierre SPITERI**

*Docteur ès sciences mathématiques*

*Professeur à l'École nationale supérieure d'électronique, d'électrotechnique, d'informatique, d'hydraulique et de télécommunication de Toulouse*

<b>1. Position du problème.....</b>	AF 502 - 2
<b>2. Méthodes directes .....</b>	— 2
<b>3. Méthodes itératives de relaxation par points et par blocs .....</b>	— 3
<b>4. Méthodes issues de la minimisation de formes quadratiques....</b>	— 5
<b>5. Méthode multigrille.....</b>	— 7
<b>6. Méthodes de décomposition de domaine .....</b>	— 8
<b>Références bibliographiques .....</b>	— 10

**O**n a vu dans l'article [AF 500] que la discréttisation d'équations aux dérivées partielles stationnaires conduisait à la **résolution de systèmes linéaires de grande dimension** dont la matrice est creuse. De même, la discréttisation d'équations aux dérivées partielles d'évolution par des schémas implicites (article [AF 501]) conduit également à la résolution de systèmes linéaires ayant les mêmes caractéristiques. Compte tenu de cette spécificité, l'inversion des matrices issues de la discréttisation d'équations aux dérivées partielles devient de plus en plus préoccupante dans le domaine de la simulation numérique et est, par conséquent, très délicate, compte tenu, en particulier, du mauvais conditionnement de ces matrices. Cet aspect dépend fortement des applications traitées et il est hors de question de donner une réponse universelle à ce problème. C'est pourquoi, dans cet article, nous allons passer en revue différentes méthodes de résolution de tels systèmes, pour essayer de dégager les algorithmes les plus performants.

Dans le cas de la résolution numérique d'une équation aux dérivées partielles non linéaire, on doit résoudre un système algébrique non linéaire ; la résolution d'un tel système s'effectuera par une méthode itérative de type méthode de Newton [1], ce qui nécessitera, à chaque itération, une linéarisation de l'application considérée autour du point courant et la résolution d'un système linéaire ; l'étude de la convergence de ce type de méthode est loin d'être triviale et les résultats théoriques garantissant la convergence de la méthode sont établis uniquement dans des situations particulières. Si l'équation aux dérivées partielles est linéaire, on aura à résoudre un système linéaire ce qui, en théorie, paraît plus simple ; cependant il subsiste des difficultés d'ordre numérique pour déterminer la solution approchée. Dans cet exposé, nous nous limiterons au cas linéaire.

On rappelle que l'étude concernant la méthode des différences finies pour résoudre des équations aux dérivées partielles se décompose en trois articles :

- [AF 500] *Méthode des différences finies pour les EDP stationnaires* ;
- [AF 501] *Méthode des différences finies pour les EDP d'évolution* ;
- [AF 502] *Algorithmes numériques pour la résolution des grands systèmes*.

## **1. Position du problème**

Avant d'exposer les grandes lignes des méthodes de résolution des systèmes linéaires, considérons un exemple issu d'applications industrielles, qui va nous permettre de comprendre la difficulté de résolution de tels systèmes.

**Exemple 1 :** considérons l'analyse des flux aérodynamiques autour d'un avion en mouvement ; le problème revient à déterminer en chaque point du milieu et à chaque instant, la valeur de paramètres comme la température, la pression, etc. Le phénomène étudié peut être modélisé par les équations de Navier-Stokes qui expriment la conservation de la masse, du moment et de l'énergie ; en coordonnées cartésiennes, et dans leurs formes complètes, ces équations comprennent plus de soixante dérivées partielles. La résolution numérique de ces équations s'avère nécessaire dans la mesure où une résolution analytique est problématique. La prise en compte des équations de Navier-Stokes dans leur totalité conduirait à des maillages comprenant de  $10^{12}$  à  $10^{15}$  points. Si l'on considère un problème simplifié décrit sur un maillage comprenant  $10^7$  points, avec 20 valeurs attachées à chaque point (paramètres du problème, éléments de géométrie, résultats intermédiaires, etc.), le modèle discret considéré comporte  $2 \times 10^8$  données ; suivant le type de problème, le volume de calculs peut atteindre voire dépasser  $10^{13}$  opérations arithmétiques. Pour un ordinateur capable d'exécuter  $10^7$  opérations arithmétiques par seconde, le temps de calcul est de l'ordre de  $10^{13}/10^7$  secondes, soit environ 278 heures ou encore près de 12 jours.

Sauf cas particulier (cf. paragraphe 5, méthode multigrille), pour simplifier les notations nous noterons dans la suite :

$$AU = F, \quad U \in \mathbb{R}^{\dim(A)}, \quad F \in I\mathbb{R}^{\dim(A)}$$

le système linéaire à inverser.

## 2. Méthodes directes

La première méthode de résolution envisageable du système linéaire précédent, est la **méthode d'élimination de Gauss** (ou ses variantes comme la méthode de Crout ou la méthode de Cholesky, etc.), dont la complexité, c'est-à-dire le nombre d'opérations arithmétiques, est de l'ordre de  $\frac{2}{3}(\dim(A))^3$  ; si  $A$  est une matrice symétrique, le nombre d'opérations arithmétiques se réduit à  $\frac{1}{3}(\dim(A))^3$  ; or on a vu que, dans le cas de résolution numérique d'équations aux dérivées partielles,  $\dim(A)$  est grand et, par conséquent, le nombre d'opérations arithmétiques devient vite important. Par ailleurs, on a vu également que, dans un grand nombre de cas, la discréttisation des équations aux dérivées partielles conduit à des matrices de discréttisation à structure bande du type suivant :

$$\begin{pmatrix} & & & & 0 \\ & & & \leftarrow 2\ell+1 \rightarrow & \\ & & & & 0 \end{pmatrix}$$

On peut tirer partie de cette situation grâce au résultat suivant :

**Théorème 1.** Pour une matrice  $A$  quelconque, soit :

$$\delta_i = \det \begin{pmatrix} a_{1,1} & \dots & a_{1,i} \\ \vdots & \ddots & \vdots \\ a_{i,1} & \dots & a_{i,i} \end{pmatrix}, \quad 1 \leq i \leq n$$

On suppose que  $\delta_i \neq 0$ ,  $1 \leq i \leq n$ . Alors la matrice  $A$  se factorise en un produit  $L \cdot R$  où  $R$  est une matrice triangulaire supérieure et  $L$  est une matrice triangulaire inférieure dont les coefficients diagonaux sont égaux à l'unité. De plus, la matrice  $R$  est inversible et la factorisation est unique.

#### **Remarque**

Le fait que la factorisation  $\mathbf{L} \cdot \mathbf{R}$  soit possible correspond à une méthode de Gauss dans laquelle on choisit comme pivot, à chaque étape  $k$ , les coefficients  $a_{k,k}$ , pour  $k = 1, 2, \dots, n-1$ , c'est-à-dire que l'on n'effectue jamais de permutation de ligne dans la mesure où cette situation permet de montrer que  $a_{k,k} \neq 0$ , pour  $k = 1, 2, \dots, n-1$ . De plus on vérifie aisément que, dans ce cas, la structure bande initiale est conservée, c'est-à-dire que si  $A = (a_{i,j})$  vérifie  $a_{i,j} = 0$  pour  $|i-j| > \ell$  où  $\ell$  est la demi-largeur de bande, alors  $\mathbf{L} = (\ell_{i,j})$  et  $\mathbf{R} = (r_{i,j})$  satisfont  $r_{i,j} = \ell_{i,j} = 0$  pour  $|i-j| > \ell$ , ce qui conduit à reformuler une version bande de la méthode de Gauss. Ainsi, on sait que la méthode d'élimination conserve la structure bande initiale, à condition toutefois de ne jamais permutez de lignes, ce qui est assuré grâce au résultat donné par le théorème 1. Par exemple, après discréttisation convenable, cette propriété se trouve encore conservée pour certain type de matrice autre que les matrices symétriques définies positives [2].

De plus, dans ce contexte, le nombre d'opérations arithmétiques est de l'ordre de  $\dim(A) \cdot \ell^2$ , ce qui diminue notablement la complexité de l'algorithme. Cependant, si l'on veut calculer la solution de l'EDP de manière très précise, il est nécessaire de considérer des maillages très fins et le coût de résolution du système linéaire peut devenir prohibitif ; il découle de cet état de fait que l'inversion de la matrice intervenant dans un système linéaire issu de la discréttisation d'une équation aux dérivées partielles conduit à :

- des temps de calculs particulièrement importants ;
  - des résultats de calcul peu précis.

En effet, pour ce dernier point, l'accumulation des erreurs d'arrondi, dues à la mauvaise représentation des nombres réels en machine, peut, comme on l'a indiqué au paragraphe 4 de l'article [AM 500] complètement dénaturer le résultat calculé, alors même que l'approximation conduit théoriquement à des résultats acceptables. Autrement dit, de petites perturbations sur le calcul des coefficients du système linéaire peuvent entraîner de grandes perturbations sur les valeurs calculées ; ce phénomène numérique, difficilement prévisible *a priori*, s'ajoute à l'imprécision découlant du procédé de discréétisation. Cette sensibilité à la propagation et à l'amplification des erreurs intervient lorsque la matrice  $A$  est mal conditionnée et le nombre de conditionnement  $C(A)$  permet d'avoir un indicateur sur la difficulté d'inversion numérique de la matrice ; si  $A$  est une matrice symétrique, et en considérant des normes matricielles induites par la norme euclidienne, on a vu au corollaire 3 de l'article [AF 500] que  $C_2(A) = \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)}$ , où  $\lambda_{\max}(A)$  et  $\lambda_{\min}(A)$  représentent respectivement la plus grande et la plus petite valeur propre de la matrice  $A$ . Une matrice bien conditionnée correspond à une valeur de  $C_2(A)$  égale ou proche de l'unité ; une

matrice mal conditionnée correspond à une valeur de  $C_2(A)$  très grande et, dans le cas du problème de Poisson discrétisé, on a montré que le nombre de conditionnement tendait vers l'infini lorsque la dimension de la matrice augmentait.

On préfère généralement inverser les systèmes linéaires issus de la discrétisation des équations aux dérivées partielles par une méthode itérative qui limite les effets du mauvais conditionnement, ainsi que le nombre d'opérations arithmétiques ; cependant, comme nous le verrons, l'effet du mauvais conditionnement se traduira par une vitesse de convergence plus lente. Il faut également avoir conscience que l'utilisation des méthodes itératives nécessite non seulement l'étude de la convergence des algorithmes mais encore celle de la vitesse de convergence ainsi que la détermination de tests d'arrêt des itérations fiables. On distingue plusieurs types de méthodes itératives :

- les méthodes de relaxation (§ 3) ;
- les méthodes issues de la minimisation de formes quadratiques (§ 4) ;
- les méthodes multigrilles (§ 5) ;
- les méthodes de décomposition de domaine (§ 6).

### 3. Méthodes itératives de relaxation par points et par blocs

Les méthodes de relaxation par points consistent à transformer l'équation  $AU = F$  en une équation de point fixe comme suit :

$$u_i = \frac{f_i - \sum_{j \neq i} a_{i,j} u_j}{a_{i,i}}, \quad \forall i$$

ce qui peut s'écrire matriciellement :

$$U = BU + c$$

avec  $B$  matrice (d'itération) de même dimension que la matrice  $A$ ,  $c$  vecteur défini par la transformation de point fixe.

Ainsi  $U^0$  étant donné, on génère l'itération :

$$U^{k+1} = BU^k + c, \text{ pour } k = 0, 1, 2, \dots$$

On a le critère général de convergence suivant :

**Théorème 2.** Une condition nécessaire et suffisante pour que l'itération :

$$\begin{cases} U^0 \text{ donné quelconque,} \\ U^{k+1} = BU^k + c, \text{ pour } k = 0, 1, 2, \dots \end{cases}$$

converge, quel que soit  $U^0$ , est que le rayon spectral de la matrice d'itération  $\rho(B) = \max_{1 \leq i \leq \dim(B)} |\lambda_i(B)|$  soit de module strictement inférieur à l'unité.

**Définition 1.** On appelle vitesse asymptotique de convergence de l'itération linéaire :

$$\begin{cases} U^0 \text{ donné quelconque,} \\ U^{k+1} = BU^k + c, \text{ pour } k = 0, 1, 2, \dots \end{cases}$$

la quantité définie par  $E(B) = -\ln(\rho(B))$ .

Les méthodes classiques de relaxation par points sont définies comme suit ; on considère la décomposition (en anglais *splitting*) suivante de la matrice  $A$  :

$$A = D - E - F$$

$$A = \begin{bmatrix} & & & & & \\ & \ddots & & & & \\ & & -F & & & \\ & & & D & & \\ & & & & \ddots & \\ & & & & & -E \\ & & & & & & \ddots \end{bmatrix}$$

avec  $D$  diagonale de la matrice  $A$ .

La **méthode de Jacobi par points** est définie par :

$$u_i^{k+1} = \frac{f_i - \sum_{j \neq i} a_{i,j} u_j^k}{a_{i,i}}, \quad \forall i \Leftrightarrow B = J = D^{-1}(E + F) = I - D^{-1}A$$

La **méthode de Gauss-Seidel par points** est définie par :

$$u_i^{k+1} = \frac{f_i - \sum_{j=1}^{i-1} a_{i,j} u_j^{k+1} - \sum_{j=i+1}^n a_{i,j} u_j^k}{a_{i,i}}, \quad \forall i \Leftrightarrow B = \mathcal{L}_1 = (D - E)^{-1}F$$

La **méthode de surrelaxation par points** (SOR : Successive Over-Relaxation) est définie par :

$$\begin{cases} u_i^{k+\frac{1}{2}} = \frac{f_i - \sum_{j=1}^{i-1} a_{i,j} u_j^{k+1} - \sum_{j=i+1}^n a_{i,j} u_j^k}{a_{i,i}}, \\ u_i^{k+1} = (1 - \omega) u_j^k + \omega u_i^{k+\frac{1}{2}}, \quad 0 < \omega < 2 \end{cases} \quad \forall i \Leftrightarrow B = \mathcal{L}_\omega = (D - \omega E)^{-1}((1 - \omega)D + \omega F)$$

#### Remarque

Il est à noter que, lorsque le paramètre  $\omega$  est égal à 1, on retrouve la méthode de Gauss-Seidel par points. Par ailleurs, l'introduction du paramètre  $\omega$  est destinée à augmenter la vitesse de convergence de l'algorithme, lorsque cela est possible.

D'autre part, on a vu dans les articles [AF 500] et [AF 501] que les matrices de discrétisation associées aux problèmes modèles avaient une structure par blocs. On peut tirer parti de cette structure particulière et définir des méthodes de relaxation par blocs ; considérons le cas où la matrice  $A$  se décompose en  $p \times p$  blocs notés  $(A_{i,j})$ ,  $i, j = 1, \dots, p$ , ce qui correspond à la structure suivante :

$$A = \begin{pmatrix} A_{1,1} & A_{1,2} & \dots & A_{1,p} \\ A_{2,1} & A_{2,2} & \dots & A_{2,p} \\ \vdots & & & \vdots \\ A_{p,1} & A_{p,2} & \dots & A_{p,p} \end{pmatrix}$$

et supposons, de plus, que les blocs diagonaux soient des matrices carrées inversibles ; décomposons également les vecteurs  $U$  et  $F$  suivant une décomposition par blocs compatible avec celle de  $A$  ;

soient  $U_i$  et  $F_i$ ,  $i = 1, \dots, p$ , les blocs composantes de ces vecteurs. Avec ces notations, on peut, comme précédemment, définir l'équation de point fixe associée au problème  $A U = F$ . On aura, par exemple :

$$A_{i,i} U_i = F_i - \sum_{j \neq i} A_{i,j} U_j, \quad \forall i \in \{1, \dots, p\}$$

■ On définit la **méthode de relaxation par blocs** (SOR) de manière analogue à la méthode par point, ce qui conduit, dans ce cas, à l'écriture suivante :

$$\begin{aligned} A_{i,i} U_i^{k+1} &= \omega \left( F_i - \sum_{j < i} A_{i,j} U_j^{k+1} \right) + (1 - \omega) A_{i,i} U_i^k \\ &\quad - \omega \sum_{j > i} A_{i,j} U_j^k, \quad \forall i \in \{1, \dots, p\}, \quad 0 < \omega < 2 \end{aligned}$$

Pour calculer les composantes  $U_i^{k+1}$ , il est nécessaire de résoudre un système linéaire du type  $A_{i,i} U_i^{k+1} = G_i$ , ce dernier vecteur  $G_i$  étant parfaitement déterminable. Il faut donc une résolution qui soit relativement aisée. Si l'on utilise une décomposition du type  $\mathbf{L}_i \cdot \mathbf{R}_i$  basée sur la décomposition de Gauss de la matrice  $A_{i,i}$ , il suffira d'effectuer une seule fois (lors de l'initialisation de l'algorithme) cette décomposition ; en effet, si l'on stocke les facteurs  $\mathbf{L}_i$  et  $\mathbf{R}_i$ , il suffira de résoudre  $\mathbf{L}_i \cdot \mathbf{R}_i U_i^{k+1} = G_i$  à chaque itération  $k$ .

#### Remarque

Dans la méthode de relaxation précédente, si  $\omega = 1$ , on retrouve la méthode de Gauss-Seidel par blocs. La formulation de la méthode de Jacobi par blocs s'écrit comme suit :

$$A_{i,i} U_i^{k+1} = F_i - \sum_{j \neq i} A_{i,j} U_j^k, \quad \forall i \in \{1, \dots, p\}$$

et le principe de résolution par blocs est le même.

#### Remarque

Les méthodes par blocs nécessitent plus d'opérations arithmétiques que les méthodes par points ; cependant, dans les applications, on vérifie que la vitesse asymptotique de convergence des itérations par blocs est supérieure à celle de leurs homologues par points.

#### Remarque

Comme pour les méthodes par points, on peut associer à la matrice  $A$  une décomposition (*splitting*) du type  $A = \mathbf{D} - \mathbf{E} - \mathbf{F}$ , où ici les matrices  $\mathbf{D}$ ,  $\mathbf{E}$  et  $\mathbf{F}$  sont définies par blocs, conformément à la structure de la matrice  $A$ .

■ Concernant l'**étude de la convergence**, on a les résultats suivants :

**Théorème 3.** Pour toute matrice  $A$ , une condition nécessaire de convergence de la méthode de relaxation par points ou par blocs est que  $0 < \omega < 2$ .

**Exemple 2 :** on peut appliquer le résultat du théorème 2 pour étudier la convergence de la méthode de Jacobi par points ; on vérifie, par un calcul simple que :

$$\rho(J_{1D}) = \rho(J_{2D}) = \rho(J_{3D}) = \max_{1 \leq i \leq \dim(B)} |\lambda_i(B)| = \cos\left(\frac{\pi}{n+1}\right) = \cos(\pi h)$$

et, par conséquent, cette itération converge.

On citera, plus loin, des résultats garantissant la convergence des méthodes de Gauss-Seidel et de surrelaxation par points ou par blocs (avec paramètre optimal) utilisables lorsque la méthode de Jacobi par points est convergente.

Compte tenu de la difficulté pratique de déterminer  $\rho(B)$  dans le cas général, et du surcoût de calcul pour déterminer cette quantité, on préfère utiliser des conditions suffisantes de convergence plus faciles à vérifier dans le cas des méthodes de relaxation classiques telles que la méthode de Jacobi, Gauss-Seidel, surrelaxation (SOR) par points ou par blocs. On donne ici un résumé de quelques résultats.

**Théorème 4.** Soit  $A$  une matrice symétrique décomposée par points ou par blocs sous la forme  $A = D - E - F$ , avec  $D$  définie positive. Alors la méthode de surrelaxation par points ou par blocs, pour  $\omega \in ]0,2[$ , est convergente si et seulement si  $A$  est définie positive.

**Corollaire 1.** Soit  $A$  une matrice symétrique définie positive ; alors la méthode de surrelaxation par points ou par blocs converge pour  $\omega \in ]0,2[$ .

#### Remarque

En particulier, la méthode de Gauss-Seidel par points ou par blocs, correspondant au cas  $\omega = 1$ , converge.

**Théorème 5.** Soit  $A$  une matrice à diagonale strictement dominante ; alors la méthode de Jacobi par points converge.

**Théorème 6.** Soit  $A$  une matrice à diagonale dominante irréductible ; alors la méthode de Jacobi par points et la méthode de Gauss-Seidel par points convergent.

#### Remarque

Les résultats des théorèmes 5 et 6 s'étendent au cas de la méthode de surrelaxation par points pour  $\omega \in ]0,1]$ .

Compte tenu des résultats obtenus au paragraphe 4 de l'article [AF 500], il est clair que l'une ou l'autre des propriétés précédentes se trouvent vérifiées par les matrices issues de la discréttisation des équations aux dérivées partielles ; il existe d'autres conditions suffisantes de convergence prenant notamment en compte les propriétés de diagonale dominance irréductible de la matrice et applicables dans le contexte de la simulation numérique que nous ne développons pas ici, renvoyant le lecteur à la littérature [2].

Les méthodes de relaxation par blocs ont été étudiées dans le cas de matrices issues de la discréttisation d'équations aux dérivées partielles ; indiquons ci-dessous quelques résultats spécifiques à ce type de situation.

**Théorème 7.** Soit  $A$  une matrice tridiagonale par blocs, dont les blocs diagonaux sont inversibles. Si  $\lambda(J)$  est valeur propre de la matrice d'itération par blocs, alors  $-\lambda(J)$  est aussi valeur propre. Les méthodes de Jacobi et Gauss-Seidel par blocs convergent ou divergent simultanément. Dans le cas de convergence,  $\rho(L_1) = (\rho(J))^2$ , où  $L_1$  est la matrice de Gauss-Seidel par blocs.

#### Remarque

Dans ces conditions, conformément à la définition de la vitesse asymptotique de convergence (cf. définition 1 ci-dessus), la méthode de Gauss-Seidel par blocs converge deux fois plus vite que la méthode de Jacobi par blocs.

**Théorème 8.** Soit  $A$  une matrice tridiagonale par blocs dont les blocs diagonaux sont inversibles. Si toutes les valeurs propres de la matrice de Jacobi par blocs associée au partitionnement de  $A$  sont réelles, alors les méthodes de Jacobi et de relaxation par blocs, pour  $0 < \omega < 2$ , convergent ou divergent simultanément. Dans le cas de la convergence, il existe une valeur optimale du paramètre  $\omega$  donnée par :

$$\omega = \omega_{\text{opt}} = \frac{2}{1 + \sqrt{1 - (\rho(J))^2}}, \quad \rho(\mathbf{L}_\omega) = \omega_{\text{opt}} - 1 \quad \text{pour } \omega = \omega_{\text{opt}}$$

qui minimise le rayon spectral de la matrice d'itération de relaxation par blocs  $\mathbf{L}_\omega$ .

#### Remarque

Pour cette valeur de  $\omega$ , la vitesse asymptotique de convergence de la méthode de relaxation par blocs est optimale, conformément à la définition 1 ci-dessus.

Grâce aux résultats précédents associés à ceux présentés au paragraphe 4 de l'article [AF 500], en particulier les valeurs propres des matrices de discréttisation, on peut calculer les rayons spectraux des matrices d'itération par points et par blocs et obtenir des estimations des vitesses asymptotiques de convergence. Cette vérification, au demeurant très simple, est laissée à titre d'exercice. On constate effectivement la supériorité des méthodes de relaxation lorsque le paramètre est optimal.

■ Expérimentalement on constate que les méthodes de relaxation ont en général de bonnes performances dans le cas de matrices non symétriques, situation que l'on retrouve en simulation numérique lorsque l'on résout le problème de convection-diffusion évolutif avec, par exemple, des conditions aux limites de Dirichlet homogènes :

$$\begin{cases} \frac{\partial u(x,t)}{\partial t} + b \frac{\partial u(x,t)}{\partial x} - a \frac{\partial^2 u(x,t)}{\partial x^2} = 0, \quad 0 < x < 1, \quad t > 0 \\ u(0, t) = u(1, t) = 0, \quad t > 0 \\ u(x, 0) = u_0(x), \quad 0 < x < 1 \end{cases}$$

Les méthodes de relaxation ont également de bonnes performances dans le cas de la résolution de problèmes non linéaires, les applications types étant relatives au traitement d'images ainsi qu'aux mathématiques financières.

## 4. Méthodes issues de la minimisation de formes quadratiques

On considère la minimisation d'une forme quadratique du type :

$$J(U) = \text{Min}(J(V), V \in \mathbb{R}^{\dim(A)})$$

avec :

$$J(U) = \frac{1}{2} U^t A U - U^t F$$

où  $A = A^t$  (symétrie), ce qui correspond à une hypothèse non restrictive, et définie positive qui assure la stricte convexité de  $J(V)$  ; dans ce contexte théorique minimal, on est assuré de l'existence et de l'unicité du minimum, ce dernier étant caractérisé, dans le cas d'optimisation sans contrainte, par l'équation d'Euler  $J'(U) = 0$  ; on vérifie alors aisément que la valeur de  $U$  qui rend minimale  $J$  est caractérisée par la relation  $J'(U) = AU - F = 0$ , c'est-à-dire que  $U$  est solution du système linéaire  $AU = F$ . Ainsi minimiser  $J(U)$  revient à résoudre ce système linéaire. Le problème revient donc à déterminer des algorithmes efficaces de minimisation de formes quadratiques.

■ La méthode la plus simple et la plus connue est la **méthode de plus profonde descente** [3] ; malheureusement, cette méthode est peu efficace dans le cas de systèmes algébriques linéaires issus de la discréttisation d'équations aux dérivées partielles, compte tenu du fait que, dans ce cas, le nombre de conditionnement  $C(A)$  est élevé, ce dernier nombre permettant d'estimer la vitesse asymptotique de convergence de l'algorithme, donnée par l'estimation :

$$E(U^k) \leq \left( \frac{C_2(A) - 1}{C_2(A) + 1} \right)^{2k} E(U^0), \quad k \geq 0$$

avec  $E(V) = \langle A(V - U), V - U \rangle$ ,

$U^k$  itéré n°  $k$ ,

$$C_2(A) = \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)} \quad \text{nombre de conditionnement.}$$

Ainsi  $C_2(A) \rightarrow \infty$  avec  $\dim(A)$  et l'algorithme de plus profonde descente est peu efficace pour inverser des systèmes algébriques linéaires issus de la discréttisation d'équations aux dérivées partielles ; c'est la raison pour laquelle nous n'insistons pas sur cette méthode, renvoyant le lecteur à [3].

■ La méthode la plus utilisée est la **méthode du gradient conjugué** (GC) ; il s'agit d'une méthode de plus profonde descente où, à chaque itération, on corrige les directions de descente de manière à accélérer la convergence de la méthode ; cet algorithme est décrit comme suit :

Initialisation  $\begin{cases} U^0 \text{ donné} \\ P^0 = R^0 = F - AU^0 \end{cases}$

Pour  $k = 0, 1, 2, \dots$

$$\alpha^k = \frac{\|R^k\|^2}{\langle AP^k, P^k \rangle}$$

$$U^{k+1} = U^k + \alpha^k P^k$$

$$R^{k+1} = R^k - \alpha^k AP^k$$

$$\beta^{k+1} = \frac{\|R^{k+1}\|^2}{\|R^k\|^2}$$

$$P^{k+1} = R^{k+1} + \beta^{k+1} P^k$$

fin pour;

avec  $\|\cdot\|$  norme euclidienne.

Cette méthode est très simple à programmer ; en effet les procédures nécessaires à son implantation sur ordinateur sont :

- une procédure optimisée de calcul du produit matrice-vecteur ;
- une procédure de calcul du produit scalaire de deux vecteurs ;
- une procédure de calcul de la combinaison linéaire de deux vecteurs.

**Théorème 9.** Soit  $A$  une matrice symétrique définie positive de dimension  $\dim(A)$ . Alors la méthode du gradient conjugué, appliquée à la résolution du système algébrique linéaire  $AU = F$ , converge, en arithmétique exacte, en  $\dim(A)$  itérations au plus. De plus, la vitesse de convergence est majorée par :

$$E(U^k) \leq 4 \left( \frac{\sqrt{C_2(A)} - 1}{\sqrt{C_2(A)} + 1} \right)^{2k} E(U^0), \quad k \geq 0$$

avec  $E$ , définie par  $E(V) = \langle A(V - U), V - U \rangle$ .

#### Remarque

Il convient de préciser également que le nombre maximal d'itérations est de  $\dim(A)$  en arithmétique exacte ; cependant, les nombres réels étant mal représentés en machine, le nombre

d'itérations pour résoudre le système linéaire peut être supérieur à  $\dim(A)$  ; cela provient d'un défaut de normalité entre les directions de descente corrigées.

#### Remarque

Si la matrice est mal conditionnée,  $C_2(A)$  est grand et la convergence de l'algorithme du gradient conjugué est lente ; par contre si  $C_2(A)$  est proche de 1, la matrice est bien conditionnée et la méthode du gradient conjugué converge rapidement.

■ Cette remarque va être le point de départ de l'étude d'un procédé d'accélération de la convergence de l'algorithme du gradient conjugué. En effet, un autre intérêt de cette méthode, réside dans le fait que l'on peut augmenter la vitesse de convergence de l'algorithme en préconditionnant la matrice  $A$ , artifice qui consiste à remplacer la résolution du système algébrique linéaire initial,  $AU = F$ , par celle du système suivant :

$$C^{-1}AU = C^{-1}F$$

que l'on peut encore écrire :

$$C^{1/2}(C^{-1}A)C^{1/2}U = C^{-1/2}F$$

où  $C^{-1}$  est la matrice de préconditionnement, matrice symétrique définie positive, de telle sorte que le système équivalent :

$$\tilde{A}\tilde{U} = \tilde{F}$$

soit défini également avec la matrice  $\tilde{A} = C^{-1/2}AC^{-1/2}$ , matrice symétrique définie positive, semblable à la matrice  $C^{-1}A$ , où  $\tilde{U} = C^{1/2}U$  et  $\tilde{F} = C^{-1/2}F$ , la matrice  $C$  étant choisie de telle sorte que :

$$C(\tilde{A}) \ll C(A)$$

La détermination de la matrice  $C$  est un problème non trivial.

La valeur idéale est  $C(\tilde{A}) = 1$  et, en pratique,  $\tilde{A}$  est une approximation de l'identité, la matrice  $C^{-1}$  étant une approximation de  $A^{-1}$  ; en pratique, il faudra trouver  $C^{-1}$  la plus proche de  $A^{-1}$ , sans que les calculs pour déterminer  $C^{-1}$  soient trop coûteux. Si l'on réécrit l'algorithme du gradient conjugué préconditionné pour la résolution du système linéaire  $\tilde{A}\tilde{U} = \tilde{F}$ , on peut exprimer les étapes de calcul directement à partir de la donnée de la matrice  $A$  et des données  $U$  et  $F$ . L'algorithme exprimé directement à partir des données  $A$ ,  $U$  et  $F$  s'écrit :

$$\text{Initialisation} \left\{ \begin{array}{l} U^0 \text{ donné} \\ R^0 = F - AU^0 \\ CP^0 = R^0 \\ Z^0 = P^0 \end{array} \right.$$

Pour  $k = 0, 1, 2, \dots$

$$\alpha^k = \frac{\langle R^k, Z^k \rangle}{\langle AP^k, P^k \rangle}$$

$$U^{k+1} = U^k + \alpha^k P^k$$

$$R^{k+1} = R^k - \alpha^k AP^k$$

$$CZ^{k+1} = R^{k+1}$$

$$\beta^{k+1} = \frac{\langle R^{k+1}, Z^{k+1} \rangle}{\langle R^k, Z^k \rangle}$$

$$P^{k+1} = Z^{k+1} + \beta^{k+1} P^k$$

fin pour;

On constate que, à chaque itération, il est nécessaire, en plus des opérations habituelles, de résoudre un système de la forme :

$$CZ = R$$

Il est donc indispensable que cette résolution soit facile et peu coûteuse, ce qui sera le cas si la matrice  $C$  est une matrice diagonale ou encore le produit de matrices triangulaires. Parmi les choix classiques de matrices de préconditionnement, on retient les trois méthodes suivantes.

- **Préconditionnement diagonal** où  $C = \text{Diag}(A) = D$ , qui, en pratique est peu efficace ;

- **Préconditionnement SSOR (Symmetric Successive Over-Relaxation)** d'Evans où :

$$C = \frac{1}{\omega(2-\omega)} (D - \omega E) D^{-1} (D - \omega E)^t$$

avec  $E$  partie strictement triangulaire inférieure de  $A$ , avec  $A = D - E - E^t$ ,

$\omega$  paramètre réel strictement compris entre 0 et 2.

On constate que, dans ce cas,  $C$  est obtenue directement en fonction de  $A$  ; aucun stockage ni calcul supplémentaire n'est nécessaire. La résolution du système  $CZ = R$  nécessite de plus la résolution d'un système triangulaire inférieur, d'un système diagonal et d'un système triangulaire supérieur. On peut, de plus, déterminer la valeur d'un paramètre  $\omega$  optimal qui minimise la valeur de  $C(\tilde{A})$  [1].

**Exemple 3** : dans le cas de l'équation de Poisson discrétisée avec un maillage différences finies uniforme de pas  $h$ , ce paramètre  $\omega$  optimal est donné par :

$$\omega = \omega_{\text{opt}} = \frac{2}{1 + \sqrt{1 - (\rho(J))^2}}$$

avec  $J = I - D^{-1}A$ ,  
 $\rho(J) = \text{Max}|\lambda_i(J)|$ .

Par un calcul direct, on vérifie que  $C(\tilde{A}) = O\left(\frac{1}{h}\right)$  alors que

$$C(A) = O\left(\frac{1}{h^2}\right) ; \text{ ainsi on obtient bien que } C(\tilde{A}) \ll C(A).$$

- **Préconditionnement basé sur la factorisation incomplète de Cholesky**

Cette technique utilise le couplage d'une méthode itérative et d'une méthode directe comme la méthode de Gauss ou la méthode de Cholesky dans le cas de matrice symétrique définie positive. On sait que si l'on considère la factorisation de Cholesky de la matrice symétrique définie positive  $A$ , on peut écrire  $A = L\Delta L^t$ , où  $L$  est triangulaire inférieure dont les éléments diagonaux sont égaux à l'unité et  $\Delta$  est une matrice diagonale ; de plus les matrices  $L$  et  $\Delta$  sont parfaitement calculables [3]. Dans le cas d'une matrice  $A$  creuse à structure bande, la matrice  $L$  ne conserve pas le caractère creux de la matrice  $A$  mais se remplit à l'intérieur de la bande ; ce phénomène de remplissage est connu, dans la terminologie anglo-saxonne sous le nom de phénomène de *fill-in* :

$$A = \begin{pmatrix} a & b & c & & & \\ b & \ddots & & & & \\ & \ddots & \ddots & 0 & & \\ & & \ddots & 0 & \ddots & \\ & & & \ddots & c & \\ c & & & & \ddots & \\ & 0 & & & & \ddots \\ 0 & . & & & & . \\ & & \ddots & b & & \\ & & & c & b & a \end{pmatrix}, \quad L = \begin{pmatrix} \alpha & & & & & \\ \beta & . & & & & 0 \\ \delta & . & . & & & \\ \varepsilon & . & . & . & & \\ \mu & . & . & . & . & \\ & . & . & . & . & . \\ & 0 & . & . & . & . \\ & & 0 & . & . & . \\ & & & \mu & \varepsilon & \delta \beta \alpha \end{pmatrix}$$

De plus on constate que, après transformation de Cholesky ou de Gauss, lorsque l'on s'éloigne de la diagonale principale et de la diagonale externe, les modules des coefficients deviennent de plus en plus petits, les termes les plus petits se trouvant au milieu de la bande ; dans le cas de la représentation précédente de la matrice  $L$ , les modules des termes  $\delta$  et  $\varepsilon$  sont négligeables devant ceux des termes  $\alpha$ ,  $\beta$  et  $\mu$ . Cette constatation suggère de négliger certains petits termes et de conserver pour la matrice  $L$  une structure quasiment semblable à celle de la matrice  $A$ . On se donne donc *a priori* un ensemble d'indices fixés  $G$  et on cherche deux matrices  $C$  et  $R$  telles que :

$$A = C - R, \quad C = TT^t$$

avec, par exemple :

$$T = \begin{pmatrix} \alpha' & & & \\ \beta' & . & & 0 \\ 0 & . & & \\ 0 & . & . & \\ \mu' & . & . & . \\ \cdot & . & . & . \\ \cdot & . & . & . \\ 0 & . & . & . \\ & \mu' & 0 & 0 & \beta' & \alpha' \end{pmatrix}$$

$T$  est alors une triangulaire inférieure, telle que  $t_{i,j} = 0$  si  $(i,j) \notin G$  et  $i \neq j$ .  $G$  définit donc l'ensemble des éléments *a priori* non nuls de  $T$ . On peut déterminer les matrices  $T$ , en écrivant que le produit  $TT^t$  correspond à la décomposition exacte de la matrice  $A + R$ . Notons que par construction on a :

$$r_{i,i} = 0 \text{ et } r_{i,j} \neq 0 \text{ si } (i,j) \in G : \text{ donc } c_{i,i} = a_{i,i} \text{ et } c_{i,j} = a_{i,j} \text{ si } (i,j) \in G$$

On sait caractériser des classes de matrices pour lesquelles la décomposition de Cholesky précédente, connue sous le nom de décomposition de Cholesky incomplète, est stable quelle que soit  $G$ , qui correspondent à une décomposition pour laquelle les pivots sont strictement positifs à chaque pas de la transformation. C'est le cas si  $A$  est diagonale dominante stricte ou diagonale dominante irréductible ou encore telle que  $a_{i,j} \leq 0$ ,  $i \neq j$  et  $A^{-1} \geq 0$ , propriété vérifiée par les matrices issues de la discréttisation du problème de Poisson avec conditions aux limites de Dirichlet [2].

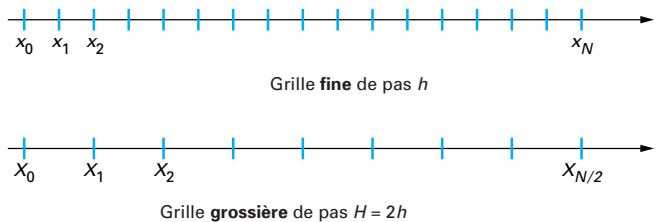
En faisant varier  $G$  on obtient une infinité de décompositions incomplètes. L'inconvénient de la factorisation incomplète de Cholesky est que l'on ne sait pas démontrer en général, même dans des cas simples, que l'on améliore le conditionnement de la matrice  $A$ . On constate, expérimentalement, par des essais numériques que c'est effectivement le cas sans disposer de raisons mathématiques précises.

#### Remarque

Le succès de la méthode du gradient conjugué a incité de nombreux auteurs à proposer certaines variantes de cette méthode pour la résolution de systèmes linéaires dont la matrice n'est pas symétrique définie positive. La plus simple consiste à remplacer l'équation  $AU = F$  par l'équation dite normale  $A^tAU = A^tF$  ; dans ce cas, la matrice  $A^tA$  est bien symétrique définie positive, et il suffit d'adapter l'algorithme du gradient conjugué à cette situation particulière. Cependant cet artifice présente de gros inconvénients ; en effet d'une part  $C_2(A^tA) = C_2(A)$  et, par conséquent, la vitesse de convergence de la méthode du gradient conjugué est plus faible ; d'autre part, à chaque itération, il est nécessaire d'effectuer deux produits supplémentaires d'une matrice par un vecteur, ce qui alourdit le calcul. Il existe d'autres variantes de la méthode du gradient conjugué mieux adaptées au cas où la matrice n'est pas symétrique [2] ; une des plus efficaces est la méthode GMRES (*Generalized Minimum Residual Method*) pour laquelle on renvoie à [6].

## 5. Méthode multigrille

Pour exposer le plus simplement possible le principe de la **méthode multigrille**, on commence par présenter la **méthode 2-Grilles** ; cette dernière combine deux méthodes peu performantes pour finalement obtenir une méthode très efficace. On considère une grille fine de pas  $h$  et une grille grossière de pas  $H = 2h$ , ainsi que la résolution du problème de Poisson monodimensionnel avec conditions aux limites de Dirichlet homogènes.



Un cycle de la méthode 2-Grilles se compose de deux phases :

— une **méthode de lissage**, correspondant à deux ou trois itérations d'une méthode de relaxation, permettant de réduire les hautes fréquences de l'erreur lorsque l'on décompose celle-ci dans la base des vecteurs propres de la matrice ; soit  $U_h$  l'approximation de  $U = U_h$  solution du système linéaire discret  $A U_h = F_h$  ;

— une **méthode de correction sur grille grossière** qui traite efficacement les basses fréquences de l'erreur dans la mesure où un mode basse fréquence sur la grille fine se transformera à terme en un mode oscillant sur la grille grossière et sera, par conséquent bien lissé par une méthode de relaxation, conformément à ce qui a été indiqué à l'alinéa précédent ; la correction  $V_h = U_h - u_h$ , qu'il faut ajouter à  $u_h$  pour obtenir  $U_h$ , doit vérifier :

$$A V_h = r_h, \quad r_h = F_h - A u_h \text{ (résidu)}$$

$V_h$  est donc la solution d'un problème du même type que celui qui définit  $U_h$ , où le second membre a été remplacé par le résidu  $r_h$  aux points de la grille fine. Le calcul de  $V_h$  est *a priori* aussi coûteux que celui de  $U_h$ . Dans la mesure où l'erreur est lissée, on peut cependant chercher à obtenir une approximation de  $V_h$  sur la grille grossière de pas  $H$ , ce qui nécessitera beaucoup moins de calculs puisque la grille de pas  $H$  possède environ deux fois moins de points dans le cas du problème monodimensionnel (quatre fois moins, dans le cas du problème bidimensionnel, huit fois moins dans le cas du problème tridimensionnel) que celle de pas  $h$  ; on pourra effectivement obtenir une bonne approximation de  $V_h$  sur la grille grossière si  $V_h$  varie lentement, ce qui est effectivement le cas grâce aux itérations de lissage. La seconde phase de chaque cycle de la méthode 2-Grilles consiste alors à :

1) définir la restriction  $r_H$  de  $r_h$  sur la grille grossière à l'aide d'un opérateur de restriction ; cet opérateur peut être, tout simplement, l'injection définie en tous points  $M$  de la grille grossière, mais on peut également utiliser une restriction par moyenne pondérée ;

2) résoudre le système linéaire  $A V_H = r_H$ , donnant la correction  $V_H$  sur la grille grossière ;

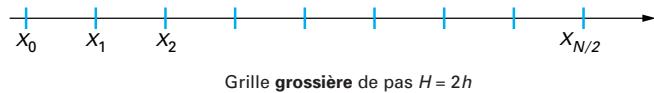
3) prolonger  $V_H$  obtenue sur la grille grossière en une fonction obtenue sur la grille fine, par exemple par interpolation bilinéaire.

On recommence ensuite un nouveau cycle de la méthode 2-Grilles.

La question principale qui se pose est de savoir si, effectivement, le prolongement de  $V_H$  est une bonne approximation de  $V_h$  et, dans ce cas, quel est le gain que l'on va attendre de l'utilisation de la méthode 2-Grilles ; il est à noter que cette méthode 2-Grilles peut s'écrire sous la forme d'une itération de point fixe linéaire du

type  $U^{k+1} = B U^k + c$ , pour  $k = 0, 1, 2, \dots$ . Pour un lissage correspondant à la méthode de Gauss-Seidel, on montre, par analyse de Fourier, que le facteur de réduction de l'erreur est égal à  $\sqrt{5}/5 = 0,447$ , ce qui correspond à un excellent taux de réduction.

■ La **méthode multigrille** correspond à l'application récursive d'une méthode 2-Grilles. En effet lors de la résolution du système d'équations sur la grille grossière  $A V_H = r_H$ , on peut de nouveau utiliser une méthode 2-Grilles, en définissant un problème sur une grille encore plus grossière de pas  $2H$ , et ainsi de suite pour finalement résoudre un système d'équations sur la grille la plus grossière ne contenant que quelques points, voire même un seul point puisque les pas de discréttisation sont choisis comme des puissances inverses de 2.



Cela nécessite de définir des stratégies de passages intergrilles que nous ne développerons pas ici dans la mesure où le lecteur trouvera de nombreux détails dans la référence [4]. Signalons également que cette méthode multigrille est bien adaptée à une utilisation sur des domaines rectangulaires ; il existe cependant des variantes de la méthode multigrille bien adaptées à la résolution numérique d'équations aux dérivées partielles sur des domaines quelconques [5].

**Exemple 4 :** à titre illustratif, considérons le problème de Poisson défini sur le carré unité avec conditions aux limites de Dirichlet homogènes :

$$\begin{cases} -\Delta u = f(x, y) \in \Omega = [0, 1] \times [0, 1] \\ u = 0, (x, y) \in \partial\Omega \end{cases}$$

La discréttisation de ce problème par différences finies classiques conduit à résoudre un système linéaire ; considérons un pas de discréttisation  $h$  suffisamment fin ainsi qu'une tolérance raisonnable  $tol$  pour arrêter les itérations, par exemple :

$$h = 256^{-1} \Rightarrow \dim(A) = 65\,025 \text{ et } tol = 10^{-4} \approx h^2$$

La comparaison des temps de calcul conduit à dresser le tableau 1.

Les résultats présentés dans le tableau 1 permettent de mesurer la différence de performance des algorithmes précédents ; il convient de noter que les méthodes multigrilles sont favorisées dans ce type de problème, dans la mesure où le domaine  $\Omega$  est ici le carré unité ; si l'on considère un domaine quelconque, les performances de la méthode multigrille diminueront et la mise en œuvre de l'algorithme sera moins simple à réaliser.

Tableau 1 – Comparaison des temps de calculs de solveurs itératifs classiques

Méthode (1)	Temps CPU de calcul (2)
Gauss-Seidel	20 h
SOR $\omega$ optimal	20 min
GC + factorisation incomplète	6 min
Multigrille standard	18 s
Full multigrille	7 s

(1) GC : gradient conjugué

SOR : Successive Over-Relaxation

(2) CPU : Central Processing Unit

## 6. Méthodes de décomposition de domaine

On a vu précédemment que la discréttisation d'équations aux dérivées partielles conduit à la résolution de systèmes algébriques de grande dimension ; les algorithmes présentés précédemment sont, en général et pour la plupart des applications, les plus performants sur les ordinateurs traditionnels. Actuellement, on assiste à une évolution de l'architecture des machines et, depuis une vingtaine d'années, on voit apparaître sur le marché des multiprocesseurs, machines équipées de plusieurs processeurs destinés à travailler en parallèle sur la même application. Bien évidemment, il est nécessaire d'adapter les algorithmes de résolution numérique des équations aux dérivées partielles à ce type de supercalculateur. Il existe plusieurs solutions. Soit on adapte les algorithmes précédents en exécutant en parallèle toutes les phases du programme qui sont indépendantes afin de tirer parti au maximum de l'architecture de la machine ; cet aspect relève plutôt de techniques informatiques et nous n'aborderons pas ici ce sujet.

Une autre façon de tirer avantage des possibilités des machines multiprocesseurs est de décomposer le domaine  $\Omega$  où est définie l'équation aux dérivées partielles en une union de sous-domaines  $\Omega_i$ , de telle sorte que  $\Omega = \cup \Omega_i$ .

■ À ce stade, plusieurs possibilités sont envisageables ; on peut soit considérer que :

$$\Omega_i \cap \Omega_k = \emptyset, \text{ si } i \neq k$$

où  $\emptyset$  est l'ensemble vide et, dans ce cas, on parle de méthode de sous-domaines sans recouvrement, la plus connue étant la **méthode du complément de Schur** qui est une méthode semi-directe correspondant à une méthode d'élimination par blocs, le domaine  $\Omega$  étant partitionné, par exemple, en quatre sous-domaines  $\Omega_i$ ,  $i = 1, \dots, 4$ , disjoints couplés à une interface  $\Gamma$ . Ce mode de décomposition est représenté sur la figure 1, dans le cas de quatre sous-domaines. La matrice de discréttisation a alors l'allure ci-dessous :

$$A = \begin{pmatrix} A_1 & & C_1 & \\ & A_2 & & C_2 \\ & & A_3 & C_3 \\ & & & A_4 C_4 \\ B_1 & B_2 & B_3 & B_4 & D \end{pmatrix}$$

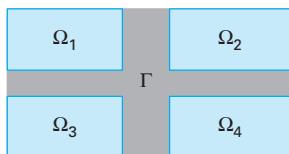


Figure 1 – Décomposition de Schur : décomposition initiale

Les matrices  $A_i$  correspondent à la discréttisation sur les sous-domaines  $\Omega_i$ ,  $i = 1, 2, 3, 4$  et les matrices  $B_i$  et  $C_i$  correspondent au couplage entre ces derniers et l'interface  $\Gamma$ , alors que  $D$  est la matrice de discréttisation définie sur  $\Gamma$ . Si l'on effectue une élimination par blocs, on vérifie qu'après élimination on obtient une matrice du type :

$$\begin{pmatrix} A_1 & C_1 \\ A_2 & C_2 \\ A_3 & C_3 \\ A_4 & C_4 \\ S \end{pmatrix}$$

avec :

$$S = D - \sum_{i=1}^4 B_i A_i^{-1} C_i$$

matrice de complément de Schur.

Donc, une fois la procédure d'élimination par blocs effectuée, on résout en premier lieu le problème sur l'interface  $\Gamma$ , puis on détermine en parallèle les valeurs sur les sous-domaines  $\Omega_i$ , dans la mesure où les quatre systèmes sont indépendants. On montre [6] que si la matrice  $A$  est inversible et si les matrices  $A_i$ ,  $i = 1, 2, 3, 4$  sont inversibles, alors la matrice de complément de Schur est aussi inversible ; de plus si la matrice  $A$  est symétrique définie positive, la matrice de complément de Schur est également symétrique définie positive [6] ; on peut donc déterminer le problème sur l'interface  $\Gamma$ , en utilisant, par exemple, un algorithme de gradient conjugué préconditionné.

On peut ainsi recommander le même type de décomposition sur chaque sous-domaine  $\Omega_i$  et on obtient la décomposition de domaines récursive représentée sur la figure 2.

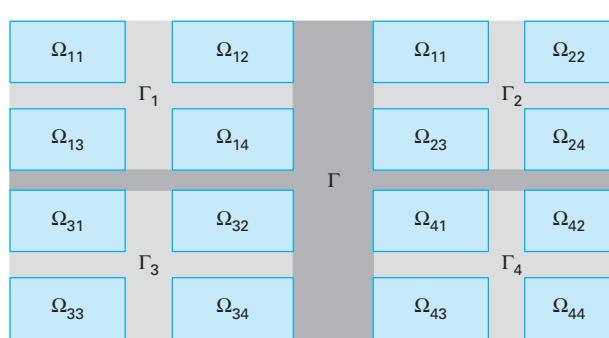


Figure 2 – Décomposition de Schur : décomposition récursive

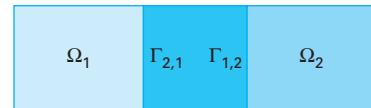


Figure 3 – Décomposition de Schwarz

■ On peut également considérer que  $\Omega_i \cap \Omega_j \neq \emptyset$  si  $i \neq j$ , et il s'agit alors de **méthode de sous-domaines avec recouvrement**, la plus classique d'entre elles étant la **méthode alternée de Schwarz**, qui est une méthode itérative de type relaxation [6]. On illustre, sur la figure 3, le cas d'une décomposition en deux sous-domaines se recouvrant. En partant d'une donnée initiale  $U^0 = (u_1^0, u_2^0)$  connue, l'algorithme consiste à résoudre alternativement le problème sur chaque sous-domaine en prenant comme valeur frontière aux points de recouvrement la restriction en ces points de la valeur obtenue sur l'autre sous-domaine.

**Exemple :** dans le cas du problème de Poisson avec conditions de Dirichlet homogènes sur la frontière, et si l'on considère une décomposition en deux sous-domaines  $\Omega_1$  et  $\Omega_2$ , on aura l'itération du type :

$$\begin{cases} -\Delta u_i^p = f_i, \text{ sur } \Omega_i, i \in \{1, 2\} \\ u_i^p = 0, \text{ sur } \gamma_i, i \in \{1, 2\} \\ u_i^p|_{\Gamma_{ij}} = u_j^{p-1}|_{\Gamma_{ji}}, i, j \in \{1, 2\}, i \neq j \end{cases}$$

avec  $f_i$  restriction de  $f$  au domaine  $\Omega_i$ ,  
 $\gamma_i = \partial\Omega_i \cap \partial\Omega$  restriction de la frontière  $\partial\Omega$  de  $\Omega$  au sous-domaine  $\Omega_i$ ,  
 $\Gamma_{ij}$  frontière de recouvrement entre les sous-domaines  $\Omega_i$  et  $\Omega_j$ .

Dans l'algorithme décrit ci-dessus, on a considéré une stratégie qui s'apparente à la méthode de Jacobi, et on parle de **méthode alternée de Schwarz additive** ; on peut également considérer des stratégies s'apparentant à la méthode de Gauss-Seidel et on parle alors de la **méthode alternée de Schwarz multiplicative**. Les critères de convergence de cette méthode se rapprochent des critères obtenus lors de l'étude des méthodes de relaxation au paragraphe 3.

L'étude des méthodes de sous-domaines soulève actuellement de nombreuses questions mathématiques, comme l'étude de la convergence ou encore de l'accélération de la convergence de ces méthodes par des techniques de préconditionnement. Le lecteur est renvoyé aux références [2] et [6], pour des compléments d'information sur ces questions en pleine évolution actuellement.

Bien sûr, nous n'avons pas fait le tour de tous les problèmes d'analyse et d'algèbre qui sont en relation avec la simulation numérique des équations aux dérivées partielles. Cependant, nous avons donné un bref aperçu d'une problématique abstraite en relation avec un très grand nombre d'applications concrètes.

Pour de plus amples renseignements concernant les algorithmes numériques pour la résolution de grands systèmes, le lecteur pourra consulter les références [7] à [17].

## Références bibliographiques

- [1] BARANGER (J.). – *Analyse numérique*. Hermann (1991).
- [2] MEURANT (G.). – *Computer solution of large linear systems*. North Holland (1999).
- [3] LASCAUX (P.) et THEODOR (R.). – *Analyse numérique matricielle appliquée à l'art de l'ingénieur*. Tomes 1 et 2, Masson (1986).
- [4] McCORMICK (S.). – *Multigrid methods*. SIAM (1987).
- [5] HACKBUSCH (W.). – *Multigrid methods and applications*. Computational Mathematics, Springer-Verlag (1980).
- [6] SAAD (Y.). – *Iterative methods for sparse linear systems*. PWS Publishing Company (1996).
- [7] AXELSON (O.) et BARKER (V.A.). – *Finite element solution of boundary value problems - Theory and computation*. Academic Press (1984).
- [8] AXELSON (O.). – *Iterative solution method*. Cambridge University Press (1996).
- [9] BEREZIN (I.S.) et ZHIDKOV (N.P.). – *Computing methods*. Pergamon Press (1965).
- [10] BRIGGS (W.). – *A multigrid tutorial*. SIAM (1988).
- [11] GASTINEL (N.). – *Analyse numérique linéaire*. Hermann (1966).
- [12] GOLUB (G.H.) et VAN LOAN (C.F.). – *Matrix computation*. The Johns Hopkins Univ. Press. (1983).
- [13] HACKBUSCH (W.) et TROTTERBERG (U.). – *Multigrid methods*. Lecture Notes in Mathematics, Springer-Verlag (1982).
- [14] SMITH (G.D.). – *Numerical solution of partial differential equations : Finite difference methods*. Clarendon Press (1984).
- [15] WESSELING (P.). – *An introduction to multi-grid methods*. John Wiley and Sons (1992).
- [16] WILKINSON (J.H.). – *The algebraic eigenvalue problem*. Clarendon Press (1965).
- [17] YOUNG (D.M.). – *Iterative solution of large linear systems*. Academic Press (1971).