TECHNIQUES
DE L'INGÉNIEUR

Réf. : **AF1375 V1**

Date de publication :
10 avril 2012

Méthodes de décomposition de domaines - Notions de base

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

par **Martin J. GANDER, Laurence HALPERN**

Mots-clés
Décomposition de domaines | Méthodes de Schwarz | Méthodes de Schur | Relaxation d'ondes | Algorithme parareél

Résumé Actuellement dans l'industrie automobile, en médecine ou astrophysique, les problèmes complexes de calculs de structures et de reconnaissance de forme sont résolus sur des calculateurs parallèles composés de centaines de nœuds de calcul. Leurs fonctionnements nécessitent l'utilisation de méthodes de décomposition de domaines. Les premiers modèles de ces méthodes ont été établis par H.A. Schwarz. Le principe consiste à morceler un problème de grande taille en une suite de sous-problèmes de taille plus petite, et donc plus facilement résolus. Depuis leur apparition, les approches ont évolué et des variantes se sont greffées aux modèles de base, aboutissant à différentes qualités de convergence.

Keywords
Domain Decomposition | Schwarz Methods | Schur's Methods | Waveform Relaxation | Parareal Algorithm

Abstract Within the automobile industry, the medical sector or in astrophysics the complex problems of structure calculations and shape recognition are currently solved on parallel calculators composed of hundreds of calculation nodes. The way in which they function requires the use of domain decomposition methods. The first models regarding such methods were defined by H.A. Schwarz. The main principle consists in the breaking down of a large scale problem into a series of smaller problems which thus become easier to solve. Since their creation, these approaches have evolved and variations have been added to the basic models thus leading to various convergence qualities.

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
Par téléphone :
00 33 (0)1 53 35 20 20

Méthodes de décomposition de domaines

Notions de base

par **Martin J. GANDER**
Professeur de mathématiques
Section de Mathématiques, Université de Genève

et **Laurence HALPERN**
Professeur de Mathématiques
Laboratoire Analyse, Géométrie et Applications, Université Paris 13

Tous les problèmes d'ingénierie aujourd'hui sont résolus en parallèle sur des ordinateurs composés de centaines, voire des milliers de noeuds de calcul. Cet article se propose d'exposer les méthodes de décomposition de domaine susceptibles de s'appliquer à ces nouveaux outils. Emile Picard nous enseigne dans [36] que pour comprendre une théorie, il est bon d'avoir en tête un problème modèle.

Les méthodes d'approximation dont nous faisons usage sont théoriquement susceptibles de s'appliquer à toute équation, mais elles ne deviennent vraiment intéressantes pour l'étude des propriétés des fonctions définies par les équations différentielles que si l'on ne reste pas dans les généralités et si l'on envisage certaines classes d'équations.

Nous choisirons donc dans tout cet exposé un fil conducteur, l'équation de la chaleur

$$\partial_t u - \Delta u = f \qquad (1)$$

représentant les variations en temps et en espace de la température d'un corps emplissant le domaine Ω, soumis à une source de chaleur f (qui sera appelée second membre), avec une température initiale donnée dans tout le domaine, et des conditions aux limites sur le bord du domaine $\partial\Omega$, par exemple de Dirichlet (la température est fixée), soit $u = g$. $\partial_t u$ est la dérivée en temps de

u, Δ *est l'opérateur de Laplace,* $\Delta u = \partial_{11}u + \partial_{22}u + \partial_{33}u$. *Pour calculer sur un ordinateur une solution approchée de cette équation, on peut commencer par une* ***semi-discrétisation en temps****. Le schéma le plus simple est le schéma d'Euler implicite (voir [AF 1 220]). Partageons l'intervalle de temps* [0, T] *en sous-intervalles* $[t_n, t_{n+1}]$ *de longueur* Δt. *Notons* $u_n(x)$ *l'approximation de u à l'instant* t_n *au point x, calculée par la formule de récurrence*

$$\frac{1}{\Delta t}u_{n+1} - \Delta u_{n+1} = \frac{1}{\Delta t}u_n + f_{n+1},$$

où f_{n+1} *représente* $f(x, t_{n+1})$. *Pour passer du temps* t_n *au temps* t_{n+1}, *il faut donc résoudre l'****équation elliptique***

$$(\eta - \Delta)u = f$$

dans le domaine Ω, *où f est maintenant une fonction indépendante du temps.*

*La discrétisation en espace de cette équation par une méthode de type éléments finis ou volumes finis mène à un système linéaire (voir [AF 500] et [AF 503]). Lorsque la taille du domaine de calcul est très grande, ou la discrétisation très fine, la taille du système linéaire excède les capacités de stockage et de calcul d'un seul ordinateur, si puissant soit-il. L'idée la plus simple pour remédier à ce problème est de décomposer le système linéaire en sous-systèmes, dont chacun est suffisamment petit pour être résolu très rapidement sur un nœud d'un système d'ordinateurs (*divide et impera*). Cela peut se faire au niveau purement informatique, mais il est plus fructueux de revenir en amont et de développer une stratégie au niveau du problème mathématique. Cette démarche est souvent réclamée par la géométrie elle-même (assemblage de structures par exemple). Le domaine de calcul est alors partagé en sous-domaines, chacun assigné à un nœud de la grappe de calcul. Les échanges entres les sous-domaines sont effectués par des conditions de transmission et traduits par des échanges entre les processeurs. La résolution du problème de départ est alors réalisée en itérant entre les sous-domaines, et les sous-domaines peuvent même être en espace-temps.*

Toutes ces méthodes sont des méthodes de décomposition de domaines. Elles ont pour fondateur H.A. Schwarz qui en écrivit une première version en 1870 [41]. Elles ont donné lieu à une intense activité scientifique depuis l'avènement des calculateurs parallèles. Elles sont utilisées pour des calculs de pneumatiques, d'automobiles, de structures sismiques, de navette spatiale, de reconnaissance de forme, d'environnement, de météorologie, d'astrophysique, de médecine, et tant d'autres.

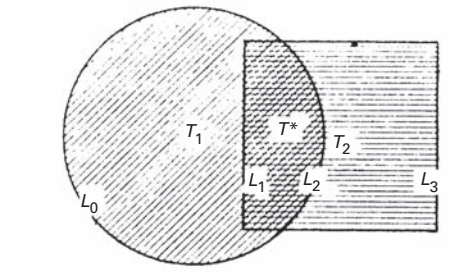
Leur champ d'utilisation est même plus large : si par exemple le modèle a des propriétés physiques différentes dans différentes parties du domaine, les méthodes de décomposition de domaines sont un outil naturel pour leur traitement. Mentionnons par exemple la jonction d'une poutre et d'une plaque, le couplage entre l'océan et l'atmosphè

1. Historique

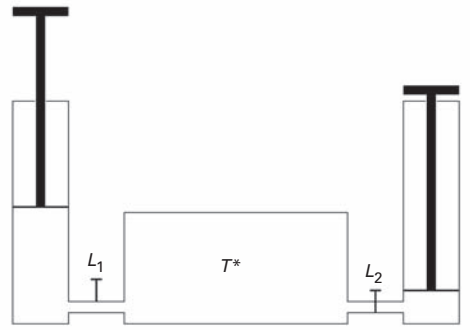
1.1 Méthode de Schwarz

Les méthodes de Schwarz trouvent leur origine dans un algorithme alterné imaginé par H.A. Schwarz pour démontrer l'existence de fonctions harmoniques (*i.e.* solutions deux fois continuement dérivables de l'équation de Laplace $\Delta u = 0$) dans un domaine composite, avec une valeur au bord prescrite *g*. Il commence par considérer un domaine composé d'un disque T_1 et d'un carré T_2 (voir figure **1*a***), domaines pour lesquels une solution explicite au moyen de séries de Fourier existe.

Les frontières respectives de ces deux domaines sont notées ∂T_1 et ∂T_2. La partie de ∂T_1 située dans T_2 est notée L_2, le complémentaire est L_0. La partie de ∂T_2 située dans T_1 est notée L_1, le complémentaire est L_3. Le domaine global T est l'union des deux domaines T_1 et T_2, sa frontière ∂T est constituée de L_0 et L_3. Pour montrer que le problème de Dirichlet



(a) figure originale



(b) interprétation physique

Figure 1 – Figure originale de 1870 pour l'algorithme alterné de H.A. Schwarz et son interprétation physique

$$\Delta u = 0 \quad \text{dans } T, \quad u = g \quad \text{sur } \partial T, \tag{2}$$

a une solution, Schwarz introduit une suite de solutions de problèmes de Dirichlet sur les domaines T_i alternativement. Pour illustrer son propos, il introduit une pompe à air où chaque T_i joue le rôle d'un cylindre, T^* est le réservoir intermédiaire, L_1 et L_2 sont les soupapes (voir figure **1***b*). L'algorithme est initialisé par une donnée constante sur L_2, $\underline{u} = \inf_{L_0} g$. La première étape définit une fonction u_1 harmonique dans T_1, égale à g sur L_0 et à $\underline{u}$ sur L_2.

Le principe du maximum (voir par exemple [14]) affirme qu'une fonction harmonique dans un domaine atteint son maximum sur le bord du domaine. De plus, si le domaine est connexe, et si le maximum est atteint en un point intérieur au domaine, la fonction est constante. En remplaçant la fonction par son opposé, on obtient les mêmes énoncés en remplaçant « maximum » par « minimum ». Cet outil fondamental de l'analyse est utilisé par Schwarz pour montrer la convergence du processus, couplé à un lemme qui en découle, que nous énonçons ici sous la forme donnée dans [29] pour des géométries plus générales.

Lemme 1.1 Soit w une fonction harmonique sur T_1, continue sauf en $\partial T_1 \cap \partial T_2$, prenant la valeur 0 sur L_0 et 1 sur L_2. Alors le maximum q_1 de w sur L_1 est strictement compris entre 0 et 1.

La seule difficulté dans ce lemme est le traitement des points de jonction entre les frontières de T_1 et T_2 (voir [29]). Le même résultat est valable dans T_2 avec une constante q_2.

Par le principe du maximum, u_1 prend ses valeurs dans T_1, et en particulier sur L_1, entre $\underline{u}$ et $\bar{u} = \sup_{L_0} g$. À partir de u_1, Schwarz construit une fonction u_2 harmonique dans T_2, fixée à la valeur g sur L_3 et assujettie sur L_1 à la valeur de u_1 nouvellement calculée. De nouveau, u_2 prend ses valeurs entre $\underline{u}$ et $\bar{u}$. Revenant maintenant dans T_1, il construit u_3, qui prend la valeur g sur L_0 et coïncide avec u_2 sur L_2. Par suite $u_3 - u_1$ est également harmonique dans T_1, s'annule sur L_0 et coïncide avec $u_2 - \underline{u}$ sur L_2. Cette dernière fonction étant positive, par le principe du maximum $u_3 - u_1$ est positive dans T_1 et plus petite que le maximum de $u_2 - \underline{u}$ sur L_2, donc que $G = \bar{u} - \underline{u}$. Par le lemme 1.1 appliqué à $(u_3 - u_1)/G$, la valeur de $u_3 - u_1$ sur L_1 est bornée par $q_1 G$. En itérant le processus, Schwarz résout alternativement pour $n \geqslant 0$ les problèmes

$$\Delta u_{2n+1} = 0 \text{ dans } T_1, \quad u_{2n+1} = g \text{ sur } L_0, \quad u_{2n+1} = u_{2n} \text{ sur } L_2, \tag{3}$$

$$\Delta u_{2n+2} = 0 \text{ dans } T_2, \quad u_{2n+2} = g \text{ sur } L_3, \quad u_{2n+2} = u_{2n+1} \text{ sur } L_1. \tag{4}$$

En appliquant le lemme 1.1 aux différences successives des itérées paires et impaires, qui sont nulles sur les bord extérieurs L_0 et L_3, il en déduit que

$$0 < u_{2n+1} - u_{2n-1} \leq G(q_1 q_2)^{n-1} \text{ sur } L_1,$$
$$0 < u_{2n+2} - u_{2n} \leq G(q_1 q_2)^{n-1} q_1 \text{ sur } L_2.$$

Il peut alors définir deux fonctions u et u' par les séries géométriques paires et impaires (convergentes !).

$$u = u_1 + (u_3 - u_1) + (u_5 - u_3) + \ldots$$
$$u' = u_2 + (u_4 - u_2) + (u_6 - u_4) + \ldots.$$

Ces fonctions sont définies dans T_1 et T_2 respectivement, elles sont toutes deux harmoniques dans T^* et coïncident sur L_1 et L_2. Elles sont donc égales dans T^* par le principe du maximum, et il a défini ainsi une fonction harmonique sur tout le domaine $T = T_1 \cup T_2$.

Près de 100 ans plus tard, Miller propose un analogue numérique à l'algorithme de Schwarz, reposant sur l'utilisation de séries de Fourier discrètes pour le calcul rapide dans des domaines simples comme des rectangles et des disques [34]. Chaque itération de l'algorithme est alors peu coûteuse.

Dans une suite de trois articles aux trois premiers congrès internationaux en décomposition de domaines [28] [29] [30], P.L. Lions propose une extension magistrale des travaux de Schwarz, adaptée au calcul parallèle sur ordinateurs. Il introduit d'abord une forme parallèle de l'algorithme où la condition $u_{2n+2} = u_{2n+1}$ de transmission sur L_1 est remplacée par $u_{2n+2} = u_{2n-1}$ (pour une écriture moderne voir (16)). Lions étend également la preuve de convergence de Schwarz à un nombre quelconque de sous-domaines avec recouvrement, et il propose une nouvelle preuve de convergence au moyen d'opérateurs de projection dans un espace de Hilbert. Notons ici que même l'algorithme de Schwarz alterné peut être exécuté en parallèle dans le cas de plus de deux sous-domaines : il suffit de colorer les domaines de différentes couleurs de façon à ce que deux sous-domaines de même couleur ne se touchent pas. Ainsi tous les sous-domaines de même couleur peuvent être calculés en parallèle, avant de prendre la couleur suivante.

Dans cette série de communications aux premiers congrès décomposition de domaines, P.L. Lions propose également d'autres conditions de transmission entre les sous-domaines. Il établit ainsi la possibilité d'utiliser des sous-domaines sans recouvrement. Ce travail a ouvert de nouvelles perspectives, créant un lien avec les méthodes de sous-structuration, et permettant plus tard la construction d'algorithmes optimisés, avec ou sans recouvrement. Pour obtenir géométriquement une décomposition avec recouvrement en sous-domaines Ω_i, il est commode de partir d'une décomposition sans recouvrement $\tilde{\Omega}_i$, et d'étendre chaque sous-domaine par une couche d'épaisseur ε (voir figure **2**).

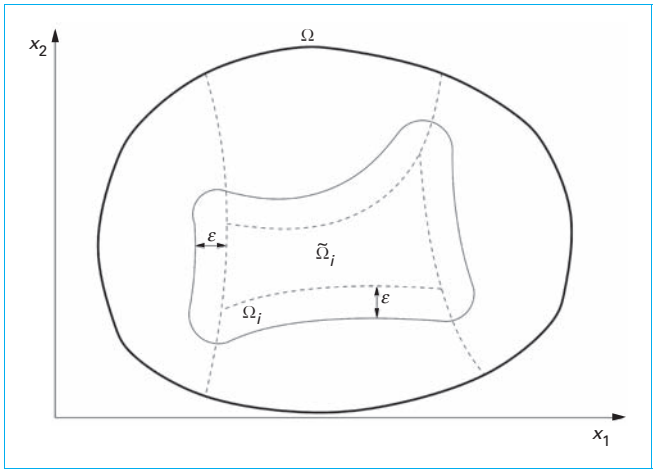


Figure 2 – Construction d'une décomposition avec recouvrement en dimension 2

Avec l'avènement des calculateurs parallèles, la méthode de Schwarz a connu un regain d'intérêt, et de nombreux développement [5] [38] [42] [43]. Les méthodes de Schwarz seront étudiées en détail dans la section 2.

1.2 Méthode de Schur

Les méthodes de Schur, historiquement aussi appelées méthodes de sous-structuration, ont été introduites par Przemieniecki en 1963, dans le contexte des calculs en aéronautique [37].

Partons d'une discrétisation par éléments finis de la structure décrite figure **3**. Dans la notation originale de Przemieniecki, P est le vecteur des forces extérieures, K la matrice de rigidité, et le vecteur des déplacements U est solution du système linéaire

$$KU = P.$$

La structure est naturellement partitionnée en **sous-structures**, en introduisant des frontières intérieures, ce qui explique le nom historique de méthode de sous-structuration. Le vecteur U_i représente l'ensemble des degrés de liberté intérieurs aux sous-domaines, et le vecteur U_b l'ensemble des degrés de liberté correspondant aux frontières intérieures. La matrice K est également partitionnée par blocs, et le système se réécrit

$$K\begin{bmatrix} U_b \\ U_i \end{bmatrix} := \begin{bmatrix} K_{bb} & K_{bi} \\ K_{ib} & K_{ii} \end{bmatrix}\begin{bmatrix} U_b \\ U_i \end{bmatrix} = \begin{bmatrix} P_b \\ P_i \end{bmatrix}.$$

Przemieniecki motive physiquement son approche en décomposant les forces appliquées P et la solution recherchée U en deux composantes

$$P = P^{(\alpha)} + P^{(\beta)} = \begin{bmatrix} P_b^{(\alpha)} \\ P_i \end{bmatrix} + \begin{bmatrix} P_b^{(\beta)} \\ 0 \end{bmatrix}, \tag{5}$$

$$U = U^{(\alpha)} + U^{(\beta)} = \begin{bmatrix} 0 \\ U_i^{(\alpha)} \end{bmatrix} + \begin{bmatrix} U_b \\ U_i^{(\beta)} \end{bmatrix}. \tag{6}$$

Par linéarité, il obtient deux systèmes

$$(\alpha):\ \begin{bmatrix} K_{bb} & K_{bi} \\ K_{ib} & K_{ii} \end{bmatrix}\begin{bmatrix} 0 \\ U_i^{(\alpha)} \end{bmatrix} = \begin{bmatrix} P_b^{(\alpha)} \\ P_i \end{bmatrix} \quad (\beta):\ \begin{bmatrix} K_{bb} & K_{bi} \\ K_{ib} & K_{ii} \end{bmatrix}\begin{bmatrix} U_b \\ U_i^{(\beta)} \end{bmatrix} = \begin{bmatrix} P_b^{(\beta)} \\ 0 \end{bmatrix},$$

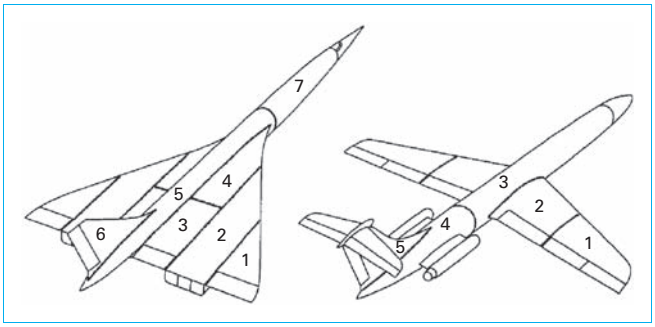


Figure 3 – Première décomposition de domaine sans recouvrement pour des calculs en aéronautiques par Przemieniecki

qui se réécrivent

$$(\alpha):\ \begin{cases} K_{bi}U_i^{(\alpha)} = P_b^{(\alpha)}, \\ K_{ii}U_i^{(\alpha)} = P_i, \end{cases} \quad (\beta):\ \begin{cases} K_{bb}U_b + K_{bi}U_i^{(\beta)} = P_b^{(\beta)}, \\ K_{ib}U_b + K_{ii}U_i^{(\beta)} = 0. \end{cases}$$

Connaissant les forces P_i intérieures aux sous-structures, il résout d'abord la deuxième équation du système (α), ce qui correspond physiquement à calculer les déplacements associés à ces forces, en fixant à zéro les déplacements sur les interfaces

$$U_i^{(\alpha)} = K_{ii}^{-1}\, P_i.$$

En reportant $U_i^{(\alpha)}$ dans la première équation du système (α), il calcule les forces $P_b^{(\alpha)}$ produites sur les interfaces par ces déplacements

$$P_b^{(\alpha)} = K_{bi}K_{ii}^{-1}P_i,$$

puis les forces restantes agissant sur les interfaces

$$P_b^{(\beta)} := P_b - P_b^{(\alpha)} = P_b - K_{bi}K_{ii}^{-1}P_i,$$

justifiant ainsi la décomposition (5). Le système (β) calcule maintenant la réponse des structures aux forces $P_b^{(\beta)}$ s'exerçant sur les interfaces. La deuxième équation de (β) permet d'exprimer les déplacement internes $U_i^{(\beta)}$ en fonction de U_b,

$$U_i^{(\beta)} = -K_{ii}^{-1}K_{ib}U_b,$$

et en reportant dans la première équation de (β), Przemieniecki obtient le **système d'interface**

$$\left(K_{bb} - K_{bi}K_{ii}^{-1}K_{ib}\right)U_b = P_b - K_{bi}K_{ii}^{-1}P_i, \tag{7}$$

qui donne la valeur U_b sur la frontière

$$U_b = \left(K_{bb} - K_{bi}K_{ii}^{-1}K_{ib}\right)^{-1}\left(P_b - K_{bi}K_{ii}^{-1}P_i\right). \tag{8}$$

Les déplacements internes dans chaque sous-domaine sont alors obtenus en sommant $U_i^{(\beta)}$ et $U_i^{(\alpha)}$.

En fait la matrice K_{ii} est une matrice diagonale par blocs, chaque bloc représentant une des sous-structures. Donc, au lieu d'inverser la grande matrice K, Przemieniecki imagine d'inverser les plus

petites matrices de rigidité de chaque sous-structure, puis la plus petite matrice $S = K_{bb} - K_{bi}K_{ii}^{-1}K_{ib}$ qui représente le couplage par l'interface, et dont la taille est une dimension inférieure à celle de K. La matrice S est la matrice du complément de Schur des sous-domaines. Cette dénomination a été introduite par E. Haynsworth dans [23] d'après le lemme du déterminant de Schur, voir [46] pour un historique du complément de Schur.

La méthode que nous venons de décrire, avec la résolution algébrique des problèmes dans les sous-domaines, est répertoriée dans la classe des « méthodes de décomposition de domaine de type Schur ». Ce nom est plus précis que le nom historique de sous-structuration, car toutes les méthodes de décomposition de domaine, y compris les méthodes de Schwarz, peuvent être écrites sous forme sous-structurée en éliminant les variables intérieures. C'est ce que nous verrons aux sections 2 et 3.

Les méthodes de Schur peuvent être employées de façon algébrique, en oubliant la physique contenue dans la présentation de Przemieniecki. La solution du système d'interface donnée explicitement dans (8) est souvent obtenue en appliquant une méthode itérative au système d'interface (7). Il n'est alors même pas nécessaire de former ce système explicitement : le produit matrice vecteur à effectuer à chaque itération revient à résoudre des problèmes aux limites dans les sous-domaines, représentés par les blocs de la matrice K_{ii} [5] [38] [42], ce que l'on peut faire à l'aide d'une méthode directe, ou de nouveau itérative. Ces méthodes seront décrites en détail dans la section 3.

1.3 Méthodes de relaxation d'onde

Ces algorithmes s'appliquent au calcul en parallèle de la solution $\boldsymbol{y} : [0,T] \to \mathbb{R}^d$ du système d'équations différentielles ordinaires

$$\frac{\mathrm{d}\boldsymbol{y}}{\mathrm{d}t} = \boldsymbol{f}(t,\boldsymbol{y}), \text{ avec la condition initiale } \boldsymbol{y}(0) = \boldsymbol{y}_0. \tag{9}$$

Ils s'appuient sur la méthode des approximations successives ou de point fixe de Picard [35] pour démontrer l'existence d'une solution à (9). Cette méthode définit une suite vectorielle $\boldsymbol{y}^n$ par

$$\frac{\mathrm{d}\boldsymbol{y}^{n+1}}{\mathrm{d}t} = \boldsymbol{f}\left(t,\boldsymbol{y}^n\right), \text{ avec la condition initiale } \boldsymbol{y}^{n+1}(0) = \boldsymbol{y}_0.$$

La première itération $\boldsymbol{y}^0$ est la fonction constante égale à $\boldsymbol{y}_0$. Cette dernière équation s'intègre sous la forme

$$\boldsymbol{y}^{n+1}(t) = \boldsymbol{y}_0 + \int_0^t \boldsymbol{f}\left(s,\boldsymbol{y}^n(s)\right)\mathrm{d}s. \tag{10}$$

On définit une condition de Lipschitz uniforme pour $\boldsymbol{f}$ (voir [AF 652])

$$\forall t \in [0,T],\ \forall (\boldsymbol{y},\boldsymbol{z}) \in \left(\mathbb{R}^d\right)^2,\quad \left\|\boldsymbol{f}(t,\boldsymbol{y}) - \boldsymbol{f}(t,\boldsymbol{z})\right\| \le L\left\|\boldsymbol{y} - \boldsymbol{z}\right\|.$$

L est la constante de Lipschitz de $\boldsymbol{f}$.

Théorème 1.2 Si f est continue par rapport à ses deux variables et uniformément lipschitzienne par rapport à y de constante L, la suite est convergente, et l'erreur est donnée par l'estimation superlinéaire de [26],

$$\left\|\boldsymbol{y} - \boldsymbol{y}^n\right\| \le \frac{(LT)^n}{n!}\left\|\boldsymbol{y} - \boldsymbol{y}^0\right\|, \tag{11}$$

où $\|\cdot\|$ est la norme du maximum sur [0, T].

Démonstration Montrons par récurrence que pour tout n

$$\left\|\boldsymbol{y}^{n+1}(t) - \boldsymbol{y}^n\right\| \le \frac{(LT)^n}{n!}\sup_{[0,T]}\left\|\boldsymbol{y}^1 - \boldsymbol{y}^0\right\|.$$

Pour cela calculons par la formule intégrale

$$\begin{aligned}
\left\|\boldsymbol{y}^{n+1}(t) - \boldsymbol{y}^n(t)\right\| &= \left\|\int_0^t f\left(s,\boldsymbol{y}^n(s)\right) - f\left(s,\boldsymbol{y}^{n-1}(s)\right)\mathrm{d}s\right\|, \\
&\le L\int_0^t \left\|\boldsymbol{y}^n(s) - \boldsymbol{y}^{n-1}(s)\right\|\mathrm{d}s \qquad (f \text{ lipschitz}), \\
&\le L \sup_{t\in[0,T]}\left\|\boldsymbol{y}^1(t) - \boldsymbol{y}^0(t)\right\|\int_0^t \frac{(Ls)^{n-1}}{(n-1)!}\mathrm{d}s \qquad (\text{hyp. de récurrence}), \\
&\le \sup_{t\in[0,T]}\left\|\boldsymbol{y}^1(t) - \boldsymbol{y}^0(t)\right\|\frac{(Lt)^n}{n!} \qquad (\text{intégration}).
\end{aligned} \tag{12}$$

Notons que $\boldsymbol{y}^1 - \boldsymbol{y}^0$ est bornée, car

$$\left\|\boldsymbol{y}^1(t) - \boldsymbol{y}^0\right\| = \left\|\int_0^t \boldsymbol{f}\left(s,\boldsymbol{y}^0\right)\mathrm{d}s\right\| \le T \sup_{s\in[0,T]}\left\|\boldsymbol{f}\left(s,\boldsymbol{y}^0\right)\right\|.$$

De l'estimation (12), nous déduisons que la suite y^n est de Cauchy. Pour tout n et tout p,

$$\begin{aligned}
\left\|\boldsymbol{y}^{n+p}(t) - \boldsymbol{y}^n(t)\right\| &\le \sum_{k=1}^{p}\left\|\boldsymbol{y}^{n+k}(t) - \boldsymbol{y}^{n+k-1}(t)\right\|, \\
&\le \sup_{[0,T]}\left\|\boldsymbol{y}^1 - \boldsymbol{y}^0\right\|\sum_{k=1}^{p}\frac{(Lt)^{n+k-1}}{(n+k-1)!}, \\
&\le \sup_{[0,T]}\left\|\boldsymbol{y}^1 - \boldsymbol{y}^0\right\|\sum_{k=n}^{n+p-1}\frac{(Lt)^k}{k!},
\end{aligned}$$

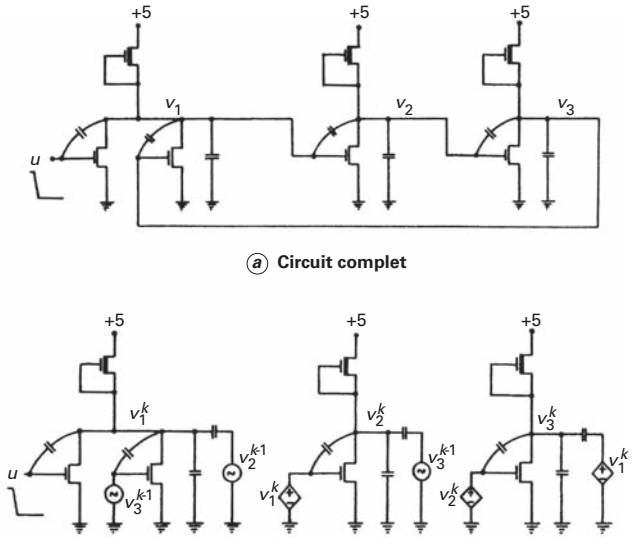
est une somme de Cauchy pour la série uniformément convergente $\sum\frac{(Lt)^j}{j!} = \exp(Lt)$. Il tend donc vers 0 et la suite est uniformément de Cauchy, elle converge vers une fonction $\tilde{y}$. Par continuité $\boldsymbol{f}(t, \boldsymbol{y}^n(t))$ tend uniformément vers $\boldsymbol{f}(t, \tilde{\boldsymbol{y}}(t))$. Passant à la limite dans (10), on peut en déduire que $\tilde{\boldsymbol{y}}$ et $\boldsymbol{y}$ coincident sur [0, T]. L'estimation (11) est obtenue par récurrence comme précédemment à partir de l'identité $\boldsymbol{y}^{n+1}(t) - \boldsymbol{y}(t) = \int_0^t\left(\boldsymbol{f}\left(s,\boldsymbol{y}^n(s)\right) - \boldsymbol{f}\left(s,\boldsymbol{y}(s)\right)\right)\mathrm{d}s$.

Remarque 1.1 On parle de convergence superlinéaire : dans les premières itérations, si $LT < 1$, la convergence est dominée par $(LT)^n$, elle est donc linéaire. Par contre après quelques itérations, le terme $1/n!$ prend le pas et la convergence devient très rapide, même si $LT > 1$.

Cet algorithme a été réinventé sous une forme plus générale un siècle plus tard par Lelarasmee, Ruehli et Sangiovanni-Vincentelli [25] pour simuler des circuits à très grande échelle (VLSI). Le modèle original traité en détail dans leur publication est l'oscillateur MOS en anneau décrit figure **4a**. Son comportement est régi par un système d'équations différentielles portant sur les tensions v_1, v_2, v_3, qui s'écrit de manière abstraite

$$\frac{\mathrm{d}v_1}{\mathrm{d}t} = f_1(v_1, v_2\, v_3), \quad \frac{\mathrm{d}v_2}{\mathrm{d}t} = f_2(v_1, v_2\, v_3), \quad \frac{\mathrm{d}v_3}{\mathrm{d}t} = f_3(v_1, v_2\, v_3).$$

Les auteurs décident de couper leurs circuits en parties cohérentes pouvant être traités sur un seul processeur et de les coller. Le processus itératif est visible sur la figure **4b**. Les trois équations différentielles des sous-circuits sont résolues à l'étape k, avec comme données le potentiel d'entrée u, et les potentiels calculés à l'étape précédente :



ⓐ Circuit complet

ⓑ Circuit partitionné

Figure 4 – Exemple original de 1982 pour les méthodes de relaxation d'ondes

$$\frac{dv_1^{k+1}}{dt} = f_1\left(v_1^{k+1}, v_2^k, v_3^k\right), \quad \frac{dv_2^{k+1}}{dt} = f_2\left(v_1^k, v_2^{k+1}, v_3^k\right),$$
$$\frac{dv_3^{k+1}}{dt} = f_3\left(v_1^k, v_2^k, v_3^{k+1}\right),$$

et ce jusqu'à convergence. Cet algorithme fait écho à l'algorithme de Jacobi pour la résolution de systèmes linéaires, qui fait partie de la famille des algorithmes de relaxation. Les représentations des signaux sur les oscilloscopes étant appelés *waveform* en anglais, ces algorithmes furent appelés *waveform relaxation algorithms* (ou relaxation d'onde en français). L'identification avec des méthodes de décomposition de domaines pour des équations aux dérivées partielles fut faite par Bjœrhus [2] pour des systèmes hyperboliques du premier ordre, et par Gander et Stuart pour des équations paraboliques comme l'équation de la chaleur [20]. C'est ainsi qu'ont été créées les *Schwarz waveform relaxation methods*, ou en français les méthodes de Schwarz relaxation d'onde.

1.4 Plan de l'exposé

Reprenons l'équation elliptique obtenue par discrétisation en temps de l'équation de la chaleur

$$(\eta - \Delta)u = f. \tag{13}$$

Dans tout l'exposé, cette équation est considérée dans un domaine Ω régulier : le segment [0,1] dans le cas monodimensionnel, le carré [0,1] × [0,1] dans le cas bidimensionnel. La fonction f est une fonction continue sur Ω, appelée second membre de l'équation. Des résultats d'analyse généraux montrent que l'équation (13) a une solution unique de classe C^2, qui s'annule sur la frontière $\partial\Omega$ de Ω. Plus précisément, nous détaillerons en dimension 1, sur l'équation

$$-\frac{d^2u}{dx^2} + \eta u = f, \tag{14}$$

tous les algorithmes dont la présentation a été faite dans cette section historique : Schwarz et ses variantes à la section 2, Schur et ses variantes à la section 3. Nous introduirons aussi en dimension 2 la notion fondamentale de conditionnement. Pour chacune de ces méthodes, nous donnerons un script Matlab, et représenterons la solution obtenue. Nous comparerons également les qualités de convergence de ces méthodes. Nous étudierons ensuite les propriétés de ces algorithmes lorsque le domaine de calcul est partagé en un grand nombre de sous-domaines. C'est la notion de scalabilité qui sera étudiée en détail dans la section 4.

Ensuite, dans la section 5, nous nous intéresserons aux problèmes d'évolution. Nous commencerons par présenter une version évolutive de la méthode de Schwarz, la méthode de Schwarz relaxation d'onde, permettant de coupler des grilles en temps différentes dans les sous-domaines. Ensuite nous aborderons la question du parallélisme en temps, en présentant l'algorithme pararéel. Nous finirons par un exemple complet de parallélisme en espace et en temps.

2. Méthodes de Schwarz

Nous exposerons d'abord dans le cas monodimensionnel de l'équation (14) les méthodes de Schwarz « historiques », présentées par H.A. Schwarz (méthode alternée) puis P.L. Lions (méthode parallèle). Ensuite nous les discrétiserons et interpréterons les algorithmes discrets obtenus en terme d'algorithme de relaxation. Puis nous présenterons les algorithmes discrets, en particulier la méthode de Schwarz additive de M. Dryja et O. Widlund [10] [11], qui constitue une invention majeure. Dans le paragraphe suivant, nous interpréterons ces méthodes de Schwarz comme des préconditionneurs pour la résolution d'un système linéaire. Ce système porte soit sur les inconnues internes, soit sur les inconnues d'interface. Nous étudierons alors en dimension 2 le conditionnement de ces systèmes linéaires préconditionnés.

2.1 Méthodes de Schwarz alternée et parallèle

Nous cherchons ici à calculer itérativement la solution du problème (14) qui prend des valeurs données g_g et g_d en 0 et 1. Le domaine de calcul Ω = [0,1] est divisé en deux sous-domaines Ω_1 = [0, β] et Ω_2 = [α,1]. La frontière de Ω_1 située dans Ω_2 est $\Gamma_1 = \{\beta\}$, et symétriquement $\Gamma_2 = \{\alpha\}$. Les domaines se recouvrent sur une distance $\delta = \beta - \alpha$.

Dans la **méthode de Schwarz alternée**, une suite $\left(u_1^n, u_2^n\right)$ pour $n \geqslant 0$ est constituée en résolvant alternativement l'équation (14) dans Ω_1 et Ω_2, en se fixant les valeurs sur le bord du recouvrement au moyen de l'itération précédente, ce qui s'écrit :

$$\begin{aligned} -\frac{d^2u_1^{n+1}}{dx^2} + \eta\, u_1^{n+1} &= f \text{ dans } \Omega_1, & -\frac{d^2u_2^{n+1}}{dx^2} + \eta\, u_2^{n+1} &= f \text{ dans } \Omega_2, \\ u_1^{n+1}(0) &= g_g, & u_2^{n+1}(1) &= g_d, \\ u_1^{n+1}(\beta) &= u_2^n(\beta), & u_2^{n+1}(\alpha) &= u_1^{n+1}(\alpha). \end{aligned} \tag{15}$$

L'algorithme est initialisée par $g \in \mathbb{R}$, avec la convention $u_2^0(\beta) \equiv g$, c'est-à-dire que u_1^1 est calculé avec la condition $u_1^1(\beta) = g$.

Dans la méthode de Schwarz parallèle [28], les calculs dans Ω_1 et Ω_2 se font en parallèle :

$$-\frac{d^2\tilde{u}_1^{n+1}}{dx^2}+\eta\,\tilde{u}_1^{n+1}=f \text{ dans } \Omega_1,\quad -\frac{d^2\tilde{u}_2^{n+1}}{dx^2}+\eta\,\tilde{u}_2^{n+1}=f \text{ dans } \Omega_2,$$
$$\tilde{u}_1^{n+1}(0)=g_g,\qquad \tilde{u}_2^{n+1}(1)=g_d, \qquad \textbf{(16)}$$
$$\tilde{u}_1^{n+1}(\beta)=\tilde{u}_2^{n}(\beta),\qquad \tilde{u}_2^{n+1}(\alpha)=\tilde{u}_1^{n}(\alpha).$$

Il faut alors se donner deux valeurs d'initialisation g_1 et g_2.

La figure **5** représente la solution de (14) dans le cas modèle qui nous accompagnera tout au long de notre étude. L'exemple que nous avons choisi reproduit la distribution de température dans un barreau de longueur 1, soumis à une source de chaleur sur une partie de sa longueur, et dont la température est fixée aux deux extrémités. Sur la figure **6** sont représentées les itérées des algorithmes de Schwarz alterné et parallèle.

Théorème 2.1 Pour tout $\eta \geqslant 0$, les algorithmes de Schwarz alterné et parallèle en dimension 1 appliqués à l'équation (14) sont convergents.

Démonstration La démonstration originale de Schwarz peut être mise en œuvre simplement ici, mais nous préférons donner une nouvelle démonstration, qui fait intervenir l'importante notion de facteur de convergence. Commençons par l'algorithme alterné (15). Par linéarité, les erreurs $e_i^n = u_i^n - u$ sont solution des mêmes équations dans les sous-domaines avec un second membre f et des données aux bords g_g et g_d nuls. Nous pouvons résoudre explicitement les équations dans les sous-domaines, au moyen de la fonction sinus hyperbolique sh $x = (e^x - e^{-x})/2$ pour $\eta > 0$, à une constante multiplicative a_i^n près :

$$\text{pour } \eta>0,\quad e_1^n=a_1^n\,\mathrm{sh}\left(\sqrt{\eta}x\right),\quad e_2^n=a_2^n\,\mathrm{sh}\left(\sqrt{\eta}(1-x)\right),$$
$$\text{pour } \eta=0,\quad e_1^n=a_1^n\,x,\qquad e_2^n=a_2^n(1-x).$$

À la première itération, a_1^1 est déterminé par la condition $u_1^1(\beta)=g$, donc $e_1^1(\beta)=g-u(\beta)$:

$$\begin{cases}\text{pour } \eta>0,\quad a_1^1\,\mathrm{sh}\left(\sqrt{\eta}\beta\right)=g-u(\beta),\\ \text{pour } \eta=0,\quad a_1^1\,\beta=g-u(\beta).\end{cases}$$

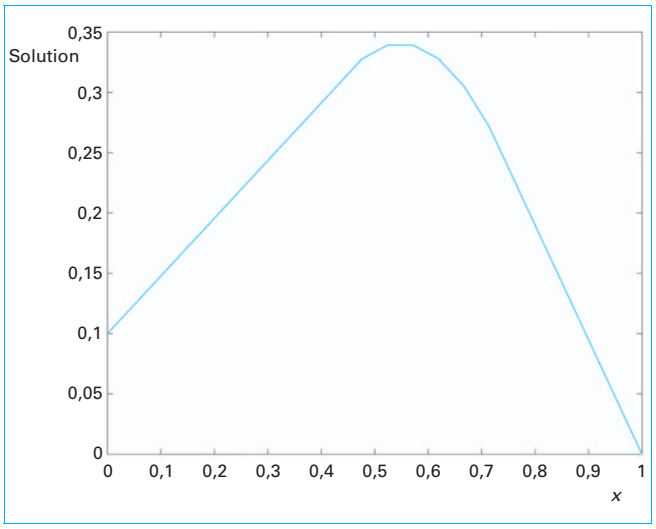


Figure 5 – Exemple de résolution de l'équation 14 par différences finies

Les conditions de transmission $e_1^{n+1}(\beta)=e_2^n(\beta)$ et $e_2^{n+1}(\alpha)=e_1^{n+1}(\alpha)$ donnent ensuite une formule de récurrence pour déterminer les coefficients a_i^n :

$$\begin{cases}\eta>0,\ a_1^{n+1}\,\mathrm{sh}\left(\sqrt{\eta}\beta\right)=a_2^n\mathrm{sh}\left(\sqrt{\eta}(1-\beta)\right),\quad a_2^{n+1}\mathrm{sh}\left(\sqrt{\eta}(1-\alpha)\right)=a_1^{n+1}\,\mathrm{sh}\left(\sqrt{\eta}\alpha\right),\\ \eta=0,\ a_1^{n+1}\,\beta=a_2^n(1-\beta),\quad a_2^{n+1}(1-\alpha)=a_1^{n+1}\alpha.\end{cases}$$

Posons

$$\rho_1=\frac{\mathrm{sh}\left(\sqrt{\eta}(1-\beta)\right)}{\mathrm{sh}\left(\sqrt{\eta}\,\beta\right)},\quad \rho_2=\frac{\mathrm{sh}\left(\sqrt{\eta}\,\alpha\right)}{\mathrm{sh}\left(\sqrt{\eta}(1-\alpha)\right)}. \qquad \textbf{(17)}$$

Ces formules sont aussi valables pour $\eta = 0$ par passage à la limite. Réécrivons la relation de récurrence sous la forme

$$a_1^{n+1}=\rho_1 a_2^n,\quad a_2^{n+1}=\rho_2 a_1^{n+1},\quad \text{ou encore}\quad a_i^{n+1}=\rho_1\rho_2 a_i^n.$$

Les suites a_1^n et a_2^n sont des suites géométriques de raison $\rho = \rho_1\rho_2$, qui est aussi appelé facteur de convergence de la

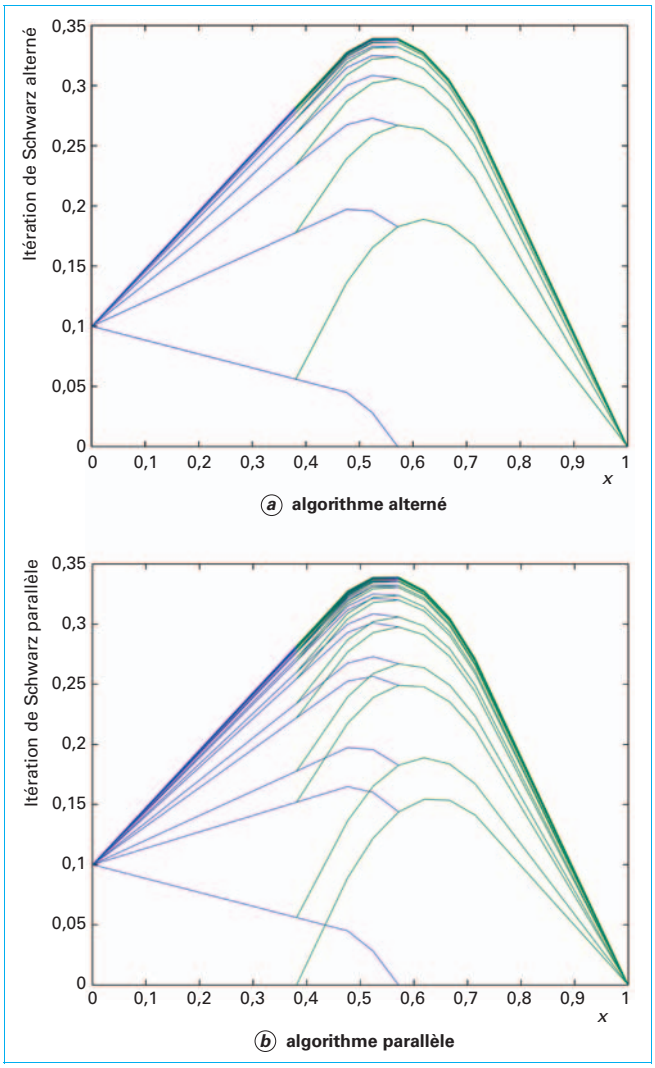


Figure 6 – Exemple de résolution de l'équation (14) par méthode de Schwarz discrétisée par différences finies

méthode. La fonction sinus hyperbolique étant croissante et $\alpha < \beta$, nous avons $\mathrm{sh}\left(\sqrt{\eta}\,\alpha\right) < \mathrm{sh}\left(\sqrt{\eta}\,\beta\right)$ et $\mathrm{sh}\left(\sqrt{\eta}\,(1-\beta)\right) < \mathrm{sh}\left(\sqrt{\eta}\,(1-\alpha)\right)$. Donc ρ est positif et strictement plus petit que 1. Les a_i^n sont alors donnés par

$$a_1^{n+1} = \rho^n a_1^1, \quad a_2^{n+1} = \rho_2\, \rho^n a_1^1. \tag{18}$$

Les fonctions u_i^n vérifient en chaque point de Ω_i :

$$u_i^{n+1}(x) - u(x) = \rho\left(u_i^n(x) - u(x)\right) = \rho^n\left(u_i^1(x) - u(x)\right).$$

À chaque étape, l'erreur en chaque point est multipliée par ρ. Dans le domaine Ω_i, la suite u_i^n converge donc uniformément vers u et la convergence est linéaire.

Dans le cas parallèle, on a une relation analogue entre les coefficients $\tilde{a}_i^n$ de $\tilde{u}_i^n$:

$$\tilde{a}_1^{n+1} = \rho_1 \tilde{a}_2^n, \quad \tilde{a}_2^{n+1} = \rho_2 \tilde{a}_1^n, \quad \text{ou encore } \tilde{a}_i^{n+1} = \rho\, \tilde{a}_i^{n-1},$$

si bien que $\tilde{a}_i^{2n+1} = \rho^n \tilde{a}_i^1$. Les itérées paires et impaires de $\tilde{u}_i^n$ convergent linéairement avec un facteur de convergence égal à ρ. •

Remarque 2.1 Si l'on pose $g = g_1$, on obtient $\tilde{u}_1^{2n-1} = u_1^n$, puis $\tilde{u}_2^{2n} = u_2^n$. Deux étapes de l'algorithme parallèle équivalent donc à une étape de l'algorithme alterné, ce que montre la figure **6**.

Remarque 2.2 La convergence de la suite est d'autant plus rapide que ρ est petit. Ceci est réalisé lorsque η est grand, auquel cas ρ est équivalent à $\exp\left(-2\sqrt{\eta}\,\delta\right)$. C'est le facteur de convergence que l'on obtiendrait aussi si l'on considérait les domaines Ω_i comme semi-infinis. Ce résultat montre aussi que les algorithmes de Schwarz sont d'autant plus rapides que le recouvrement δ est grand.

2.2 Méthodes de Schwarz alternée et parallèle discrétisées

L'intervalle [0,1] est partagé en $J + 1$ sous-intervalles de longueur h. Les nœuds de la discrétisation sont les $x_j = jh$ pour $0 \leqslant j \leqslant J + 1$. Le schéma aux différences finies associé à (14), où u_j est une approximation de $u(x_j)$ et f_j une approximation de $f(x_j)$, s'écrit

$$-\frac{u_{j+1} - 2u_j + u_{j-1}}{h^2} + \eta u_j = f_j, \quad 1 \leq j \leq J.$$

À ces J équations s'ajoutent les valeurs aux deux extrémités $u_0 = g_g$ et $u_{J+1} = g_d$, pour obtenir un système linéaire portant sur les inconnues $\boldsymbol{u} = (u_1, \ldots, u_J)^T$ aux nœuds internes, que l'on peut écrire sous forme matricielle

$$Au = f, \quad A = \begin{pmatrix} \frac{2}{h^2} + \eta & -\frac{1}{h^2} & & \\ -\frac{1}{h^2} & \frac{2}{h^2} + \eta & \ddots & \\ & \ddots & \ddots & -\frac{1}{h^2} \\ & & -\frac{1}{h^2} & \frac{2}{h^2} + \eta \end{pmatrix}, \quad f = \begin{pmatrix} f_1 + \frac{1}{h^2} g_g \\ f_2 \\ \vdots \\ f_{J-1} \\ f_J + \frac{1}{h^2} g_d \end{pmatrix}. \tag{19}$$

La matrice A s'écrit en format *sparse* en Matlab au moyen de la routine spdiags :

```
function A=A1d(eta,a,b,J)
% A1D one dimensional finite difference approximation
%   A=A1d(eta,a,b,J) computes a sparse finite difference approximation
%   of the one dimensional operator eta-Delta on the domain
%   Omega=(a,b) using J interior points
h=(b-a)/(J+1); e=ones(J,1);
A=spdiags([-e/h^2 (eta+2/h^2)*e -e/h^2],[-1 0 1],J,J);
```

Le script suivant détaille la résolution en Matlab du problème discret avec conditions aux limites non homogènes. Le système linéaire est résolu par la routine "\" de Matlab, qui est basée sur une méthode de Gauss, optimisée pour des matrices creuses. Notons que, la matrice A étant symétrique définie positive, nous pourrions également choisir comme solveur du système linéaire un algorithme itératif, le gradient conjugué, qui est extrêmement efficace, surtout pour les matrices creuses de grande taille (voir [AF 488]).

```
function u=Solve1d(f,eta,a,b,gg,gd)
% SOLVE1D solves eta-Delta in 1d using finite differences
%   u=Solve1d(f,eta,a,b,gg,gd) solves the one dimensional equation
%   (eta-Delta)u=f on the domain Omega=(a,b) with Dirichlet boundary
%   conditions u=gg at x=a and u=gd at x=b using a finite
%   difference approximation with length(f) interior grid points

J=length(f);
A=A1d(eta,a,b,J);          % construct 1d finite difference operator
h=(b-a)/(J+1);
f(1)=f(1)+gg/h^2;          % add boundary conditions into rhs
f(end)=f(end)+gd/h^2;
u=A\f;
u=[gg;u;gd];               % add boundary values to solution
```

L'exemple que nous avons choisi reproduit la distribution de température dans un barreau de longueur 1, soumis à une source de chaleur sur une partie de sa longueur (le segment [0,4 0,7]), et dont la température est fixée aux deux extrémités. Le code Matlab Bar.m ci-dessous appelle le sous-programme de résolution Solve1d pour ces données et produit la courbe représentée figure **5**.

```
eta=0;J=20;                % J number of interior mesh points
x=0:1/(J+1):1;             % finite difference mesh, including boundary
f=zeros(J,1);              % source term zero, except for a
f(x>0.4&x<0.7)=5;          % heater in this position
gg=0.1; gd=0;              % put warm wall on the left, cold on the right
u=Solve1d(f,eta,0,1,gg,gd);
plot(x,u,'-'); xlabel('x'); ylabel('solution');
```

Pour discrétiser l'algorithme de Schwarz alterné, le point α est repéré par son indice a et β par $b = a + d$ où d représente le recouvrement, $\delta = dh$. Les points $x_1, \ldots, x_J$ sont intérieurs à Ω, les points $x_1, \ldots, x_{b-1}$ sont intérieurs à Ω_1, tandis que les points $x_{a+1}, \ldots, x_J$ sont intérieurs à Ω_2. La discrétisation de l'algorithme de Schwarz alterné (15) s'écrit

$$\begin{aligned} &-\frac{\left(u_1^{n+1}\right)_{j+1} - 2\left(u_1^{n+1}\right)_j + \left(u_1^{n+1}\right)_{j-1}}{h^2} + \eta\left(u_1^{n+1}\right)_j = f_j,\ 1 \leq j \leq b-1,\ \left(u_1^{n+1}\right)_b = \left(u_2^n\right)_b, \\ &-\frac{\left(u_2^{n+1}\right)_{j+1} - 2\left(u_2^{n+1}\right)_j + \left(u_2^{n+1}\right)_{j-1}}{h^2} + \eta\left(u_2^{n+1}\right)_j = f_j,\ a+1 \leq j \leq J,\ \left(u_2^{n+1}\right)_a = \left(u_1^{n+1}\right)_a, \end{aligned} \tag{20}$$

avec les conditions aux limites extérieures $\left(u_1^{n+1}\right)_0 = g_g$ et $\left(u_2^n\right)_{J+1} = g_d$. Le script suivant réalise l'algorithme (20) (la première ligne de ce script, Bar; exécute les commandes de l'exemple précédent, qui doivent se trouver dans le fichier Bar.m) :

```
Bar;                              % to include problem parameters
a=8; d=4;                         % subdomain decomposition
f1=f(1:a+d-1); f2=f(a+1:J);       % subdomain source terms
u1=[gg; zeros(a+d,1)];            % zero initial guess, except boundary value
u2=[zeros(J-a+1,1); gd];
x1=x(1:a+d+1); x2=x(a+1:end);     % finite difference meshes
for i=1:20                        % alternating Schwarz iteration
   u1=Solve1d(f1,eta,x1(1),x1(end),gg,u2(d+1));
   u2=Solve1d(f2,eta,x2(1),x2(end),u1(end-d),gd);
   plot(x1,u1,'-',x2,u2,'-'); xlabel('x');
   ylabel('Alternating Schwarz iterates');
   hold on; pause
end
hold off
```

Exécuté en Matlab, il produit la suite de courbes représentée figure **6*a***.

Pour obtenir l'algorithme parallèle de Schwarz, et les résultats de la figure **6*b***, il faut modifier la boucle précédente en

```
ulold=ul;
ul=Solveld(fl,eta,xl(l),xl(end),gg,u2(d+l));
u2=Solveld(f2,eta,x2(l),x2(end),ulold(end-d),gd);
plot(xl,ul,'-',x2,u2,'-'); xlabel('x');
ylabel('Parallel Schwarz iterates');
```

Interprétation algébrique Nous allons maintenant écrire l'algorithme (20) sous forme d'un algorithme algébrique portant sur les vecteurs $\boldsymbol{u}_1^n = \left(\left(u_1^n\right)_1, \cdots, \left(u_1^n\right)_{b-1}\right)^T$ et $\boldsymbol{u}_2^n = \left(\left(u_2^n\right)_{a+1}, \cdots, \left(u_2^n\right)_J\right)^T$. La matrice A peut être décomposée par blocs sous la forme

$$A = \left(\begin{array}{cccc|ccccc} \frac{2}{h^2}+\eta & -\frac{1}{h^2} & & & & & & & \\ -\frac{1}{h^2} & \frac{2}{h^2}+\eta & \ddots & & & & & & \\ & \ddots & \ddots & -\frac{1}{h^2} & & & & & \\ & & -\frac{1}{h^2} & \frac{2}{h^2}+\eta & -\frac{1}{h^2} & & & & \\ \hline & & & -\frac{1}{h^2} & \frac{2}{h^2}+\eta & -\frac{1}{h^2} & & & \\ & & & & -\frac{1}{h^2} & \frac{2}{h^2}+\eta & \ddots & & \\ & & & & & \ddots & \ddots & -\frac{1}{h^2} & \\ & & & & & & -\frac{1}{h^2} & \frac{2}{h^2}+\eta \end{array}\right). \quad (21)$$

Introduisons deux décompositions de ce type :

$$A = \begin{pmatrix} A_1 & B_1 \\ C_1 & D_1 \end{pmatrix} = \begin{pmatrix} D_2 & C_2 \\ B_2 & A_2 \end{pmatrix}, \quad (22)$$

où A_1 est de taille $(b-1) \times (b-1)$, et D_2 de taille $a \times a$, et donc A_2 est de taille $(J-a) \times (J-a)$. Les matrices A_1 et D_2 coïncident lorsque $b = a+1$, c'est-à-dire $d = 1$. Le recouvrement géométrique est alors minimal et le recouvrement algébrique est vide. Les matrices A_1 et A_2 sont les matrices de l'opérateur $\eta - \Delta$ discrétisé, avec données de Dirichlet homogène sur les bords, elles sont donc inversibles.

Les matrices B_i sont complétées par des zéros en

$$\tilde{B}_1 = \left[0_{b-1,d-1}\, B_1\right], \quad \tilde{B}_2 = \left[B_2\, 0_{J-a,d-1}\right].$$

Ainsi $\tilde{B}_1$ est une matrice $(b-1) \times (J-a)$ qui à un vecteur défini sur Ω_2 associe un vecteur défini sur Ω_1, prolongé par 0 hors de Ω_2. De même $\tilde{B}_2$ est une matrice $(J-a) \times (b-1)$ qui à un vecteur défini sur Ω_1 associe un vecteur défini sur Ω_2, prolongé par 0 hors de Ω_1. Avec ces notations, l'algorithme alterné (20) prend la forme

$$A_1 \boldsymbol{u}_1^{n+1} = \boldsymbol{f}_1 - \tilde{B}_1 \boldsymbol{u}_2^n, \quad A_2 \boldsymbol{u}_2^{n+1} = \boldsymbol{f}_2 - \tilde{B}_2 \boldsymbol{u}_1^{n+1}, \quad (23)$$

qui n'est autre qu'une méthode de Gauss-Seidel par blocs

$$\begin{pmatrix} A_1 & 0 \\ \tilde{B}_2 & A_2 \end{pmatrix} \begin{pmatrix} \boldsymbol{u}_1 \\ \boldsymbol{u}_2 \end{pmatrix}^{n+1} = \begin{pmatrix} 0 & -\tilde{B}_1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{u}_1 \\ \boldsymbol{u}_2 \end{pmatrix}^n + \begin{pmatrix} \boldsymbol{f}_1 \\ \boldsymbol{f}_2 \end{pmatrix} \quad (24)$$

pour le système augmenté

$$\tilde{A}\tilde{\boldsymbol{u}} = \tilde{\boldsymbol{f}} \ : \begin{pmatrix} A_1 & \tilde{B}_1 \\ \tilde{B}_2 & A_2 \end{pmatrix} \begin{pmatrix} \boldsymbol{u}_1 \\ \boldsymbol{u}_2 \end{pmatrix} = \begin{pmatrix} \boldsymbol{f}_1 \\ \boldsymbol{f}_2 \end{pmatrix}. \quad (25)$$

Notons que lorsque le recouvrement est minimal, la matrice augmentée coïncide avec la matrice A de départ.

L'algorithme de Schwarz parallèle discrétisé peut aussi être exprimé sous forme algébrique :

$$A_1 \boldsymbol{u}_1^{n+1} = \boldsymbol{f}_1 - \tilde{B}_1 \boldsymbol{u}_2^n, \quad A_2 \boldsymbol{u}_2^{n+1} = f_2 - \tilde{B}_2 \boldsymbol{u}_1^n, \quad (26)$$

qui est cette fois une méthode de Jacobi par blocs pour le système augmenté (25), i.e.

$$\begin{pmatrix} A_1 & 0 \\ 0 & A_2 \end{pmatrix} \begin{pmatrix} \boldsymbol{u}_1 \\ \boldsymbol{u}_2 \end{pmatrix}^{n+1} = \begin{pmatrix} 0 & -\tilde{B}_1 \\ -\tilde{B}_2 & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{u}_1 \\ \boldsymbol{u}_2 \end{pmatrix}^n + \begin{pmatrix} \boldsymbol{f}_1 \\ \boldsymbol{f}_2 \end{pmatrix}. \quad (27)$$

Une itération de l'algorithme se comprend de la manière suivante : $\tilde{B}_2 \boldsymbol{u}_1^n$ est le vecteur de taille $J-a$ dont toutes les composantes sont nulles sauf la première qui vaut $-\frac{1}{h^2}\left(u_1^n\right)_a$. De même $\tilde{B}_1 \boldsymbol{u}_2$ est le vecteur de taille $b-1$ dont toutes les composantes sont nulles sauf la dernière qui vaut $-\frac{1}{h^2}\left(u_2^n\right)_b$. Maintenant $A_1^{-1}\left(\boldsymbol{f}_1 - \tilde{B}_1 \boldsymbol{u}_2\right)$ représente la résolution du problème de Dirichlet dans Ω_1, avec second membre $\boldsymbol{f}_1$, et donnée à la limite en b égale à $\left(u_2^n\right)_b$. De même $A_2^{-1}\left(\boldsymbol{f}_2 - \tilde{B}_2 \boldsymbol{u}_1\right)$ représente la résolution du problème de Dirichlet dans Ω_2, avec second membre $\boldsymbol{f}_2$, et donnée à la limite en a égale à $\left(u_1^n\right)_a$. Si bien qu'une itération de l'algorithme revient à résoudre un problème aux limites dans chaque sous-domaine.

Notons que, bien que la matrice A soit symétrique, la matrice augmentée ne l'est pas, car les matrices $\tilde{B}_2^T$ et $\tilde{B}_1$ sont différentes sauf si le recouvrement est minimal, i.e. $d = 1$ (en effet le seul terme non nul dans $\tilde{B}_1$ est $\left(\tilde{B}_1\right)_{b-1,d}$, alors que $\left(\tilde{B}_2^T\right)_{b-1,d} = \left(\tilde{B}_2\right)_{d,b-1} = 0$ si $d \neq 1$). Ceci interdit l'utilisation de la méthode du gradient conjugué pour résoudre le système préconditionné par la méthode de Schwarz parallèle,

$$\begin{pmatrix} A_1^{-1} & 0 \\ 0 & A_2^{-1} \end{pmatrix} \begin{pmatrix} A_1 & \tilde{B}_1 \\ \tilde{B}_2 & A_2 \end{pmatrix} \begin{pmatrix} \boldsymbol{u}_1 \\ \boldsymbol{u}_2 \end{pmatrix} = \begin{pmatrix} A_1^{-1} & 0 \\ 0 & A_2^{-1} \end{pmatrix} \begin{pmatrix} \boldsymbol{f}_1 \\ \boldsymbol{f}_2 \end{pmatrix}, \quad (28)$$

car la méthode du gradient conjugué ne peut s'appliquer qu'à une matrice symétrique définie positive préconditionnée par matrice symétrique définie positive. Toutefois des méthodes de Krylov ont été développées pour les systèmes non symétriques, et ces méthodes, malgré leur coût supérieur à celui de la méthode du gradient conjugué et l'absence d'une théorie de convergence complète, sont plus performantes que les méthodes simples de Jacobi ou Gauss-Seidel par blocs (par exemple GMRES, QMR, BiCGStab, voir l'excellent livre d'analyse numérique matricielle de Youssef Saad [40] ou [AF 488]). D'autre part d'autres algorithmes de décomposition de domaine ont été développés pour rétablir la symétrie, comme nous allons le voir dans le paragraphe suivant.

2.3 Méthodes de Schwarz discrètes : AS, MS et RAS

Pour comprendre l'invention majeure que constitue la méthode de Schwarz additive (AS), revenons à la méthode de Schwarz parallèle discrétisée écrite sous forme de Jacobi par blocs en (27). Dans le cas où $d = 1$, $\tilde{B}_i$ n'est autre que B_i, et l'itération est identique à

$$\begin{pmatrix} \boldsymbol{u}_1 \\ \boldsymbol{u}_2 \end{pmatrix}^{n+1} = \begin{pmatrix} \boldsymbol{u}_1 \\ \boldsymbol{u}_2 \end{pmatrix}^n + \begin{pmatrix} A_1^{-1} & 0 \\ 0 & A_2^{-1} \end{pmatrix} \left(\boldsymbol{f} - A \begin{pmatrix} \boldsymbol{u}_1 \\ \boldsymbol{u}_2 \end{pmatrix}^n \right). \quad (29)$$

Cette écriture nous permet dans ce cas d'interpréter l'algorithme de Schwarz parallèle discrétisé (27) comme une méthode itérative pour le **système préconditionné**

$$\begin{pmatrix} A_1^{-1} & 0 \\ 0 & A_2^{-1} \end{pmatrix}\begin{pmatrix} A_1 & B_1 \\ B_2 & A_2 \end{pmatrix}\begin{pmatrix} \boldsymbol{u}_1 \\ \boldsymbol{u}_2 \end{pmatrix} = \begin{pmatrix} A_1^{-1} & 0 \\ 0 & A_2^{-1} \end{pmatrix}\begin{pmatrix} \boldsymbol{f}_1 \\ \boldsymbol{f}_2 \end{pmatrix}. \quad (30)$$

Si la matrice A est symétrique et définie positive, le **préconditionneur** $\begin{pmatrix} A_1^{-1} & 0 \\ 0 & A_2^{-1} \end{pmatrix}$ l'est aussi, et (30) peut être résolu par la méthode du gradient conjugué.

Introduisons maintenant les matrices de restriction

$$R_1 = \begin{bmatrix} I_{b-1} & 0_{b-1,J-b+1} \end{bmatrix}, \quad R_2 = \begin{bmatrix} 0_{J-a,a} & I_{J-a} \end{bmatrix}. \quad (31)$$

Le préconditionneur s'écrit au moyen de ces matrices de restriction :

$$\begin{pmatrix} A_1^{-1} & 0 \\ 0 & A_2^{-1} \end{pmatrix} = \sum_{i=1}^{2} R_i^T A_i^{-1} R_i.$$

Puisque le vecteur $(\boldsymbol{u}_1, \boldsymbol{u}_2)$ avec recouvrement minimal $d = 1$ n'est autre que le vecteur global $\boldsymbol{u}$, nous en déduisons une nouvelle forme pour l'algorithme de Schwarz parallèle discrétisé (27) dans le cas où $d = 1$, *i.e.* (29) :

$$\boldsymbol{u}^{n+1} = \boldsymbol{u}^n + \sum_{i=1}^{2} R_i^T A_i^{-1} R_i \left(\boldsymbol{f} - A\boldsymbol{u}^n\right). \quad (32)$$

Cet algorithme a encore un sens pour un recouvrement quelconque, mais il n'est pas très utile, comme le montre le contre-exemple suivant :

Théorème 2.2 Si le recouvrement n'est pas minimal, $d > 1$, l'algorithme (32) appliqué à la matrice de différences finies (19) n'est pas convergent : il existe une donnée initiale $\boldsymbol{u}^0$ telle que l'algorithme oscille entre $\boldsymbol{u}^0$ et $-\boldsymbol{u}^0$.

Démonstration Décomposons la matrice A de deux façons comme en (22). À chaque itération, le vecteur $\boldsymbol{u}^n$ est décomposé en $\left(\boldsymbol{u}_{11}^n, \boldsymbol{u}_{12}^n\right)$ en accord avec la première décomposition de A, et en $\left(\boldsymbol{u}_{21}^n, \boldsymbol{u}_{22}^n\right)$ en accord avec la seconde décomposition. Le second membre $\boldsymbol{f}$ est décomposé de la même façon. Nous avons alors :

$$R_1 A\boldsymbol{u}^n = \begin{bmatrix} A_1 & B_1 \end{bmatrix}\boldsymbol{u}^n = A_1\boldsymbol{u}_{11}^n + B_1\boldsymbol{u}_{12}^n, \quad R_2 A\boldsymbol{u}^n = \begin{bmatrix} B_2 & A_2 \end{bmatrix}\boldsymbol{u}^n = B_2\boldsymbol{u}_{21}^n + A_2\boldsymbol{u}_{22}^n.$$

Calculons maintenant

$$R_1^T A_1^{-1} R_1 A\boldsymbol{u}^n = R_1^T\boldsymbol{u}_{11}^n + R_1^T A_1^{-1} B_1\boldsymbol{u}_{12}^n,$$
$$R_2^T A_2^{-1} R_2 A\boldsymbol{u}^n = R_2^T\boldsymbol{u}_{22}^n + R_2^T A_2^{-1} B_2\boldsymbol{u}_{21}^n.$$

L'équation (32) s'écrit alors

$$\boldsymbol{u}^{n+1} = \boldsymbol{u}^n - \begin{pmatrix} \boldsymbol{u}_{11}^n \\ 0 \end{pmatrix} - \begin{pmatrix} 0 \\ \boldsymbol{u}_{22}^n \end{pmatrix} - \begin{pmatrix} A_1^{-1} B_1\boldsymbol{u}_{12}^n \\ 0 \end{pmatrix} - \begin{pmatrix} 0 \\ A_2^{-1} B_2\boldsymbol{u}_{21}^n \end{pmatrix} + \begin{pmatrix} A_1^{-1}\boldsymbol{f}_1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ A_2^{-1}\boldsymbol{f}_2 \end{pmatrix}.$$

Regardons maintenant la convergence de l'algorithme. Pour cela choisissons un second membre $\boldsymbol{f}$ nul, et pour un indice j compris strictement entre a et $a + d$, une donnée initiale $\boldsymbol{u}^0 = \boldsymbol{e}_j$, le j-ème vecteur de la base canonique de $\mathbb{R}^J$. Les seuls termes non nuls à droite de l'équation précédente sont les trois premiers, chacun vaut $\boldsymbol{u}^0$, si bien que $\boldsymbol{u}^1 = -\boldsymbol{u}^0$. En itérant l'argument, on voit que l'algorithme oscille entre $\boldsymbol{u}^0$ et $-\boldsymbol{u}^0$. •

Le script Matlab suivant BarAS.m réalise les itérations de la méthode (32) pour le même exemple que précédemment. Les itérations sont représentées figure **7**. Le défaut de convergence dans le recouvrement apparaît nettement.

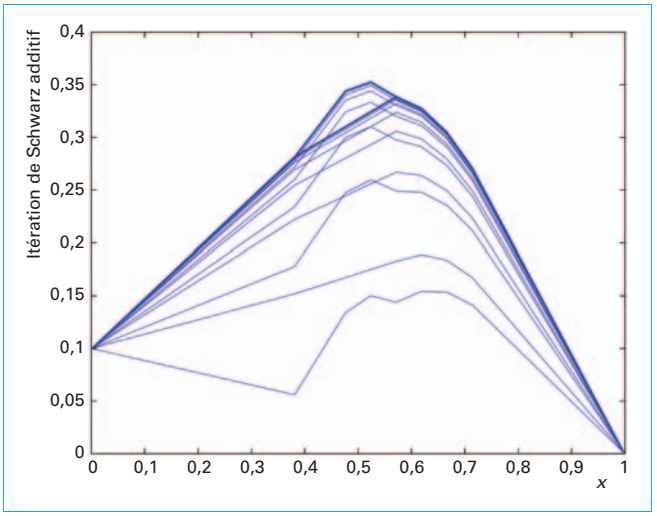


Figure 7 – Essai de résolution du système (19) par l'algorithme (32)

```
Bar;                                        % to include problem parameters
α=8; d=4;                                   % subdomain decomposition
h=1/(J+1);
f(1)=f(1)+gg/h^2; f(end)=f(end)+gd/h^2; % add boundary conditions into rhs
A=A1d(eta,0,1,J);                           % construct finite difference operator
R1=[speye(α+d-1) sparse(α+d-1,J-α-d+1)];
R2=[sparse(J-α,α) speye(J-α)];
A1=R1*A*R1'; A2=R2*A*R2';
u=zeros(J,1);
for i=1:20
   r=f-A*u;
   u=u+(R1'*(A1\(R1*r))+R2'*(A2\(R2*r)));
   plot(x,[gg;u;gd],'-'); xlabel('x'); ylabel('Additive Schwarz stationnary iterates');
   hold on; pause
   ri(i)=norm(r);                           % keep residual for plotting later
end
hold off
```

La véritable **méthode de Schwarz additive ou AS** est basée sur l'itération (32), mais considérée comme préconditionneur. Elle consiste à résoudre le système préconditionné obtenu à la limite

$$M_{AS}^{-1} A\boldsymbol{u} := \sum_{i=1}^{2} R_i^T A_i^{-1} R_i A\boldsymbol{u} = \sum_{i=1}^{2} R_i^T A_i^{-1} R_i\boldsymbol{f}. \quad (33)$$

Lorsque A est symétrique, le préconditionneur de Schwarz additif $M_{AS}^{-1} = \sum_i R_i^T A_i^{-1} R_i$ est symétrique, ce qui permet la résolution par l'algorithme du gradient conjugué, voir paragraphe 2.4. C'est la seule méthode de Schwarz connue à ce jour qui ait cette propriété, mais elle est obtenue au prix de la perte de convergence dans la version itérative (32) pour les modes présents dans le recouvrement. Cet algorithme est devenu l'algorithme de Schwarz le plus communément utilisé pour les problèmes symétriques, couplé avec une grille grossière, nous en parlerons à la section 4. Pour une approche différente qui utilise du recouvrement et reste symétrique, voir [9].

La **méthode de Schwarz multiplicative ou MS** (voir [5]) est la version séquentielle de la méthode de Schwarz additive. Pour notre exemple, elle s'écrit

$$\boldsymbol{u}^{n+\frac{1}{2}} = \boldsymbol{u}^n + R_1^T A_1^{-1} R_1\left(\boldsymbol{f} - A\boldsymbol{u}^n\right),$$
$$\boldsymbol{u}^{n+1} = \boldsymbol{u}^{n+\frac{1}{2}} + R_2^T A_2^{-1} R_2\left(\boldsymbol{f} - A\boldsymbol{u}^{n+\frac{1}{2}}\right). \quad (34)$$

Pour l'implémenter en Matlab, il suffit de remplacer dans la boucle du programme précédent le calcul de r et u par

```
r=f-A*u; u=u+R1'*(A1\(R1*r));
r=f-A*u; u=u+R2'*(A2\(R2*r));
```

ce qui produit les itérations tracées sur la figure **8**.

Il est intéressant de constater que les modes oscillants dans le recouvrement ont disparu : l'algorithme multiplicatif itératif est convergent. Ceci peut être démontré d'une manière rigoureuse pour des décompositions très générales, voir [16] où l'on montre également que cette méthode est équivalente à la méthode de Schwarz alternée discrétisée. Mais par contre le système préconditionné associé est non symétrique, puisqu'il correspond à une méthode de Gauss-Seidel par blocs, comme nous l'avons vu, il ne peut donc pas être résolu par un gradient conjugué.

En 1998, Cai et Sarkis [4] introduisirent une nouvelle méthode discrète, l'**algorithme de Schwarz additif restreint ou RAS**, défini par

$$\boldsymbol{u}^{n+1} = \boldsymbol{u}^n + \sum_{i=1}^{2} \tilde{R}_i^T A_i^{-1} R_i \left(\boldsymbol{f} - A\boldsymbol{u}^n \right). \qquad (35)$$

Les deux nouvelles matrices de restriction $\tilde{R}_i$ sont obtenues en changeant des 1 en 0 dans les R_i, de façon à correspondre à une décomposition sans recouvrement, i.e. $\tilde{R}_1^T R_1 + \tilde{R}_2^T R_2 = I$. Le principe est décrit figure **9**, et les matrices $\tilde{R}_i$ sont de même taille que les R_i, avec

$$\tilde{R}_1 = \begin{bmatrix} I_{a+\left[\frac{d}{2}\right]} & 0 \\ 0 & 0 \end{bmatrix}, \quad \tilde{R}_2 = \begin{bmatrix} 0 & 0 \\ 0 & I_{J-a-\left[\frac{d}{2}\right]} \end{bmatrix},$$

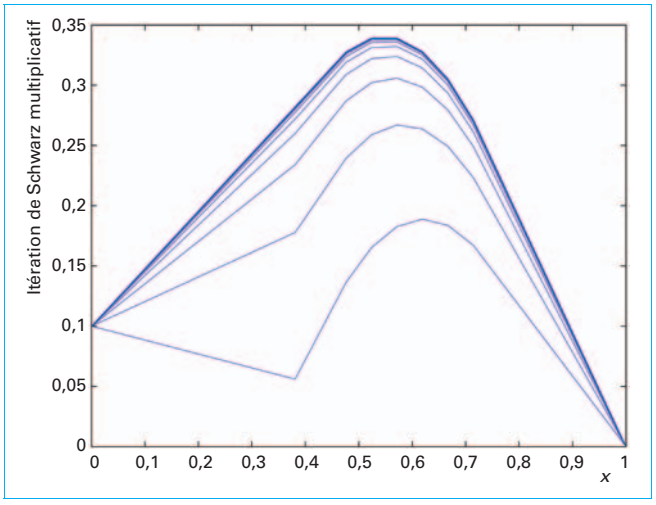


Figure 8 – Exemple de la méthode de Schwarz multiplicative

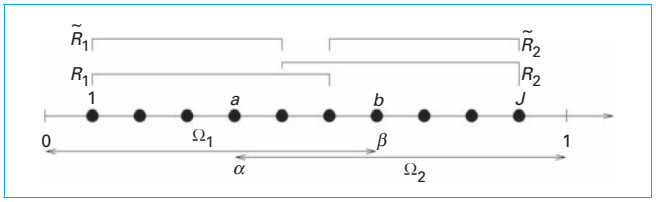


Figure 9 – Définition des $\tilde{R}_i$ en dimension 1

où $[x]$ désigne la partie entière de x. Cette modification a comme conséquence de supprimer l'erreur commise dans le recouvrement par l'itération (32) de Schwarz additif, et la méthode est convergente et équivalente à la méthode de Schwarz parallèle discrétisée, voir [13] [16].

Pour obtenir BarRAS.m, implémentation en Matlab de la méthode RAS, il suffit d'ajouter dans BarAS.m le calcul des opérateurs $\tilde{R}_i$ par les commandes

```
m=a+ceil(d/2);              % construct the Rtilde operators
R1t=R1;R1t(m:a+d-1,m:a+d-1)=0;
R2t=R2;R2t(1:m-a-1,a+1:m-1)=0;
```

et de les remplacer dans la formule pour u. On voit clairement sur les courbes de la figure **10** que l'algorithme converge maintenant dans le recouvrement, mais les itérées sont discontinues.

Malheureusement cette modification qui améliore la performance de AS détruit la symétrie du préconditionneur, car $M_{RAS}^{-1} := \sum_{i=1}^{2} \tilde{R}_i^T A_i^{-1} R_i$ n'est clairement pas symétrique. Cependant, parmi les méthodes de Schwarz, RAS est devenu le standard pour les problèmes non symétriques.

Remarque 2.3 On pourrait également construire une méthode de Schwarz multiplicative restreinte, mais cela changerait peu les performances, puisque Schwarz multiplicatif est déjà convergent dans le recouvrement.

Tous les algorithmes de Schwarz discrets calculent une approximation globale de la solution, tandis que les algorithmes discrétisés (23) et (26) calculent des approximations de $\boldsymbol{u}_1$ et $\boldsymbol{u}_2$, dans les sous-domaines. Il est cependant facile de reconstituer une solution globale à partir de ces approximations : introduisons les matrices de partition de l'unité X_i pour $i = 1,2$:

$$X_1 = \begin{pmatrix} I_a & 0 & 0 \\ 0 & M_{d-1} & 0 \\ 0 & 0 & 0_{J-a+d+1} \end{pmatrix} R_1^T, \quad X_2 = \begin{pmatrix} 0_a & 0 & 0 \\ 0 & I - M_{d-1} & 0 \\ 0 & 0 & I_{J-a+d+1} \end{pmatrix} R_2^T,$$

où la matrice M est par exemple diagonale avec $M_{ii} = (a + d - i)/d$. Une approximation discrète $\boldsymbol{u}$ de u aux points de grille est alors définie par

$$\boldsymbol{u} = X_1 \boldsymbol{u}_1 + X_2 \boldsymbol{u}_2.$$

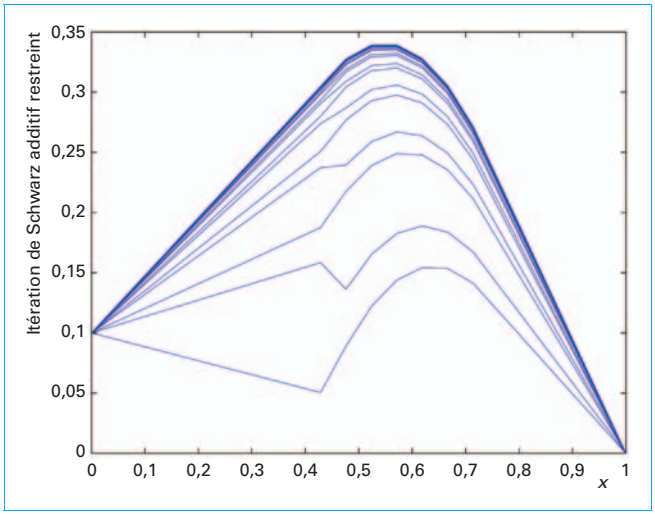


Figure 10 – Exemple de résolution de l'équation (14) par la méthode de Schwarz additive restreinte

Cette construction est également possible au niveau continu, au moyen d'une partition de l'unité, c'est-à-dire deux fonctions χ_i pour $i = 1,2$ indéfiniment différentiable sur Ω, de somme égale à 1, à support dans $\overline{\Omega_i}$. L'approximation globale est alors donnée par

$$u = \chi_1 u_1 + \chi_2 u_2.$$

La matrice M précédente est une discrétisation affine de χ_1.

2.4 Méthodes de Schwarz considérées comme préconditionneur

Une des motivations les plus fortes pour l'utilisation des méthodes de décomposition de domaines est la taille des systèmes en jeu. En effet lorsque la taille du système croit, les méthodes itératives traditionnelles deviennent de plus en plus lentes. La qualité de la méthode est mesurée par le conditionnement (voir [40] et [AF 485]).

Conditionnement et gradient conjugué

Le conditionnement de la résolution d'un système linéaire $\mathcal{A}x = b$ est mesuré par le conditionnement de la matrice $\mathcal{A}$. Ce dernier, relatif à la norme euclidienne notée $\|\cdot\|$, est défini par la formule

$$\kappa(\mathcal{A}) = \|\mathcal{A}\| \|\mathcal{A}^{-1}\|.$$

La matrice A des différences finies en dimension 1 donnée par (19) est symétrique définie positive. Sa norme est donc égale à son rayon spectral, c'est-à-dire sa plus grande valeur propre. Il en va de même de son inverse. Les valeurs propres de A sont

$$\lambda_j^d = \eta + \frac{4}{h^2}\sin^2\frac{j\pi h}{2}, \quad 1 \leqslant j \leqslant J. \tag{36}$$

Le conditionnement de A est donc donné par

$$\kappa(A) = \frac{\max_{1\leqslant j\leqslant J}\lambda_j^d}{\min_{1\leqslant j\leqslant J}\lambda_j^d} = \frac{\lambda_J^d}{\lambda_1^d} = \frac{\eta + \frac{4}{h^2}\sin^2\frac{J\pi h}{2}}{\eta + \frac{4}{h^2}\sin^2\frac{\pi h}{2}}.$$

Lorsque h est petit, $\lambda_J^d \sim \frac{4}{h^2}$, tandis que $\lambda_1^d \sim \eta + \pi^2$. Notons que les valeurs propres du problème continu (14) avec des conditions aux limites de Dirichlet homogène sont égales à $\lambda_j^c = \eta + j^2\pi^2$ pour tout j entier naturel. On a donc pour de faibles valeurs de h, $\lambda_1^d \sim \lambda_1^c$ et $\lambda_J^d \sim \lambda_{\frac{2}{\pi h}}^c$. La gamme de fréquences du problème continu correspondant à des modes propres du problème discret est $\left(1, \frac{2}{\pi h}\right)$. Maintenant le conditionnement de la matrice A pour h petit est asymptotiquement égal à

$$\kappa(A) \sim \frac{4}{\eta + \pi^2}\frac{1}{h^2}. \tag{37}$$

Il est bien connu que l'algorithme de Richardson, défini par

$$\boldsymbol{x}^{n+1} = \boldsymbol{x}^n + \lambda\left(\boldsymbol{b} - \mathcal{A}\boldsymbol{x}^n\right), \tag{38}$$

converge linéairement pour des matrices symétriques définies positives, avec un facteur de convergence égal à

$$\frac{\kappa(\mathcal{A}) - 1}{\kappa(\mathcal{A}) + 1},$$

si l'on utilise un paramètre de relaxation optimal

$$\lambda^* = \frac{2}{\lambda_{\min}(\mathcal{A}) + \lambda_{\min}(\mathcal{A})}.$$

Dans le cas des différences finies en dimension 1, avec l'estimation (37), le facteur de convergence est équivalent à $1 - \frac{\sqrt{n+\pi^2}}{2}h^2$. Pour l'algorithme du gradient conjugué, le facteur de convergence est égal à

$$\frac{\sqrt{\kappa(\mathcal{A})} - 1}{\sqrt{\kappa(\mathcal{A})} + 1},$$

et pour la matrice A des différences finies en dimension 1, cette quantité est équivalente à $1 - \sqrt{\eta + \pi^2}h$, ce qui améliore de façon frappante la convergence. En effet la convergence est d'autant plus lente que le facteur de convergence est proche de 1, et lorsque h est petit, $1 - \mathcal{O}(h^2)$ est beaucoup plus proche de 1 que $1 - \mathcal{O}(h)$. Ce résultat est aussi valable en dimension supérieure, comme nous le verrons plus loin en dimension 2 dans la formule (45). Pour l'améliorer encore, il faudrait rendre le conditionnement le moins dépendant de h possible. Cela peut être obtenu en préconditionnant le système, c'est-à-dire en remplaçant le système $\mathcal{A}\boldsymbol{x} = \boldsymbol{b}$ par $\mathfrak{M}^{-1}\mathcal{A}\boldsymbol{x} = \mathfrak{M}^{-1}\boldsymbol{b}$ où la matrice $\mathfrak{M}$ est une matrice facile à inverser, telle que le conditionnement de $\mathfrak{M}^{-1}\mathcal{A}$ soit meilleur que celui de $\mathcal{A}$. Parmi beaucoup de choix possibles, les méthodes de Schwarz que nous avons présentées plus haut sont d'excellents candidats parallèles.

Préconditionnement en volume

Nous avons vu que le préconditionneur de Schwarz additif présenté en (33) est symétrique pour les problèmes symétriques. Le système préconditionné peut alors être résolu par le même algorithme que le problème de départ, le gradient conjugué, qui est l'algorithme le plus performant pour les problèmes symétriques. Sur la figure **11** nous voyons comment, à partir de l'algorithme itératif (32) qui ne converge pas dans le recouvrement, nous sommes arrivés à une convergence très rapide avec le gradient conjugué. En abscisse est représenté le numéro n de l'itération, en ordonnée la norme euclidienne du résidu $\boldsymbol{f} - A\boldsymbol{u}_n$. Le script Matlab correspondant est inséré ci-après.

```
BarAS ;                                  % first part like in BarAS.m
Mfun=@(x)R1'*(A1\(R1*x))+R2'*(A2\(R2*x));
[u, fl, r, it, rcg]=pcg(A, f, 1e-6, 10, Mfun);
semilogy(0 : length(ri)-1,ri,'-o',0 : length(rcg)-1,rcg,'-+');
xlabel('iteration') ; ylabel('residual') ;
legend('Iterative','CG');
```

L'algorithme du gradient conjugué converge à la quatrième itération. En effet la matrice du système linéaire préconditionné issu de l'algorithme de Schwarz s'écrit sous la forme $(I - M)$ où M est de rang $r = 3$ pour notre exemple, car dans ce cas monodimensionnel le problème d'interface résolu est de taille 2, comme nous le verrons plus tard, et Schwarz additif a encore un mode dans le recouvrement. Or on sait que pour une telle matrice, les méthodes de Krylov convergent en $r + 1$ itérations.

L'algorithme de Schwarz additif restreint peut aussi être utilisé comme préconditionneur. Les matrices R_i à gauche sont alors remplacées par les matrices $\tilde{R}_i$. Le préconditionneur obtenu n'est plus symétrique, la méthode du gradient conjugué ne semble pas converger, comme le montre la figure **12**. On doit alors faire appel à un algorithme de Krylov adapté aux problèmes non symétriques, par exemple GMRES voir [40]. Le script Matlab est inséré ci-après.

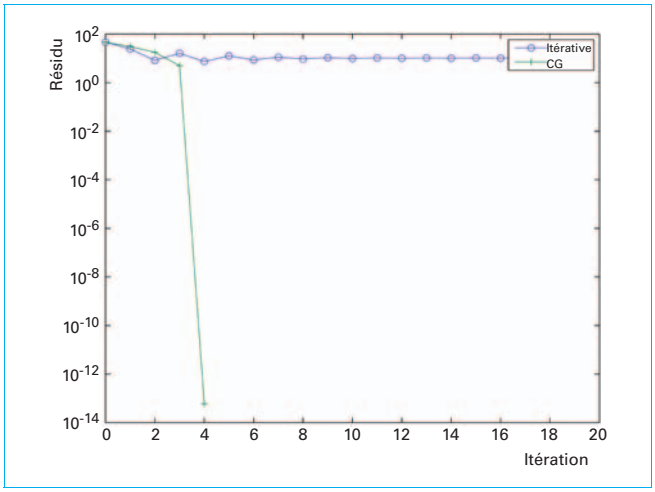


Figure 11 – Schwarz additif utilisé comme préconditionneur comparé à la méthode itérative

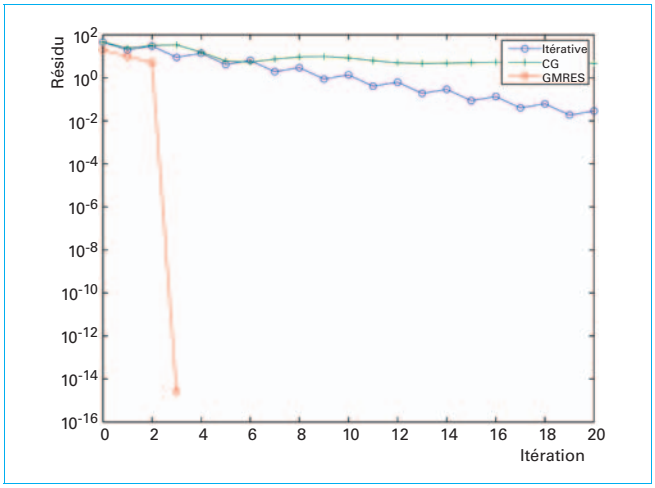


Figure 12 – Schwarz additif restreint utilisé comme préconditionneur pour le gradient conjugué (CG) ou GMRES, comparé à la méthode itérative

```
BarRAS;                                  % first part like in BarRAS.m
Mfun Y@(x)RNt'*(AN\(RN*x))+ROt'*(AO\(RO*x));
[u, fl, r, it, rcg]Y pcg(A, f, NeXS, OW, Mfun);
[u, fl, r, it, rgmresZY gmres(A, f, [Z, NeXS, NW, Mfun);
rgmres Y rgmres*ri(N);                   % plot absolute residual
semilogy( W:length(ri)XN,ri,'Xo', W:length(rcg)XN,rcg,'X+',
          W:length(rgmres)XN,rgmres,'Xd' );
xlabel('iteration')
ylabel('residual')
legend('Iterative','CG','GMRES')
```

Ici la méthode de Krylov GMRES converge à la troisième itération, une de moins que pour la méthode de Schwarz additif avec le gradient conjugué, car dans la matrice $I - M$ du système préconditionné, la matrice M est maintenant de rang 2.

Préconditionnement par sous-structuration

Une autre façon d'accélérer les méthodes de Schwarz itératives par des méthodes de Krylov est de les interpréter comme des algorithmes portant sur les variables d'interface, ici u_a et u_b. Pour cela, définissons les opérateurs de trace discrets

$$G_1 : \mathbb{R}^{a+d-1} \to \mathbb{R}, \quad (u_1, \ldots, u_a, \ldots, u_{b-1}) \mapsto u_a,$$
$$G_2 : \mathbb{R}^{J-a} \to \mathbb{R}, \quad (u_{a+1}, \ldots, u_b, \ldots, u_J) \mapsto u_b,$$

et les relèvements

$$E_1 : \mathbb{R} \to \mathbb{R}^{b-1}, \quad u_b \mapsto (0, \ldots, 0, u_b),$$
$$E_2 : \mathbb{R} \to \mathbb{R}^{J-a}, \quad u_a \mapsto (u_a, 0, \ldots, 0).$$

Appliquons à l'itération de Schwarz parallèle (27) l'opérateur de trace $\begin{pmatrix} G_1 & 0 \\ 0 & G_2 \end{pmatrix}$. Puisque

$$\tilde{B}_1 \boldsymbol{u}_2 = -\frac{1}{h^2} E_1 (u_2)_b, \quad \tilde{B}_2 \boldsymbol{u}_1 = -\frac{1}{h^2} E_2 (u_1)_a,$$

nous pouvons écrire un algorithme portant sur les seules inconnues d'interface $\left(u_1^n\right)_a$ et $\left(u_2^n\right)_b$:

$$\left(u_1^{n+1}\right)_a = \frac{1}{h^2} G_1 A_1^{-1} E_1 \left(u_2^n\right)_b + G_1 A_1^{-1} \boldsymbol{f}_1,$$
$$\left(u_2^{n+1}\right)_b = \frac{1}{h^2} G_2 A_2^{-1} E_2 \left(u_1^n\right)_a + G_2 A_2^{-1} \boldsymbol{f}_2.$$

L'algorithme de Schwarz parallèle se réécrit donc sous la forme d'un algorithme portant uniquement sur les inconnues d'interface $\left(g_1^n, g_2^n\right) = \left(\left(u_1^n\right)_a, \left(u_2^n\right)_b\right)$:

$$\begin{pmatrix} g_1^{n+1} \\ g_2^{n+1} \end{pmatrix} = \frac{1}{h^2} \begin{pmatrix} 0 & G_1 A_1^{-1} E_1 \\ G_2 A_2^{-1} E_2 & 0 \end{pmatrix} \begin{pmatrix} g_1^n \\ g_2^n \end{pmatrix} + \begin{pmatrix} G_1 A_1^{-1} \boldsymbol{f}_1 \\ G_2 A_2^{-1} \boldsymbol{f}_2 \end{pmatrix}. \tag{39}$$

Cet algorithme peut être vu, de nouveau, comme un algorithme de Jacobi par blocs (ou de Richardson, car la diagonale est l'identité) pour le système

$$\begin{pmatrix} I & -\frac{1}{h^2} G_1 A_1^{-1} E_1 \\ -\frac{1}{h^2} G_2 A_2^{-1} E_2 & I \end{pmatrix} \begin{pmatrix} g_1 \\ g_2 \end{pmatrix} = \begin{pmatrix} G_1 A_1^{-1} \boldsymbol{f}_1 \\ G_2 A_2^{-1} \boldsymbol{f}_2 \end{pmatrix}. \tag{40}$$

Ce dernier système peut alors être résolu par un algorithme de Krylov. Le script suivant contient les algorithmes de Schwarz et Krylov en version sous-structurée.

```
Bar;                                     % to include problem parameters
α=8; d=4;                                % subdomain decomposition
fl=f(1:α+d-1); f2=f(α+1:J);              % subdomain source terms
x1=x(1:α+d+1); x2=x(α+1:end);            % finite difference meshes
G1=zeros(1, α+d+1); G1(end-d)=1;         % construct substructured system :
G2=zeros(1, J-α+2); G2(d+1)=1;           %
b=[G1*Solve1d(f1, eta, x1(1)x1(end), gg, 0);   % T*g = b, g unknows at interfaces
   G2*Solve1d(f2, eta, x2(1), x2(end), 0, gd);]
T=@(g)[g(1)-G1*Solve1d(zeros(size(f1)), eta, x1(1), x1(end), 0, g(2));
       g(2)-G2*Solve1d(zeros(size(f2)), eta, x2(1), x2(end), g(1), 0)];
g=[0;0];                                 % zero initial guess
ri(1)=norm(b-T(g));                      % intial residual
for i=1:20                               % classial Schwarz iteration
   g=g-T(g)+b;                           % gn = (I-T)*g+b
   ri(i+1)=norm(b-T(g));                 % keep residual for plotting
end
[g, fl, r, it, rk]=gmres(T, b);          % use Krylov solver
rk(end+1)=norm(b-T(g));                  % add residual of re sult
semilogy(0:length(ri)-1, ri, '-o', 0:length(rk)-1, rk, '-+');
xlabel('iteration'); ylabel('residual'); legend('Iterative','Krylov')
```

La figure **13** compare la convergence des deux algorithmes, itératif ou préconditionneur pour Krylov, pour un recouvrement de 4 points de grille.

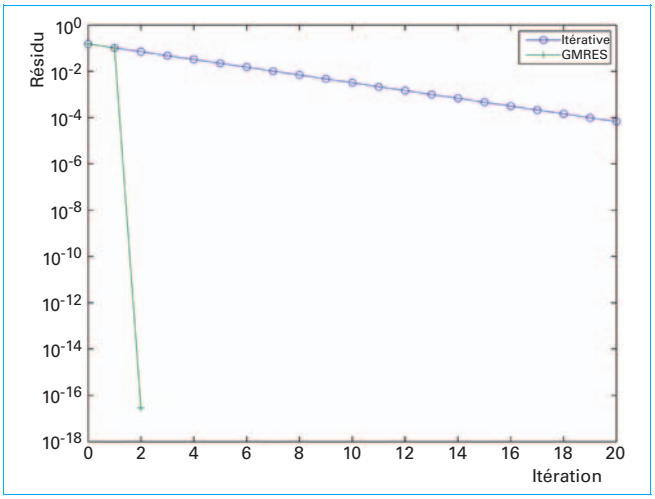


Figure 13 – Système sous-structuré résolu par méthode de Krylov (GMRES) et itérative

Le problème sous-structuré est un système de deux équations à deux inconnues g_1 et g_2. La méthode de Krylov converge donc en deux itérations.

Remarque 2.4 ***1.*** Dans le cas où la matrice A est symétrique définie positive, le préconditionneur AS dans (33) l'est aussi, ce qui permet d'utiliser la méthode du gradient conjugué, qui est la méthode de Krylov de choix pour les matrices symétriques définies positives, car elle converge d'une manière optimale avec une récurrence courte. Il n'en est plus de même pour le préconditionneur RAS, voir (35), et le problème d'interface (40). Cet avantage de AS disparaît lorsque le problème de départ n'est pas symétrique (équation d'advection-diffusion par exemple), ou pas défini positif (équation de Helmholtz par exemple).

2. En dimension $d \geqslant 1$, si le nombre de points dans chaque direction est de l'ordre de J, la dimension du système d'interface (35) est J^{d-1} tandis que celle du système complet préconditionné en volume (par exemple (33) pour AS) est J^d. Nous voyons maintenant que la taille du système d'interface (35) le rend plus avantageux dans l'utilisation d'une méthode de Krylov, où il faut stocker les directions de descente. Ceci est vrai en particulier si l'on utilise la méthode GMRES, où il faut stocker toutes les directions de descente.

3. Il est possible d'utiliser des solveurs approchés pour la résolution dans les sous-domaines dans les formulations non sous-structurées, i.e. les méthodes de Schwarz discrètes, car la solution du problème de départ n'en est pas affectée. On voit cela clairement par exemple dans la formulation de Schwarz additif (33) : si l'on remplace les matrices A_i^{-1} par des approximations, la solution u de (33) ne change pas, car le système préconditionné est équivalent au système de départ. Par contre l'utilisation de solveurs approchés dans les formulations sous-structurées des méthodes de Schwarz changent la solution, comme on le voit clairement dans (40) : si l'on remplace maintenant les matrices A_i^{-1} par des approximations, la matrice du système d'interface change, et donc la solution (g_1, g_2) à l'interface également.

La formulation sous-structurée des méthodes de Schwarz peut également être décrite sur le problème continu. Pour g_1 et g_2 dans $\mathbb{R}$, définissons $\mathfrak{U}_1^{\mathcal{D}}(g_2, g_g, f)$ et $\mathfrak{U}_2^{\mathcal{D}}(g_1, g_d, f)$ solutions des problèmes aux limites

$$-\frac{d^2u_1}{dx^2}+\eta u_1 = f \text{ dans } \Omega_1, \quad -\frac{d^2u_2}{dx^2}+\eta u_2 = f \text{ dans } \Omega_2, \qquad (41)$$
$$u_1(0) = g_g,\ u_1(\beta) = g_2, \quad u_2(\alpha) = g_1,\ u_2(1) = g_d.$$

On a donc pour la méthode alternée de Schwarz

$$u_1^{n+1} = \mathfrak{U}_1^{\mathcal{D}}\left(u_2^n(\beta), g_g, f\right), \quad u_2^{n+1} = \mathfrak{U}_2^{\mathcal{D}}\left(u_2^{n+1}(\alpha), g_d, f\right),$$

et le problème en variable d'interface s'écrit au moyen des opérateurs de trace $\gamma_1 : u \to u(\alpha)$ et $\gamma_2 : u \to u(\beta)$:

$$u_1^{n+1}(\alpha) = \gamma_1\mathfrak{U}_1^{\mathcal{D}}\left(u_2^n(\beta), g_g, f\right), \quad u_2^{n+1}(\beta) = \gamma_2\mathfrak{U}_2^{\mathcal{D}}\left(u_1^{n+1}(\alpha), g_d, f\right),$$

ou encore, en notant $g_2^n = u_2^n(\beta)$ et $g_1^n = u_1^n(\alpha)$,

$$g_1^{n+1} = \gamma_1\mathfrak{U}_1^{\mathcal{D}}\left(g_2^n, g_g, f\right), \quad g_2^{n+1} = \gamma_2\mathfrak{U}_2^{\mathcal{D}}\left(g_1^{n+1}, g_d, f\right),$$

ce qui donne matriciellement l'itération

$$\begin{pmatrix} 1 & 0 \\ -\gamma_2\mathfrak{U}_2^{\mathcal{D}}(\cdot,0,0) & 1 \end{pmatrix}\begin{pmatrix} g_1^{n+1} \\ g_2^{n+1} \end{pmatrix} = \begin{pmatrix} 0 & \gamma_1\mathfrak{U}_1^{\mathcal{D}}(\cdot,0,0) \\ 0 & 0 \end{pmatrix}\begin{pmatrix} g_1^n \\ g_2^n \end{pmatrix} + \begin{pmatrix} \gamma_1\mathfrak{U}_1^{\mathcal{D}}(0,g_g,f) \\ \gamma_2\mathfrak{U}_2^{\mathcal{D}}(0,g_d,f) \end{pmatrix}.$$

Nous reconnaissons un algorithme de Gauss-Seidel pour la résolution du problème d'interface

$$\begin{pmatrix} 1 & -\gamma_1\mathfrak{U}_1^{\mathcal{D}}(\cdot,0,0) \\ -\gamma_2\mathfrak{U}_2^{\mathcal{D}}(\cdot,0,0) & 1 \end{pmatrix}\begin{pmatrix} g_1 \\ g_2 \end{pmatrix} = \begin{pmatrix} \gamma_1\mathfrak{U}_1^{\mathcal{D}}(0,g_g,f) \\ \gamma_2\mathfrak{U}_2^{\mathcal{D}}(0,g_d,f) \end{pmatrix}. \qquad (42)$$

Avec les notations précédentes, l'algorithme de Schwarz parallèle s'écrit aussi en formulation sous-structurée à l'interface.

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} g_1^{n+1} \\ g_2^{n+1} \end{pmatrix} = \begin{pmatrix} 0 & \gamma_1\mathfrak{U}_1^{\mathcal{D}}(\cdot,0,0) \\ \gamma_2\mathfrak{U}_2^{\mathcal{D}}(\cdot,0,0) & 0 \end{pmatrix}\begin{pmatrix} g_1^n \\ g_2^n \end{pmatrix} + \begin{pmatrix} \gamma_1\mathfrak{U}_1^{\mathcal{D}}(0,g_g,f) \\ \gamma_2\mathfrak{U}_2^{\mathcal{D}}(0,g_d,f) \end{pmatrix}.$$

ce qui est un algorithme de Jacobi pour la résolution du problème d'interface (42).

Rayon spectral et conditionnement en dimension 2

En dimension 1, le problème sous-structuré en variables d'interface ne comporte que deux inconnues et les méthodes de Krylov convergent alors en au plus deux iterations. Pour obtenir un problème d'interface intéressant, il faut passer en dimension 2. Considérons donc l'équation $\eta u - \Delta u = f$ dans le rectangle $\Omega = [0, 1] \times [0, 1]$. Le domaine est partagé en deux sous-domaines constitués de deux rectangles $\Omega_1 = [0, \beta] \times [0, 1]$ et $\Omega_2 = [\alpha, 1] \times [0, 1]$.

Les méthodes de Schwarz alternée et parallèle gardent la même forme, avec les conditions aux limites portant maintenant sur toute l'étendue des segments $\Gamma_1 = \{\beta\} \times [0, 1]$ et $\Gamma_2 = \{\alpha\} \times [0, 1]$. Les erreurs peuvent être développées en séries de sinus dans la variable y :

$$e_i^n(x,y) = \sum_{k=0}^{\infty} \hat{e}_i^n(x,k)\sin k\pi y.$$

L'équation $\eta e_i^n - \Delta e_i^n = 0$ devient après développement en série de sinus l'équation monodimensionnelle (14), où η est remplacé par $\pi + k^2\pi^2$. Cette équation est résolue comme dans la démonstration du théorème 2.1, et l'erreur dans le sous-domaine i à l'étape n est alors donnée par

$$e_i^n(x,y) = \sum_k \hat{e}_i^n(k)\,\mathrm{sh}\sqrt{\eta + k^2\pi^2}\,x \sin k\pi y.$$

Pour l'algorithme de Schwarz parallèle par exemple, on obtient

$$\hat{e}_1^{n+1}(k) = \rho_1(k)\hat{e}_2^n(k), \quad \hat{e}_2^{n+1}(k) = \rho_2(k)\hat{e}_1^n(k),$$

avec

$$\rho_1(k) = \frac{\mathrm{sh}\left(\sqrt{\eta + k^2\pi^2}\,(1-\beta)\right)}{\mathrm{sh}\left(\sqrt{\eta + k^2\pi^2}\,\beta\right)}, \quad \rho_2(k) = \frac{\mathrm{sh}\left(\sqrt{\eta + k^2\pi^2}\,\alpha\right)}{\mathrm{sh}\left(\sqrt{\eta + k^2\pi^2}\,(1-\alpha)\right)}. \qquad (43)$$

On obtient de nouveau une formule de récurrence sur une double itération :

$$\hat{e}_i^{n+1}(k) = \rho^{(\mathcal{D})}(k)\hat{e}_i^{n-1}(k), \quad \text{avec } \rho^{(\mathcal{D})}(k) = \rho_1(k)\rho_2(k). \qquad (44)$$

Théorème 2.3 L'algorithme de Schwarz parallèle en dimension 2 est convergent. Avec un recouvrement $\delta := \beta - \alpha$, le facteur de convergence $\rho^{(\mathcal{D})}(k)$ satisfait

$$\sup_k \rho^{(\mathcal{D})}(k) = 1 - \mathbb{O}(\delta).$$

Démonstration Comme en dimension 1, il est facile de voir que $\rho^{(\mathcal{D})}(k)$ est strictement compris entre 0 et 1. Mieux encore, un calcul direct montre que la fonction $k \mapsto \rho^{(\mathcal{D})}(k)$ est décroissante. Elle atteint son maximum en $k = 1$:

$$\rho^{(\mathcal{D})}(k) \leq \rho^{(\mathcal{D})}(1) < 1.$$

Étudions par exemple la suite e_1^{2n}. Puisque, pour tout k, $\hat{e}_1^{2n}(k) = \left(\rho^{(\mathcal{D})}(k)\right)^n \hat{e}_1^0(k)$, nous pouvons écrire

$$e_1^{2n}(x,y) = \sum_k \left(\rho^{(\mathcal{D})}(k)\right)^n \hat{e}_1^0(k) \operatorname{sh}\sqrt{\eta + k^2\pi^2}\, x \sin k\pi y,$$

et la norme L^2 de e_1^{2n} peut être calculée explicitement :

$$\begin{aligned}
\left\|e_1^{2n}\right\|_{\Omega_1}^2 &= \int_{\Omega_1} \left|e_1^{2n}(x,y)\right|^2 dxdy \\
&= \sum_{k\in\mathbb{Z}} \left(\rho^{(\mathcal{D})}(k)\right)^{2n} \left|\hat{e}_1^0(k)\right|^2 \int_{\Omega_1} sh^2\left(\sqrt{\eta + k^2\pi^2}\, x\right) \sin^2(k\pi y)\, dxdy \\
&\leq \left(\rho^{(\mathcal{D})}(1)\right)^{2n} \sum_{k\in\mathbb{Z}} \left|\hat{e}_1^0(k)\right|^2 \int_{\Omega_1} sh^2\left(\sqrt{\eta + k^2\pi^2}\, x\right) \sin^2(k\pi y)\, dxdy \\
&\leq \left(\rho^{(\mathcal{D})}(1)\right)^{2n} \left\|e_1^0\right\|_{\Omega_1}^2
\end{aligned}$$

et nous obtenons finalement

$$\left\|e_1^{2n}\right\|_{\Omega_1} \leq \left(\rho^{(\mathcal{D})}(1)\right)^n \left\|e_1^0\right\|_{\Omega_1}.$$

Puisque $\rho^{(\mathcal{D})}(1) < 1$, la suite e_1^{2n} tend vers 0 dans $L^2(\Omega_1)$, et donc la suite u_1^{2n} tend vers u. Il en est de même pour les itérées paires, et pour l'autre sous-domaine.

Le facteur $\rho^{(\mathcal{D})}(1)$ mesure la vitesse de convergence de la suite. Il est intéressant de connaître son comportement lorsque le recouvrement tend vers 0. Pour cela effectuons un développement limité de $\rho^{(\mathcal{D})}(1)$ pour δ petit en remplaçant β par $\alpha + \delta$. Commençons par ρ_1 :

$$\begin{aligned}
\rho_1(1) &= \frac{\operatorname{sh}\left(\sqrt{\eta+\pi^2}\,(1-\alpha-\delta)\right)}{\operatorname{sh}\left(\sqrt{\eta+\pi^2}\,(\alpha+\delta)\right)} \\
&= \frac{\operatorname{sh}\left(\sqrt{\eta+\pi^2}\,(1-\alpha)\right)\operatorname{ch}\left(\sqrt{\eta+\pi^2}\,\delta\right) - \operatorname{ch}\left(\sqrt{\eta+\pi^2}\,(1-\alpha)\right)\operatorname{sh}\left(\sqrt{\eta+\pi^2}\,\delta\right)}{\operatorname{sh}\left(\sqrt{\eta+\pi^2}\,\alpha\right)\operatorname{ch}\left(\sqrt{\eta+\pi^2}\,\delta\right) + \operatorname{ch}\left(\sqrt{\eta+\pi^2}\,\alpha\right)\operatorname{sh}\left(\sqrt{\eta+\pi^2}\,\delta\right)} \\
&= \frac{\operatorname{sh}\left(\sqrt{\eta+\pi^2}\,(1-\alpha)\right) - \delta\sqrt{\eta+\pi^2}\operatorname{ch}\left(\sqrt{\eta+\pi^2}\,(1-\alpha)\right) + \mathbb{O}(\delta^2)}{\operatorname{sh}\left(\sqrt{\eta+\pi^2}\,\alpha\right) + \sqrt{\eta+\pi^2}\operatorname{ch}\left(\sqrt{\eta+\pi^2}\,\alpha\right) + \mathbb{O}(\delta^2)} \\
&= \frac{\operatorname{sh}\left(\sqrt{\eta+\pi^2}\,(1-\alpha)\right)}{\operatorname{sh}\left(\sqrt{\eta+\pi^2}\,\alpha\right)} \frac{1 - \delta\sqrt{\eta+\pi^2}\coth\left(\sqrt{\eta+\pi^2}\,(1-\alpha)\right) + \mathbb{O}(\delta^2)}{1 + \delta\sqrt{\eta+\pi^2}\coth\left(\sqrt{\eta+\pi^2}\,\alpha\right) + \mathbb{O}(\delta^2)} \\
&= \frac{1}{\rho_2(1)}\left(1 - \delta\sqrt{\eta+\pi^2}\left(\coth\left(\sqrt{\eta+\pi^2}\,(1-\alpha)\right) + \coth\left(\sqrt{\eta+\pi^2}\,\alpha\right)\right) + \mathbb{O}(\delta^2)\right),
\end{aligned}$$

si bien que

$$\begin{aligned}
\rho^{\mathcal{D}}(1) &= \rho_1(1)\rho_2(1) \\
&= 1 - \delta\sqrt{\eta+\pi^2}\left(\coth\left(\sqrt{\eta+\pi^2}\,(1-\alpha)\right) + \coth\left(\sqrt{\eta+\pi^2}\,\alpha\right)\right) + \mathbb{O}(\delta^2),
\end{aligned}$$

ce qui établit le développement limité du théorème. •

Comme en dimension 1, cet algorithme correspond à un algorithme de Jacobi pour le problème d'interface (42).

On se donne maintenant une grille (i, j), isotrope pour simplifier, c'est-à-dire $h_x = h_y = h$, et le schéma aux différences finies pour $u_{i,j} \sim u(x_i, y_j)$,

$$-\frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} - \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{h^2} + \eta\, u_{i,j} = f_{i,j},$$

avec les conditions aux limites $u_{0,\,j} = (g_g)_j$ et $u_{J+1,j} = (g_d)_j$ pour $0 \leqslant j \leqslant J+1$, et $u_{i,0} = u_{i,\,J+1} = 0$ pour $0 \leqslant i \leqslant J+1$. Voici notre code Matlab qui construit la matrice associée en dimension deux à partir de la matrice A1d en dimension 1, à l'aide du produit tensoriel, ou produit de Kronecker.

```
function A=A2d(eta, Jx, Jy);
% A2D finite difference approximation of eta-Delta in 2d
%   A=A2d(eta, Jx, Jy); constructs the finite difference approximation to
%   eta-Delta on a Jx x Jy grid with spacing h=1/(Jy+1) and homogeneous
%   Dirichlet conditions all around

h=1/(Jy+1); Dxx=A1d(eta, 0, h*(Jx+1), Jx); Dyy=A1d(0, 0, 1, Jy);
A=kron(speye(size(Dxx)), Dyy)+kron(Dxx, speye(size(Dyy)));
```

Les valeurs propres de la matrice A associée sont

$$\lambda_{j,k}^d = \eta + \frac{4}{h^2}\left(\sin^2\frac{j\pi\, h}{2} + \sin^2\frac{k\pi\, h}{2}\right) = \lambda_j^d + \lambda_k^d - \eta. \quad 1 \leqslant j, k \leqslant J,$$

où λ_j^d est définie dans (36) et contient η, ce qui explique pourquoi il faut enlever de nouveau un η après la somme de l'expression à droite. Le conditionnement de la matrice A est donné par

$$\kappa(A) = \frac{2\lambda_J^d - \eta}{2\lambda_1^d - \eta} \sim \frac{2}{\eta + 2\pi^2}\frac{1}{h^2}. \tag{45}$$

Voici une implémentation en dimension 2 de notre solveur en Matlab :

```
function u=Solve2d(f, eta, ai, bi, gg, gd)
% SOLVE2D solves 2d eta-Delta using a finite difference approximation
%   u=Solve2d(f, eta, a, b, gg, gd) solves the two dimensional equation
%   (eta-Delta) u=f on the domain Omega=(ai*h, bi*h) (0,1) with
%   Dirichlet boundary conditions u=gg at x=ai*h and u=gd at x=bi*h
%   and u=0 at y=0 and y=1 using a finite difference approximation
%   with interior grid points (bi-ai) times length(gg) using the same
%   mesh size h=1/(length(gg)+1) in both x and y.

nx=bi-ai-1;
ny=length(gg);
h=1/(length(gg)+1);
A=A2d(eta, nx, ny);
f(1:ny, 1)=f(1:ny,1)+gg/h^2;        % add boundary conditions into rhs
f(1:ny, end)=f(1:ny,end)+gd/h^2;
u=A\f(:);
u=reshape(u, ny, nx);
u=[gg u gd];                        % add boundary values to solution
```

Nous l'utilisons maintenant pour calculer un exemple modèle, mais déjà intéressant : la distribution de température dans une pièce chauffée par un poêle situé en son centre (représenté par le second membre f). Les trois murs extérieurs sont froids, le mur ouest est chauffé par une pièce voisine, et contient une porte. Le

petit programme Matlab Room.m ci-après effectue ce calcul et trace le résultat figure **14**.

```
eta=0 ; J=20;                    % number of interior mesh points in x and y direction
x=0:1/(J+1) : 1; y=x ;           % finite difference mesh, including boundary
f=zeros(J, J) ;                  % source term does not include boundar
xi=x(2 : end-1) ; yi=xi ;
f([yi>0.4 & yi<0.6], [xi > 0.4 & xi < 0.6])=50 ;
gg=0.3*ones(J, 1) ; gg(yi > 0.5 & yi < 0.9)=1 ;
gd=zeros(J, 1) ;
u=Solve2d(f, eta, 0, J+1, gg, gd) ;
u=[zeros(1, J+2) ; u ; zeros(1, J+2)] ;
mesh(x(1 : end), y, u) ; xlabel('x') ; ylabel('y') ; zlabel('solution') ;
```

Voici maintenant le programme Matlab de la méthode de Schwarz parallèle discrétisée en dimension 2. Le recouvrement est de $d = 4$, soit $\beta - \alpha = 4h$. Les six premières itérations sont représentées figure **15**.

```
Room ;                                % include problem parameters
α=8; d=4;                             % decomposition
f1=f(:, 1: α+d-1) ; f2=f(:, α+1 : end) ;
u1=[gg zeros(J, α+d)] ;               % zero initial guess, except boundary value
u2=[zeros(J, J-α+1) gd] ;
h=1/(J+1) ; y=0 : h : 1 ;             % finite difference meshes
x1=0 : h : (α+d)*h ; x2=α*h : h : 1 ;
z1=zeros(1, α+d+1) ; z2=zeros(1, J-α+2) ;  % for plotting purposes
for i=1 : 20
   u1n=Solve2d(f1, eta, 0, α+d, gg, u2(:, d+1)) ;
   u2n=Solve2d(f2, eta, α, J+1, u1(:, end-d), gd) ;
   u1=u1n ;
   u2=u2n ;
   mesh(x1, y, [z1 ; u1n ; z1]) ; hold on ; mesh(x2, y, [z2 ; u2n ; z2]) ; hold off
   xlabel('x') ; ylabel('y') ; zlabel('Schwarz iterates') ;
   pause
end
```

Si l'on essaie d'utiliser la méthode de Schwarz additive stationnaire (32), on constate, comme en dimension 1, des oscillations dans le recouvrement, représentées sur la figure **16**.

```
Room ;
α=8 ; d=4 ;                           % decomposition
h=1/(J+1) ;
f=f(:)   ;
f(1 : J)=f(1 : J)+gg/h^2 ;            % add boundary conditions into rhs
f(end-J+1 : end)=f(end-J+1 : end)+gd/h^2 ;
A=A2d(eta, J, J) ;
R1=[speye(J*(α+d-1)) sparse(J*(α+d-1), J*(J-α-d+1))] ;
R2=[sparse(J*(J-α), J*α) speye(J*(J-α))] ;
A1=R1*A*R1' ;
A2=R2*A*R2' ;
u=zeros(J^2, 1) ;
x=0 : h : 1 ; y=x ;                   % finite difference mesh, including boundary
for i=1 : 20
   r=f-A*u ;
   u=u+(R1'*(A1\(R1*r))R2'*(A2\(R2*r))) ;
   mesh(x, y, [zeros(1, J+2) ; gg reshape(u, J, J) gd ; zeros(1, J+2)]) ;
   xlabel('x') ; ylabel('y') ; zlabel('Additive Schwarz iterates') ;
   pause
end
```

Nous abordons maintenant les formulations de Schwarz considéré comme préconditionneur. Nous ne donnons ici que les résultats graphiques, les scripts sont analogues aux formulations en dimension 1.

D'abord les calculs en sous-structuration : nous comparons la résolution du système correspondant à (40) en dimension deux par méthode itérative ou méthode de Krylov. Les deux courbes de convergence sont représentées figure **17**. Elles montrent bien le gain qu'apporte l'utilisation de l'algorithme de Krylov.

Pour le même cas test, avec la méthode de Schwarz additive (33), les résultats sont représentés figure **18**. Ici, comme le problème est symétrique, l'algorithme du gradient conjugué peut être utilisé. Par contre, comme en dimension 1, l'itération stationnaire ne converge pas.

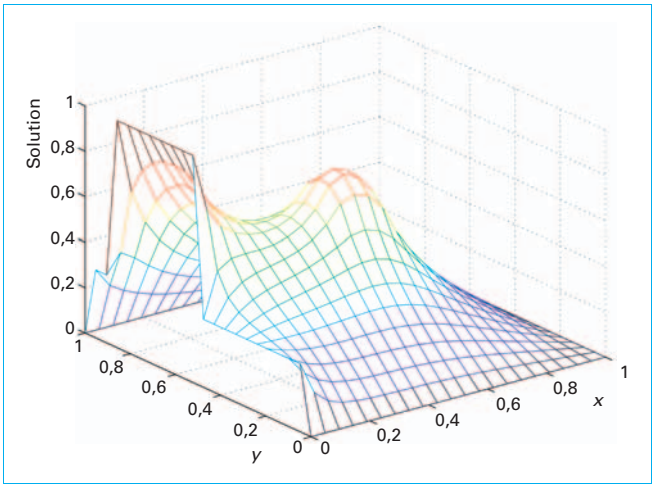


Figure 14 – Chauffage d'une pièce par un poêle

Pour finir, nous avons représenté figure **19** les calculs correspondant à l'algorithme de Schwarz restreint (RAS). Nous comparons ici trois méthodes : itérative, gradient conjugué et GMRES qui est adapté aux matrices non symétriques.

Dans cet exemple, nous voyons que pour RAS, la méthode du gradient conjugué converge plutôt bien, compte-tenu du fait que RAS est un préconditionneur non symétrique. Ce n'était pas le cas en dimension 1. La méthode GMRES donne une convergence encore un peu meilleure, ce qui est logique dans la mesure où elle est adaptée aux préconditionneurs non symétriques, cependant elle a un coût plus élevé que la méthode du gradient conjugué (voir [40]).

Nous représentons maintenant les spectres des matrices préconditionnées $M_{AS}^{-1}A$ et $M_{RAS}^{-1}A$, et nous étudions leur comportement lorsque le pas h tend vers 0. Rappelons que si le spectre de la matrice préconditionnée est proche de 1, un algorithme de Krylov pour la résolution du système est en général rapide. D'abord nous considérons dans la figure **20** le cas où le recouvrement est maintenu constant.

De ***a*** à ***d***, dans la figure **20**, le pas est divisé par deux à chaque fois. Il y a donc de plus en plus de points dans le recouvrement.

Les spectres de AS et de RAS sont tous deux contenus dans le segment [0, 2], et présentent peu de différences. Dans le cas d'un recouvrement minimal à gauche, les deux méthodes sont identiques, mais dès qu'il y a des points de grille dans le recouvrement, le mode non convergent de AS apparaît comme associé à la valeur propre supplémentaire de valeur 2. On voit aussi que si l'on raffine le maillage, l'intervalle du spectre reste inchangé. Ceci illustre le fait que ces méthodes convergent indépendamment de h lorsque le recouvrement est indépendant de h, voir théorème 2.3.

Dans la série de courbes de la figure **21**, nous fixons maintenant le recouvrement à 2 points de grille, *i. e.* $\delta = 2h$, et nous faisons varier h comme dans la figure **20**. La taille du recouvrement tend donc vers 0 avec h.

De nouveau, les spectres de AS et RAS sont très similaires, sauf pour les modes non convergents de AS. Mais cette fois les spectres tendent à remplir tout l'intervalle [0, 2] lorsque h diminue : les méthodes de Krylov utilisant ces préconditionneurs deviennent alors plus lents lorsque l'on raffine le maillage et que le recouvrement est de l'ordre de h.

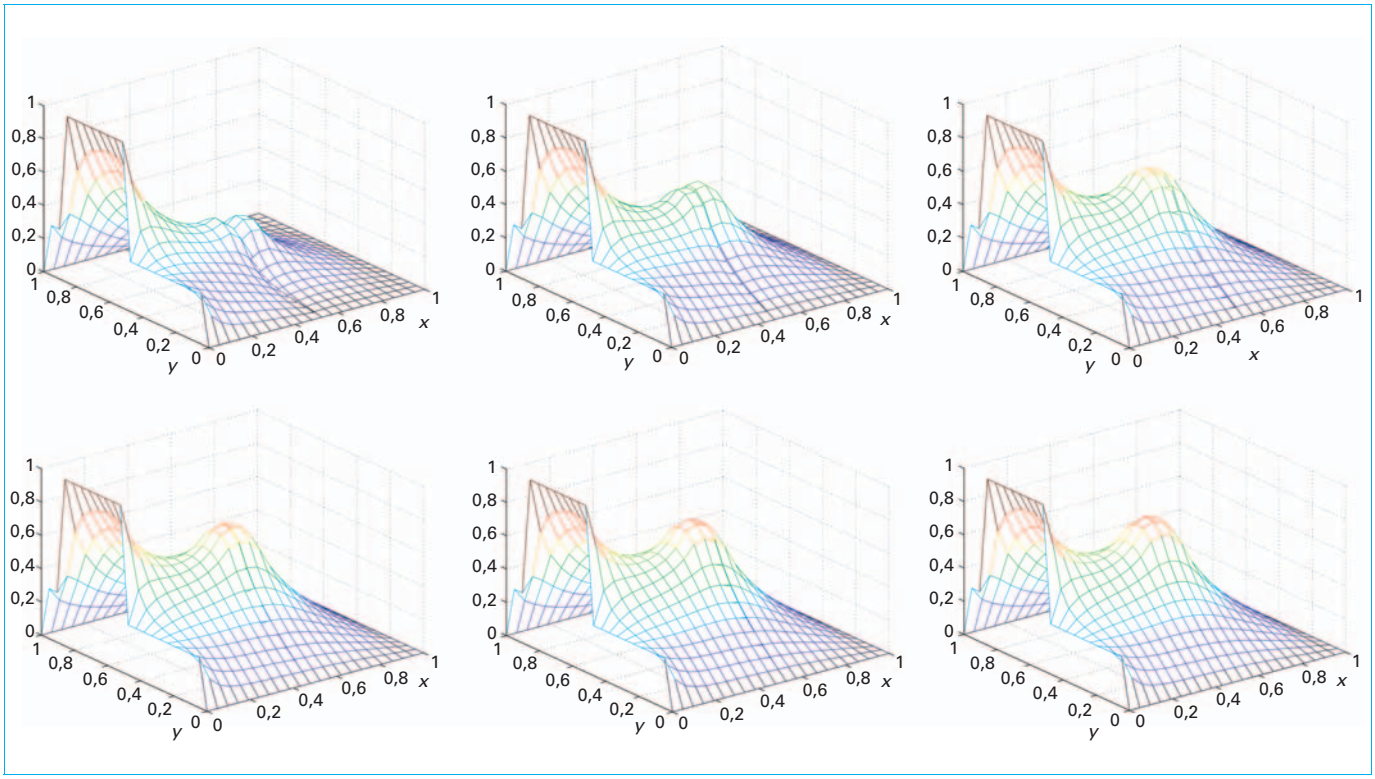


Figure 15 – Résolution du problème de chauffage par l'algorithme de Schwarz parallèle : 6 premières itérations

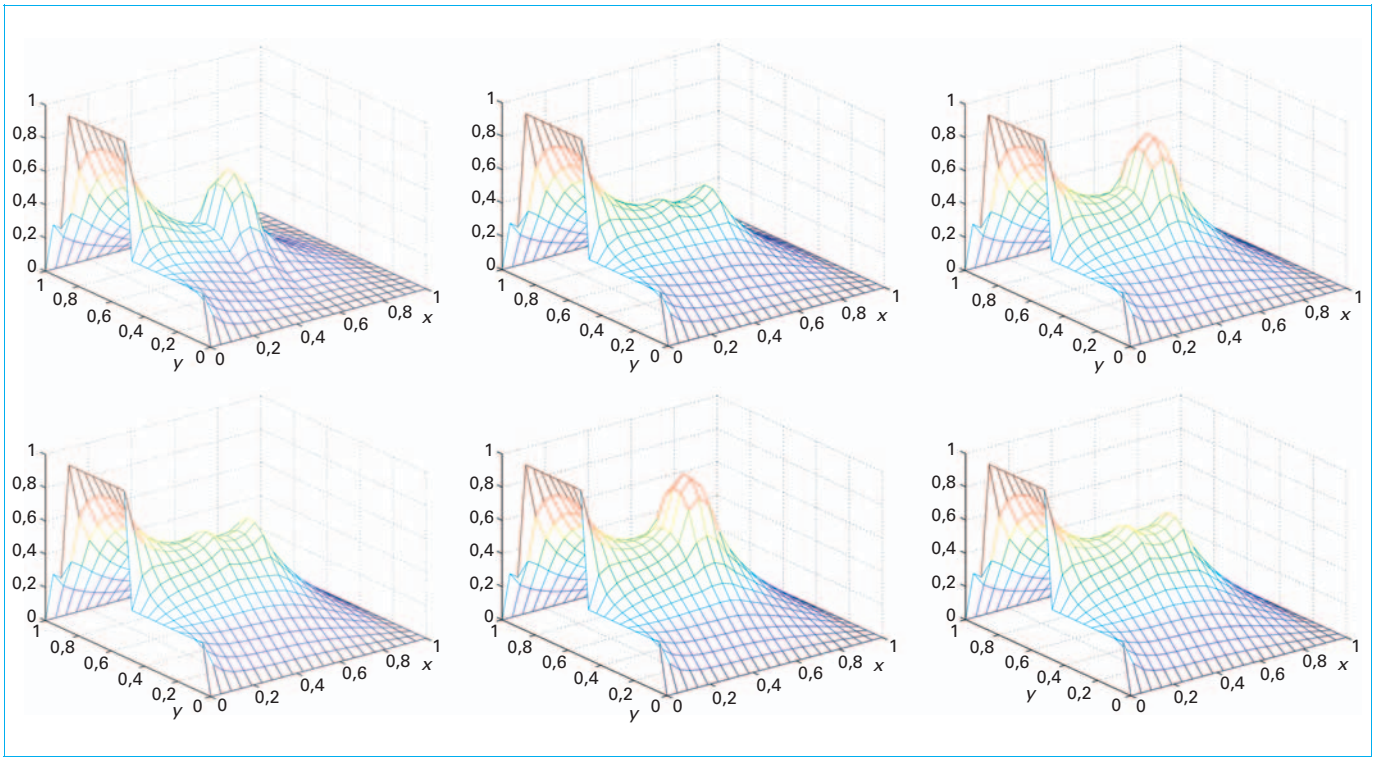


Figure 16 – Essai de résolution avec Schwarz additif, mettant en évidence le mode oscillatoire dans le recouvrement

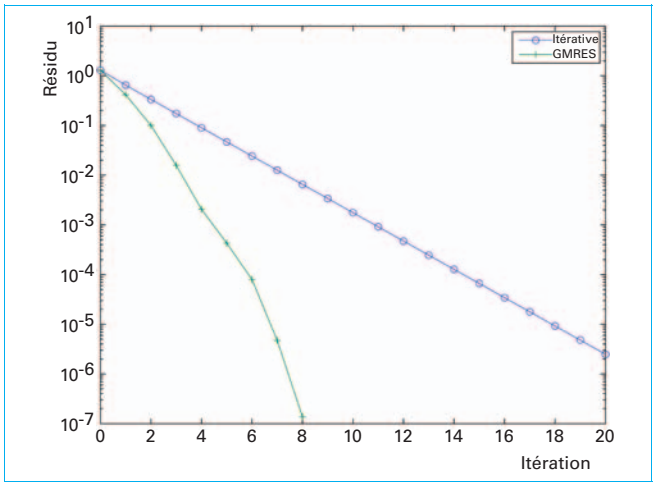


Figure 17 – Résolution du problème sous-structuré (40) en dimension 2 par méthode itérative ou GMRES

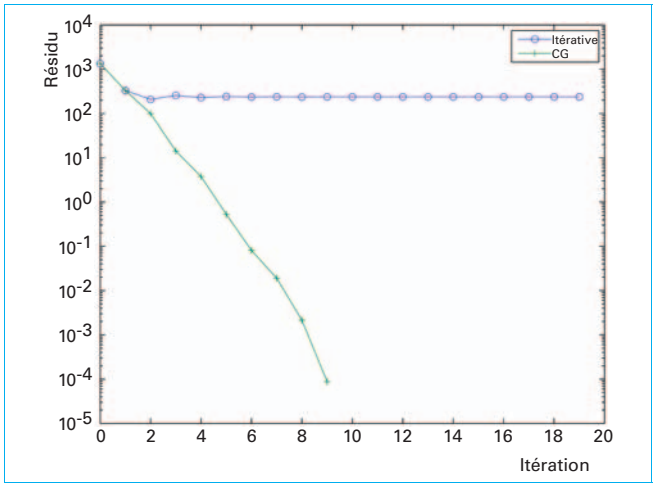


Figure 18 – Résolution en dimension 2 par Schwarz additif : version itérative ou gradient conjugué

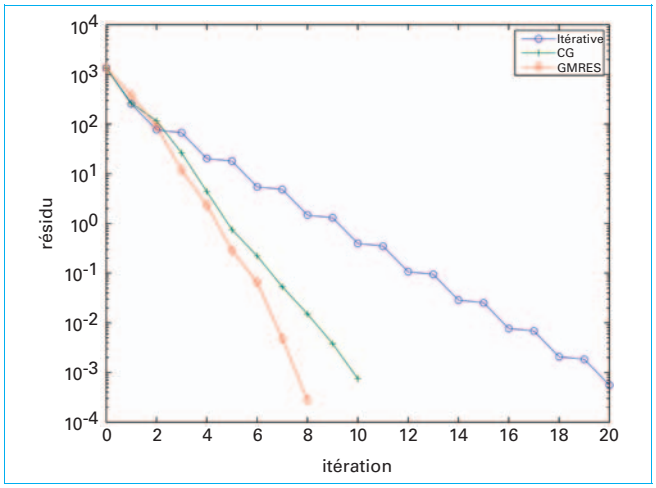


Figure 19 – Résolution en dimension 2 par Schwarz additif restreint : version itérative, gradient conjugué ou GMRES

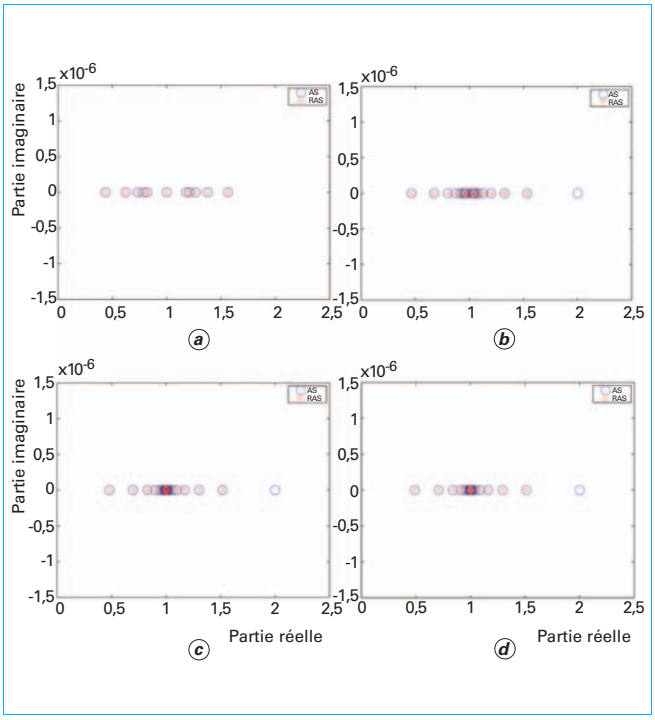


Figure 20 – Comparaison des spectres de AS and RAS comme préconditionneurs, avec recouvrement δ constant, indépendant du pas du maillage

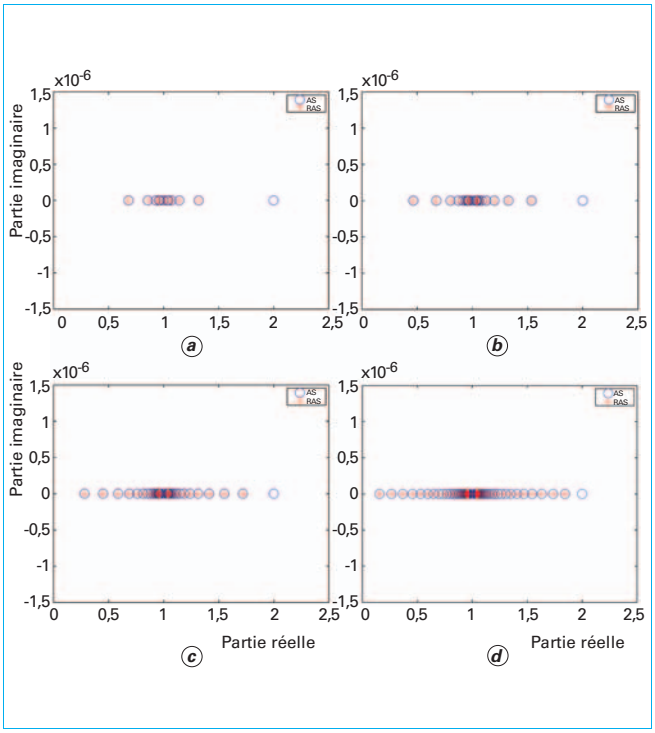


Figure 21 – Comparaison des spectres de AS et RAS comme préconditionneurs, avec recouvrement dépendant du pas de maillage, $\delta = 2h$

2.5 Méthodes de Schwarz optimisées

Les méthodes de Schwarz optimisées établissent un pont entre les méthodes de Schwarz classiques, qui fonctionnent avec du recouvrement, et les méthodes de Schur que nous verrons plus tard, et qui n'utilisent pas de recouvrement. Elles permettent d'améliorer nettement les facteurs de convergence à la fois des méthodes stationnaires et des méthodes de Krylov ; elles peuvent être utilisées sans recouvrement, pour une introduction, voir [15]. Nous nous plaçons dès maintenant en dimension 2. Dans [28], P.L. Lions propose de remplacer les conditions de transmission portant sur la valeur de la fonction par des conditions de type Robin (ou encore Fourier en thermique), ce qui donne l'algorithme suivant :

$$\begin{aligned} \eta\, u_1^n - \Delta u_1^n &= f \quad \text{dans } \Omega_1, \\ u_1^n(0,\cdot) &= g_g, \\ (\partial_x + p)u_1^n(\beta,\cdot) &= (\partial_x + p)u_2^{n-1}(\beta,\cdot), \end{aligned} \tag{46}$$

$$\begin{aligned} \eta\, u_2^n - \Delta u_2^n &= f \quad \text{dans } \Omega_2, \\ (\partial_x - p)u_2^n(\alpha,\cdot) &= (\partial_x - p)u_1^{n-1}(\alpha,\cdot), \\ u_2^n(1,\cdot) &= g_d. \end{aligned}$$

Pour tout $p > 0$, les problèmes aux limites dans les sous-domaines sont bien posés : il existe une et une seule solution. On peut mettre les problèmes sous forme variationnelle et les traiter par une méthode d'éléments finis. On peut aussi les résoudre par différences finies, c'est ce que nous faisons dans le script ci-après, où nous avons utilisé une modification de notre procédure Solve pour tenir compte de conditions de Robin sur les bords à gauche et à droite :

```
function u=Solve2dR(f, eta, ai, bi, gg, gd, p1, p2)
% SOLVE2DR solves 2d eta-Delta using Robin conditions
%   u=Solve2dR(f, eta, a, b, gg, gd, n, p1, p2) solves the two dimensional
%   equation (eta-Delta)u=f on the domain Omega=(ai*h, bi*h)x(0, 1) with
%   Robin boundary conditions (dn+p1)u=gg at x=ai*h and (dn+p2)u=gd at
%   x = bi*h and u=0 at y=0 and y=1 using a finite difference
%   approximation with interior grid points (bi-ai) timeslength(gg)
%   using the same mesh size h=1/(length(gg)+1) in both x and y.

nx=bi-ai+1;
ny=length(gg);
h=1/(length(gg)+1);
A=A2d(eta, nx, ny);
A(1:ny, 1:ny)=A(1:ny, 1:ny)/2+p1/h*speye(ny);
A(end-ny+1:end, end-ny+1:end)
                    =A(end-ny+1:end, end-ny+1:end)/2+p2/h*speye(ny);
f(1:ny, 1)=f(1:ny, 1)/2+gg/h;      % add boundary conditions into rhs
f(1:ny, end)=f(1:ny, end)/2+gd/h;
u=A\f(:);
u=reshape(u, ny, nx);
```

Nous pouvons maintenant résoudre un problème encore plus réaliste de thermique dans une pièce, avec un mur isolant à l'est, modélisé par une condition de Robin. Avec le petit script Matlab suivant nommé RoomR, nous obtenons la distribution de température représentée figure **22**.

```
eta=0;
J=20;                          % number of interior mesh points in x and y direction
x=0:1/(J+1):1; y=x;            % finite difference mesh, including boundary
f=zeros(J, J+2);               % source term includes Robin boundary
xi=x(2:end-1); yi=xi;
f([yi > 0.4 & yi < 0.6], [xi > 0.4 & xi < 0.6])=50;
gd=zeros(J, 1);
gg=0.3*ones(J, 1);
gg(yi > 0.5 & yi < 0.9)=1;
pe=1e5;                        % 1e5 to emulate a Dirichlet condition
pin=1;                         % 1 to emulate insulation
u=Solve2dR(f, eta, 0, J+1, gg*pe, gd, pe, pin);
u=[zeros(1, J+2); u; zeros(1, J+2)];
mesh(x(1:end), y, u); xlabel('x'); ylabel('y'); zlabel('solution');
```

Remarquons dans ce code que la condition de Dirichlet à l'ouest est réalisée avec une condition de Robin de paramètre 10^5.

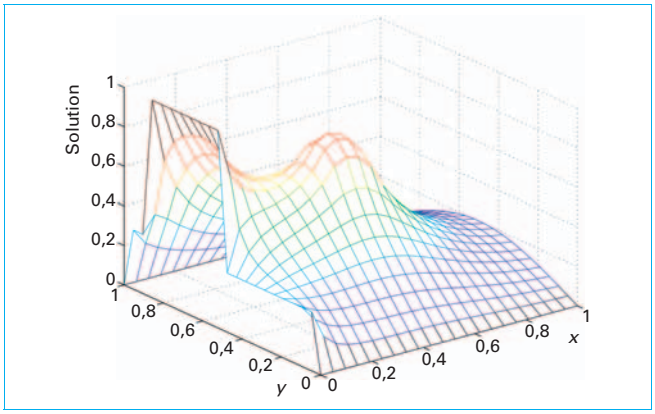


Figure 22 – Exemple de distribution de température avec une condition de Robin à l'est, représentant un mur isolé

La température sur le mur n'est plus nulle comme dans l'exemple précédent figure **14**, mais varie en fonction de la température intérieure.

La difficulté supplémentaire dans l'algorithme de Schwarz optimisé réside dans le choix du paramètre p. Il est calculé de la manière suivante. Supposons pour simplifier que la taille des sous-domaines est infinie dans la direction x. Nous pouvons reprendre l'analyse développée dans le théorème 2.3. Elle est un peu plus simple, puisque nous pouvons écrire

$$e_1^n(x,y) = \sum_k \hat{e}_1^n(k) e^{\sqrt{\eta + k^2\pi^2}\,x} \sin k\pi y,$$
$$e_2^n(x,y) = \sum_k \hat{e}_2^n(k) e^{-\sqrt{\eta + k^2\pi^2}\,x} \sin k\pi y.$$

Les coefficients $\hat{e}_i^n(k)$ sont maintenant donné par $\hat{e}_i^n(k) = \rho^{(\mathfrak{R})}\hat{e}_i^{n-2}(k)$, avec

$$\rho^{(\mathfrak{R})}(k,p) = \rho^{(\mathfrak{D})}(k)\left(\frac{p - \sqrt{\eta + k^2\pi^2}}{p + \sqrt{\eta + k^2\pi^2}}\right)^2, \quad \rho^{(\mathfrak{D})}(k) = e^{-2\sqrt{\eta + k^2\pi^2}\,\delta}.$$

Ici $\rho^{(\mathfrak{D})}$ est le facteur de convergence de l'algorithme classique, il mesure l'importance du recouvrement. Le facteur $\rho^{(\mathfrak{R})}$ peut être minimisé en choisissant judicieusement le paramètre p, voir [15], c'est-à-dire en résolvant le problème de min-max

$$\min_{p>0} \max_{1 \leqslant k \leqslant J} \left|\frac{p - \sqrt{\eta + k^2\pi^2}}{p + \sqrt{\eta + k^2\pi^2}}\right|^2 e^{-2\sqrt{\eta + k^2\pi^2}\,\delta}.$$

Une étude approfondie de ce problème par des procédés classiques d'analyse (voir [15] montre qu'il a une solution p^*, qui est donnée par les formules asymptotiques :

– **Cas avec recouvrement**. Si $\delta > 0$ et J grand, le paramètre p^* optimisé et le facteur de convergence correspondant ont un comportement asymptotique donné par

$$p^* \sim \left(\frac{\pi^2 + \eta}{2\delta}\right)^{\frac{1}{3}}, \quad \max_{1 \leqslant k \leqslant J} \left|\rho^{(\mathfrak{R})}(k, p^*)\right| \sim 1 - 4\left(2\delta\sqrt{\pi^2 + \eta}\right)^{\frac{1}{3}} = 1 - \mathcal{O}\left(\delta^{1/3}\right).$$

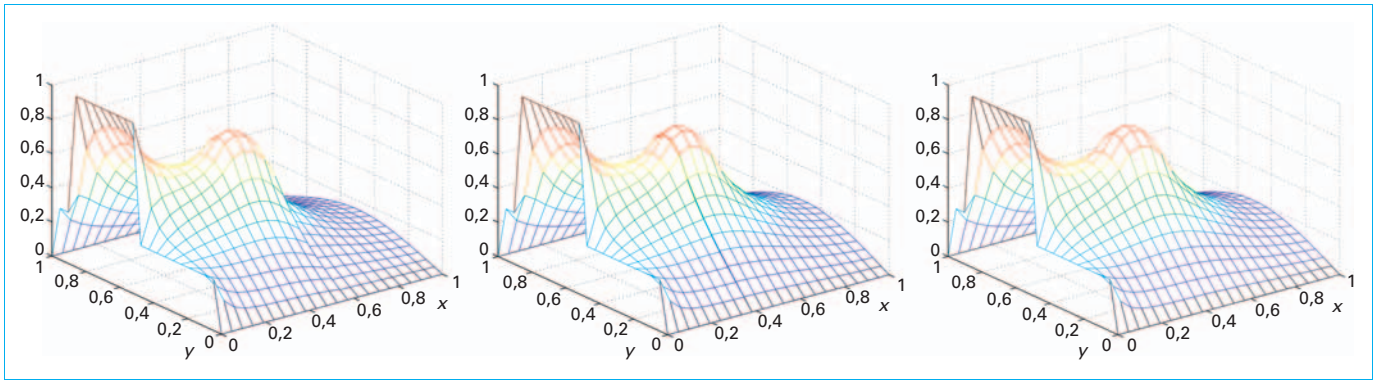


Figure 23 – Résolution du problème de chauffage de la figure 22 par la méthode de Schwarz optimisée

Ces formules impliquent que si le recouvrement dépend de h, par exemple $\delta = \mathcal{O}(h)$, le facteur de convergence est asymptomatiquement en $1-\mathcal{O}\left(h^{1/3}\right)$;

– **Cas sans recouvrement**. Si $\delta = 0$, le résultat de l'optimisation est donné par

$$p^* = \left(\left(\pi^2+\eta\right)\left(\pi^2 J^2+\eta\right)\right)^{1/4},$$

$$\max_{1\leqslant k\leqslant J}\left|\rho^{(\Re)}\left(k, p^*\right)\right| = \left(\frac{\sqrt[4]{\pi^2J^2+\eta}-\sqrt[4]{\pi^2+\eta}}{\sqrt[4]{\pi^2J^2+\eta}+\sqrt[4]{\pi^2+\eta}}\right)^2 = 1-\mathcal{O}\left(J^{-1/2}\right).$$

Puisque $(J+1)h = 1$, nous avons maintenant un taux de convergence égal à $1-\mathcal{O}\left(h^{1/2}\right)$.

Ces résultats doivent être mis en regard du facteur de convergence pour l'algorithme de Schwarz classique qui est $1-\mathcal{O}(h)$: la vitesse de convergence dépend plus faiblement du pas du maillage, et donc de la taille de la matrice. Voici une implémentation en Matlab de cette méthode de Schwarz optimisée.

```
RoomR ;
ue=u ;
α=8 ; d=4 ;
f1=f(:, 1 : α+d+1) ; f2=f(:, α+1 : end) ;
u1=zeros(J, α+d+1) ; u2=zeros(J, J-α+2) ;          %  zero initial guess
h=1/(J+1) ;
x1=0 : h : (α+d)*h ;   x2=α*h : h : 1 ;   y=0 : h : 1 ;   %  finite difference meshes
z1=zeros(1, α+d+1) ; z2=zeros(1, J-α+2) ;          %  for plotting purposes
p=((pi^2+eta)/(d*h))^(1/3) ;
e=ones(J, 1) ;                                     %  construct normal derivatives
Nα=[speye(J)-spdiags([-e(eta*h^2+4)*e-e]/2, [-1  0  1], J, J)]/h ;
Nb=[-spdiags([-e(eta*h^2+4)*e-e]/2, [-1  0  1], J, J) speye(J)]/h ;
for i=1 : 20
   tb=Nb*[u2(:, d+1) ; u2(:, d+2)]+f2(:, d+1)*h/2+p*u2(:,d+1) ;
   tα=Nα*[u1(:, end-d-1) ; u1(:, end-d)]+f1(:, end-d)*h/2+p*u1(:, end-d) ;
   u1n=Solve2dR(f1, eta, 0, α+d, pe*gg, tb, pe, p) ;
   u2n=Solve2dR(f2, eta, α, J+1, tα, gd, p, pin) ;
   u1=u1n ;
   u2=u2n ;
   mesh(x1, y, [z1 ; u1n ; z1]) ; hold on ; mesh(x2, y, [z2 ; u2n ; z2]) ; hold off
   xlabel('x') ; ylabel('y') ; zlabel('Optimized Schwarz iterates') ;
   pause
end
```

Pour le cas test décrit sur la figure **22**, nous représentons sur la figure **23** les trois premières itérations de cette méthode avec une valeur de p donnée par la formule asymptotique précédente, soit p = 3,7281. On voit bien que la convergence est beaucoup plus rapide que pour les méthodes de Schwarz classiques illustrées figure **15** : la première itération est déjà une excellente approximation de la solution et la troisième itération est indiscernable à l'œil nu de la solution « convergée ».

Comme pour les méthodes de Schwarz précédentes, nous allons mettre l'algorithme sous forme d'un algorithme de Jacobi par blocs pour le problème d'interface portant sur $g_2 = (\partial_x + p)(u_2)(\beta,\cdot)$ et $g_1 = (\partial_x - p)(u_1)(\alpha,\cdot)$. Introduisons les problèmes

$$\begin{aligned}
&\eta\, u_1 - \Delta u_1 = f \text{ dans } \Omega_1,\\
&\eta\, u_2 - \Delta u_2 = f \text{ dans } \Omega_2,\\
&u_1(0,\cdot) = g_g\,,\ (\partial_x u_1 + p u_1)(\beta,\cdot) = g_2,\\
&(\partial_x u_2 - p u_2)(\alpha,\cdot) = g_1\,,\ u_2(1,\cdot) = g_d.
\end{aligned} \qquad \textbf{(47)}$$

Pour $p \geqslant 0$, ces deux problèmes ont une solution unique, que nous appellerons $\mathcal{U}_1^{\Re}(g_2, g_g, f)$ et $\mathcal{U}_2^{\Re}(g_1, g_d, f)$. Supposons $g_1^n = (\partial_x - p)\left(u_1^n\right)(\alpha,\cdot)$ et $g_2^n = (\partial_x + p)\left(u_2^n\right)(\beta,\cdot)$ calculés à l'étape n. Alors une étape de l'algorithme de Schwarz optimisé s'écrit

$$g_1^{n+1} = \left((\partial_x - p)\mathcal{U}_1^{\Re}\left(g_2^n, g_d, f\right)\right)(\alpha,\cdot),\quad g_2^{n+1} = \left((\partial_x - p)\mathcal{U}_2^{\Re}\left(g_1^n, g_g, f\right)\right)(\beta,\cdot).$$

Il nous est maintenant facile d'écrire le problème d'interface sous forme de système linéaire

$$\begin{pmatrix} I & -(\partial_x - p)\mathcal{U}_1^{\Re}(\cdot, 0, 0)(\alpha) \\ -(\partial_x + p)\mathcal{U}_2^{\Re}(\cdot, 0, 0)(\beta) & I \end{pmatrix}\begin{pmatrix} g_1 \\ g_2 \end{pmatrix} = \begin{pmatrix} (\partial_x - p)\mathcal{U}_1^{\Re}(0, g_g, f)(\alpha) \\ (\partial_x + p)\mathcal{U}_2^{\Re}(0, g_d, f)(\beta) \end{pmatrix}, \qquad \textbf{(48)}$$

et de le traiter par l'algorithme GMRES. Sur la figure **24**, nous traçons l'historique de convergence de l'algorithme sous-structuré itératif d'une part, accéléré par la méthode GMRES (donc où on résout le système sous-structuré (48) par GMRES), et les résultats correspondants pour la méthode de Schwarz classique de la figure **17**.

En comparant ces courbes de convergence, nous constatons deux choses : la méthode de Schwarz optimisée converge maintenant plus vite que la méthode de Schwarz classique accélérée par GMRES. Si l'on accélère maintenant la méthode de Schwarz optimisée par GMRES, on gagne encore en nombre d'itérations, mais le gain est moins important que pour la méthode de Schwarz classique : la méthode de Schwarz optimisée est déjà un très bon solveur itératif, même sans accélération par Krylov.

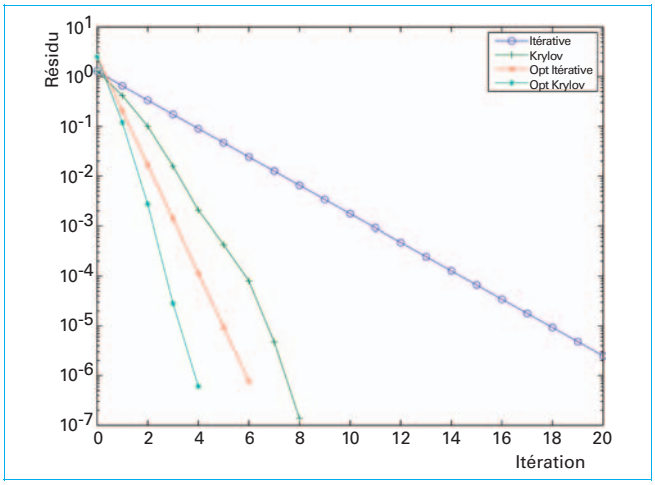


Figure 24 – Comparaison de Schwarz optimisé comme algorithme itératif et comme préconditionneur, et pour référence les résultats correspondants de la méthode de Schwarz classique. Historique de convergence

3. Méthodes de Schur

Historiquement les méthodes de Schur ont souvent été appelées méthodes de sous-structuration. Néanmoins nous avons vu que les méthodes de Schwarz peuvent aussi s'écrire sous forme sous-structurée, c'est pourquoi nous préférons utiliser la terminologie de **méthodes de Schur**. Il en existe deux variantes principales, la méthode de Schur primal et la méthode de Schur dual. Nous verrons que les deux méthodes sont intimement liées, et chacune constitue un excellent préconditionneur pour l'autre.

3.1 Méthode de Schur primal

Suivons la démarche de Przemieniecki esquissée au paragraphe 1.2, avec les notations modernes. Reprenons le système linéaire (19) en dimension 1, $A\boldsymbol{u} = \boldsymbol{f}$, en écrivant d'abord les inconnues de $\Omega_1 =]0, \alpha[$, $\boldsymbol{u}_1 = (u_1 \ldots . u_{a-1})^T$, puis celles de $\Gamma = \{\alpha\}$, $u_\Gamma = u_a$, puis celles de $\Omega_2 =]\alpha, 1[$, $\boldsymbol{u}_2 = (u_{a+1} \ldots . u_J)^T$. Ces trois blocs constituent le vecteur

$$\boldsymbol{u} = \begin{pmatrix} \boldsymbol{u}_1 \\ u_\Gamma \\ \boldsymbol{u}_2 \end{pmatrix}.$$

Le second membre $\boldsymbol{f}$ est décomposé de la même façon, et la matrice A est décomposée par blocs en

$$A = \left(\begin{array}{ccc|c|ccc} \frac{2}{h^2}+\eta & -\frac{1}{h^2} & & & & & \\ -\frac{1}{h^2} & \frac{2}{h^2}+\eta & \ddots & & & & \\ & \ddots & \ddots & \frac{1}{h^2} & & & \\ & & -\frac{1}{h^2} & \frac{2}{h^2}+\eta & -\frac{1}{h^2} & & \\ \hline & & & -\frac{1}{h^2} & \frac{2}{h^2}+\eta & -\frac{1}{h^2} & \\ \hline & & & & -\frac{1}{h^2} & \frac{2}{h^2}+\eta & -\frac{1}{h^2} \\ & & & & & -\frac{1}{h^2} & \frac{2}{h^2}+\eta \end{array}\right).$$

soit sous forme compacte

$$A = \begin{pmatrix} A_{11} & A_{1\Gamma} & 0 \\ A_{\Gamma 1} & A_{\Gamma\Gamma} & A_{\Gamma 2} \\ 0 & A_{2\Gamma} & A_{22} \end{pmatrix}.$$

Les matrices A_{11} et A_{22} ont la même forme que A mais sont de taille respective $a-1$ et $J-a$, elles sont inversibles. $A_{\Gamma\Gamma}$ représente les échanges entre les points de Γ. En dimension 1 c'est une matrice scalaire, ce sera une matrice bande en dimension 2. Les matrices $A_{i\Gamma}$ représentent le couplage entre les points intérieurs au domaine Ω_i et la frontière commune $\{\alpha\}$, et $A_{\Gamma i} = A_{i\Gamma}^T$ puisque la matrice A est symétrique. En dimension 1 ce sont des vecteurs. Les blocs concernant Γ sont donnés par

$$A_{\Gamma 1} = -\frac{1}{h^2}(0 \ldots . 0, 1), \quad A_{\Gamma 2} = -\frac{1}{h^2}(1, 0 \ldots . 0), \quad A_{\Gamma\Gamma} = \frac{2}{h^2} + \eta.$$

Le problème se réécrit par blocs

$$\begin{aligned} A_{11}\boldsymbol{u}_1 + A_{1\Gamma}u_\Gamma &= \boldsymbol{f}_1, \\ A_{22}\boldsymbol{u}_2 + A_{2\Gamma}u_\Gamma &= \boldsymbol{f}_2, \\ A_{\Gamma 1}\boldsymbol{u}_1 + A_{\Gamma 2}\boldsymbol{u}_2 + A_{\Gamma\Gamma}u_\Gamma &= \boldsymbol{f}_\Gamma. \end{aligned} \tag{49}$$

Nous pouvons extraire $\boldsymbol{u}_1$ et $\boldsymbol{u}_2$ des deux premières équations,

$$\boldsymbol{u}_1 = A_{11}^{-1}(\boldsymbol{f}_1 - A_{1\Gamma}u_\Gamma), \quad \boldsymbol{u}_2 = A_{22}^{-1}(\boldsymbol{f}_2 - A_{2\Gamma}u_\Gamma), \tag{50}$$

et les reporter dans la troisième. Nous obtenons alors une équation (scalaire en dimension 1) portant sur u_Γ,

$$\left(A_{\Gamma\Gamma} - A_{\Gamma 1}A_{11}^{-1}A_{1\Gamma} - A_{\Gamma 2}A_{22}^{-1}A_{2\Gamma}\right)u_\Gamma = f_\Gamma - A_{\Gamma 1}A_{11}^{-1}\boldsymbol{f}_1 - A_{\Gamma 2}A_{22}^{-1}\boldsymbol{f}_2 := \tilde{f}_\Gamma. \tag{51}$$

Nous reconnaissons le système (8) de Przemienecki, nous l'appellerons formulation *Schur primal*. La matrice $S_P = A_{\Gamma\Gamma} - A_{\Gamma 1}A_{11}^{-1}A_{1\Gamma} - A_{\Gamma 2}A_{22}^{-1}A_{2\Gamma}$ est la **matrice du complément de Schur** introduite dans l'historique. Puisque le système (49) a une solution unique, la matrice S_P est inversible. En contraste avec l'algorithme de Schwarz, la méthode de Schur introduit un nouveau système, plus petit (puisque sa taille est égale au nombre de nœuds sur l'interface), mais ne fournit pas d'algorithme pour le résoudre. Le système (51) peut être formé et résolu par une méthode directe, ce qui était historiquement souvent le cas. Les matrices A_{ii} admettent chacune une décomposition $L_{ii}U_{ii}$, ces calculs peuvent être faits en parallèle et sont regroupés pour calculer la matrice S_P. Ce système peut aussi être résolu par une méthode itérative, de préférence une méthode de Krylov. Dans ce cas, on ne forme pas la matrice S_P, puisqu'une itération revient à un produit matrice vecteur, qui correspond à une résolution par sous-domaine.

La méthode de Schur est par définition une méthode discrète, mais pour avoir une vision plus globale, nous allons en déduire une formulation continue. Pour cela, explicitons la troisième équation de (49) :

$$-\frac{1}{h^2}(u_1)_{a-1} - \left(\frac{2}{h^2} + \eta\right)u_\Gamma - \frac{1}{h^2}(u_2)_1 = f_\Gamma. \tag{52}$$

Étendons le vecteur $\boldsymbol{u}_1$ par $(u_1)_a = u_\Gamma$ et $\boldsymbol{u}_2$ par $(u_2)_0 = u_\Gamma$. Lorsque le pas h tend vers 0, $\boldsymbol{u}_1$ tend vers u_1, solution de l'équation continue (14) dans Ω_1, avec $u_1(0) = g_g$ et $u_1(\alpha) = u_\Gamma$, et de même pour $\boldsymbol{u}_2$. L'équation précédente se réécrit

$$-\frac{1}{h^2}u_1(\alpha - h) + \left(\frac{2}{h^2} + \eta\right)u_\Gamma - \frac{1}{h^2}u_2(\alpha + h) = f(\alpha) + \mathbb{O}(h^2),$$

ou encore

$$\frac{1}{h^2}(u_1(\alpha) - u_1(\alpha - h)) - \frac{1}{h^2}(u_2(\alpha + h) - u_2(\alpha)) + \eta u = f(\alpha) + \mathbb{O}(h^2). \tag{53}$$

Calculons par développements limités

$$u_1(\alpha-h)=u_1(\alpha)-h\frac{\mathrm{d}u_1}{\mathrm{d}x}(\alpha)+\frac{h^2}{2}\frac{\mathrm{d}^2u_1}{\mathrm{d}x^2}(\alpha)+o\left(h^2\right),$$

et utilisons l'équation différentielle sur u_1 pour obtenir

$$u_1(\alpha-h)=u_1(\alpha)-h\frac{\mathrm{d}u_1}{\mathrm{d}x}(\alpha)+\frac{h^2}{2}\left(\eta u_1(\alpha)-f(\alpha)\right)+o\left(h^2\right),$$

ce qui fournit l'approximation

$$\frac{\mathrm{d}u_1}{\mathrm{d}x}(\alpha)=\frac{u_1(\alpha)-u_1(\alpha-h)}{h}+\frac{h}{2}\left(\eta u_1(\alpha)-f(\alpha)\right)+o(h). \tag{54}$$

De même

$$\frac{\mathrm{d}u_2}{\mathrm{d}x}(\alpha)=\frac{u_2(\alpha+h)-u_2(\alpha)}{h}-\frac{h}{2}\left(\eta u_2(\alpha)-f(\alpha)\right)+o(h). \tag{55}$$

Ces deux relations nous indiquent que

$$\frac{u_1(\alpha)-u_1(\alpha-h)}{h^2}-\frac{u_2(\alpha+h)-u_2(\alpha)}{h^2}+\eta u_\Gamma-f(\alpha)$$
$$=\frac{1}{h}\left(\frac{\mathrm{d}u_1}{\mathrm{d}x}(\alpha)-\frac{\mathrm{d}u_2}{\mathrm{d}x}(\alpha)\right)+o(1).$$

Ainsi (52) est la discrétisation de la condition de couplage

$$\frac{\mathrm{d}u_2}{\mathrm{d}x}(\alpha)-\frac{\mathrm{d}u_1}{\mathrm{d}x}(\alpha)=0.$$

Par conséquent, le système (49) est une version discrète de la formulation continue

$$\begin{aligned}-\frac{\mathrm{d}^2u_1}{\mathrm{d}x^2}+\eta u_1&=f \text{ dans } \Omega_1,\\ -\frac{\mathrm{d}^2u_2}{\mathrm{d}x^2}+\eta u_2&=f \text{ dans } \Omega_2,\\ u_1(\alpha)=u_2(\alpha),&\quad \frac{\mathrm{d}u_1}{\mathrm{d}x}(\alpha)=\frac{\mathrm{d}u_2}{\mathrm{d}x}(\alpha).\end{aligned} \tag{56}$$

C'est un système de deux équations différentielles dans les sous-domaines, couplées sur la frontière commune par l'égalité des fonctions et de leurs dérivées. Nous allons en déduire la formulation *Schur primal continue*, en appliquant sur le problème continu la démarche précédente. Pour cela nous définissons l'opérateur de Dirichlet-Neumann ou de Steklov-Poincaré comme suit. Reprenons les notations du paragraphe 2.4 en dimension 1, avec $\delta = 0$. Pour une donnée g, nous calculons la solution du problème aux limites dans Ω_i avec donnée g en $x = \alpha$ notée u_i, puis nous définissons les opérateurs $S_i^{\mathcal{D},\mathcal{N}}$ par

$$S_1^{\mathcal{D},\mathcal{N}}\left(g,g_g,f\right)=\frac{\mathrm{d}u_1}{\mathrm{d}x}(\alpha),\quad S_2^{\mathcal{D},\mathcal{N}}\left(g,g_d,f\right)=\frac{\mathrm{d}u_2}{\mathrm{d}x}(\alpha). \tag{57}$$

Les opérateurs $S_i^{\mathcal{D},\mathcal{N}}$ sont affines en la variable g, $S_1^{\mathcal{D},\mathcal{N}}\left(g,g_g,f\right)=S_1^{\mathcal{D},\mathcal{N}}(g,0)+S_1^{\mathcal{D},\mathcal{N}}\left(0,g_g,f\right)$ et de même pour $S_2^{\mathcal{D},\mathcal{N}}$, si bien que l'équation de continuité des dérivées à l'interface dans (56) se réécrit, en posant $g = u_1(\alpha) = u_2(\alpha)$,

$$S_P\,g:=S_1^{\mathcal{D},\mathcal{N}}(g,0,0)-S_2^{\mathcal{D},\mathcal{N}}(g,0,0)=-S_1^{\mathcal{D},\mathcal{N}}\left(0,g_g,f\right)+S_2^{\mathcal{D},\mathcal{N}}(0,g_d,f) \tag{58}$$

Le système linéaire (58) peut être résolu soit par méthode directe, soit par méthode itérative. En dimension 1, cela ne représente qu'une équation scalaire, mais en dimension supérieure, c'est un système qui couple toutes les inconnues à l'interface. Avant d'en expliciter une méthode de résolution, nous allons calculer le conditionnement du système en dimension 2. Pour cela développons g en séries de sinus dans la variable y :

$$g(y)=\sum_{k=1}^{+\infty}\hat{g}(k)\sin k\pi y.$$

Les images $S_i^{\mathcal{D},\mathcal{N}}(g,0,0)$ se décomposent de la même façon et les coefficients sont donnés par :

$$\begin{aligned}\widehat{S_1^{\mathcal{D},\mathcal{N}}(g,0,0)}(k)&=\sqrt{\eta+k^2\pi^2}\,\coth\left(\sqrt{\eta+k^2\pi^2}\alpha\right)\hat{g}(k),\\ \widehat{S_2^{\mathcal{D},\mathcal{N}}(g,0,0)}(k)&=-\sqrt{\eta+k^2\pi^2}\,\coth\left(\sqrt{\eta+k^2\pi^2}\,(1-\alpha)\right)\hat{g}(k).\end{aligned} \tag{59}$$

Le symbole de l'opérateur S_P (défini par $\widehat{S_Pg}(k)=\widehat{S_P}(k)\hat{g}(k)$) est donc

$$\widehat{S_P}(k)=\sqrt{\eta+k^2\pi^2}\left(\coth\left(\sqrt{\eta+k^2\pi^2}\alpha\right)+\coth\left(\sqrt{\eta+k^2\pi^2}\,(1-\alpha)\right)\right).$$

La fonction $\widehat{S_P}$ est croissante. Le conditionnement de $\widehat{S_P}$ discrétisé sur la grille de taille J peut donc être estimé par

$$\mathcal{K}\left(\widehat{S_P}\right):=\frac{\max\limits_{1\leqslant k\leqslant J}\left(\widehat{S_P}(k)\right)}{\min\limits_{1\leqslant k\leqslant J}\left(\widehat{S_P}(k)\right)}=\frac{\widehat{S_P}(J)}{\widehat{S_P}(1)},$$

et son comportement asymptotique en fonction de h est facile à calculer. Il est donné par

$$\mathcal{K}\left(\widehat{S_P}\right)=\frac{2\sqrt{\eta+J^2\pi^2}}{\sqrt{\eta+\pi^2}\left(\coth\left(\sqrt{\eta+\pi^2}\alpha\right)+\coth\left(\sqrt{\eta+\pi^2}\,(1-\alpha)\right)\right)}\left(1+\mathbb{O}\left(e^{-\sqrt{\eta+J^2\pi^2}\alpha}\right)\right)$$
$$=\mathbb{O}(J)=\mathbb{O}\left(h^{-1}\right)\quad\text{puisque}\quad(J+1)h=1.$$

Rappelons que le conditionnement du problème de résolution du système linéaire de matrice (19) est $\mathbb{O}\left(h^{-2}\right)$: la formulation sous forme de problème d'interface améliore donc le conditionnement d'un ordre de grandeur. Nous verrons plus tard sur la figure **26**

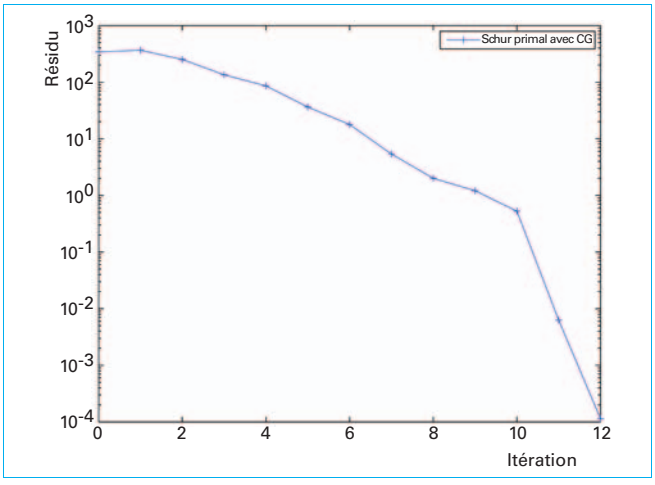


Figure 25 – Convergence de la méthode de Schur primal pour le problème modèle bidimensionnel de la figure 14

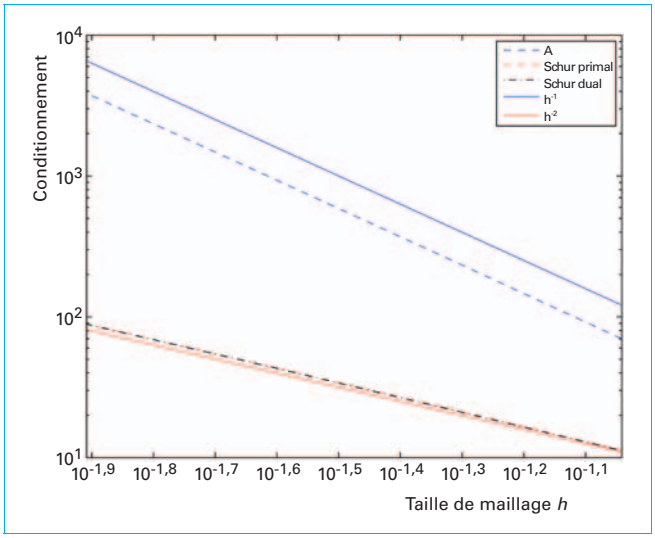


Figure 26 – Comparaison en dimension 2 des conditionnements du problème de départ et des méthodes de Schur primal et Schur dual

une comparaison numérique des conditionnements sur notre problème modèle de la figure **14**.

Voici maintenant un script Matlab de résolution de ce problème modèle par la méthode du complément de Schur primal. Le problème d'interface est résolu par la méthode du gradient conjugué, car il est symétrique pour une matrice de départ symétrique. La courbe de convergence est représentée figure **25**.

```
Room ;
α=10 ;                                  % decomposition
h=1/(J+1) ;
f=f(:) ;
f(1:J)=f(1:J)+gg/h^2 ;                  % add boundary conditions into rhs
f(end-J+1:end)=f(end-J+1:end)+gd/h^2 ;
A=A2d(eta, J, J) ;

A11=A(1:J*(α-1), 1:J*(α-1)) ;           % form block decomposition
AGG=A(J*(α-1)+1:J*α, J*(α-1)+1:J*α) ;
A22=A(J*α+1:J*J, J*α+1:J*J) ;
A1G=A(1:J*(α-1), J*(α-1)+1:J*α) ;
AG1=A(J*(α-1)+1:J*α, 1:J*(α-1)) ;
A2G=A(J*α+1:J*J, J*(α-1)+1:J*α) ;
AG2=A(J*(α-1)+1:J*α, J*α+1:J*J) ;

f1=f(1:J*(α-1)) ;
fG=f(J*(α-1)+1:J*α) ;
f2=f(J*α+1:J*J) ;

Afun=@(x)  AGG*x-AG1*(A11\(A1G*x))-AG2*(A22\(A2G*x)) ;
ft=fG-AG1*(A11\f1)-AG2*(A22\f2) ;
[uG, fl, r, it, rcg]=pcg(Afun, ft, 1e-6, 100) ;
u1=A11\(f1-A1G*uG) ; u2=A22\(f2-A2G*uG) ;
u=[zeros(1, J+2) ; gg reshape([u1 ; uG ; u2]J, J) gd ; zeros(1, J+2) ;]

mesh(0:h:1, 0:h:1, u) ; xlabel('x') ; ylabel('y')
pause

semilogy(0:length)(rcg)-1, rcg,"-+' ;
xlabel('iteration') ; ylabel('residual') ; legend('Primal Schur with CG')
```

Considérons de nouveau le système (56). Dans la méthode précédente, on considère que u_1 et u_2 ont la même valeur g sur l'interface et on calcule le saut des dérivées en ce même point en fonction de g. Annuler cette quantité fournit l'équation sur g à résoudre.

On peut imaginer de faire l'inverse, c'est ce que l'on appelle la **méthode de Schur dual** que nous présentons maintenant.

3.2 Méthode de Schur dual

Donnons-nous un réel g et calculons les analogues de (41) à ceci près que les conditions sur l'interface sont maintenant des conditions de Neumann :

$$\begin{aligned} &-\frac{d^2u_1}{dx^2}+\eta u_1 = f \text{ dans } \Omega_1, && -\frac{d^2u_2}{dx^2}+\eta u_2 = f \text{ dans } \Omega_2, \\ &u_1(0)=g_g,\ \frac{du_1}{dx}(\alpha)=g, && \frac{du_2}{dx}(\alpha)=g,\ u_2(1)=g_d. \end{aligned} \quad \textbf{(60)}$$

Les équations précédentes définissent une solution unique dans chaque sous-domaine, elles sont notées $u_1 = \mathcal{U}_1^{\mathcal{N}}(g, g_g, f)$ et $u_2 = \mathcal{U}_2^{\mathcal{N}}(g, g_d, f)$.

Remarque 3.1 Lorsque $\eta = 0$, l'existence et l'unicité dans les sous-domaines proviennent de la condition de Dirichlet sur un bord, qui permet d'utiliser l'inégalité de Poincaré. Dans le cas de trois sous-domaines, le problème du milieu est un problème de Neumann « pur », qui réclame une relation de compatibilité pour en assurer l'existence. L'unicité est alors obtenue modulo les constantes, ce qui fut au début considéré comme un défaut de la méthode. Une étape astucieuse dans le cadre des méthodes de FETI fut par la suite d'utiliser ces constantes libres dans chaque sous-domaine pour former une composante grille grossière naturelle, ce qui contribua grandement à la célébrité des méthodes FETI, que nous exposerons au paragraphe 3.3.

Nous allons déduire maintenant la **formulation Schur dual continue**. Pour cela définissons les opérateurs de Neumann-Dirichlet.

$$S_1^{\mathcal{N}\mathfrak{D}}(g, g_d, f) = \mathcal{U}_1^{\mathcal{N}}(g, g_g, f)(\alpha) \quad S_2^{\mathcal{N}\mathfrak{D}}(g, g_d, f) = \mathcal{U}_2^{\mathcal{N}}(g, g_d, f)(\alpha) \quad \textbf{(61)}$$

L'équation $u_1(\alpha) = u_2(\alpha)$ prend la forme d'une équation linéaire portant sur la variable d'interface g :

$$S_{\mathfrak{D}} g := S_1^{\mathcal{N}\mathfrak{D}}(g, 0, 0) - S_2^{\mathcal{N}\mathfrak{D}}(g, 0, 0) = S_1^{\mathcal{N}\mathfrak{D}}(0, g_g, f) + S_2^{\mathcal{N}\mathfrak{D}}(0, g_d, f). \quad \textbf{(62)}$$

Étudions maintenant ce système en dimension 2. Il est symétrique et sa résolution par la méthode du gradient conjugué requiert à chaque étape la résolution de deux problèmes avec conditions aux limites de Neumann. Développons g en série de sinus dans la variable y comme précédemment. Les images $S_i^{\mathfrak{D}\mathcal{N}}$ se décomposent de la même façon, avec

$$\begin{aligned} \widehat{S_1^{\mathcal{N}\mathfrak{D}}(g, 0, 0)}(k) &= \frac{\operatorname{th}\left(\sqrt{\eta + k^2\pi^2}\,\alpha\right)}{\sqrt{\eta + k^2\pi^2}}\hat{g}(k), \\ \widehat{S_2^{\mathcal{N}\mathfrak{D}}(g, 0, 0)}(k) &= -\frac{\operatorname{th}\left(\sqrt{\eta + k^2\pi^2}\,(1-\alpha)\right)}{\sqrt{\eta + k^2\pi^2}}\hat{g}(k). \end{aligned} \quad \textbf{(63)}$$

Le symbole de l'opérateur $S_{\mathfrak{D}}$ est donc

$$\widehat{S_{\mathfrak{D}}}(k) = \frac{1}{\sqrt{\eta + k^2\pi^2}}\left(\operatorname{th}\left(\sqrt{\eta + k^2\pi^2}\,\alpha\right) + \operatorname{th}\left(\sqrt{\eta + k^2\pi^2}\,(1-\alpha)\right)\right).$$

Cette fois la fonction $\widehat{S_{\mathfrak{D}}}$ est décroissante. Le conditionnement de $\widehat{S_{\mathfrak{D}}}$ sur la grille de taille J peut être estimé par

$$\begin{aligned} \mathcal{K}\left(\widehat{S_{\mathfrak{D}}}\right) &= \frac{\widehat{S_{\mathfrak{D}}}(1)}{\widehat{S_{\mathfrak{D}}}(J)} = \frac{\operatorname{th}\left(\sqrt{\eta+\pi^2}\,\alpha\right) + \operatorname{th}\left(\sqrt{\eta+\pi^2}\,(1-\alpha)\right)}{2\sqrt{\eta+\pi^2}}\sqrt{\eta + J^2\pi^2}\left(1 + \mathcal{O}\left(e^{-\sqrt{\eta+J^2\pi^2}\,\alpha}\right)\right) \\ &= \mathcal{O}(J) = \mathcal{O}\left(h^{-1}\right). \end{aligned}$$

Le conditionnement est du même ordre que celui de Schur primal. Voici un exemple en Matlab qui permet de comparer en

dimension 2 le conditionnement de la matrice A aux conditionnement des matrices des méthodes de Schur primal et Schur dual.

```
for j=1:4
    J=10*2^(j-1); α=5*2^(j-1);
    eta=0;
    h(j)=1/(J+1);
    A=A2d(eta,J,J);

    A11=A(1:J*(α-1),1:J*(α-1));          % form block decomposition
    AGG=A(J*(α-1)+1:J*α,J*(α-1)+1:J*α);
    A22=A(J*α+1:J*J,J*α+1:J*J);
    A1G=A(1:J*(α-1),J*(α-1)+1:J*α);
    AG1=A(J*(α-1)+1:J*α,1:J*(α-1));
    A2G=A(J*α+1:J*J,J*(α-1)+1:J*α);
    AG2=A(J*(α-1)+1:J*α,J*α+1:J*J);

    A1=[A11  A1G
        AG1  AGG/2];
    A2=[AGG/2 AG2
        A2g  A22];

    R1=[zeros(size(AG1))    speye(size(AGG))];
    R2=[speye(size(AGG))  zeros(size(AG2))];

    Acond(j)=condest(A);
    SPcond(j)=condest(AGG-AG1*(A11\A1G)-AG2*(A22\A2G));
    SDcond(j)=condest(R1*(A1\R1')+R2*(A2\R2'));
end;
loglog(h,Acond,'--b',h,SPcond,'--r',h,SDcond,'-.k',h,1./h.^2,'-b',h,1./h,'-r');
xlabel('mesh size h')
ylabel('condition number')
legend('A','Primal Schur','Dual Schur','h^{-1}','h^{-2}')
```

Les résultats sont portés sur la figure **26** où nous traçons les conditionnements de la matrice A, des matrices $S_{\mathcal{D}}$ et $S_{\mathcal{R}}$, en fonction du pas du maillage h. Les résultats sont en accord avec les estimations théoriques.

Écrivons maintenant en dimension 1 la formulation discrète associée à la méthode de Schur dual, avec les notations du paragraphe 3.1. Pour cela nous écrivons l'équation habituelle en tous les points intérieurs et nous ajoutons sur l'interface commune une relation obtenue au moyen de (54) et (55) :

$$g=\frac{(u_1)_a-(u_1)_{a-1}}{h}+\frac{h}{2}\left(\eta(u_1)_a-f_a\right)=\frac{(u_2)_{a+1}-(u_2)_a}{h}-\frac{h}{2}\left(\eta(u_2)_a-f_a\right),$$

ce qui donne un système global

$$\begin{aligned}
&-\frac{(u_1)_{j+1}-2(u_1)_j+(u_1)_{j-1}}{h^2}+\eta(u_1)_j=(f_1)_j,\quad 1\leqslant j\leqslant a-1,\\
&-\frac{1}{h^2}(u_1)_{a-1}+\frac{1}{2}\left(\eta+\frac{2}{h^2}\right)(u_1)_a=\frac{1}{h}g+\frac{1}{2}f_a,\\
&-\frac{(u_2)_{j+1}-2(u_2)_j+(u_2)_{j-1}}{h^2}+\eta(u_2)_j=(f_2)_j,\quad a+1\leqslant j\leqslant J,\\
&-\frac{1}{h^2}(u_2)_{a+1}+\frac{1}{2}\left(\eta+\frac{2}{h^2}\right)(u_2)_a=-\frac{1}{h}g+\frac{1}{2}f_a.
\end{aligned}\tag{64}$$

Ce système s'écrit sous forme matricielle :

$$\begin{pmatrix}A_{11} & A_{1\Gamma}\\ A_{\Gamma 1} & \frac{1}{2}A_{\Gamma\Gamma}\end{pmatrix}\begin{pmatrix}\boldsymbol{u}_1\\ (u_1)_a\end{pmatrix}=\begin{pmatrix}\boldsymbol{f}_1\\ \frac{1}{2}f_a+\frac{1}{h}g\end{pmatrix},\tag{65}$$

$$\begin{pmatrix}\frac{1}{2}A_{\Gamma\Gamma} & A_{\Gamma 2}\\ A_{2\Gamma} & A_{22}\end{pmatrix}\begin{pmatrix}(\boldsymbol{u}_2)_a\\ u_2\end{pmatrix}=\begin{pmatrix}\frac{1}{2}f_a-\frac{1}{h}g\\ \boldsymbol{f}_2\end{pmatrix}.\tag{66}$$

Définissons les opérateurs de trace discrets $\tilde{G}_i$ (distincts des G_i définis au paragraphe 2.4 dans le cas avec recouvrement)

$$\begin{aligned}
\tilde{G}_1:\mathbb{R}^a &\to\mathbb{R},\quad (u_1,\dots,u_a)^T\mapsto u_a,\\
\tilde{G}_2:\mathbb{R}^{J-a+1} &\to\mathbb{R},\quad (u_a,\dots,u_J)^T\mapsto u_a.
\end{aligned}$$

Extrayons $(u_1)_a$ de (65) et $(u_2)_a$ de (66),

$$(u_1)_a=\tilde{G}_1\begin{pmatrix}\boldsymbol{u}_1\\ (u_1)_a\end{pmatrix}=\tilde{G}_1\begin{pmatrix}A_{11} & A_{1\Gamma}\\ A_{\Gamma 1} & \frac{1}{2}A_{\Gamma\Gamma}\end{pmatrix}^{-1}\begin{pmatrix}\boldsymbol{f}_1\\ \frac{1}{2}f_a+\frac{1}{h}g\end{pmatrix}.\tag{67}$$

$$(u_2)_a=\tilde{G}_2\begin{pmatrix}(u_2)_a\\ \boldsymbol{u}_2\end{pmatrix}=\tilde{G}_2\begin{pmatrix}\frac{1}{2}A_{\Gamma\Gamma} & A_{\Gamma 2}\\ A_{2\Gamma} & A_{22}\end{pmatrix}^{-1}\begin{pmatrix}\frac{1}{2}f_a-\frac{1}{h}g\\ \boldsymbol{f}_2\end{pmatrix}.\tag{68}$$

En écrivant que ces deux expressions sont égales, nous obtenons une équation portant sur g :

$$\tilde{G}_1\begin{pmatrix}A_{11} & A_{1\Gamma}\\ A_{\Gamma 1} & \frac{1}{2}A_{\Gamma\Gamma}\end{pmatrix}^{-1}\begin{pmatrix}\boldsymbol{f}_1\\ \frac{1}{2}f_a+\frac{1}{h}g\end{pmatrix}-\tilde{G}_2\begin{pmatrix}\frac{1}{2}A_{\Gamma\Gamma} & A_{\Gamma 2}\\ A_{2\Gamma} & A_{22}\end{pmatrix}^{-1}\begin{pmatrix}\frac{1}{2}f_a-\frac{1}{h}g\\ \boldsymbol{f}_2\end{pmatrix}=0.\tag{69}$$

et par linéarité

$$\begin{aligned}
&\left(\tilde{G}_1\begin{pmatrix}A_{11} & A_{1\Gamma}\\ A_{\Gamma 1} & \frac{1}{2}A_{\Gamma\Gamma}\end{pmatrix}^{-1}\tilde{G}_1^T+\tilde{G}_2\begin{pmatrix}\frac{1}{2}A_{\Gamma\Gamma} & A_{\Gamma 2}\\ A_{2\Gamma} & A_{22}\end{pmatrix}^{-1}\tilde{G}_2^T\right)g=\\
&-h\tilde{G}_1\begin{pmatrix}A_{11} & A_{1\Gamma}\\ A_{\Gamma 1} & \frac{1}{2}A_{\Gamma\Gamma}\end{pmatrix}^{-1}\begin{pmatrix}\boldsymbol{f}_1\\ \frac{1}{2}f_a\end{pmatrix}+h\tilde{G}_2\begin{pmatrix}\frac{1}{2}A_{\Gamma\Gamma} & A_{\Gamma 2}\\ A_{2\Gamma} & A_{22}\end{pmatrix}^{-1}\begin{pmatrix}\frac{1}{2}f_a\\ \boldsymbol{f}_2\end{pmatrix}.
\end{aligned}\tag{70}$$

Voici une implementation de la méthode Schur dual en Matlab, avec résolution du système par l'algorithme du gradient conjugué. La courbe de convergence est représentée figure **27**.

```
Room;
α=10;                                % decomposition
h=1/(J+1);
f=f(:);
f(1:J)=f(1:J)+gg/h^2;                % add boundary conditions into rhs

f(end-J+1:end)=f(end-J+1:end)+gd/h^2;
A=A2d(eta,J,J);

A11=A(1:J*(α-1),1:J*(α-1));          % form block decomposition
AGG=A(J*(α-1)+1:J*(α-1)+1:J*α);
A22=A(J*α+1:J*J,J*α+1:J*J);
A1G=A(1:J*(α-1),J*(α-1)+1:J*α);
AG1=A(J*(α-1)+1:J*α,1:J*(α-1));
A2G=A(J*α+1:J*J,J*(α-1)+1:J*α);
AG2=A(J*(α-1)+1:J*α,J*α+1:J*J);

f1=f(1:J*(α-1));
fG=f(J*(α-1)+1:J*α);
f2=f(J*α+1:J*J);

A1=[A11  A1G
    AG1 AGG/2];
A2=[AGG/2 AG2
    A2G  A22];

G1=[zeros(size(AG1)) speye(size(AGG))];
G2=[zeros(size(AGG)) zeros(size(AG2))];

Afun=@(x) G1*(A1\(G1'*x))+G2*(A2\(G2'*x));
ft=-G1*(A1\[f1;fG/2])+G2*(A2\[fG/2;f2]);
[lα,fl,r,it,rcg]=pcg(Afun,ft,1e-6,100);
u1=A1\(G1'*lα+[f1;fG/2]);
u2=A2\(-G2'*lα+[fG/2;f2]);
u=[zeros(1,J+2);gg reshape([u1;u2(J+1:end)],J,J) gd;zeros(1,J+2)];

mesh(0:h:1,0:h:1,u);xlabel('x');ylabel('y');
pause

semilogy(0:length(rcg)-1,rcg,'-+');
xlabel('iteration');ylabel('residual');legend('Dual Schur with CG')
```

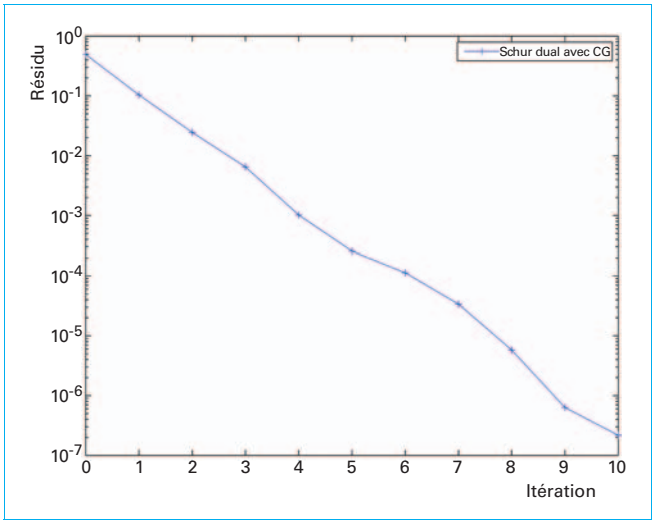


Figure 27 – Exemple de la méthode de Schur dual

3.3 FETI et Neumann-Neumann

La méthode FETI (*Finite Element Tearing and Interconnect*) de Farhat et Roux est à la base une méthode de Schur dual, mais dans sa formulation originale elle s'écrit sous forme variationnelle (voir les premiers exposés à la conférence internationale sur les méthodes de décomposition de domaines disponibles dans la rubrique Sites Internet du *Pour en savoir plus*). La solution du problème $-\Delta u = f$ dans Ω, nulle au bord de Ω, réalise le minimum de la fonctionnelle quadratique

$$J(v) = \frac{1}{2}\int_\Omega |\nabla v|^2 \mathrm{d}x - \int_\Omega fv\,\mathrm{d}x$$

sur l'espace V des fonctions, nulles au bord de Ω, dont le gradient est de carré intégrable sur Ω. Décomposons maintenant J sur les sous-domaines Ω_i, sans recouvrement,

$$J_1(v) = \frac{1}{2}\int_{\Omega_1} |\nabla v|^2 \mathrm{d}x - \int_{\Omega_1} fv\,\mathrm{d}x,\ J_2(v) = \frac{1}{2}\int_{\Omega_2} |\nabla v|^2 \mathrm{d}x - \int_{\Omega_2} fv\,\mathrm{d}x,$$
$$J(v) = J_1(v) + J_2(v),$$

et minimisons sur l'espace des couples de fonctions (v_1, v_2) telles que le gradient de v_i est de carré intégrable sur Ω_i et que $v_1 = v_2$ sur l'interface commun Γ. C'est un problème de minimisation sous contraintes égalité que l'on peut écrire au moyen du lagrangien

$$\mathfrak{L}(v_1, v_2, h) = J_1(v_1) + J_2(v_2) + \int_\Gamma h(v_2 - v_1)\mathrm{d}s.$$

Le minimum de J est atteint au point (u_1, u_2, g) où toutes les dérivées partielles du lagrangien sont nulles :

$$\partial_{v_i}\mathfrak{L}(u_1, u_2, g) = 0,\ i = 1, 2 \quad \text{et} \quad \partial_h \mathfrak{L}(u_1, u_2, g) = 0.$$

Les dérivées de ces fonctions quadratiques ou linéaires sont simples à calculer, par exemple pour la dérivée par rapport à la variable v_1 en développant $\mathfrak{L}(u_1 + v_1, u_2, g) - \mathfrak{L}(u_1, u_2, g)$ et en négligeant le terme $J_1(v_1)$. On obtient

$$\partial_{v_i}\mathfrak{L}(u_1, u_2, g)\cdot v_i = \int_{\Omega_i} \nabla u_i \cdot \nabla v_i\,\mathrm{d}x - \int_{\Omega_i} fv_i\,\mathrm{d}x,\ i = 1, 2$$
$$\partial_g \mathfrak{L}(u_1, u_2, g)\cdot h = \int_\Gamma h(u_2 - u_1)\mathrm{d}s.$$

La nullité des dérivées peut donc se traduire par

$$\begin{aligned}
&\forall v_1 \in V_1,\ \int_{\Omega_1} \nabla u_1 \nabla v_1\,\mathrm{d}x - \int_{\Omega_1} fv_1\,\mathrm{d}x - \int_\Gamma gv_1\,\mathrm{d}s = 0,\\
&\forall v_2 \in V_2,\ \int_{\Omega_2} \nabla u_2 \nabla v_2\,\mathrm{d}x - \int_{\Omega_2} fv_2\,\mathrm{d}x - \int_\Gamma gv_2\,\mathrm{d}s = 0,\\
&\forall h \text{ défini sur } \Gamma,\ \int_\Gamma h(u_2 - u_1)\mathrm{d}s = 0.
\end{aligned} \qquad \textbf{(71)}$$

Les deux premières équations du système (71) constituent les formulations variationnelles en dimension 2 des équations du système (60) avec donnée de Neumann g sur l'interface. La troisième équation est la forme faible de l'équation de continuité $u_1 = u_2$ sur Γ. La résolution des deux premières équations et l'insertion dans la troisième donne la formulation variationnelle de la méthode de Schur dual (62) . Elle est souvent résolue par une méthode d'éléments finis, ce qui mène à un système discret similaire à (70) .

La méthode FETI contient deux ingrédients supplémentaires. Nous avons déjà évoqué la grille grossière naturelle à la remarque 3.1. Nous y reviendrons à la section 4. Le second ingrédient est lié au préconditionnement. Nous avons vu que les méthodes de Schur primal et dual ont un conditionnement similaire. Mais il existe un lien beaucoup plus important entre ces deux méthodes : regardons leurs symboles $\hat{S}_P$ et $\hat{S}_D$. Leur produit est égal à

$$\hat{S}_P\hat{S}_D = \left(\coth\left(\sqrt{\eta + k^2\pi^2}\alpha\right) + \coth\left(\sqrt{\eta + k^2\pi^2}\,(1-\alpha)\right)\right)$$
$$\left(\mathrm{th}\left(\sqrt{\eta + k^2\pi^2}\alpha\right) + \mathrm{th}\left(\sqrt{\eta + k^2\pi^2}\,(1-\alpha)\right)\right)$$

qui est une fonction décroissante comprise entre 4 et $\left(\coth\left(\sqrt{\eta}\alpha\right) + \coth\left(\sqrt{\eta}\,(1-\alpha)\right)\right)\left(\mathrm{th}\left(\sqrt{\eta}\alpha\right) + \mathrm{th}\left(\sqrt{\eta}\,(1-\alpha)\right)\right)$. Le conditionnement de S_PS_D **est donc indépendant de** h : la méthode de Schur primal est un préconditionneur idéal pour la méthode de Schur dual (et réciproquement), obtenant ainsi un conditionnement indépendant du maillage. Plus précisément, au lieu de résoudre par une méthode itérative le système de Schur dual (62) avec un conditionnement $O\left(\frac{1}{h}\right)$, il est beaucoup plus avantageux de résoudre le système préconditionné

$$\begin{aligned}
&\left(S_1^{\mathcal{D},\mathcal{N}} - S_2^{\mathcal{D},\mathcal{N}}\right)\left(S_1^{\mathcal{N}\mathcal{D}}(g, 0, 0) - S_2^{\mathcal{N}\mathcal{D}}(g, 0, 0), 0, 0\right)\\
&= \left(S_1^{\mathcal{D},\mathcal{N}} - S_2^{\mathcal{D},\mathcal{N}}\right)\left(-S_1^{\mathcal{N}\mathcal{D}}(0, g_g, f) + S_2^{\mathcal{N}\mathcal{D}}(0, g_d, f), 0, 0\right),
\end{aligned} \qquad \textbf{(72)}$$

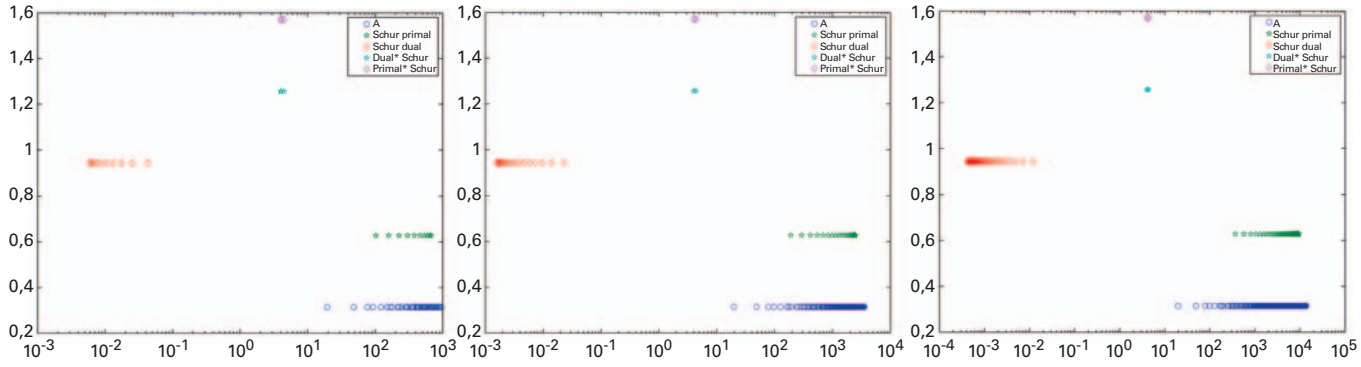
dont le conditionnement est $O(1)$, par la méthode de Krylov. Joint à la grille grossière naturelle, c'est la méthode FETI.

On peut évidemment inverser les deux processus, c'est-à-dire préconditionner Schur primal par Schur dual, ce qui a été présenté dans la littérature sous le nom de méthode de Neumann-Neumann, voir [3] . Elle peut aussi être implémentée avec une grille grossière naturelle et prend alors le nom de *Balancing Neumann-Neumann*.

Nous représentons sur la figure **28** les spectres de la matrice A de départ, des méthodes de Schur primal et dual, de la méthode de Schur primal préconditionnée par la méthode de Schur dual et vice versa, en fonction du pas du maillage. Dans cette représentation, les spectres sont tous réels et nous utilisons l'axe y uniquement pour ne pas dessiner les spectres les uns sur les autres.

Nota : Pour ces simulations, nous avons choisi de décaler légèrement l'interface du milieu du segment, car si l'interface est au milieu du domaine, ce qui donne une décomposition géométrique symétrique, la méthode de Schur primal est exactement l'inverse de la méthode de Schur dual, et ainsi le spectre de Schur primal préconditionné par Schur dual est réduit au point 1 (et vice versa).

Il est intéressant de voir comment le spectre de la méthode de Schur primal se décale vers l'infini, comme le spectre de la matrice de départ, et en contraste le spectre de la méthode de Schur dual s'approche de zéro. Les deux méthodes ont un conditionnement comparable, et nettement meilleur que celui de la matrice de départ, comme le montrent les trois premières colonnes du tableau **1**, pour diverses valeurs du nombre de points de grille J.



Les spectres sont réels, l'axe y est seulement utilisé pour les distinguer.

Figure 28 – Spectres de A, Schur primal, Schur dual, Schur primal préconditionné par Schur dual et vice versa en fonction du pas du maillage

Tableau 1 – Conditionnement des matrices des systèmes linéaires

J	*A*	Schur Primal	Schur Dual	Dual Primal	Primal Dual
10	48.37	6.55	7.28	1.11	1.11
20	178.06	13.04	14.31	1.10	1.10
40	680.62	25.91	28.26	1.09	1.09

Par contre pour les méthodes préconditionnées, le spectre reste concentré et le conditionnement est proche de 1, indépendamment du pas du maillage, ce qui montre que la méthode de Schur primal est un préconditionneur idéal pour la méthode de Schur dual, et vice versa. Cette propriété peut être démontrée pour des décompositions beaucoup plus générales que les deux sous-domaines utilisés ici, voir par exemple [43].

3.4 Méthodes de Dirichlet-Neumann et Neumann-Dirichlet

Au cours de l'histoire des méthodes de décomposition de domaines, le système (56) a été également résolu par des méthodes itératives stationnaires. Dans l'algorithme de Dirichlet-Neumann, une suite $\left(g^n, u_1^n, u_2^n\right)$ est définie par l'algorithme alterné, dont une étape est donnée par

$$\eta u_1^{n+1} - \Delta u_1^{n+1} = f \text{ dans } \Omega_1, \qquad \eta u_2^{n+1} - \Delta u_2^{n+1} = f \text{ dans } \Omega_2,$$
$$u_1^{n+1}(0,\cdot) = g_g, \qquad u_2^{n+1}(1,\cdot) = g_d,$$
$$u_1^{n+1}(\alpha,\cdot) = g^n, \qquad \frac{du_2^{n+1}}{dx}(\alpha,\cdot) = \frac{du_1^{n+1}}{dx}(\alpha,\cdot),$$
$$g^{n+1} = \theta u_2^{n+1}(\alpha,\cdot) + (1-\theta) g^n.$$

Voici une implémentation en Matlab de la méthode de Dirichlet-Neumann en dimension 2, sur l'exemple de la figure **22** :

```
eta=0 ; J=2c ;                          % number of interior mesh points
x=0:1/(J+1):1 ; y=x ;                   % finite difference mesh, including boundary
f=zeros(J, J+2) ;                       % source term, include right boundary
f([y > 0.4 & y < 0.6], [x > 0.4 & x < 0.6])=50 ;
gg=0.3*ones(J, 1) ; gg(y > 0.5 & y < 0.9)=1 ;
gd=zeros(J, 1) ;

α=10 ;                                  % decomposition
f1=f(:, 2:α) ;
f2=f(:, α+1:end) ;
g=zeros(J, 1);                          % zero initial guess
h=1/(J+1) ;
x1=0:h:α*h ;                            % finite difference meshes
x2=α*h:h:1 ;
y=0:h:1 ;
z1=zeros(1, α+1) ; z2=zeros(1, J-α+2) ; % for plotting purposes
th=0.5 ;                                % relaxation parameter
e=ones(J, 1) ;                          % construct normal derivative
pe=1e12 ;
Nα=[speye(J) - spdiags([- e (eta*h^2+4)*e - e]/2, [-1 0 1], J, J)]/h ;
ue=Solve2dR(f, eta, 0, J+1, gg*pe, gd, pe, 0) ;
for i=1:20
   u1=Solve2d(f1, eta, 0, α, gg, g) ;
   tα=Nα*[u1(:, end-1) ; u1(:, end)]+f2(:, 1)*h/2 ;
   u2=Solve2dR(f2, eta, α, J+1, tα, gd, 0, 0) ;
   g=th*u2(:, 1)+(1-th)*g ;
   mesh(x1, y, [z1 ; u1, z1]) ; hold on ; mesh(x2, y, [z2 ; u2, z2]) ; hold off
   xlabel('x') ; ylabel('y') ; zlabel('Dirichlet Neumann iterates') ;
   pause
end
```

Nous montrons sur la figure **29** que le choix du **paramètre de relaxation** θ est crucial et le meilleur choix dépend malheureusement de la position de l'interface. Si les sous-domaines et le problème sont symétriques, le choix $\theta = \frac{1}{2}$ est optimal, comme on peut le voir sur la première ligne de la figure **29**. Par contre, lorsque l'interface est plus à gauche en $\alpha = 3$, la méthode converge très mal pour ce choix de θ et peut même diverger (deuxième ligne). Il faut choisir θ plus petit pour rétablir la convergence optimale (troisième ligne).

En termes d'opérateurs de Dirichlet-Neumann et Neumann-Dirichlet, l'algorithme s'écrit

$$g^{n+1} = g^n + \theta\left(S_2^{\mathcal{N}\mathcal{D}}\left(S_1^{\mathcal{D},\mathcal{N}}\left(g^n, g_g, f\right), g_d, f\right) - g^n\right).$$

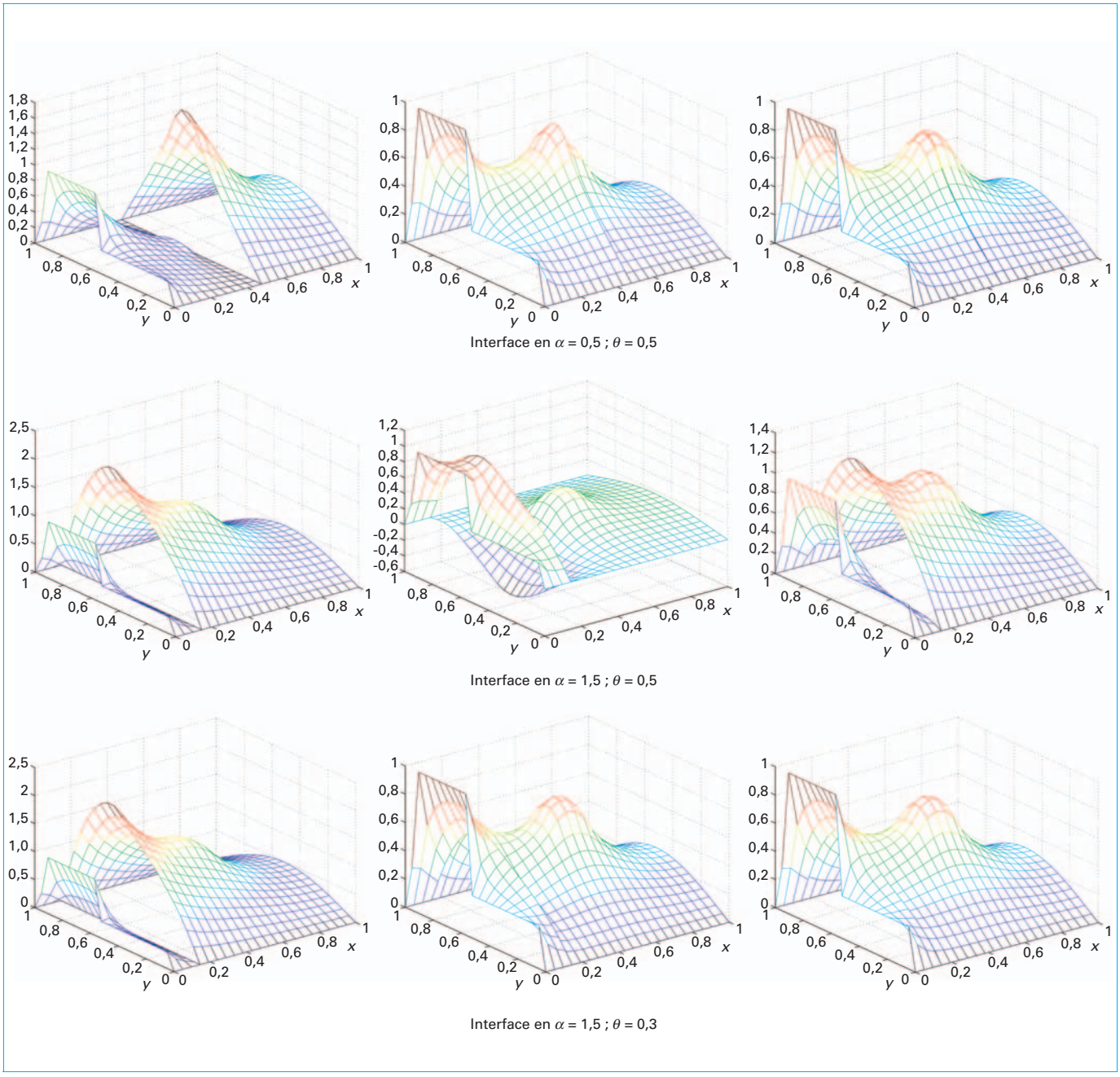


Figure 29 – Méthode de Dirichlet-Neumann : représentation des trois premières itérations de gauche à droite, pour trois couples de valeurs (α, θ)

C'est un algorithme de Richardson de paramètre θ pour la résolution du système linéaire d'inconnue g,

$$S_2^{\mathcal{N}\mathcal{D}}\left(S_1^{\mathcal{D},\mathcal{N}}\left(g, g_g, f\right), g_d, f\right) = g.$$

Nous utilisons maintenant le fait que, pour toutes fonctions f, g_d et g, $S_2^{\mathcal{N}\mathcal{D}}\left(S_2^{\mathcal{D},\mathcal{N}}\left(g, g_g, f\right), g_d, f\right) = g$ pour écrire le système linéaire sous la forme

$$S_2^{\mathcal{N}\mathcal{D}}\left(S_1^{\mathcal{D},\mathcal{N}}\left(g, g_g, f\right) - S_2^{\mathcal{D},\mathcal{N}}\left(g, g_d, f\right), g_d, f\right) = 0 \qquad \textbf{(73)}$$

C'est donc un algorithme de Richardson pour la résolution du problème de Schur primal (58) préconditionné par $S_2^{\mathcal{N}\mathcal{D}}$.

Pour l'algorithme de Neumann-Dirichlet, une suite $\left(g^n, u_1^n, u_2^n\right)$ est définie par

$$\eta u_1^{n+1} - \Delta u_1^{n+1} = f \text{ dans } \Omega_1, \qquad \eta u_2^{n+1} - \Delta u_2^{n+1} = f \text{ dans } \Omega_2,$$
$$u_1^{n+1}(0,\cdot) = g_g, \qquad u_2^{n+1}(1,\cdot) = g_d,$$
$$\frac{du_1^{n+1}}{dx}(\alpha,\cdot) = g^n, \qquad u_2^{n+1}(\alpha,\cdot) = u_1^{n+1}(\alpha,\cdot).$$

$$g^{n+1} = \theta \frac{du_2^{n+1}}{dx}(\alpha,\cdot) + (1-\theta) g^n.$$

En termes d'opérateurs de Dirichlet-Neumann et Neumann-Dirichlet, cet algorithme s'écrit

$$g^{n+1} = g^n + \theta\left(S_2^{\mathcal{D},\mathcal{N}}\left(S_1^{\mathcal{N}\mathcal{D}}\left(g^n, g_g, f\right), g_d, f\right) - g^n\right),$$

ce qui est de nouveau un algorithme de Richardson pour la résolution du système linéaire pour g :

$$S_2^{\mathcal{D},\mathcal{N}}\left(S_1^{\mathcal{N}\mathcal{D}}\left(g, g_g, f\right), g_d, f\right) = g,$$

ou encore de

$$S_2^{\mathcal{D},\mathcal{N}}\left(S_1^{\mathcal{N}\mathcal{D}}\left(g, g_g, f\right) - S_2^{\mathcal{N}\mathcal{D}}\left(g, g_d, f\right), g_d, f\right) = 0. \qquad \textbf{(74)}$$

C'est maintenant un algorithme de Richardson pour la résolution du problème de Schur dual (62) préconditionné par $S_2^{\mathcal{D},\mathcal{N}}$.

Une décomposition en modes de Fourier comme pour l'analyse des méthodes de Schur montre que la condition des deux systèmes (73) et (74) ne dépend pas du pas du maillage h.

Méthodes de décomposition de domaines

par **Martin J. GANDER**
Professeur de mathématiques
Section de Mathématiques, Université de Genève

et **Laurence HALPERN**
Professeur de Mathématiques
Laboratoire Analyse, Géométrie et Applications, Université Paris 13

Sources bibliographiques

[1] ARNOLD (V.). – *Équations différentielles ordinaires*. Éditions MIR, Moscou (1974).

[2] BJORHUS (M.). – *Semi-discrete subdomain iteration for hyperbolic systems*. Tech. Rep. 4, NTNU (1995).

[3] BOURGAT (J.-F.), GLOWINSKI (R.), LE TALLEC (P.) et VIDRASCU (M.). – *Variational formulation and algorithm for trace operator in domain decomposition calculations*. in Domain Decomposition Methods, T. Chan, R. Glowinski, J. Périaux, and O. Widlund, eds., Philadelphia, PA, SIAM, pp. 3-16 (1989).

[4] CAI (X.-C.) et SARKIS (M.). – *A restricted additive Schwarz preconditioner for general sparse linear systems*. SIAM Journal on Scientific Computing, 21, pp. 239-247 (1999).

[5] CHAN (T.F.) et MATHEW (T.P.). – *Domain decomposition algorithms*. in Acta Numerica 1994. Cambridge University Press, pp. 61-143 (1994).

[6] CHARTIER (P.) et PHILIPPE (B.). – *A parallel shooting technique for solving dissipative ODEs*. Computing, 51, pp. 209-236 (1993).

[7] CHENEY (E.W.). – *Introduction to Approximation Theory*. McGraw-Hill Book Co., New York (1966).

[8] DESPRÉS (B.). – *Méthodes de décomposition de domaines pour les problèmes de propagation d'ondes en régime harmonique*. PhD thesis, Université Paris IX Dauphine (1991).

[9] DINH (Q.V.), MANTEL (B.), PERIAUX (J.) et GLOWINSKI (R.). – *Approximate solution of the Navier-Stokes equations for incompressible viscous fluids, related domain decomposition methods*. vol. 1005 of Lect. notes in Math., Springer, pp. 46-86 (1983).

[10] DRYJA (M.). – *A capacitance matrix method for Dirichlet problem on polygon region*. Numer. Math., 39, pp. 51-64 (1982).

[11] DRYJA (M.) et WIDLUND (O.B.). – *An additive variant of the Schwarz alternating method for the case of many subregions*. Tech. Rep. 339, also Ultracomputer Note 131, Department of Computer Science, Courant Institute (1987).

[12] —. – *Schwarz methods of Neumann-Neumann type for three-dimensional elliptic finite element problems*. Comm. Pure Appl. Math., 48, pp. 121-155 (1995).

[13] EFSTATHIOU (E.) et GANDER (M.J.). – *Why Restricted Additive Schwarz converges faster than Additive Schwarz*. BIT Numerical Mathematics, 43, pp. 945-959 (2003).

[14] EVANS (L.C.). – *Partial differential equations*. vol. 19 of Graduate studies in Mathematics, AMS (1998).

[15] GANDER (M.J.). – *Optimized Schwarz methods*. SIAM J. Numer, Anal., 44, pp. 699-731 (2006).

[16] —. – *Schwarz methods over the course of time*. Electron. Trans. Numer. Anal, 31, pp. 228-255 (2008).

[17] GANDER (M.J.) et HAIRER (E.). – *Nonlinear convergence analysis for the parareal algorithm*. in Domain Decomposition Methods in Science and Engineering XVII, O. B. Widlund and D. E. Keyes, eds., vol. 60 of Lecture Notes in Computational Science and Engineering, Springer, pp. 45-56 (2008).

[18] GANDER (M.J.), HALPERN (L.) et NATAF (F.). – *Optimal convergence for overlapping and non-overlapping Schwarz waveform relaxation*. in Eleventh international Conference of Domain Decomposition Methods, C.-H. Lai, P. Bjørstad, M. Cross, and O. Widlund, eds., ddm. org (1999).

[19] GANDER (M.J.), JIANG (Y.-L.) et LI (R.-J.). – *Parareal schwarz waveform relaxation methods*. in Domain Decomposition Methods in Science and Engineering XX, O. B. Widlund and D. E. Keyes, eds., Lecture Notes in Computational Science and Engineering, Springer, submitted (2011).

[20] GANDER (M.J.) et STUART (A.M.). – *Space-time continuous analysis of waveform relaxation for the heat equation*. SIAM J. Sci. Comput., 19, pp. 2014-2031 (1998).

[21] GANDER (M.J.) et VANDEWALLE (S.). – *Analysis of the parareal time-parallel time-integration method*. SIAM J. Sci. Comput., 29, pp. 556-578 (2007).

[22] GANDER (M.J.) et ZHAO (H.). – *Overlapping Schwarz waveform relaxation for the heat equation in n-dimensions*. BIT, 42, pp. 779-795 (2002).

[23] HAYNSWORTH (E.V.). – *On the Schur complement*. Tech. Rep. 20, Basel Mathematical Notes (June 1968).

[24] KELLER (H.B.). – *Numerical Solution for Two-Point Boundary-Value Problems*. Dover Publications, Inc., New York (1992).

[25] LELARASMEE (E.), RUEHLI (A.E.) et SANGIOVANNI-VINCENTELLI (A. L.). – *The waveform relaxation method for time-domain analysis of large scale integrated circuits*. IEEE Trans. on CAD of IC and Syst., 1, pp. 131-145 (1982).

[26] LINDELOF (E.). – *Sur l'application des méthodes d'approximations successives à l'étude des intégrales réelles des équations différentielles ordinaires*. Journal de Mathématiques Pures et Appliquées, 10, pp. 117-128 (1894).

[27] LIONS (J.-L.), MADAY (Y.) et TURINICI (G.). – *A parareal in time discretization of PDE's*. C. R. Acad. Sci. Paris, Serie I, 332, pp. 661-668 (2001).

[28] LIONS (P.-L.). – *On the Schwarz alternating method I*. in First International Symposium on Domain Decomposition Methods for Partial Differential Equations, R. Glowinski, G. H. Golub, G. A. Meurant, and J. Périaux, eds., Philadelphia, PA, SIAM, pp. 1-42 (1988).

[29] —. – *On the Schwarz alternating method II : Stochastic interpretation and orders properties*. in Domain Decomposition Methods, T. Chan, R. Glowinski, J. Périaux, and O. Widlund, eds., Philadelphia, PA, SIAM, pp. 47-70 (1989).

[30] —. – *On the Schwarz alternating method III : A variant for nonoverlapping subdomains*. in Third International Symposium on Domain Decomposition Methods for Partial Differential Equations, held in Houston, Texas, March 20-22, 1989, T. F. Chan, R. Glowinski, J. Périaux, and O. Widlund, eds., Philadelphia, PA, SIAM, pp. 202-223 (1990).

[31] MADAY (Y.) et TURINICI (G.). – *The parareal in time iterative solver : a further direction to parallel implementation*. in Proceedings of the 15th international domain decomposition conference, R. Kornhuber, R. H. W. Hoppe, J. Périaux, O. Pironneau, O. B. Widlund, and J. Xu, eds., Springer LNCSE, pp. 441-448 (2003).

[32] MANDEL (J.) et BREZINA (M.). – *Balancing domain decomposition for problems with large jumps in coefficients*. Math. Comp., 65, pp. 1387-1401 (1996).

[33] MANDEL (J.) et TEZAUR (R.). – *Convergence of a substructuring method with Lagrange multipliers*. Numer. Math., 73, pp. 473-487 (1996).

[34] MILLER (K.). – *Numerical analogs to the Schwarz alternating procedure*. Numer. Math., 7, pp. 91-103 (1965).

[35] PICARD (E.). – *Mémoire sur la théorie des équations aux dérivée partielles et la méthodes des approximations successives*. Journal de Mathématiques Pures et Appliquées, 6, pp. 145-210 (1890).

[36] —. – *Sur l'application des méthodes d'approximations successives à l'étude de certaines équations différentielles ordinaires*. Journal de Mathématiques Pures et Appliquées, 9, pp. 217-271 (1893).

[37] PRZEMIENIECKI (J.S.). – *Matrix structural analysis of substructures*. Am. Inst. Aero. Astro. J., 1, pp. 138-147 (1963).

[38] QUARTERONI (A.) et VALLI (A.). – *Domain Decomposition Methods for Partial Differential Equations*. Oxford Science Publications (1999).

[39] RUDIN (W.). – *Real and Complex Analysis*. Mc Graw-Hill (1966).

[40] SAAD (Y.). – *Iterative Methods for Sparse Linear Systems*. PWS Publishing Company (1996).

[41] SCHWARZ (H.A.). – *Über einen Grenzübergang durch alternierendes Verfahren*. Vierteljahrsschrift der Naturforschenden Gesellschaft in Zürich, 15, pp. 272-286 (1870).

[42] SMITH (B.F.), BJØRSTAD (P.E.) et GROPP (W.). – *Domain Decomposition : Parallel Multilevel Methods for Elliptic Partial Differential Equations*. Cambridge University Press (1996).

[43] TOSELLI (A.) et WIDLUND (O.). – *Domain Decomposition Methods – Algorithms and Theory*. vol. 34 of Springer Series in Computational Mathematics, Springer (2004).

[44] TRAN (M.B.). – *Schwarz domain decomposition methods for linear and nonlinear problems*. PhD thesis, Université Paris 13 (September 2011).

[45] WEISS (R.). – *Convergence of shooting methods*. BIT, 13, pp. 470-475 (1973).

[46] ZHANG (F.). – *The Schur complement and its applications*. vol. 4 of Numerical Methods and Algorithms, Springer (December 2005).

À lire également dans nos bases

CABANE (R.). – *Méthodes numériques en algèbre linéaire*. [AF 485] Mathématiques pour l'ingénieur (1995).

MEURANT (G.). – *Méthodes de Krylov pour la résolution du systèmes linéaires*. [AF 488] Mathématiques pour l'ingénieur (2007).

SPITERI (P.). – *Méthode des différences finies pour les EDP stationnaires*. [AF 500] Mathématiques pour l'ingénieur (2002).

SPITERI (P.). – *Approche variationnelle pour la méthode des éléments finis*. [AF 503] Mathématiques pour l'ingénieur (2003).

BREZINSKI (C.). – *Méthodes numériques de base - Analyse numérique*. [AF 1 220] Mathématiques pour l'ingénieur (2006).

Sites Internet

Premiers exposés de la conférence internationale sur les méthodes de décomposition de domaines — http://www.ddm.org/conferences.html