Eleventh International Conference on Domain Decomposition Methods

Eleventh International Conference on Domain Decomposition Methods

Edited by CHOI-HONG LAI, PETTER E. BJØRSTAD, MARK CROSS AND OLOF WIDLUND

DDM.org

Contents

Part III: Applications 369

Part IV: Student Papers 503

Preface

This volume represents the Proceedings of the Eleventh International Conference on Domain Decomposition Methods for Science and Engineering held at the University of Greenwich - Avery Hill Campus, July 20th - 24th, 1998.

This almost annual conference series has now become a major event in Applied Mathematics and Computational Science. At this 11th Conference, there were two to three parallel sessions being run daily and a total of six mini-symposia addressing various aspects of this subject involving both theory and applications. There were also sixteen invited presentations delivered by internationally well known scientists. Some of these invited presentations are included in this volume.

A new feature at this Conference was a Graduate Student Paper Competition. There were thirteen graudate student papers at the final competition being heard by the entire conference and examined by the Judging Committee which consisted of M Cross, I G Graham, J E Roberts, O Widlund and J Xu. Prizes were offered covering international return flights to attend the Conference.

In addition to the invited and student presentations, there were a total 102 mini-symposium and contributed talks. Many presentations represented an interdiciplinary view of the subject with papers ranging from the use of advanced computer architectures to the development of related software tools, from the theoretical formulation to the design of robust algorithms and from engineering to medical science applications.

The participants who have their papers presented at the Conference were invited to submit their manuscripts for the Proceedings. All submitted manuscripts have gone through a reviewing process. Keynote speakers were invited to submit manuscripts of 12 pages for each article. Speakers for contributed sessions and minisymposia were invited to submit manuscripts of 8 pages for each article. We believe that the traditional partitioning of the Proceedings into Theory, Algorithms, and Applications still works well for the present volume. We have also included a new section with Student Papers. Papers in each section are alphabetically ordered as according to the first author in the author list of each paper.

The Conference was organised by M Cross (Greenwich), K Chen (Liverpool), A Craig (Durham), M G Everett (Greenwich), I G Graham (Bath), C - H Lai (Greenwich), and K A Pericleous (Greenwich) with the help from the three European Affair Advisors, P Deuflhard (ZIB - Berlin), J Periaux (Dassault), and J E Roberts (INRIA). The technical direction of the Conference was provided by the Scientific Committee which consisting of P E Bjørstad, J H Bramble, T F Chan, P Deuflhard,

R Glowinski, R Hoppe, H Kawarada, D E Keyes, Yu Kuznetsov, J Periaux, O Pirronneau, Z - C Shi, O Widlund, and J Xu. The Conference received support from the Engineering and Physical Sciences Research Council (UK), the American Pacific Technology Group (Hong Kong), and the University of Greenwich.

We wish to thank the Conference Secretary, Mrs F Barkshire, for her enomous hard work towards the success of the Conference. We are also grateful to the organisers of the mini-symposia for attracting high quality presentations. Timely production of these Proceedings has only been possible through the cooperation of the authors and numerous referees. The editors wish to thank the effort of all referees and the quick response from the contributors to the editors regarding comments and modifications of their manuscripts.

The web page http://dd11.gre.ac.uk contains the complete programme of the Conference and an electronic version of these Proceedings. The Official Domain Decomposition Web site is http://www.ddm.org, from which further details for future conferences, previous conference proceedings, and the present proceedings may be obtained.

Choi-Hong Lai
Greenwich, UK

Petter Bjørstad
Bergen, Norway

Mark Cross
Greenwich, UK

Olof Widlund
New York, USA

Part I

Theory

Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

1

Domain Decomposition Methods for Non-Symmetric Problems

Y. Achdou[1], C. Japhet[2], P. Le Tallec[3], F. Nataf[4], F. Rogier[5] & M. Vidrascu[6]

Introduction

The two algorithms presented below are especially fitted for non-symmetric elliptic problems. The model equation is the convection-diffusion equation:

$$\frac{\partial u}{\partial t} + \vec{a}.\nabla u - \nu \Delta u = f \tag{1}$$

This equation is important in itself in engineering or environnemental sciences for instance. It models the transport and diffusion of species (pollutant in air or water, electrons in semiconductor devices, ...) in a given flow (with velocity field $\vec{a}$) . It is also a key ingredient in Navier-Stokes equations. An implicit scheme in time will demand at each time step the solving of

$$\mathcal{L}(u) \equiv \frac{u}{\Delta t} + \vec{a}.\nabla u - \nu \Delta u = f \tag{2}$$

1 Insa Rennes, 20 Av. des Buttes de Coesmes, 35043 Rennes, France. yves.achdou@insa-rennes.fr
2 ONERA and CMAP, Ecole Polytechnique, 91128 Palaiseau, France. japhet@onera.fr
3 Universty of Paris IX, Ceremade, Paris, France, letallec@ceremade.dauphine.fr
4 address to whom correspondance should be sent: CMAP, Ecole Polytechnique, 91128 Palaiseau, France. nataf@cmapx.polytechnique.fr
5 ONERA-CERT, 2 av. Edouard Belin - BP 4025 - 31055 TOULOUSE CEDEX 4 - FRANCE , Francois.Rogier@cert.fr
6 INRIA Rocquencourt, domaine de Voluceau, BP105, 78153 Le Chesnay, France, Marina.Vidrascu@inria.fr
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

The first algorithm is a preconditioner for the Schur formulation of domain decomposition problems. It is an extension of the well-known Neumann-Neumann preconditioner [BGLTV89] to non-symmetric problems. The second algorithm is an optimized Schwarz method. The Dirichlet boundary conditions on the interfaces are replaced by more general boundary conditions. The algorithm can then be used on non-overlapping (and/or overlapping) subdomains and has a fast convergence.

In this presentation, we emphasize the use of the Fourier transform on the continuous problem as a tool of analysis and design. Hence, we consider the simple geometry of the plane $\mathbf{R}^2$ divided into two or more vertical strips. This might seem inappropriate since Fourier analysis is essentially limited to constant coefficients operators and since computations are made on discretized problems and not on continuous models. Moreover, real life geometries are more complex. Variational settings or matrix analysis, for instance, do not have these limitations. Nevertheless, some ideas come up much more easily in the Fourier space and are then independent of the discretization scheme. Moreover, as we shall see, the methods that we propose are not limited to constant coefficient operators and can be used with various numerical schemes.

More precisely, in § 1 we study the Neumann-Neumann preconditioner when applied to a non-symmetric operator. We explain why it is not adapted. In the next section, we extend it to non-symmetric operators by modifying the Neumann boundary conditions. We shall also see how to write it by using a variational formulation for variable coefficients. Numerical results are shown to illustrate the efficiency of the preconditioner. We mention also the fact that it can be used in a FETI framework, see [FMR94].

In § 1, we turn to the classical Schwarz algorithm. We show that by changing the interface conditions, much better convergence rates can be reached. In § 1, we present an optimization procedure for choosing efficient and easy to implement interface conditions. A frozen coefficient approximation enables its use in a variable coefficient context. Numerical results are shown to illustrate the efficiency of the preconditioner.

The Neumann-Neumann preconditioner

We consider the case of a decomposition of the plane $\mathbf{R}^2$ into two subdomains $\Omega_1 = (-\infty, 0] \times \mathbf{R}$ and $\Omega_2 = [0, \infty) \times \mathbf{R}$. The interface $\{0\} \times \mathbf{R}$ is denoted Γ. The Schur formulation is defined as follows:

Find u_0 such that the solution of the following two Dirichlet problems ($i = 1, 2$):

$$\mathcal{L}(u_i) = f \text{ in } \Omega_i,\ \ u_i = u_0 \text{ on } \Gamma \tag{3}$$

have matching normal derivatives on the interface Γ. Let us denote $\mathcal{S}(u_0, f) = \nu(\frac{\partial u_1}{\partial n_1} + \frac{\partial u_2}{\partial n_2})$ the jump of the normal derivatives. The Schur formulation is thus:

Find u_0 such that

$$\mathcal{S}(u_0, 0) = -\mathcal{S}(0, f). \tag{4}$$

The problem (4) once discretized is solved by a Krylov type method. Preconditioning (4) amounts to finding an approximate inverse for the operator $\mathcal{S}(., 0)$. The Neumann-Neumann preconditioner $\mathcal{T}_{NN}$ is defined as follows: let g be a function on Γ, a Neumann boundary value problem is solved in each subdomain ($i = 1, 2$)

Find v_i such that

$$\mathcal{L}(v_i) = 0 \text{ in } \Omega_i,\ \nu \frac{\partial v_i}{\partial n_i} = g \text{ on } \Gamma \tag{5}$$

The preconditioner $\mathcal{T}_{NN}$ maps g to $\frac{1}{2}(v_1 + v_2)$. By performing a Fourier transform along the interface Γ, it is possible to find the symbol of these operators (see [AN97] or [ATNV]):

$$\mathcal{S}(u_0, 0) = \mathcal{F}_\xi^{-1}\left(\sqrt{\frac{4\nu}{\Delta t} + a_x^2 + 4Ia_y\xi\nu + 4\xi^2\nu^2}\, \hat{u}_0(\xi)\right) \tag{6}$$

and

$$\mathcal{T}_{NN}(\mathcal{S}(u_0, 0)) = \mathcal{F}_\xi^{-1}\left((1 + \frac{a_x^2}{\frac{4\nu}{\Delta t} + 4Ia_y\xi\nu + 4\xi^2\nu^2})\, \hat{u}_0(\xi)\right) \tag{7}$$

where ξ is the Fourier variable, $I^2 = -1$ and $\mathcal{F}_\xi^{-1}$ denotes the inverse Fourier transform. The condition number of the discrete version of $\mathcal{TS}$ can therefore be estimated by $\frac{symbol(\mathcal{TS})(0)}{symbol(\mathcal{TS})(\infty)} = 1 + \frac{a_x^2 \Delta t}{4\nu}$. For a small time step or a very diffusive problem, the Neumann-Neumann preconditioner should perform well. For a large time step or a convection dominated flow, the condition number of $\mathcal{TS}$ can be very large.

A Robin-Robin Preconditioner

By modifying the Neumann boundary condition in (5), it is possible to have an optimal result for two semi-infinite subdomains. Indeed, let g be a function on Γ, a Robin boundary value problem is solved in each subdomain ($i = 1, 2$)

Find v_i such that

$$\mathcal{L}(v_i) = 0 \text{ in } \Omega_i,\ \nu \frac{\partial v_i}{\partial n_i} - \frac{\vec{a}.\vec{n}_i}{2} v_i = g \text{ on } \Gamma \tag{8}$$

The preconditioner $\mathcal{T}_{RR}$ maps g to $\frac{1}{2}(v_1 + v_2)$. It can be seen that (8) is well posed. By performing a Fourier transform, one sees that $\mathcal{T}_{RR}$ is an exact preconditioner, $\mathcal{T}_{RR}\mathcal{S}(., 0) = Id$.

It is interesting to note that the Robin boundary condition in (8) may arise from a variational formulation, see [ATNV]. Indeed let us consider a skew-symmetric variational formulation of (2) for some domain Ω (forgetting on purpose boundary terms):

Find u such that

$$\int_\Omega \frac{uw}{\Delta t} - \frac{1}{2}\nabla \cdot \vec{a}\ uw + \frac{1}{2}(\vec{a}.\nabla uw - \vec{a}.\nabla wu) + \nu \nabla u.\nabla w = \int_\Omega fw,\quad \forall w \tag{9}$$

Integrating by parts (9), (and this time keeping boundary terms), we get as a boundary term the Robin boundary condition in (8) :

$$\int_\Omega (\frac{u}{\Delta t} + \vec{a}.\nabla u - \nu \Delta u) w + \int_{\partial\Omega} (\nu \frac{\partial u}{\partial n} - \frac{\vec{a}.\vec{n}}{2}) w = \int_\Omega fw,\quad \forall w \tag{10}$$

This remark is very useful when implementing this method in a finite element framework.
It is worth noticing that the above Robin boundary condition may also be used in a FETI context. The dual Schur formulation would consist of:

Find λ such that the solution of the following two Robin boundary value problems $(i = 1, 2)$:

$$\mathcal{L}(u_i) = f \text{ in } \Omega_i,\ \nu\frac{\partial u_i}{\partial n_i} - \frac{\vec{a}.\vec{n}_i}{2}u_i = (-1)^i\lambda \text{ on } \Gamma \tag{11}$$

match on the interface Γ. Let us denote $\mathcal{DR}(\lambda, f) = \frac{1}{2}(u_2 - u_1)$ half the jump on the interface. The modified Schur formulation is thus,

Find λ such that

$$\mathcal{DR}(\lambda, 0) = -\mathcal{DR}(0, f). \tag{12}$$

The problem (12), once discretized, is solved by a Krylov type method. The optimal preconditioner is then the usual Dirichlet-Dirichlet preconditioner.

More precisely, let v be a function on Γ, a Dirichlet boundary value problem is solved in each subdomain $(i = 1, 2)$

Find v_i such that

$$\mathcal{L}(v_i) = 0 \text{ in } \Omega_i,\ v_i = v \text{ on } \Gamma \tag{13}$$

The preconditioner $\mathcal{T}_{DD}$ maps v to $\nu(\frac{\partial v_1}{\partial n_1} + \frac{\partial v_2}{\partial n_2})$. By performing a Fourier transform, one sees that $\mathcal{T}_{DD}$ is an exact preconditioner, $\mathcal{T}_{DD}\mathcal{DR}(.,0) = Id$.
Let us also mention the fact that the algorithm can be used with non matching grids on the interface and the mortar formulation see [BMP94], [Ach95], see [AN97].

An interesting feature of the Robin-Robin preconditioner is its nilpotency property for a domain decomposed into N vertical strips. Indeed, the preconditioner is then no more exact but $\mathcal{T}_{RR}\mathcal{S}(.,0) - Id$ is close to a nilpotent operator. We assume that the component of the velocity normal to the interface is positive $a_x \geq 0$. Let H denote the minimum of the widths of the subdomains. For $e^{-(a_x+\sqrt{a_x{}^2+\frac{4\nu}{\Delta t}})\,H/\nu}$ small enough, the preconditioned system is close to an idempotent operator of order $\left[\frac{N}{2}\right]$ where $[x]$ denotes the integer part of x. This has an important effect on the convergence of GMRES applied to the preconditioned system since it means that a decrease of the residual should occur only after $\left[\frac{N}{2}\right]$ iterations, see Figure 1.

Theorem 1 *Assume that the velocity field $\vec{a}$ is uniform. Let $\Omega = (0, H_\Omega) \times \mathbf{R}$ be decomposed into N nonoverlapping vertical strips $\Omega_k = (l_k, l_{k+1}) \times \mathbf{R}$, $0 \leq k \leq N-1$ and $\Gamma_{k,k+1} = \{l_{k+1}\} \times \mathbf{R}$.*
Let $\mathcal{H}_N^s = \prod_{k=1}^{N-1} H^s(\Gamma_{k,k+1})$, $s \in \mathbf{R}$ be the space of H^s functions on the $N-1$ interfaces. Let $U \in \mathcal{H}_N^s$ be denoted by $U = (u_k)_{1\leq k\leq N-1}$. The space $\mathcal{H}_N^s$ is endowed with the norm $\|U\|_{\mathcal{H}_{N,\infty}^s} = \sup_{1\leq k\ N-1}\|u_k\|_{H^s(\Gamma_{k,k+1})}$.
Let H be the size of the smallest subdomain, $H = \min_k(l_{k+1} - l_k)$. Assume $\epsilon \equiv e^{-(a_x+\sqrt{a_x{}^2+\frac{4\nu}{\Delta t}})\,H/\nu} < 1$ and that $\rho \equiv 3N\epsilon/(1-\epsilon)^{N+1} < 1$.

Then, for $n \geq [\frac{N}{2}]$, we have

$$\|(\mathcal{T} \circ \mathcal{S}(.,0) - Id)^n\|_{L(\mathcal{H}^s_N)} \leq \frac{N}{2(1-\rho)} \frac{1}{(1-\epsilon)^{N-2}} \; \rho^{\left[\frac{n}{[\frac{N}{2}]}\right] - 1} \tag{14}$$

where $[x]$ denotes the integer part of x.

Remark 1 *In fact, as soon as $e^{(a_x - \sqrt{{a_x}^2 + \frac{4\nu}{\Delta t}})\, H/\nu} \ll 1$, $\mathcal{T} \circ \mathcal{S}$ is already very close to identity and a better estimate can easily be obtained:*

$$\|(\mathcal{T} \circ \mathcal{S}(.,0) - Id)\|_{L(\mathcal{H}^s_N)} \leq \frac{6 e^{(a_x - \sqrt{{a_x}^2 + \frac{4\nu}{\Delta t}})\, H/\nu}}{(1-\epsilon)^3}. \tag{15}$$

The above theorem is thus interesting when $e^{(a_x - \sqrt{{a_x}^2 + \frac{4\nu}{\Delta t}})\, H/\nu}$ is close to 1. This corresponds for instance to the case of a very large time step with a strongly laminar flow.
On the other hand, the above results cannot be applied when the operator is symmetric and is very close to a Laplace operator ($\frac{1}{\Delta t} \ll 1$). Indeed, in order to have good convergence rates, it is then necessary to modify the Robin-Robin (which amounts to the Neumann-Neumann preconditioner) by adding a coarse space and a pseudo inverse for the Neumann problem, see [ATNV].

Numerical Results in two dimensions for the Robin-Robin preconditioner

The advection-diffusion is discretized on a Cartesian grid by a Q1-streamline-diffusion method. The system for the nodal values at the interface is solved by a preconditioned GMRES algorithm and the stopping criterion is to reduce the initial residual by a factor 10^{-10}. The preconditioners are either of the type Robin-Robin (R-R), Neumann-Neumann (N-N) or the identity (–).

A first comparison between the preconditioners

The first series of tests will be to compare the performances of the Robin-Robin (R-R), Neumann-Neumann (N-N) preconditioners, and the non-preconditioned method, in some very simple typical situations. Here, the domain is the rectangle $[0,1] \times [0,0.2]$ partitioned into five square vertical strips of sizes 0.2×0.2. In each subdomain, there is a uniform grid of 60×60 quadrangular elements. We choose $\Delta t = 1$ and $\nu = 0.001$ or $\nu = 1$, and four velocities:

1. $\vec{a} = \vec{e_1}$. In this case, the velocity is perpendicular to the interfaces between the subdomains.
2. $\vec{a} = \vec{e_2}$. In this case, the velocity is parallel to the interfaces.
3. $\vec{a} = \frac{\sqrt{2}}{2}(\vec{e_1} + \vec{e_2})$. We refer to this convecting field as *oblique*.
4. $\vec{a} = 2\pi\left((x_1 - 0.5)\vec{e_2} - (x_2 - 0.1)\vec{e_1}\right)$. Here, the convecting field can be seen as a vortex.

viscosity	Precond.\ velocity.	normal	parallel	oblique	rotating
$\nu = 0.001$	R-R	3	2	5	36
	N-N	52	2	42	> 100
	–	14	34	13	71
$\nu = 1$	R-R	9	9	10	10
	N-N	9	9	10	11
	–	30	38	41	41

Table 1: Different velocity fields and five subdomains

Based on these results, several remarks can be done:
The Robin-Robin preconditioner performs much better than the Neumann-Neumann preconditioner when the viscosity is small while the performances are equivalent for large viscosity.
For small viscosities, when the velocity is not parallel to the interface, the Neumann-Neumann preconditioner gives very poor results (poorer than when no preconditioner is used). On the contrary, when the velocity is parallel to the interfaces, both the Neumann-Neumann and the Robin-Robin preconditioners work very well (2 iterations), (note that they are equivalent in this case). These results are in complete agreement with the Fourier analysis above.
Thus, the Robin-Robin adapts smoothly to the different situations presented in the table.
In our implementation, one iteration of the Robin-Robin or Neumann-Neumann method costs twice as much as when no preconditioner is used. Even though, the Robin-Robin preconditioner remains always faster.

Influence of the number of subdomains: case of strips

The purpose of these tests is to assess the nilpotency properties of the algorithm. To illustrate the theory above, the domain is partitioned into vertical strips, the velocity is uniform and normal to the interfaces. The dependence of the number of iterations with respect to the number of strips will be investigated.

Here, the domain is $\Omega = (0, \frac{N_{sd}}{4}) \times (0,1)$. It is partitioned into N_{sd} square subdomains, which are rectangles of sizes 0.25×1. In each subdomain, there is a uniform grid of size 20×40.

Partition		4×1	8×1	12×1	24×1	36×1
Grid		20×40	20×40	20×40	20×40	20×40
$\Delta t = 1$,	R-R	5	8	12	23	30
$\vec{a} = 3\vec{e}_1$,	–	13	18	22	33	44
$\nu = 0.001$	N-N	43	> 100	> 100	> 100	> 100

Table 2: Influence of the number of subdomains

On Figure 1(left), the convergence of the GMRES algorithms with and without the Robin-Robin preconditioner are plotted. It is very clear on Figure 1(left), that for the

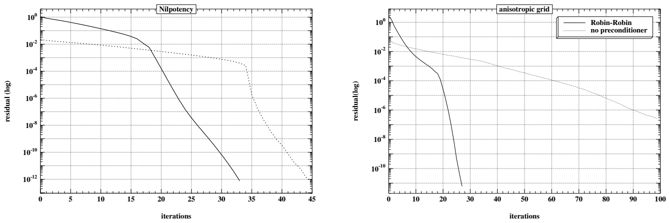


Figure 1 The convergence with and without preconditioner: 36 subdomains. Left:uniform grid. Right:anisotropic grid.

Robin-Robin preconditioner, the residual does not vary significantly for $\frac{N_{sd}}{2} = 18$ iterations, then drops quickly. This confirms very well the theory above on the idempotency of the preconditioned Schur complement matrix. For GMRES without preconditioner, the same kind of phenomenon is observed, but the decay of the residual is obtained only after $N_{sd} = 36$ iterations, see [ATNV] for details. Note that, since the number of grid nodes is not large enough, the rates of convergence after the idempotency threshold are of the same order. This would not be the case for a finer grid, as we can see Figure 1(right) and in § 1. On Figure 1(right), the convergence plots correspond to the same experiment except that the grid has been refined with a geometrical progression of ratio 0.9 in the x_2 direction. Therefore, the grid is very fine near the boundary $x_2 = 0$. We see that the idempotency properties of the preconditioned operator are conserved, although they appear less clearly. On the contrary, the non preconditioned operator yields a very poor convergence.

Influence of the grid anisotropy and boundary layers

Consider a velocity with a boundary layer near a wall

$$\begin{array}{ll} \vec{a} = 3 - (300 * (x_2 - 0.1)^2)\vec{e}_1 & \text{if } x_2 < 0.1 \\ \vec{a} = 3\vec{e}_1 & \text{if } x_2 \geq 0.1 \end{array} \tag{16}$$

To capture the boundary layer, the mesh is refined in the x_2-direction, near the wall $x_2 = 0$ with a geometric progression of ratio 0.9.

Partition		4×1	8×1	12×1	24×1	36×1
Grid		20×40	20×40	20×40	20×40	20×40
$\Delta t = 1$	R-R	11	18	25	39	51
$\nu = 0.001$	–	51	75	91	> 100	> 100
	N-N	49	> 100	> 100	> 100	> 100

Table 3 : Anisotropic grids

In comparison with the tests of § 1, we see that the performances deteriorate due to the change of velocity and grid, but is is very clear that the Robin-Robin method is the less severely affected.

Rotating velocity: influence of the number of unknowns

The domain is the unit square, which is partitioned into 4×4 subdomains. In each subdomain, the grid varies from 20×20 up to 60×60. The velocity is $\vec{a} = \vec{e_3} \times (\vec{x} - \vec{x_0})$, ($x_0$ is the center of Ω) and $\nu = 0.001$, $\Delta t = 10^7$ (almost a steady state computation). A coarse space solver of BPS type is used here, but will not be discussed.

	Partition	4×4	4×4	4×4	4×4
	Grid	20×20	30×30	40×40	60×60
$\Delta t = 10^7, \nu = 0.001$	R-R	34	34	34	34
$\vec{a} = \vec{e}_3 \times (\vec{x} - \vec{x_0})$	–	80	82	89	> 100

Table 4: Influence of the number of grid points

The convergence is not affected by the grid size.

Results in three dimensions

In Table 5, we give results for a simple geometry with different partitions and velocity fields. In Table 6, the domain $[0,1] \times [0,1] \times [0,.01]$ is decomposed into 100 subdomains. The number of elements is $60,000$ and the number of nodes is $121,203$. We compare the Robin-Robin with and without coarse grid preconditioner.

Ω	$[0,1]^2 \times [0,0.5]$	$[0,1]^2 \times [0,0.33]$	$[0,1]^3$
Partition	$2 \times 2 \times 1$	$3 \times 3 \times 1$	$3 \times 3 \times 3$
$\vec{a} = (y/.5 - .5, -x/.5 + 5.0, 0)$, $\nu = 10^{-3}$, $c = 10^{-4}$			
R-R	12	21	35
N-N	24	51	165
$\vec{a} = 300 * \min(x_2^2, 0.01)\ \vec{e}_1$, $\nu = 10^{-2}$, $c = 10^{-4}$			
R-R	9	13	19
N-N	23	65	92
$\vec{a} = 3 - 300 * (\min(x_2, 0.1) - 0.1)^2\ \vec{e}_1$, $\nu = 10^{-2}$, $c = 10^{-4}$			
R-R	9	16	25
N-N	37	88	> 200

Table 5 : Iteration Count for different parameters

velocity	ν	c	R-R	R-R + coarse grid
rotating	10^{-2}	10^{-7}	51	54
null	10^{-2}	10^{-7}	78	28

Table 6: Influence of the coarse grid

Figure 2. shows the unit cube containing 24576 tetrahedric second order finite elements split into 45 subdomains by an automatic mesh partitioner. This is why the

boundaries between subdomains are less regular than for the 2D computations. For this decomposition the algorithm converges in 48 iterations with the R-R preconditioner. The last numerical example aims to be a more realistic computation : the diffusion of a pollutant in a fluid contained in a T-shape reservoir. The fluid is incompressible. For such a model if the viscosity is small, the advection is dominant, on the opposite if the viscosity increases the convection phenomena is more important than the diffusion. If, for the flow, the given initial velocity has a parabolic profile and, in the advection diffusion step, the pollutant concentration given on a boundary is linear the convection and diffusion phenomena are coupled in the hole domain, a boundary layer is expected in the large part ot the T-shape domain and a vortex region in the bottom of the domain.

The mesh considered here is too coarse to expect realistic results, it contains 29,448 elements and 43,937 nodes. The initial domain was split in 50 subdomains and the velocity field was obtained by a Stokes computation. This initial computation is in fact more expensive than the advection-diffusion one and this was the limiting point. It was performed on the same grid and same decomposition using a Neumann-Neumann domain decomposition algorithm. In this case the number of degrees of freedom is 131,811.

On this advection-diffusion problem, the Robin-Robin preconditioner converged in 81 iterations (compared to 78 iterations for an elasticity problem on the same grid). Looking at the level lines of the concentration on a transversal section (figure 3) one can see that the advection diffusion phenomena corresponds to the given boundary conditions and the vortex region appears. In addition the values are small were the boundary layer is expected. The results obtained are thus concordant with the prediction.

Optimized Schwarz algorithm

We consider in the sequel a very different type of algorithm, namely, the classical Schwarz algorithm in a simple case: the plane $\mathbf{R}^2$ is divided into two subdomains $\Omega_1 = (-\infty, \delta) \times \mathbf{R}$ and $\Omega_2 = (0, \infty) \times \mathbf{R}$. Numerical results will be given for general decompositions. The size of the overlap is $\delta \geq 0$. In order to solve (2), the classical additive Schwarz method is based on the use of Dirichlet boundary conditions

$$\mathcal{L}(u_i^{k+1}) = f \text{ in } \Omega_\mathrm{i}, \mathrm{u}_\mathrm{i}^{\mathrm{k}+1} = \mathrm{u}_{3-\mathrm{i}}^{\mathrm{k}}. \tag{17}$$

A Fourier analysis shows that

$$\hat{u}(x,\xi) - \hat{u}_i^{k+1}(x,\xi) = e^{-\sqrt{\frac{4\nu}{\Delta t/\nu} + a_x^2 + 4I a_y \xi\nu + 4\xi^2\nu^2}\,\delta} (\hat{u}(x,\xi) - \hat{u}_i^{k-1}(x,\xi)) \tag{18}$$

and overlap is necessary for convergence. For a small overlap ($\delta \ll 1$), the convergence will be very slow. In order to remedy to this situation, it has been proposed to use more general interface conditions. The algorithm reads

$$\mathcal{L}(u_i^{k+1}) = f \text{ in } \Omega_\mathrm{i}, \mathcal{B}_\mathrm{i}(\mathrm{u}_\mathrm{i}^{\mathrm{k}+1}) = \mathcal{B}_\mathrm{i}(\mathrm{u}_{3-\mathrm{i}}^{\mathrm{k}}). \tag{19}$$

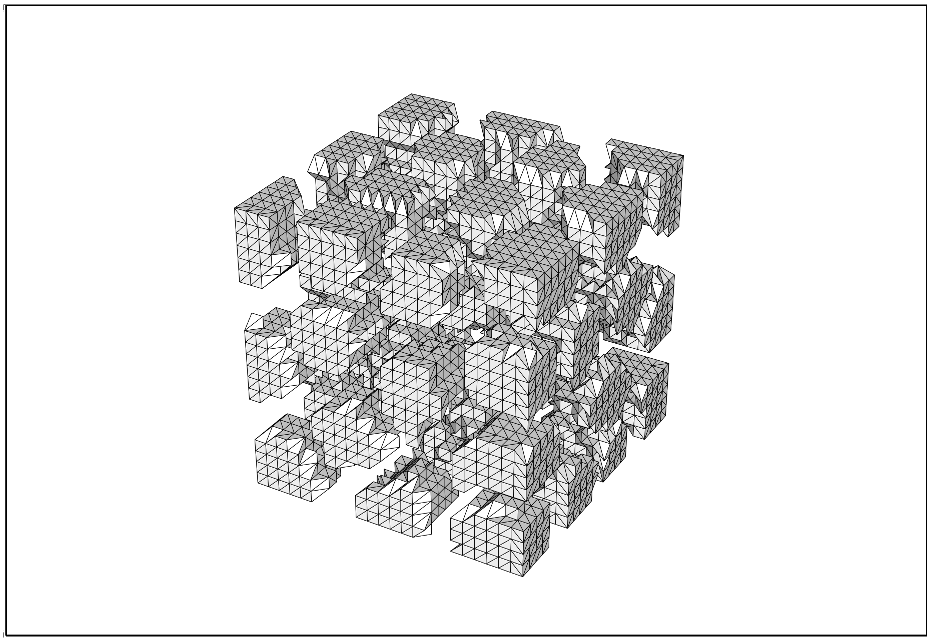


Figure 2 Three dimensional triangulation and automatic decomposition into 45 subdomains

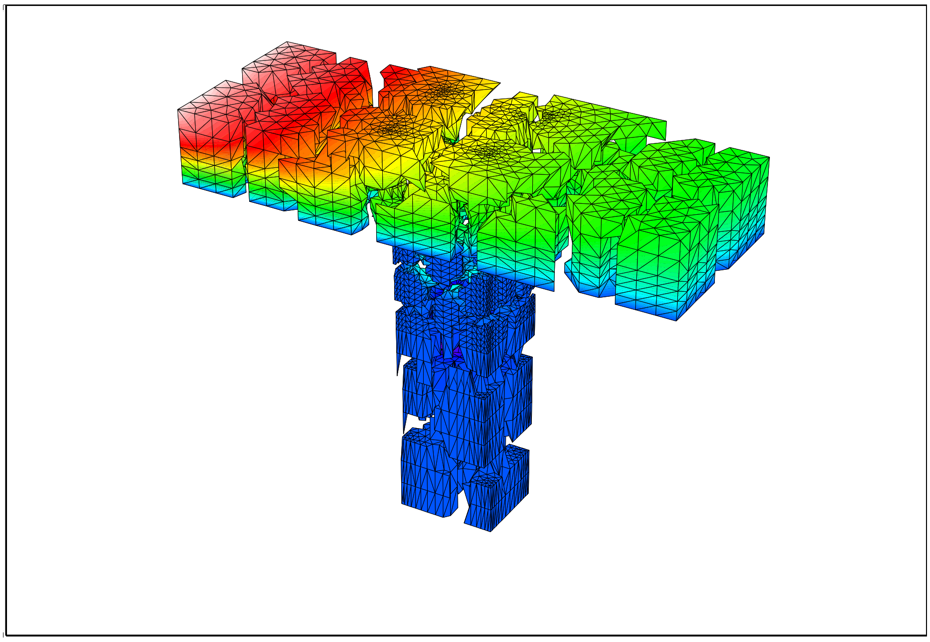


Figure 3 Three dimensional triangulation and automatic decomposition into 50 subdomains

where $\mathcal{B}_i(u) = \nu \frac{\partial u}{n_i} - \mathcal{F}_\xi^{-1}(\lambda_i(\xi)\hat{u}(x,\xi))$. A Fourier analysis shows that (see [NR94])

$$\hat{u}(x,\xi) - \hat{u}_i^{k+1}(x,\xi) = \frac{\lambda_1^e(\xi) - \lambda_1(\xi)}{\lambda_2^e(\xi) + \lambda_1(\xi)} \frac{\lambda_2^e(\xi) - \lambda_2(\xi)}{\lambda_1^e(\xi) + \lambda_2(\xi)} \tag{20}$$

$$\times\ e^{-\sqrt{\frac{4\nu}{\Delta t} + a_x^2 + 4Ia_y\xi\nu + 4\xi^2\nu^2}\,\delta/\nu} (\hat{u}(x,\xi) - \hat{u}_i^{k-1}(x,\xi)) \tag{21}$$

where

$$\lambda_1^e(\xi) = \frac{a_x + \sqrt{\frac{4\nu}{\Delta t} + a_x^2 + 4Ia_y\xi\nu + 4\xi^2\nu^2}}{2}, \tag{22}$$

$$\lambda_2^e(\xi) = \frac{-a_x + \sqrt{\frac{4\nu}{\Delta t} + a_x^2 + 4Ia_y\xi\nu + 4\xi^2\nu^2}}{2}. \tag{23}$$

If $\lambda_i = \lambda_i^e$, we have convergence in two steps (see [HTJ88] or [FNdS94]). These optimal interface conditions are also exact absorbing boundary conditions which are used for truncating infinite domains. The fact that these interface conditions are optimal for domain decomposition methods is quite general since it applies to variable coefficient operators and more general decompositions as well, see [FNdS94]. In the case of the Laplace operator, their use had also been suggested in [Lio90]. Because of the square root in formulas (22) and (23), these boundary operators are not not differential operators. They are of integro-differential (or pseudo-differential) type and thus difficult to implement numerically. As it is the case when they are used for truncating infinite domains, they are approximated by partial differential operators. In [DJR92], and then in [Des93], [BD97], [GGQ96] [dLBFM+98], [MSRKA98] and [CCEW98], the lowest order approximations used for the truncation of domains are used as interface conditions for domain decomposition domains. It amounts to taking $\lambda_i = \lambda_i^e(0)$ (*Taylor approximation at order 0*). In [NR94] higher order approximations are used for the convection-diffusion equation, namely *Taylor approximation at order 2* of $\lambda_i^e(\xi)$ in the vicinity of $\xi = 0$.

Optimized of Order 2 (OO2) Interface Conditions

A more suited and efficient possibility is to optimize the choice of the interface conditions $\mathcal{B}_i$ with respect to the convergence of the domain decomposition method (see [TB94], [EZ98] for the operator $\eta - \Delta$, and [Jap97] for the convection-diffusion equation and [CN98] for the wave equation). For ease of implementation, $\mathcal{B}_i$ is sought in the form

$$\mathcal{B}_i = \nu \frac{\partial}{\partial n_i} + \alpha_i + \beta_i \frac{\partial}{\partial \tau_i} - \gamma_i \frac{\partial^2}{\partial \tau_i^2}. \tag{24}$$

The parameters are chosen in order to minimize the maximum of the convergence rate over all the wavenumbers which can be represented on the computational grid, $|\xi| \leq 1/h$ (h is the typical mesh size):

$$\min_{\alpha_1,\beta_1,\gamma_1,\alpha_2,\beta_2,\gamma_2} \max_{|\xi|\leq 1/h} \rho(\xi,\alpha_1,\beta_1,\gamma_1,\alpha_2,\beta_2,\gamma_2) \tag{25}$$

where

$$\rho(\xi,\alpha_1,\beta_1,\gamma_1,\alpha_2,\beta_2,\gamma_2) = \left|\frac{\lambda_1^e(\xi) - (\alpha_1 + \beta_1 I\xi + \gamma_1\xi^2)}{\lambda_2^e(\xi) + (\alpha_1 + \beta_1 I\xi + \gamma_1\xi^2)}\right| \times \left|\frac{\lambda_2^e(\xi) - (\alpha_2 - \beta_2 I\xi + \gamma_2\xi^2)}{\lambda_1^e(\xi) + (\alpha_2 - \beta_2 I\xi + \gamma_2\xi^2}\, e^{-\sqrt{\frac{4\nu}{\Delta t}+a_x^2+4I a_y\xi\nu+4\xi^2\nu^2}\delta/\nu}\right| \tag{26}$$

This minimization problem is difficult since it is not differentiable nor convex. It can be proved that choosing $\alpha_i, \gamma_i \geq 0$, $\alpha_2 - \alpha_1 = a_x$, $\beta_1 = -\beta_2$, $\gamma_1 = \gamma_2$ and $sgn(\beta_1) = -sgn(\beta_2) = sgn(a_y)$ is sufficient for having $|\rho| < 1$. Moreover, when $a_y = 0$, the min-max problem (25) is equivalent to a 1D minimization problem over a wavenumber ξ_{int} such that $\rho(\xi_{int}) = 0$. In the general case $a_y \neq 0$ this last procedure is used, see [Jap97]. In the limit $h \to 0$, a theoretical estimate shows that for the Taylor interface conditions at order 0 or 2 , $\rho \simeq 1 - C^t|a|h/\nu$ while for the OO2 interface condition $\rho \simeq 1 - C^t(|a|h/\nu)^{1/3}$. For a variable coefficient operator, a frozen coefficient approximation is used for computing the parameters of the interface condition.
In order to improve the convergence, the iterative algorithm (19) is replaced by a Krylov type method applied to the substructured problem with $\mathcal{B}_i(u)$ as unknowns on the interfaces, see [NR94].

Numerical Results for the OO2 method

Dirichlet, Taylor of order zero, Taylor of order 2 and OO2 interface conditions are compared. A BICGSTAB algorithm is used for the substructured problem. The stopping criterion was the maximum error between the converged solution and the iterative solution to be smaller than 10^{-6}. The problems in the subdomains are solved by a direct method.
We first consider an upwind finite difference scheme with a small overlap of size h. The time step is taken very large ($\Delta t = 10^9$) so that it corresponds to a stationary equation. For a decomposition into 16×1 subdomains, we have the following iteration count for two velocity fields: one normal to the interface and one tangential to the interface:

Nb of iterations	**Dirichlet**	**Taylor 0**	**Taylor 2**	**002**
$a_y = 0,\ a_x = y$	60	137	15	15
$a_x = 0,\ a_y = x$	60	90	265	9

Table 5: Overlapping subdomains - $\nu = 10^{-2}$, $h = 1/241$

We see that the OO2 interface conditions lead to the fastest algorithms.
We now consider a finite volume discretization with no overlap between the subdomains ($\delta = 0$). The Dirichlet interface conditions cannot be used anymore. For the same parameters as in Table 5, we have

Nb of iterations	**Taylor 0**	**Taylor 2**	**002**
$a_y = 0,\ a_x = y$	140	122	18
$a_x = 0,\ a_y = x$	85	divergence	20

Table 6: Non Overlapping subdomains - $\nu = 10^{-2}$, $h = 1/241$

Once more, the OO2 interface conditions give the best results. We now focus on the robustness of the algorithms which are illustrated by the following tables. These results can be related to the estimate of § 1.

Nb of points	65×65	129×129	241×241
OO2	25	26	30
Taylor 0	76	130	224

Table 7: Non Overlapping subdomains– **4** by **4** subdomains–rotating velocity field

$\vert a\vert\Delta t/h$	10^0	10^3	10^5	10^9
OO2	3	12	15	15
Taylor 0	3	18	48	141
Taylor 2	2	21	58	123

Table 8 Non Overlapping subdomains– **16** by **1** subdomains – boundary layer velocity field

When the operator degenerates to a Laplace operator, a coarse grid preconditioner has to be used in order to keep the robustness of the method, see [JNR98].

Conclusion

We have presented two very different methods which are adapted to non symmetric scalar problems. It seems to us that they perform essentially equally well (see Tables 4 and 7) although a thorough study should be made. Through its variational formulation (see (10), the Robin-Robin preconditioner can be easily implemented in a FEM. As for the OO2 approach, its extension to other type of equations seems more easily feasible. An interesting perspective is the extension of both approaches to systems of equations.

REFERENCES

[Ach95] Achdou Y. (1995) The mortar element method for convection diffusion problems. *C. R. Acad. Sci. Paris Sér. I Math.* 321(1): 117–123.

[AN97] Achdou Y. and Nataf F. (1997) A robin-robin preconditioner for an advection-diffusion problem. *C.R. Acad. Sci. Série I* 325: 1211–1216.

[ATNV] Achdou Y., Tallec P. L., Nataf F., and Vidrascu M.A domain decomposition preconditioner for an advection-diffusion problem. to apper.

[BD97] Benamou J.-D. and Desprès B. (1997) A domain decomposition method for the Helmholtz equation and related optimal control problems. *J. Comput. Phys.* 136(1): 68–82.

[BGLTV89] Bourgat J.-F., Glowinski R., Le Tallec P., and Vidrascu M. (1989) Variational formulation and algorithm for trace operator in domain decomposition calculations. In Chan T., Glowinski R., Périaux J., and Widlund O. (eds) *Domain Decomposition Methods*, pages 3–16. SIAM, Philadelphia, PA.

[BMP94] Bernardi C., Maday Y., and Patera A. T. (1994) A new non conforming approach to domain decomposition: The mortar element method. In Brezis H. and Lions J.-L. (eds) *Collège de France Seminar*. Pitman. This paper appeared as a technical report about five years earlier.

[CCEW98] Cai X.-C., Casarin M. A., Elliott Frank W. J., and Widlund O. B. (1998) Overlapping Schwarz algorithms for solving Helmholtz's equation. In *Domain decomposition methods, 10 (Boulder, CO, 1997)*, pages 391–399. Amer. Math. Soc., Providence, RI.

[CN98] Chevalier P. and Nataf F. (1998) Symmetrized method with optimized second-order conditions for the Helmholtz equation. In *Domain decomposition methods, 10 (Boulder, CO, 1997)*, pages 400–407. Amer. Math. Soc., Providence, RI.

[Des93] Després B. (1993) Domain decomposition method and the Helmholtz problem. II. In *Second International Conference on Mathematical and Numerical Aspects of Wave Propagation (Newark, DE, 1993)*, pages 197–206. SIAM, Philadelphia, PA.

[DJR92] Després B., Joly P., and Roberts J. E. (1992) A domain decomposition method for the harmonic Maxwell equations. In *Iterative methods in linear algebra (Brussels, 1991)*, pages 475–484. North-Holland, Amsterdam.

[dLBFM+98] de La Bourdonnaye A., Farhat C., Macedo A., Magoulès F., and Roux F.-X. (1998) A non-overlapping domain decomposition method for exterior Helmholtz problem. In *Domain decomposition methods, 10 (Boulder, CO, 1997)*, pages 42–66. Amer. Math. Soc., Providence, RI.

[EZ98] Engquist B. and Zhao H.-K. (1998) Absorbing boundary conditions for domain decomposition. *Appl. Numer. Math.* 27(4): 341–365. Absorbing boundary conditions.

[FMR94] Farhat C., Mandel J., and Roux F.-X. (1994) Optimal convergence properties of the FETI domain decomposition method. *Comput. Methods Appl. Mech. Engrg.* 115(3-4): 365–385.

[FNdS94] F. Nataf F. R. and de Sturler E. (1994) Optimal interface conditions for domain decomposition methods. Technical Report 301, CMAP (Ecole Polytechnique).

[GGQ96] Gastaldi F., Gastaldi L., and Quarteroni A. (1996) Adaptive domain decomposition methods for advection dominated equations. *East-West J. Numer. Math.* 4(3): 165–206.

[HTJ88] Hagstrom T., Tewarson R. P., and Jazcilevich A. (1988) Numerical experiments on a domain decomposition algorithm for nonlinear elliptic boundary value problems. *Appl. Math. Lett.* 1(3): 299–302.

[Jap97] Japhet C. (1997) Conditions aux limites artificielles et décomposition de domaine: Méthode oo2 (optimisé d'ordre 2). application à la résolution de problèmes en mécanique des fluides. Technical Report 373, CMAP (Ecole Polytechnique).

[JNR98] Japhet C., Nataf F., and Roux F.-X. (1998) Extension of a coarse grid preconditioner to non-symmetric problems. In *Domain decomposition methods, 10 (Boulder, CO, 1997)*, pages 279–286. Amer. Math. Soc., Providence, RI.

[Lio90] Lions P. L. (1990) On the Schwarz alternating method. III: a variant for nonoverlapping subdomains. In Chan T. F., Glowinski R., Périaux J., and Widlund O. (eds) *Third International Symposium on Domain Decomposition Methods for Partial Differential Equations , held in Houston, Texas, March 20-22, 1989.* SIAM, Philadelphia, PA.

[MSRKA98] McInnes L. C., Susan-Resiga R. F., Keyes D. E., and Atassi H. M. (1998) Additive Schwarz methods with nonreflecting boundary conditions for the parallel computation of Helmholtz problems. In *Domain decomposition methods, 10 (Boulder, CO, 1997)*, pages 325–333. Amer. Math. Soc., Providence, RI.

[NR94] Nataf F. and Rogier F. (1994) Outflow boundary conditions and domain decomposition method. In *Domain decomposition methods in scientific and engineering computing (University Park, PA, 1993)*, pages 289–293. Amer. Math. Soc., Providence, RI.

[TB94] Tan K. H. and Borsboom M. J. A. (1994) On generalized Schwarz coupling applied to advection-dominated problems. In *Domain decomposition methods in scientific and engineering computing (University Park, PA, 1993)*, pages 125–130. Amer. Math. Soc., Providence, RI.

2

A Multigrid Method for the Complex Helmholtz Eigenvalue Problem

Tilmann Friese[1], Peter Deuflhard[2], Frank Schmidt[3]

Introduction

The paper deals with the solution of the eigenvalue problem of the complex Helmholtz equation. We present an adaptive multigrid method for solving the nonselfadjoint algebraic eigenproblem arising from discretization with finite elements. A technological relevant numerical example, the simulation of an integrated optical component containing Multi Quantum Well layers, is included.

The task is to find a few eigenvalues λ and corresponding eigenfunctions u of the Helmholtz equation with Dirichlet boundary condition

$$\begin{array}{rll} -\Delta u(x,y) - f(x,y)\,u(x,y) = \lambda\, u(x,y), & (x,y) \in \Omega \\ u(x,y) = 0, & (x,y) \in \partial\Omega \end{array} ,$$

where the region Ω is an open, bounded, and connected subset of $\mathbf{R}^2$, and the function f is bounded and in general complex valued. As usual, we transform this problem in its weak formulation: determine $u \in H_0^1(\Omega) \setminus \{0\}$ and $\lambda \in \mathbf{C}$, such that the relation

$$a(v,u) = \lambda\,(v,u) \quad \forall v \in H_0^1(\Omega) \tag{1}$$

1 Konrad-Zuse-Zentrum für Informationstechnik Berlin, Takustr. 7, D-14195 Berlin, Germany, E-mail: friese@zib.de. Research supported by the BMBF under grant 03-DE7ZIB-5.

2 Address same as above, E-mail: deuflhard@zib.de.
3 Address same as above, E-mail: frank.schmidt@zib.de.

Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

with the sesquilinear form $a(v,u) = (\nabla v, \nabla u) - (v, fu)$ is fulfilled. The inner product $(\cdot,\cdot)$ is herein given by the L^2 scalar product. In the special case of a pure real valued function f the eigenvalue problem (1) is selfadjoint. In the general case of a complex valued function f we have to solve a nonselfadjoint eigenproblem. Finally, in many cases the eigenvalues with lowest real parts and the corresponding eigenfunctions are of interest.

By using the Rayleigh quotient

$$R(u) = \frac{a(u,u)}{(u,u)}$$

we can conclude from a simple computation [Sch95] that the eigenvalues λ lie in a half stripe Σ of the complex plane as shown in Figure 1. Furthermore, from a general

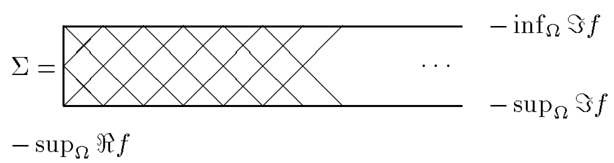


Figure 1 Position of the eigenvalues in the complex plane.

completeness result of Katsnelson [Kat67] we obtain the following

Theorem 1. *Let the region Ω and the function f be given as above. Then the spectrum of the associated Helmholtz operator is discrete, and there exists a basis $\{u_j\}_{j=1}^{\infty}$ of $L^2(\Omega)$, such that the relation*

$$a(v,u_j) = \lambda_j\,(v,u_j) + \sum_{k=1}^{j-1} \tau_{kj}\,(v,u_k) \quad \forall v \in H_0^1(\Omega) \tag{2}$$

holds. The eigenvalues fulfill the inequalities $\Re\lambda_1 \le \Re\lambda_2 \le \ldots \to \infty$.

Remark 1. In the special selfadjoint case the Courant minimax principle [CH53] shows that this basis is even orthonormal. In addition, the values τ_{kj} are zero, i. e., the basis consists only of eigenfunctions.

The statements of Theorem 1 enable us to carry out a perturbation analysis as for matrix eigenvalue problems, see, e. g., [GV89]. It turns out that the sensitivity of the eigensolutions with respect to perturbations in f depends on the spectral gap, i. e., on the separation of the eigenvalues from the remaining part of the spectrum. We can conclude that we should determine the interesting eigensolutions in terms of invariant subspaces.

The discretization of (2) with finite elements leads to the generalized matrix eigenvalue problem

$$AU = BUT \tag{3}$$

with the sparse system matrix A, the sparse, selfadjoint, and positive definite mass matrix B, and the unknown matrices U and T. The column vectors of the matrix U represents a basis of the interesting discrete invariant subspace. From the partial Schur decomposition of matrices [GV89] we see that the matrix U may be chosen orthonormal, i. e., $U^*BU = I$, and that the matrix T is upper triangular with the discrete eigenvalues in the diagonal. It can be shown [Fri98] that the discrete eigenvalues lie also in the half stripe Σ of the complex plane. Furthermore, in the selfadjoint case the upper triangular matrix T reduces to a diagonal one. Hence the discrete problem (3) reflects all properties of the continuous problem.

A Multigrid Method for the Helmholtz Eigenproblem

In this section we present a multigrid algorithm for the efficient solution of the discrete eigenproblem (3). We restrict ourselves to conforming methods, i. e., we assume a nested sequence of finite element spaces

$$S_0 \subset S_1 \subset \ldots \subset S_{l_{\max}} \subset H_0^1(\Omega) \quad .$$

We begin with the description of the multigrid principle for the special selfadjoint case. The starting point of our considerations is the following minimal property of the invariant subspace corresponding to the q lowest eigenvalues: the matrix U is given as the minimal point

$$R(U) = \min_V R(V)$$

of the generalized Rayleigh quotient $R(V) = \operatorname{trace}\big((V^*BV)^{-1}(V^*AV)\big)$, where the matrix V has q columns and full rank. This characterization leads to the following general algorithmic idea: given an approximation $\widetilde{U}$ of U and search directions $\widetilde{P}$, determine a new approximation $\widehat{U}$ by minimization of the generalized Rayleigh quotient over the subspace spanned by the column vectors of the matrices $\widetilde{U}$ and $\widetilde{P}$, i. e., by

$$R(\widehat{U}) = \min_{\binom{\Phi}{\Psi}} R(\widetilde{U}\Phi + \widetilde{P}\Psi) \quad , \tag{4}$$

where the matrix $\binom{\Phi}{\Psi}$ has q columns and full rank. If we iterate this procedure we obtain an algorithm which traces the solution of the original large scale problem back to the successive solution of low dimensional problems of the same type. This general method produces a sequence of invariant subspace approximations with decreasing functional values, i. e., it is a monotone method.

From this general principle we can derive a multigrid algorithm for the selfadjoint case. The resulting method is a generalization of the multigrid minimization of Mandel, McCormick [MM89] to invariant subspace computations. In the smoothing step we apply a classical Rayleigh quotient minimization method such as, e. g., a simultaneous gradient [LM80] or conjugate gradient method [Döh82]. This means, the matrix $\widetilde{P}$ of search directions is formed by the gradient or by a generalized conjugate gradient of the Rayleigh quotient. The numerical experience shows that these algorithms have a smoothing property. Furthermore, in the coarse grid correction step the matrix $\widetilde{P}$ is given by the usual prolongation matrix which describes the transition between the coarse and the fine grid. Hence both parts of the multigrid algorithm are based on the above described minimization principle which implies the monotonicity of the method. This property ensures a high numerical robustness of the algorithm.

Remark 2. Optimal complexity results for a variant of the described method with a different smoothing procedure were given by McCormick [McC94] and Cai, Mandel, McCormick [CMM97]. A different proof of optimality has been suggested by Chan, Sharapov [CS96] in connection with domain decomposition techniques. A numerical comparison between the multigrid minimization and the multigrid method of Hackbusch [Hac85] in the context of adaptive generated meshes may be found in [DFS+96]. It turns out that, on one hand, the above described method is more robust, and, on the other hand, the two methods have asymptotically the same convergence rates.

For the generalization of the above method to the nonselfadjoint case we use the fact that the solution of the minimization problem (4) is equivalent to the solution of an eigenvalue problem

$$\widetilde{A}\Theta = \widetilde{B}\Theta\widehat{\Lambda}$$

with the projected matrices

$$\widetilde{A} = \begin{pmatrix} \widetilde{U} & \widetilde{P} \end{pmatrix}^* A \begin{pmatrix} \widetilde{U} & \widetilde{P} \end{pmatrix} \quad \text{and} \quad \widetilde{B} = \begin{pmatrix} \widetilde{U} & \widetilde{P} \end{pmatrix}^* B \begin{pmatrix} \widetilde{U} & \widetilde{P} \end{pmatrix} \quad .$$

The essential idea for the extension of the multigrid minimization to nonselfadjoint problems is to replace these projected eigenvalue problems in each smoothing and coarse grid correction step by projected Schur problems

$$\widetilde{A}\Theta = \widetilde{B}\Theta\widehat{T} \quad .$$

By this modification, we obtain the following smoothing algorithm which resembles a Block Arnoldi method [Saa92].

Algorithm 1.

- Initialization:
  - ∗ given a $(N \times q)$-matrix U with $U^*BU = I$, $U^*AU = T_U$
  - ∗ set $P = R = -(AU - BUT_U)$
- Iteration:
  - ∗ set $V = \begin{pmatrix} U & P \end{pmatrix}$
  - ∗ solve
  $$\begin{aligned} (V^*AV)\Theta &= (V^*BV)\Theta T \\ \Theta^*(V^*BV)\Theta &= I \end{aligned}$$
  with $\Re(\lambda_1) \leq \ldots \leq \Re(\lambda_q) < \Re(\lambda_{q+1}) \leq \ldots \leq \Re(\lambda_{2q})$
  - ∗ set $\begin{pmatrix} U & P \end{pmatrix} = V\Theta$
  - ∗ compute $R = -(AU - BUT_U)$

* determine X from

$$XT_U - T_P X = P^*(AR - BRT_U)$$

with $T_P = P^*AP$
* set $P = R + PX$

A detailed description of this procedure is given in [DFS97, Fri98]. Again, the numerical experience shows that this method has a smoothing property. Furthermore, as above, the search directions in the coarse grid correction step are given by the usual prolongation matrix. Since, in particular, the arising coarse grid problems are of the same type as the original problem (3), a recursive application of a multigrid procedure as given in Algorithm 2 is possible.

Algorithm 2.

$[U_l, T_l] = \mathrm{MGM}(A_l, B_l, U_l, T_l, l)$

1. presmoothing using Algorithm 1: $U_l \to \widetilde{U}_l$, $T_l \to \widetilde{T}_l$
2. coarse grid correction: $\widetilde{U}_l \to \widehat{U}_l$, $\widetilde{T}_l \to \widehat{T}_l$
   - compute $A_{l-1} = V_l^* A_l V_l$ and $B_{l-1} = V_l^* B_l V_l$, where in case
     * $l = l_{\max}$: $V_l = \begin{pmatrix} \widetilde{U}_l & P_l \end{pmatrix}$
     * $l < l_{\max}$: $V_l = \begin{pmatrix} \widetilde{U}_l & \begin{matrix} 0 \\ P_l \end{matrix} \end{pmatrix}$
   - if
     * $l > 1$: $[U_{l-1}, T_{l-1}] = \mathrm{MGM}(A_{l-1}, B_{l-1}, \begin{pmatrix} I \\ 0 \end{pmatrix}, \widetilde{T}_l, l-1)$
     * $l = 1$: solve

$$\begin{aligned} A_0 U_0 &= B_0 U_0 T_0 \\ U_0^* B_0 U_0 &= I \end{aligned}$$

   - set $\widehat{U}_l = V_l U_{l-1}$, $\widehat{T}_l = T_{l-1}$
3. postsmoothing using Algorithm 1: $\widehat{U}_l \to U_l$, $\widehat{T}_l \to T_l$

In Algorithm 2, the matrices A_l and B_l are the (augmented) system and mass matrix corresponding to the finite element space S_l. The matrix U_l with q columns and the upper triangular matrix T_l are the matrices of unknowns. The matrix P_l is the prolongation matrix for the interpolation between the spaces S_l and S_{l-1}.

Remark 3. Note, that this multigrid method for the nonselfadjoint case is not based on a minimization principle. An elaborate theoretical investigation of the method, especially a proof of optimality, is an open topic for future research. However, numerical experiments as presented in the following section suggest the optimal complexity of the algorithm.

Numerical Example

For the illustration of the performance of the above described multigrid algorithm we have computed the guided modes of an integrated optical component which is invariant in z-direction. The cross section of the interesting structure is drawn in Figure 2. The

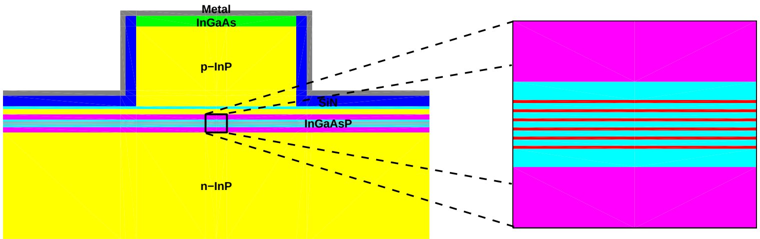


Figure 2 Cross section of an integrated optical structure with zoom.

function f can be represented as $f(x,y) = k_0^2 n^2(x,y)$, where k_0 is the vacuum wave number of light, and the function $n(x,y)$ is given by the refractive indexes of the materials. The refractive index is complex valued and parameter dependent in the Multi Quantum Well (MQW) layers (the narrow stripes in the zoom of Figure 2), complex valued in the metal layer, and real valued otherwise. The exact parameters of the structure are technologically relevant and therefore not documented here.

For the construction of the hierarchy of meshes we used a nested iteration technique in connection with a triangle based error indicator. At first, for each triangle t_j a relative L^2 error ε_j with respect to the whole invariant subspace was determined by computation of a correction in the edge midpoints of the current triangulation with a scheme analogously to the Jacobi method for linear systems. Then, the triangles t_k with

$$\varepsilon_k > \frac{1}{2} \max \varepsilon_j$$

were refined uniformly, and neighbouring triangles were divided by a bisection method to remove hanging nodes.

In a first experiment, we were interested in the two eigenvalues with lowest real parts and their corresponding Schur functions. A difficulty of this problem was to find a proper starting triangulation. Since we must resolve the very thin MQW layers with normally sized triangles, the coarsest mesh (shown in Figure 3) consists already of 2515 nodes and 4956 triangles. The final triangulation with 15246 nodes and 30396 triangles reached after 6 refinements is also given in Figure 3. The numerical experiments have been performed with a tolerance $\mathrm{tol} = 10^{-3}$ for all multigrid levels. The whole computation with a MATLAB program took 16 minutes on a SUN ULTRA 1 workstation. The coarsest grid problems were solved with a Block Arnoldi procedure [Saa92]. Logarithmic contour plots for the resulting Schur functions on the final mesh are represented in Figure 4. The convergence history of the full multigrid run is given

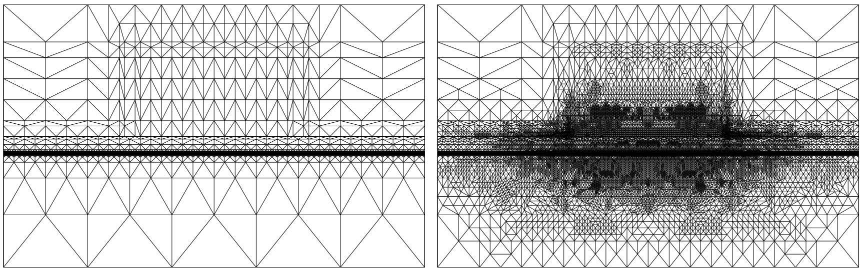


Figure 3 Starting and final triangulation.

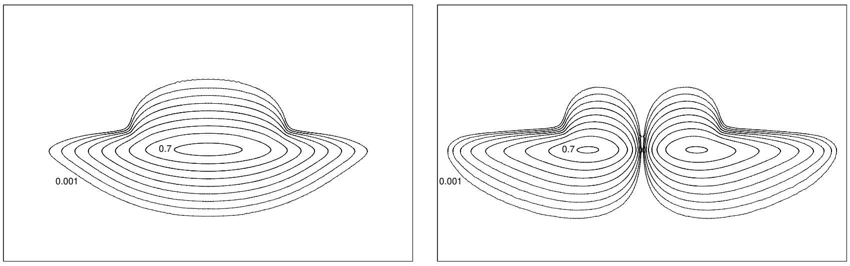


Figure 4 Logarithmic contour plots of $|u_1|^2$ and $|u_2|^2$.

Table 1 Number of inner points and number of multigrid iterations per grid.

Grid	1	2	3	4	5	6
Inner points	2564	2888	3987	5536	9422	15152
MG Iterations	3	3	4	4	4	4

in Table 1.

In a second experiment, we carried out a parameter study to demonstrate the dependence of the eigenvalues on the imaginary part of the refractive index in the MQW layers. We computed the invariant subspace corresponding to the four eigenvalues with lowest real parts for the parameter values $\alpha = 0, 0.25, 0.5, 0.75, 1$. The computed eigenvalue trajectories are shown in Figure 5. The two eigenvalues from the previous computation are marked with arrows. This type of diagram is of technological interest.

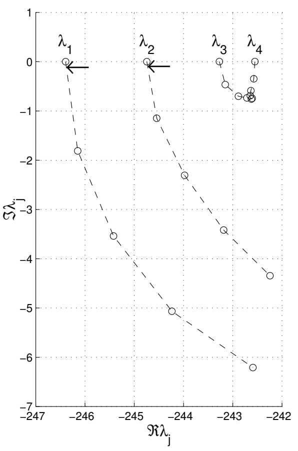


Figure 5 Dependence of the eigenvalues on the imaginary part of the refractive index in the Multi Quantum Well layers.

Acknowledgment. The authors wish to thank Dr. Reinhard März from Siemens AG, Corporate Research and Development, Munich for helpful discussions and a stimulating cooperation.

REFERENCES

[CH53] Courant R. and Hilbert D. (1953) *Methods of Mathematical Physics*, volume 1. Wiley Interscience New York.

[CMM97] Cai Z., Mandel J., and McCormick S. F. (1997) Multigrid methods for nearly singular linear equations and eigenvalue problems. *SIAM J. Numer. Anal.* 34: 178–200.

[CS96] Chan T. F. and Sharapov I. (1996) Subspace correction multilevel methods for elliptic eigenvalue problems. CAM Report 96-34, University of California Los Angeles.

[DFS+96] Deuflhard P., Friese T., Schmidt F., März R., and Nolting H.-P. (1996) Effiziente Eigenmodenberechnung für den Entwurf integriert-optischer Chips. In Hoffmann K.-H., Jäger W., Lohmann T., and Schunck H. (eds) *Mathematik-*

Schlüsseltechnologie für die Zukunft, pages 267–280. Springer-Verlag Berlin Heidelberg New York.

[DFS97] Deuflhard P., Friese T., and Schmidt F. (1997) A nonlinear multigrid eigenproblem solver for the complex Helmholtz equation. Preprint SC 97-55, Konrad-Zuse-Zentrum für Informationstechnik Berlin.

[Döh82] Döhler B. (1982) Ein neues Gradientenverfahren zur simultanen Berechnung der kleinsten oder größten Eigenwerte des allgemeinen Eigenwertproblems. *Numer. Math.* 40: 79–91.

[Fri98] Friese T. (1998) *Eine Mehrgitter-Methode zur Lösung des Eigenwertproblems der komplexen Helmholtzgleichung.* PhD thesis, Freie Universität Berlin.

[GV89] Golub G. H. and Van Loan C. F. (1989) *Matrix Computations.* The Johns Hopkins University Press Baltimore London, second edition.

[Hac85] Hackbusch W. (1985) *Multi-Grid Methods and Applications.* Springer-Verlag Berlin Heidelberg New York.

[Kat67] Katsnelson V. E. (1967) Conditions under which systems of eigenvectors of some classes of operators form a basis. *Funct. Anal. Appl.* 1: 122–132.

[LM80] Longsine D. E. and McCormick S. F. (1980) Simultaneous Rayleigh-quotient minimization methods for $Ax = \lambda Bx$. *Lin. Alg. Appl.* 34: 195–234.

[McC94] McCormick S. F. (1994) Multilevel adaptive methods for elliptic eigenproblems: A two-level convergence theory. *SIAM J. Numer. Anal.* 31: 1731–1745.

[MM89] Mandel J. and McCormick S. F. (1989) A multilevel variational method for $Au = \lambda Bu$ on composite grids. *J. Comp. Phys.* 80: 442–452.

[Saa92] Saad Y. (1992) *Numerical Methods for Large Eigenvalue Problems.* Manchester University Press.

[Sch95] Schmidt F. (1995) Simultaneous computation of the lowest eigenvalues and eigenvectors of the Helmholtz equation. Technical Report TR 95-13, Konrad-Zuse-Zentrum für Informationstechnik Berlin.

3

Optimal Convergence for Overlapping and Non-Overlapping Schwarz Waveform Relaxation

M.J. Gander[1], L. Halpern[2], & F. Nataf[3]

INTRODUCTION

We are interested in solving time dependent problems of parabolic and hyperbolic type using domain decomposition techniques. Contrary to the classical approach where one discretizes time to obtain a sequence of steady problems to which the domain decomposition algorithms are applied (see [Cai91, Meu91, Cai94] for parabolic and [BGT97, WCK98] for hyperbolic problems), we formulate algorithms directly for the original problem without discretization. We decompose the spatial domain into subdomains and solve iteratively time dependent problems on subdomains, exchanging information at the boundary. Thus the algorithm is defined as in the classical Schwarz case, but like in waveform relaxation, time dependent subproblems are solved, which explains the name of these methods. In [Gan96, Gan97b, GS98] and [GK97] the overlapping version of such an algorithm has been studied for different types of parabolic problems. We investigate the algorithm applied to two new problems in this paper, the wave equation

$$\mathcal{L}_1(u) := u_{tt} - c^2 u_{xx} = f(x,t), \quad c > 0$$

1 CMAP, Ecole Polytechnique, 91128 Palaiseau, France. mgander@cmapx.polytechnique.fr
2 Département de Mathématiques, Université Paris XIII, 93430 Villetaneuse and CMAP, Ecole Polytechnique, 91128 Palaiseau, France. halpern@math.univ-paris13.fr
3 CMAP, Ecole Polytechnique, 91128 Palaiseau, France. nataf@cmapx.polytechnique.fr
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

and the linear convection reaction diffusion equation

$$\mathcal{L}_2(u) := u_t - \nu u_{xx} - a u_x - bu = f(x,t), \quad \nu > 0,\ a, b \in \mathbb{R}$$

on $\mathbb{R} \times [0,T]$ with appropriate initial conditions. Without loss of generality we assume for the convection reaction diffusion equation $a > 0$. We first analyze the convergence behavior of the overlapping Schwarz waveform relaxation algorithm applied to the above problems. We then show that the Dirichlet conditions at the artificial interfaces inhibit the information exchange between subdomains and therefore slow down the convergence of the algorithms. Using ideas introduced in [Hal86] and [NRdS95] we derive optimal transmission conditions for the convergence of the algorithms. These transmission conditions coincide with the absorbing boundary conditions studied in great detail to truncate computational domains in [EM77] for hyperbolic problems and in [Hal86] for convection diffusion problems. They lead to non-overlapping Schwarz waveform relaxation algorithms which converge in a finite number of steps, identical to the number of subdomains. In general however the exact absorbing boundary conditions are not available or expensive to compute. Similar to the approach for stationary problems in [NR95] and [Jap96] and for control problems in [Ben97] we approximate the exact absorbing boundary conditions locally. We optimize the convergence rate including an overlap in the optimization if desired. Numerical experiments show that the convergence rates are improved by orders of magnitudes.

OVERLAPPING SCHWARZ WAVEFORM RELAXATION

We decompose the spatial domain $\mathbb{R}$ into two overlapping subdomains $(-\infty, L]$ and $[0,\infty)$. By linearity it suffices to analyze the overlapping Schwarz waveform relaxation algorithm for the homogeneous problems with zero initial conditions,

$$\begin{array}{rcll} \mathcal{L}_i(v^{k+1}) & = & 0, & x \in (-\infty, L) \\ v^{k+1} & = & w^k, & x = L \\ \mathcal{L}_i(w^{k+1}) & = & 0, & x \in (0,\infty) \\ w^{k+1} & = & v^k, & x = 0 \end{array} \tag{1}$$

for $i = 1, 2$ and prove convergence to zero. Existence and uniqueness of the iterates is easily ensured by classical methods.

Theorem 1 *For the wave equation, $i = 1$ in (1), the algorithm converges in a finite number of iterations, $v^{2k+1} \equiv w^{2k+1} \equiv 0$ as soon as*

$$k \geq \frac{Tc}{2L}.$$

Proof Applying the Laplace transform with parameter $s \in \mathbb{C}$, $\Re(s) > 0$, we find the transformed solutions to be

$$\hat{v}^{k+1}(x,s) = \hat{w}^k(L,s)e^{s(x-L)/c} \tag{2}$$

$$\hat{w}^{k+1}(x,s) = \hat{v}^k(0,s)e^{-sx/c} \tag{3}$$

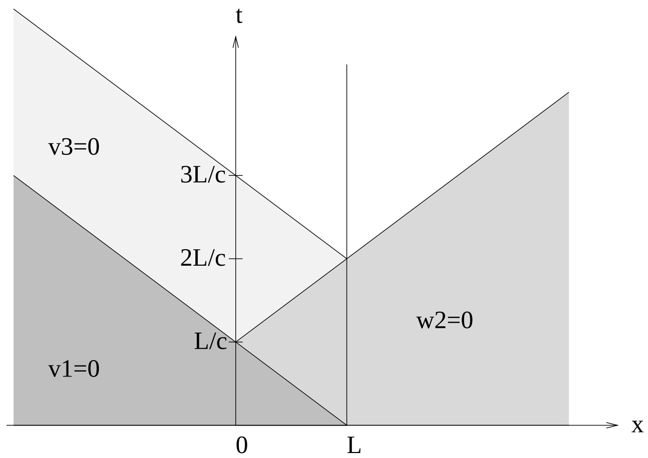


Figure 1 Regions where the iterates of the overlapping Schwarz waveform relaxation algorithm for the wave equation vanish due to the finite speed of propagation.

Evaluating (3) at iteration step k for $x = L$ and inserting it into (2), evaluated at $x = 0$, we find

$$\hat{v}^{k+1}(0,s) = e^{-2sL/c}\hat{v}^{k-1}(0,s).$$

Defining the convergence rate $\rho := e^{-2sL/c}$ we find by induction

$$\hat{v}^{2k}(0,s) = \rho^k\, \hat{v}^0(0,s). \tag{4}$$

A similar result holds for $\hat{w}^{2k}(L,s)$ and thus the iteration converges for all frequencies with $\Re(s) > 0$. To obtain the convergence result for bounded time intervals, we back-transform (4). Since

$$e^{-2kLs/c} = \int_0^\infty e^{-st}\delta(t - 2kL/c)dt$$

we find on using the convolution theorem of the Laplace transform

$$v^{2k}(0,t) = \int_0^t \delta(t - \tau - 2kL/c)v^0(0,\tau)d\tau = v^0(0, t - 2kL/c).$$

A similar result holds for $w^{2k}(L,t)$ and hence if $k > \frac{Tc}{2L}$ the transmission conditions imposed are identically zero and thus the next step leads to convergence. ■

Figure 1 shows intuitively why the overlapping Schwarz waveform relaxation algorithm for the wave equation converges in a finite number of steps. It is due to the finite speed of propagation: the iterates are identically zero before the arrival of the first disturbance from the artificial interfaces.

Theorem 2 *For the convection reaction diffusion equation, $i = 2$ in (1), the asymptotic convergence rate is superlinear and governed by the diffusion parameter ν,*

$$\frac{||v^{2k}(0,\cdot) + w^{2k}(L,\cdot)||_T}{||v^0(0,\cdot) + w^0(L,\cdot)||_T} \leq C erfc(\frac{kL}{\sqrt{\nu T}}).$$

with the constant $C = \max(1, e^{(b-a^2/4\nu)T})$.

Proof We take again a Laplace transform in time with parameter $s \in \mathbb{C}$, $\Re(s) > b$ and find the transformed solutions to be

$$\begin{aligned} \hat{v}^{k+1}(x,s) &= \hat{w}^k(L,s)e^{\frac{-a+\sqrt{a^2+4\nu(s-b)}}{2\nu}(x-L)} \quad (5)\\ \hat{w}^{k+1}(x,s) &= \hat{v}^k(0,s)e^{\frac{-a-\sqrt{a^2+4\nu(s-b)}}{2\nu}x}. \quad (6)\end{aligned}$$

Evaluating (6) at $x = L$ for iteration index k, inserting it into (5) and evaluating at $x = 0$ we find

$$\hat{v}^{k+1}(0,s) = e^{-2\sqrt{\frac{a^2}{4\nu^2}+\frac{s-b}{\nu}}L}\hat{v}^{k-1}(0,s).$$

Defining the convergence rate $\rho := e^{-2\sqrt{\frac{a^2}{4\nu^2}+\frac{s-b}{\nu}}L}$ we find by induction

$$\hat{v}^{2k}(0,s) = \rho^k \hat{v}^0(0,s). \quad (7)$$

A similar result holds for $\hat{w}^{2k}(L,s)$ and thus the additive Schwarz method converges for all frequencies $\Re(s) > b$. To obtain the desired convergence result for bounded time, we back-transform (7) on noting that [AS64]

$$e^{-x\sqrt{s+q}} = \int_0^\infty e^{-st}K(x,t)e^{-qt}dt$$

where the kernel K is given by

$$K(x,t) = \frac{x}{2\sqrt{\pi}t^{3/2}}e^{-\frac{x^2}{4t}}.$$

We find on using the convolution theorem for the Laplace transform

$$v^{2k}(0,t) = \int_0^t K_x(\frac{2kL}{\sqrt{\nu}}, t-\tau)e^{-(a^2/4\nu-b)(t-\tau)}v^0(0,\tau)d\tau.$$

Taking the supremum in time on a bounded time interval $0 < t < T$,

$$||v^{2k}(0,\cdot)||_T := \sup_{0<t<T} |v^{2k}(0,t)|$$

and estimating the exponential with $\max(1, e^{(b-a^2/4\nu)T})$ we get

$$||v^{2k}(0,\cdot)||_T \le \max(1, e^{(b-a^2/4\nu)T})\int_0^T K_x(\frac{2kL}{\sqrt{\nu}}, T-\tau)d\tau||v^0(0,\cdot)||_T.$$

Now applying the variable transform $y = kL/\sqrt{\nu(t-\tau)}$ in the integration leads to

$$||v^{2k}(0,\cdot)||_T \le \max(1, e^{(b-a^2/4\nu)T})\mathrm{erfc}(\frac{kL}{\sqrt{\nu T}})||v^0(0,\cdot)||_T.$$

By a similar argument for the second subdomain, the result follows. ■

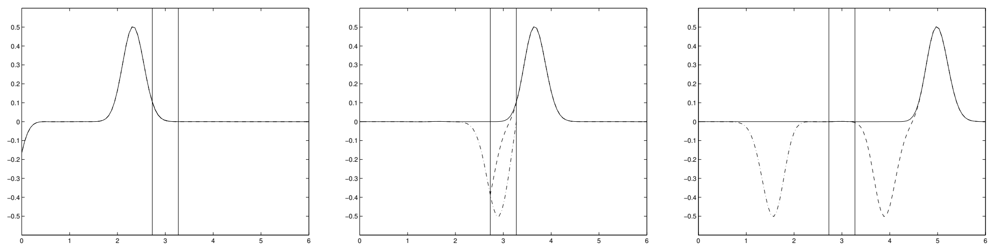


Figure 2 Snapshots of $v^1(x,t)$ (dash-dot) and $w^2(x,t)$ (dashed) in the wave equation case, together with the exact solution (solid) showing the erroneous reflections caused by the Dirichlet transmission conditions.

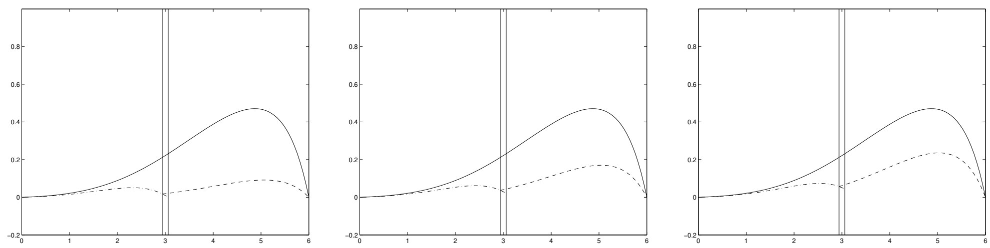


Figure 3 Iterates $v^k(x,T)$ (dash-dot) and $w^{k+1}(x,T)$ (dashed) at the end of the time interval for $k = 1,3,5$ in the convection reaction diffusion case together with the exact solution (solid) showing how the Dirichlet transmission conditions inhibit the information transport.

Both results differ from the classical linear convergence of the overlapping Schwarz method for elliptic problems. The convergence in a finite number of steps in the wave equation case and the superlinear convergence in the convection reaction diffusion case depend both on the time interval under consideration. For the wave equation it is evident that the convergence rate does not depend on the number of subdomains, whereas for the convection reaction diffusion equation one can show that the convergence rate depends only lower order on the number of subdomains [Gan97a]. This shows that coarse grid preconditioners are not necessary in this case.

In both cases however the Dirichlet transmission conditions at the interfaces are responsible for slow convergence, as one can see in Figure 2 where wrong reflected waves are created and in Figure 3 where the convection and diffusion of the information across the interface is inhibited. We are thus looking for a remedy of this by investigating the transmission conditions in the next section.

NON-OVERLAPPING SCHWARZ WAVEFORM RELAXATION

We are using the same algorithm as before, but with different transmission conditions, namely

$$\begin{array}{rcll}\mathcal{L}_i(v^{k+1}) & = & 0, & x \in (-\infty, L)\\ v_x^{k+1} + \Lambda_v(v^{k+1}) & = & w_x^k + \Lambda_v(w^k), & x = L\\ \mathcal{L}_i(w^{k+1}) & = & 0, & x \in (0, \infty)\\ w_x^{k+1} + \Lambda_w(w^{k+1}) & = & v_x^k + \Lambda_w(v^k), & x = 0,\end{array} \tag{8}$$

where Λ_v and Λ_w are linear operators acting along the boundary in time. Well-posedness is ensured in the case $i = 1$ by [EM77] and in the case $i = 2$ by [Hal86].

Theorem 3 *For the wave equation, $i = 1$ in (8), the algorithm converges in two iterations, independently of the size of the overlap, if*

$$\Lambda_v = \frac{1}{c}\partial_t, \quad \Lambda_w = -\frac{1}{c}\partial_t.$$

Proof Taking a Laplace transform of the above equations for $i = 1$ with parameter s, $\Re(s) > 0$, we find

$$\begin{array}{rcl} s^2\hat{v}^{k+1} & = & c^2\hat{v}_{xx}^{k+1},\\ \hat{v}_x^{k+1}(L,s) + \lambda_v(s)(\hat{v}^{k+1}(L,s)) & = & \hat{w}_x^k(L,s) + \lambda_v(s)(\hat{w}^k(L,s)),\\ s^2\hat{w}^{k+1} & = & c^2\hat{w}_{xx}^{k+1},\\ \hat{w}_x^{k+1}(0,s) + \lambda_w(s)(\hat{w}^{k+1}(0,s)) & = & \hat{v}_x^k(0,s) + \lambda_w(s)(\hat{v}^k(0,s)).\end{array}$$

Solving for $\hat{w}$ at iteration k for $x = L$ and inserting into the solution for $\hat{v}^{k+1}$ one obtains after evaluating at $x = 0$

$$\hat{v}^{k+1}(0,s) = \frac{-\frac{s}{c} + \lambda_v}{\frac{s}{c} + \lambda_v} \cdot \frac{\frac{s}{c} + \lambda_w}{-\frac{s}{c} + \lambda_w} e^{-2\frac{s}{c}L}\hat{v}^{k-1}(0,s).$$

A similar result holds for $\hat{w}^{k+1}(L,s)$. Thus choosing $\lambda_v = \frac{s}{c}$ and $\lambda_w = -\frac{s}{c}$ the iteration converges in two steps, $v^2 \equiv w^2 \equiv 0$, and the factor $e^{-2\frac{s}{c}L}$ stemming from the overlap becomes irrelevant. Back-transforming this choice, the result follows. ■

Theorem 4 *For the convection reaction diffusion equation, $i = 2$ in (8), the above algorithm converges in two iterations independently of the size of the overlap, if the operators Λ_v and Λ_w have the corresponding symbols*

$$\lambda_v = \frac{a}{2\nu} + \frac{\sqrt{a^2 + 4\nu(s-b)}}{2\nu}, \quad \lambda_w = \frac{a}{2\nu} - \frac{\sqrt{a^2 + 4\nu(s-b)}}{2\nu}.$$

Proof Using the Laplace transform as before with parameter $s \in \mathbb{C}$, $\Re(s) > b$, we find

$$\begin{array}{rcl} s\hat{v}^{k+1} & = & \nu\hat{v}_{xx}^{k+1} + a\hat{v}_x^{k+1} + b\hat{v}^{k+1},\\ \hat{v}_x^{k+1}(L,s) + \lambda_v(s)(\hat{v}^{k+1}(L,s)) & = & \hat{w}_x^k(L,s) + \lambda_v(s)(\hat{w}^k(L,s)),\\ s\hat{w}^{k+1} & = & \nu\hat{w}_{xx}^{k+1} + a\hat{w}_x^{k+1} + b\hat{w}^{k+1},\\ \hat{w}_x^{k+1}(0,s) + \lambda_w(s)(\hat{w}^{k+1}(0,s)) & = & \hat{v}_x^k(0,s) + \lambda_w(s)(\hat{v}^k(0,s))\end{array}$$

Solving for $\hat{w}$ at iteration k for $x = L$ and inserting into the solution $\hat{v}^{k+1}$ one obtains after evaluating at $x = 0$

$$\hat{v}^{k+1}(0,s) = \frac{\lambda_2 + \lambda_v}{\lambda_1 + \lambda_v} \cdot \frac{\lambda_1 + \lambda_w}{\lambda_2 + \lambda_w} e^{(\lambda_2 - \lambda_1)L} \hat{v}^{k-1}(0,s)$$

where λ_{12} denote the characteristic roots,

$$\lambda_{12} = -\frac{a}{2\nu} \pm \frac{\sqrt{a^2 + 4\nu(s-b)}}{2\nu}.$$

A similar result holds for $\hat{w}^{k+1}(L,s)$. Thus by choosing $\lambda_v = -\lambda_2$ and $\lambda_w = -\lambda_1$ the algorithm converges in two steps, $v^2 \equiv w^2 \equiv 0$, independently of the overlap. ■

Note that in this case however, the symbols lead to nonlocal operators in time, which are more expensive to implement in an algorithm than the local ones found for the wave equation. It is therefore of interest to approximate the nonlocal operators by local ones, whose symbols are polynomials. We propose here four different approximations: Taylor approximations of zeroth and first order

$$\Lambda_{vw} = \frac{a \pm \sqrt{a^2 - 4\nu b}}{2\nu}, \quad \Lambda_{vw} = \frac{a \pm \sqrt{a^2 - 4\nu b}}{2\nu} \pm \frac{1}{\sqrt{a^2 - 4\nu b}} \partial t$$

and optimized constant and first order polynomials

$$\Lambda_{vw} = \frac{a \pm p}{2\nu} \qquad , \quad \Lambda_{vw} = \frac{a \pm \sqrt{a^2 - 4\nu b}}{2\nu} \pm \frac{q}{2\nu} \partial t$$

where we optimize the convergence rate using p in

$$\min_{p>0} \left(\max_{\Re(s)>0} \left| \frac{(p - \sqrt{a^2 + 4\nu(s-b)})^2}{(p + \sqrt{a^2 + 4\nu(s-b)})^2} e^{-\frac{L}{\nu}\sqrt{a^2+4\nu(s-b)}} \right| \right)$$

and q in

$$\min_{q>0} \left(\max_{\Re(s)>0} \left| \frac{(qs + \sqrt{a^2 - 4\nu b} - \sqrt{a^2 + 4\nu(s-b)})^2}{(qs + \sqrt{a^2 - 4\nu b} + \sqrt{a^2 + 4\nu(s-b)})^2} e^{-\frac{L}{\nu}\sqrt{a^2+4\nu(s-b)}} \right| \right).$$

The optimization is performed numerically to obtain the convergence results in the next section.

NUMERICAL RESULTS

We show numerical results for the parabolic problem to test the effectiveness of the approximately absorbing transmission conditions. We consider the model problem

$$u_t = u_{xx} - 2u_x + \frac{1}{2}u \quad 0 < x < 6, \; 0 < t < T = 2$$

with given data

$$u(0,t) = 0, \quad u(6,t) = 0, \quad u(x,0) = e^{-3(\frac{3}{2} - x)^2}$$

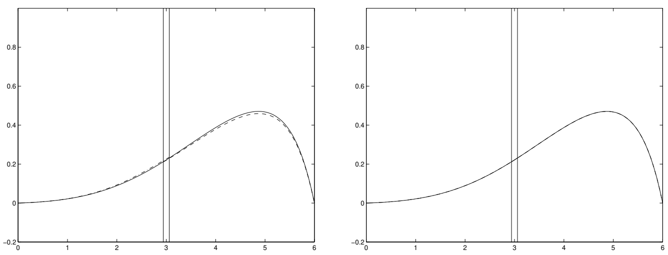


Figure 4 Iterates $v^k(x,T)$ (dash-dot) and $w^{k+1}(x,T)$ (dashed) at the end of the time interval for $k=1,3$ in the convection reaction diffusion case together with the exact solution (solid) using optimized first order transmission conditions.

and to compare the performance with the Dirichlet case, we employ an overlap of 2%. Figure 4 shows the iterates $v^1(x,T)$ and $w^2(x,T)$ on the left and $v^3(x,T)$ and $w^4(x,T)$ on the right at the end of the time interval with the optimized first order transmission conditions and should be compared with the results in Figure 3. Clearly the information is now convected and diffused across the artificial interface with the new transmission conditions. Figure 5 shows the performance of the same algorithm when the transmission conditions are changed from Dirichlet to the new approximately absorbing transmission conditions.

REFERENCES

[AS64] Abramowitz M. and Stegun I. A. (1964) *Handbook of mathematical functions with formulas, graphs, and mathematical tables.* U.S. Govt. Print. Off., Wahington, USA.

[Ben97] Benamou J. (1997) Décomposition de domaine pour le contrôle de systèmes gouvernés par des équations d'évolution. *C. R. Acad. Sci. Paris, Série I* 324: 1065–1070.

[BGT97] Bamberger A., Glowinski R., and Tran U. (April 1997) A domain decomposition method for the acoustic wave equation with discontinuous coefficients and grid chance. *SIAM Journal on Numerical Analysis* 34(2): 603–639.

[Cai91] Cai X.-C. (1991) Additive Schwarz algorithms for parabolic convection-diffusion equations. *Numer. Math.* 60(1): 41–61.

[Cai94] Cai X.-C. (1994) Multiplicative Schwarz methods for parabolic problems. *SIAM J. Sci Comput.* 15(3): 587–603.

[EM77] Engquist B. and Majda A. (1977) Absorbing boundary conditions for the numerical simulation of waves. *Math. Comp.* 31(139): 629 – 651.

[Gan96] Gander M. J. (1996) Overlapping Schwarz for linear and nonlinear parabolic problems. In Bjørstad P., Espedal M., and Keyes D. (eds) *Proceedings of the Ninth International Conference on Domain Decomposition Methods*, pages 97–104. DDM.org. Held in Bergen, Norway, June 3-8 1996.

[Gan97a] Gander M. J. (September 1997) *Analysis of Parallel Algorithms for Time Dependent Partial Differential Equations.* PhD thesis, Stanford University, Stanford, CA 94305, USA.

[Gan97b] Gander M. J. (1997) Overlapping Schwarz waveform relaxation for

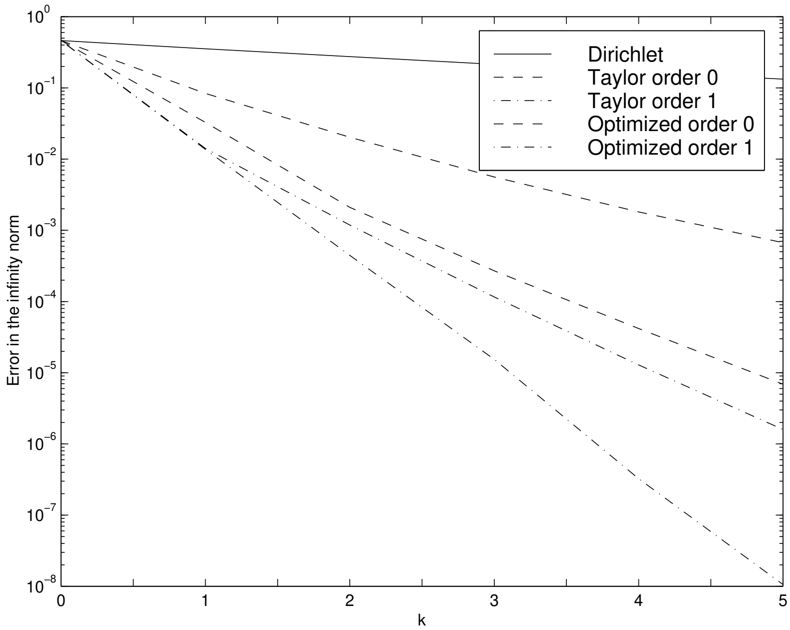


Figure 5 Convergence rates of the classical Schwarz with Dirichlet transmission conditions compared to the same algorithm with the new transmission conditions.

parabolic problems. In Mandel J., Farhat C., and Cai X.-C. (eds) *Proceedings of the Tenth International Conference on Domain Decomposition Methods*, Contemporary Mathematics. AMS. Held in Boulder, Colorado, August 10-14, 1997.

[GK97] Giladi E. and Keller H. (1997) Space time domain decomposition for parabolic problems. Technical Report 97-4, Center for research on parallel computation CRPC, Caltech.

[GS98] Gander M. J. and Stuart A. (1998) Space-time continuous analysis of waveform relaxation for the heat equation. *SIAM Journal on Scientific Computing* 19(6): 2014–2031.

[Hal86] Halpern L. (1986) Artificial boundary conditions for the advection-diffusion equations. *Math. Comp.* 174: 425–438.

[Jap96] Japhet C. (1996) Optimized Krylov-Ventcell method. Application to convection-diffusion problems. In Bjørstad P., Espedal M., and Keyes D. (eds) *Proceedings of the Ninth International Conference on Domain Decomposition Methods*, pages 382–389. DDM.org. Held in Bergen, Norway, June 3-8 1996.

[Meu91] Meurant G. A. (1991) Numerical experiments with a domain decomposition method for parabolic problems on parallel computers. In Glowinski R., Kuznetsov Y. A., Meurant G. A., Périaux J., and Widlund O. (eds) *Fourth International Symposium on Domain Decomposition Methods for Partial Differential Equations*. SIAM, Philadelphia, PA.

[NR95] Nataf F. and Rogier F. (1995) Factorization of the convection-diffusion operator and the Schwarz algorithm. M^3AS 5(1): 67–93.

[NRdS95] Nataf F., Rogier F., and de Sturler E. (1995) Domain decomposition methods for fluid dynamics, Navier-Stokes equations and related nonlinear analysis. *Edited by A. Sequeira, Plenum Press Corporation* pages 367–376.

[WCK98] Wu Y., Cai X.-C., and Keyes D. E. (1998) Additive Schwarz methods for hyperbolic equations. In Mandel J., Farhat C., and Cai X.-C. (eds) *Tenth International Conference on Domain Decomposition Methods*, pages 513–521. AMS, Contemporary Mathematics 218.

4

Fast Integration Techniques in 3D Boundary Elements

I.G. GRAHAM [1], W. HACKBUSCH [2], S.A. SAUTER[3]

Introduction

Boundary element methods are popular in numerical engineering, especially for solving classical PDEs on complicated and or/infinite 3D regions, where (for example in scattering problems) they are often preferred to finite elements. More generally, they can be combined with finite elements in order to efficiently handle problems in which different PDEs on different parts of a domain are coupled together. Boundary elements can be used on the boundaries of linear homogeneous regions, with (domain) finite element approximation in nonlinear or inhomogeneous regions. There is a large literature on this topic in which fast solution strategies based on this type of decomposition have been widely reported (e.g., [5]).

The inclusion of boundary elements in such a coupled solution strategy poses two problems not found in the stand-alone finite element method: the calculation of complicated (sometimes singular) integrals to form the stiffness matrix, and the solution of full systems. Much recent work has been done on system solution, yielding almost optimal solvers with close to $O(N)$ complexity, where N is the number of degrees of freedom. Less work has been done on optimising the assembly of the stiffness matrix which (if it is fully assembled) costs $C_1 N^2 + O(N)$ operations, where C_1 is

1 Department of Mathematical Sciences, University of Bath, UK. email: igg@maths.bath.ac.uk
2 Max-Planck-Institut "Mathematik in den Naturwissenschaften", 04103 Leipzig, Germany. email: wh@mis.mpg.de
3 Mathematisches Institut, Universität Leipzig, 04109 Leipzig, Germany. email: sas@mathematik.uni-leipzig.de
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

moderately large and driven by the number of kernel evaluations required by the integrator. In modern boundary element codes the cost of stiffness matrix assembly can be the principle bottleneck. In this paper we describe a new procedure ([2, 3]) for computing approximations to Galerkin stiffness matrices using only $N^2 + O(N^\gamma)$ kernel evaluations with $\gamma \in (0,2)$. The approximate stiffness matrix is guaranteed to be accurate enough so that the corresponding numerical solution inherits the same stability and convergence properties as are enjoyed by conventional Galerkin methods.

A key point in our method is the decomposition of the (closed) surface $\Gamma \subset \mathbf{R^3}$ on which the integral equation is posed into a number of smooth closed components $\{\Gamma_\ell : \ell = 1, \dots L\}$ (with the interiors of the Γ_ℓ assumed pairwise disjoint). As is typical in the description of 2D manifolds in 3D space, each Γ_ℓ is here assumed to be parametrised by a polygonal planar chart $\tilde{\Gamma}_\ell$. We shall devise fast integration techniques on the $\tilde{\Gamma}_\ell$, which then induce corresponding fast methods on Γ. One of the novel contributions of this paper is to provide a more efficient way of implementing the new algorithm when several parametrised pieces of a smooth surface have to be stitched together. In order to explain the basic principles of the method and to present the novel contribution in the least technical context we assume here that Γ is a $\mathcal{C}^\infty$ surface. Note that this is by no means essential and the technical details for general piecewise smooth surfaces are given in ([2, 3, 4]). The new implementation described here can also be used for integration over smooth portions of such piecewise smooth surfaces.

We shall consider general boundary integral equations of the form

$$(\lambda I + \mathcal{K})u(\boldsymbol{x}) := \lambda u(\boldsymbol{x}) + \int_\Gamma k(\boldsymbol{x}, \boldsymbol{y}) u(\boldsymbol{y}) dy = f(\boldsymbol{x}), \quad \boldsymbol{x} \in \Gamma \ , \tag{1}$$

where the real scalar λ and function f are given and u is to be found. For convenience we assume that $\tilde{\Gamma} := \cup_\ell \tilde{\Gamma}_\ell$ forms the surface of a polyhedron (although, in principle any standard chart system would also do) and we assume that the parametrisation is via a bi-Lipschitz bijectiion $\eta : \tilde{\Gamma} \to \Gamma$. (Only *action* of the mapping η is explicitly required in practice.)

A standard boundary element method for (1) then begins by triangulating $\tilde{\Gamma}$ using a family of meshes $\tilde{\mathcal{T}}_h$, with parameter $h \to 0$ denoting the mesh diameter. As the meshes are refined we assume that they are conforming and shape-regular (see, e.g. [2]). We denote the mesh nodes by $\{\tilde{\boldsymbol{x}}_p : p \in \mathcal{N}\}$ (where $\mathcal{N}$ is an index set), and we denote the corresponding nodal basis for the continuous piecewise linear functions with respect to $\tilde{\mathcal{T}}_h$ on $\tilde{\Gamma}$ by $\{\tilde{\phi}_p : p \in \mathcal{N}\}$. The induced functions $\phi_p := \tilde{\phi}_p \circ \eta^{-\mathbf{1}} : \mathbf{\Gamma} \to \mathbf{R}$ are used to approximate (1) using Galerkin's method. The ϕ_p are piecewise smooth with respect to the curvilinear triangular mesh on Γ with triangles $\eta(\tau) : \tau \in \tilde{\mathcal{T}}_\mathbf{h}$ and nodes $\boldsymbol{x}_p := \eta(\tilde{\boldsymbol{x}}_\mathbf{p}) : \mathbf{p} \in \mathcal{N}$.

The chief computational task in assembling the resulting stiffness matrix is computing the N^2 entries (where $N = |\mathcal{N}|$):

$$K_{p,q} = \int_\Gamma \int_\Gamma k(\mathbf{x}, \mathbf{y}) \phi_\mathbf{q}(\mathbf{y}) \phi_\mathbf{p}(\mathbf{x}) \mathbf{dydx} \ , \quad \mathbf{p}, \mathbf{q} \in \mathcal{N}. \tag{2}$$

Our new method adaptively partitions K into a "conventional Galerkin" part in which $K_{p,q}$ is computed by conventional quadrature and another part which is done by very cheap unconventional rules (using typically between 10 and 100 times fewer kernel evaluations). The complexity theory of the resulting "hybrid" algorithm (see §4) shows that under appropriate conditions the cheap part dominates. In §4 we give a new set of numerical results (for computation of the 3D harmonic Neumann-Dirichlet map) which show that the algorithm can compute more than 90% of the matrix K using the cheap method, without damaging the underlying convergence properties of the numerical solution.

To motivate the quadrature techniques for (2), let us first of all ignore the fact that the "kernel function" k is normally singular at $\boldsymbol{x} = \boldsymbol{y}$ and imagine that k is globally smooth. Then both conventional and unconventional approaches to computing (2) can be obtained as two applications of rules which compute, for general smooth F, the "fundamental integral":

$$\int_{\Gamma} F(\mathbf{x})\phi_{\mathbf{p}}(\mathbf{x})\mathbf{dx} = \int_{\tilde{\Gamma}} \tilde{\mathbf{F}}(\tilde{\mathbf{x}})\tilde{\phi}_{\mathbf{p}}(\tilde{\mathbf{x}})\mathbf{d}\tilde{\mathbf{x}} \ . \tag{3}$$

Here $\tilde{F}(\tilde{\boldsymbol{x}}) := F(\eta(\tilde{\boldsymbol{x}}))\mathrm{g}(\tilde{\boldsymbol{x}})$, g is the "Gram determinant" of η appearing in the area element transformation $dx = g(\tilde{\boldsymbol{x}})d\tilde{x}$ and we have used the fact that $\phi_p(\eta(\tilde{\boldsymbol{x}})) = \tilde{\phi}_{\mathbf{p}}(\tilde{\boldsymbol{x}})$. Note our assumptions ensure that g is a smooth and uniformly positive function on each $\tilde{\Gamma}_\ell$ (although g may not be smooth globally).

The conventional approach to (3) separates the right-hand side into a sum of integrals over each of the planar triangles in supp $\tilde{\phi}_p$, on which standard rules (for example from [8]) can be applied. This approach results in a fairly high complexity in terms of kernel evaluations. For example, a popular rule which will be exact for (3) when $\tilde{F}$ is a bivariate polynomial on each triangle requires evaluations of $\tilde{F}$ at the nodes, edge mid-points and centroids of a triangle. Recalling that N is the number of nodes in the mesh, there are about $2N$ triangles and $3N$ edges and so use of this rule to compute (3) for all p requires about $6N$ evaluations of F. Computing all elements of the matrix K by iterating this rule would then require about $36N^2$ kernel evaluations. Higher order conventional rules will typically require CN^2 evaluations with C higher still.

However we can obtain the same degree of precision in $N^2+O(N)$ kernel evaluations using the following more unconventional approach. Note that the main reason for separating (3) into a sum over triangles is the fact that $\tilde{\phi}_p$ is only triangle-wise smooth. However $\tilde{\phi}_p$ is simple enough so that it can be integrated easily against any polynomial over all of its support. Since k is (temporarily assumed) globally smooth, this suggests that we should devise rules for (3) which work well for smooth $\tilde{F}$, and in which $\tilde{\phi}_p$ is treated as a weight-function. This gives us the freeedom to use evaluations of F at any points, and if we use only *nodal* evaluations then we can compute (3) for all p using only N evaluations of F. Iterating this means that K can be computed using exactly N^2 evaluations, namely $k(\boldsymbol{x}_p, \boldsymbol{y}_q)$, $p, q \in \mathcal{N}$. Note also that, in contrast to conventional rules, the N^2 evaluation count remains the same irrespective of the degree of precision required, and each nodal evaluation of k may be used in the computation of $K_{p,q}$ for several p, q.

Our task then is to devise quadrature approximations of the right-hand side of (3) which take some subset of the nodes $\{\tilde{\boldsymbol{x}}_p : p \in \mathcal{N}\}$ as quadrature points and which are exact for polynomials of some specified degree. However, since we are dealing here with general meshes, and therefore quadrature rules with general abscissae (in 2D regions), the weights of these quadrature rules are not known *a priori* and have to be computed as part of the algorithm. For example, suppose supp $\tilde{\phi}_p$ is entirely contained inside one of the charts $\tilde{\Gamma}_\ell$ and suppose we wish to construct a quadrature rule which is exact for (3) when $\tilde{F}$ is a bivariate polynomial on $\tilde{\Gamma}_\ell$. Then we would choose at least 6 nodes $\{\tilde{\boldsymbol{x}}_j : j \in J_p\}$ in $\tilde{\Gamma}_\ell$ (with J_p denoting a suitable index set) and (using the method of undetermined coefficients), seek weights $\{\tilde{w}_j : j \in J_p\}$ such that the equation

$$\sum_{j \in J_p} \tilde{w}_j \tilde{\Pi}_i(\tilde{\boldsymbol{x}}_j) = \int_{\tilde{\Gamma}_\ell} \tilde{\Pi}_i(\tilde{\mathbf{x}}) \tilde{\phi}_p(\tilde{\mathbf{x}}) d\tilde{x} \ , \tag{4}$$

where $\{\tilde{\Pi}_i : i = 1, \ldots 6\}$ are a suitable basis for the bivariate quadratic polynomials (e.g. $1, \tilde{x}_1, \ldots \tilde{x}_2^2$, in local coordiantes on $\tilde{\Gamma}_\ell$). This yields 6 equations to solve for the weights $\{\tilde{w}_j : j \in J_p\}$.

More generally, if supp $\tilde{\phi}_p$ overlaps with two charts $\tilde{\Gamma}_\ell$ and $\tilde{\Gamma}_{\ell'}$ say, then $\tilde{\Gamma}_\ell \cup \tilde{\Gamma}_{\ell'}$ may not be a smooth subset of $\tilde{\Gamma}$, even though Γ itself is smooth. In [2, 3] this situation was dealt with by separating supp $\tilde{\phi}_p$ into smooth components, each from one $\tilde{\Gamma}_\ell$, then selecting at least 6 nodal quadrature points in each relevant $\tilde{\Gamma}_\ell$, and solving systems of form (4) in each component for the weights. This approach is natural when an edge of parameter space $\tilde{\Gamma}$ corresponds to a true edge in Γ. However when a smooth (part of) Γ is parametrised by a piecewise smooth (part of) $\tilde{\Gamma}$ it is more natural to use an alternative approach, based on overlapping subdivisions of Γ, which is described in §4. In any case it should be clear that we can in principle construct node-based rules of precision 2 (or indeed any precision) for (3) and that the computation of the corresponding weights requires the solution of (possibly underdetermined) systems such as (4). The theory of solvability of these systems is an interesting problem in its own right and in [3] stability is demonstrated in a wide range of practical situations for rules with degree of precision 1 and 2. Computing one (or several) such node-based rules for all $p \in \mathcal{N}$ is an $O(N)$ process.

Now recall that the "real" kernel function k in (1) is a fundamental solution of a PDE (or a derivative of a fundamental solution) and so it can be expected to blow up as $\boldsymbol{x} \to \boldsymbol{y}$. To allow for this we shall assume here that $k(\boldsymbol{x}, \boldsymbol{y})$ is $\mathcal{C}^\infty$ for $\boldsymbol{x} \neq \boldsymbol{y}$ (more general kernels are allowable [2, 3]), and also that there exists $\alpha > 0$ such that for all integers $m \geq 0$, there exists a constant $B_m > 0$ such that

$$|D^m\{k(\eta(\tilde{\boldsymbol{x}}), \eta(\tilde{\boldsymbol{y}}))\}| \leq \mathbf{B_m} |\tilde{\boldsymbol{x}} - \tilde{\boldsymbol{y}}|^{-\alpha - \mathbf{m}} \ , \quad \text{for } \tilde{\boldsymbol{x}} \neq \tilde{\boldsymbol{y}} \ , \quad \tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}} \in \tilde{\boldsymbol{\Gamma}} \ . \tag{5}$$

Here D^m denotes any mth order partial differential operator with respect to $(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}})$. Most kernels in practice satisfy this and the general theory in [2] shows that, in order for the quadrature error in computing K not to damage the overall convergence rate of the Galerkin scheme, the degree of precision of the quadrature rules needed to compute $K_{p,q}$ must be higher if supp $\tilde{\phi}_p$ is close to supp $\tilde{\phi}_q$ compared to when they are well-separated. Moreover special regularising transforms (e.g. [1]) are needed when supp $\tilde{\phi}_p \cap$ supp $\tilde{\phi}_q \neq \emptyset$. But (most importantly for this paper), only low degree of precision rules (e.g., 1 or 2) are typically needed when the distance between supp $\tilde{\phi}_p$ and supp $\tilde{\phi}_q$ is $O(1)$ (as $h \to 0$). This motivates the hybrid algorithm given in §4, which uses cheap low order node-based rules when supp $\tilde{\phi}_p$ and supp $\tilde{\phi}_q$ are far enough apart, and uses the conventional approach elsewhere. In the experiments in §4 we see that in standard applications the cheap method strongly dominates the computation, yielding a complexity of exactly N^2 kernel evaluations (at pairs of node-points) together with a additional $O(N^\gamma)$ evaluations with $\gamma \in (1,2)$, due to the small part of the matrix done by conventional methods. The solutions computed by the cheap method exhibit the same rate of convergence as those computed by the conventional method. To the cost of the cheap method we should also add $O(N)$ operations needed to compute the weights of the node-based rules in the first phase of the algorithm.

Overlapping Decompositions

In this section we describe an efficient method for computing node-based rules when $\tilde{\Gamma}$ contains edges which do not correspond to edges of Γ. We assume that each $\tilde{\Gamma}_\ell$ is extended to a slightly larger planar polygon $\tilde{\Gamma}_\ell^e$, with the distance between the boundaries of $\tilde{\Gamma}_\ell$ and $\tilde{\Gamma}_\ell^e$ being bounded below by $\epsilon > 0$. Assume also that each parametrisation map $\eta_\ell := \eta|_{\tilde{\Gamma}_\ell}$ can be extended to a smooth map $\eta_\ell^{\mathbf{e}}$ which takes $\tilde{\Gamma}_\ell^e$ into a neighbourhood Γ_ℓ^e of Γ_ℓ. (Note that such an extended chart system would normally be part of the geometric description of a surface.) In order to use these extended charts in the computation of (3), we need also to assume that the inverse maps $(\eta_\ell^{\mathbf{e}})^{-\mathbf{1}} : \mathbf{\Gamma}_\ell^{\mathbf{e}} \to \tilde{\mathbf{\Gamma}}_\ell^{\mathbf{e}}$ are known, and that, for each $p \in \mathcal{N}$, there exists ℓ (depending on p) such that supp $\phi_p \subset \Gamma_\ell^e$. Clearly the latter assumption holds if the overlap ϵ is sufficiently large compared to h (e.g. $\epsilon = Ch$ for a suitably large constant C would suffice). Then we can write, analogously to (3),

$$\int_\Gamma F(\mathbf{x})\phi_{\mathbf{p}}(\mathbf{x})\mathbf{dx} = \int_{\tilde{\mathbf{\Gamma}}_\ell^{\mathbf{e}}} \tilde{\mathbf{F}}^{\mathbf{e}}(\tilde{\mathbf{x}}^{\mathbf{e}})\phi_{\mathbf{p}}(\eta_\ell^{\mathbf{e}}(\tilde{\mathbf{x}}^{\mathbf{e}}))\mathbf{d}\tilde{\mathbf{x}}^{\mathbf{e}} \ , \tag{6}$$

with $\tilde{F}^e(\tilde{\boldsymbol{x}}^e) := F(\eta_\ell^{\mathbf{e}}(\tilde{\boldsymbol{x}}^{\mathbf{e}}))\mathbf{g}_\ell^{\mathbf{e}}(\tilde{\boldsymbol{x}}^{\mathbf{e}})$, and g_ℓ^e denoting the Gram determinant of $\eta_\ell^{\mathbf{e}}$.

Now since $\tilde{\Gamma}_\ell^e$ is planar, we can apply the procedure introduced in the previous section to generate node-based rules for (6). To compute a rule with degree of precision d, select at least $(d+1)(d+2)/2$ nodes $\{\boldsymbol{x}_j : j \in J_p\}$ on Γ_ℓ^e, pull them back to the set $\{\tilde{\boldsymbol{x}}_j^e = (\eta_\ell^{\mathbf{e}})^{-\mathbf{1}}(\boldsymbol{x}_{\mathbf{j}}) : \mathbf{j} \in \mathbf{J_p}\}$ of nodes on $\tilde{\Gamma}_\ell^e$, and then seek weights $\tilde{w}_j$ so that

$$\sum_{j \in J_p} \tilde{w}_j \tilde{\Pi}_i(\tilde{\boldsymbol{x}}_j^e) = \int_{\tilde{\Gamma}_\ell^e} \tilde{\Pi}_i(\tilde{\mathbf{x}}^e)\phi_p(\eta_\ell^{\mathbf{e}}(\tilde{\mathbf{x}}^{\mathbf{e}}))\mathbf{d}\tilde{\mathbf{x}}^{\mathbf{e}} \ , \quad \mathbf{i} = \mathbf{1}, \ldots, (\mathbf{d}+\mathbf{1})(\mathbf{d}+\mathbf{2})/\mathbf{2} \ , \tag{7}$$

where the $\tilde{\Pi}_i$ are a basis for the polynomials of degree d on the planar domain $\tilde{\Gamma}_\ell^e$. This procedure yields a (possibly underdetermined) linear system for $\tilde{\boldsymbol{w}}$:

$$M\tilde{\boldsymbol{w}} = \tilde{\boldsymbol{b}} \ , \tag{8}$$

where M is the trivially computed generalised Vandermonde matrix containing the values of the basis functions evaluated at the quadrature points. From (7) and the definition of ϕ_p, each element of the right-hand side $\tilde{\boldsymbol{b}}$ can be written

$$\tilde{b}_i = \int_{\tilde{\Gamma}_\ell^e} \tilde{\Pi}_i(\tilde{\mathbf{x}}^e)\tilde{\phi}_p((\eta^{-\mathbf{1}} \circ \eta_\ell^{\mathbf{e}})(\tilde{\mathbf{x}}^{\mathbf{e}}))\mathbf{d}\tilde{\mathbf{x}}^{\mathbf{e}} \ .$$

In the case when supp $\tilde{\phi}_p \subset \tilde{\Gamma}_\ell$, the map $\eta^{-1} \circ \eta^e_\ell$ is the identity on supp $\tilde{\phi}_p$, and so $\tilde{b}_i$ is simply the integral of a piecewise polynomial over a set of planar triangles which can be done exactly using triangle-based quadrature rules of appropriate order. By contrast, if supp $\tilde{\phi}_p$ contains triangles which are in other charts besides $\tilde{\Gamma}_\ell$, then we have to write

$$\tilde{b}_i = \int_{\tilde{\Gamma}} \tilde{\Pi}_i((\eta^e_\ell)^{-1} \circ \eta)(\boldsymbol{y}))\mathbf{G}(\boldsymbol{y})\tilde{\phi}_{\mathbf{p}}(\boldsymbol{y})\mathbf{dy} \ , \tag{9}$$

where G is the Gram determinant of the map $\boldsymbol{y} \mapsto ((\eta^e_\ell)^{-1} \circ \ \eta)(\boldsymbol{y})$. The first two terms in the integrand in (9) are smooth on the intersection of supp $\tilde{\phi}_p$ with each of the $\tilde{\Gamma}_\ell$ and so $\tilde{b}_i$ can be approximated to arbitrary accuracy by appropriate triangle-based rules on $\tilde{\Gamma}$. We emphasise that the computation of $\tilde{b}_i$ is part of the process of computing the weights for the node-based rules and it does not require any kernel evaluations. We shall see in §4 that the computation of the weights (since it is an $O(N)$ process) is a small part of the overall computation time and the extra work that has to be done is small compared to that saved in using the node-based rules for the assembly of K instead of conventional rules.

In the following section we review the hybrid Galerkin algorithm and we also extend its theory ([2, 3]) to the case where the weights of the node-based rules are taken to be the solution $\hat{\boldsymbol{w}}$ of the approximation to (8):

$$M\hat{\boldsymbol{w}} = \hat{\boldsymbol{b}} \ , \tag{10}$$

where $\hat{\boldsymbol{b}}$ is an appropriately accurate approximation of $\tilde{\boldsymbol{b}}$. This extension of the theory is used in the computations in §4. In §4 we need following generalisation of [2]Lemma 2.7 which can be proved by reworking the proof in this more general situation. To state this lemma we let h_τ denote the diameter of any (curvilinear) triangle $\tau \in \mathcal{T}_h$ and we define the mesh diameter "local" to the node $\mathbf{x_p}$ by $h_p = \max\{h_\tau : \tau \in \mathcal{T}_(, \tau \subset \text{supp } \phi_{\mathrm{p}}\}$.

Lemma 1 *Suppose that the solution $\hat{\boldsymbol{w}}$ of (10) is stable in the sense that $\sum_{j\in J_p} |\hat{w}_j| \leq \sigma \sum_{j \in J_p} \hat{w}_j$, and that the points $\{\boldsymbol{x}_j \ : \ j \ \in \ J_p\}$ are not too far from $\boldsymbol{x}_p$, i.e., $\max\{|\boldsymbol{x}_j - \boldsymbol{x}_p| \ : \ j \ \in \ J_p\} \ \leq \ \delta h_p$. Suppose also that $\hat{\boldsymbol{b}}$ is a sufficiently accurate approximation to $\tilde{\boldsymbol{b}}$ in the sense that*

$$||\hat{\boldsymbol{b}} - \tilde{\boldsymbol{b}}||_\infty \leq \beta h_p^{d+3} \ , \tag{11}$$

where $\sigma, \delta, \beta > 0$ are bounded independently of h. Then the resulting node-based quadrature rule for (6) satisfies:

$$| \int_{\tilde{\Gamma}^e_\ell} \tilde{F}^e \phi_p \circ \eta^e_\ell - \sum_{\mathbf{j} \in \mathbf{J_p}} \hat{\mathbf{w}}_{\mathbf{j}} \tilde{\mathbf{F}}^e(\tilde{\boldsymbol{x}}^e_{\mathbf{j}})| \leq \mathbf{C} \mathbf{h}_{\mathbf{p}}^{\mathbf{d+1}} \int_{\boldsymbol{\Gamma}} \phi_{\mathbf{p}} \ ,$$

where C depends on σ, δ, β and the $d+1st$ derivatives of $\tilde{F}^e$ on $\tilde{\Gamma}^e_\ell$.

Remark. Conditions sufficient to ensure the above stability requirement follow from the analysis in [3].

Hybrid Galerkin Algorithm and Complexity

In this section we give a more precise description of the hybrid Galerkin algorithm outlined in §4, together with an extension of its theory to the case of the overlapping decompositions introduced in §4. Letting H^μ, $\mu \in [-1,1]$, denote the usual the Sobolev space on Γ, we shall assume that the operator in (1) satisfies $\lambda I + \mathcal{K} : H^\mu \to H^{-\mu}$, and that the corresponding bilinear form $a(u,v) = ((\lambda I + \mathcal{K})u, v)$ is H^μ−elliptic (where $(\cdot,\cdot)$ denotes the dual pairing between $H^{-\mu}$ and H^μ). This assumption is widely satisfied in practice (see, e.g. [2]). For the Laplace differential operator the standard boundary integral equations are the single-layer potential and hypersingular equations (where $\lambda = 0$ in (1) and $\mu = -1/2, 1/2$ respectively) and the classical second-kind equation with the double-layer potential operator (where $\lambda = \pm 1/2$ and $\mu = 0$).

The usual Galerkin method seeks $U \in S_h := \text{span}\{\phi_p : p \in \mathcal{N}\}$ such that $a(U,V) = (f,V)$ for all $V \in \mathcal{S}_h$. This is equivalent to a linear system with matrix $\lambda M + K$, with M an easy to compute sparse mass matrix and K given in (2) above. It is well-known that U satisfies the quasi-optimal error estimate: $||u - U||_\mu \le Ch^{2-\mu}$ as $h \to 0$. The algorithm presented below provides an automatic procedure for computing an approximation $\tilde{K}$ to K in such a way that the corresponding discrete Galerkin solution $\tilde{U}$ also enjoys this convergence rate. The algorithm has a first phase in which the weights of (relatively low-order) node-based rules are computed on the charts described above and a second phase where these are used (where possible) to compute $K_{p,q}$, with conventional rules used otherwise. In determining what degree of precision should be employed the following quantities become important: $h_{p,q} := \max\{h_p, h_q\}$, and

$$\rho_{p,q} = \min\{|\boldsymbol{x} - \boldsymbol{y}| : \boldsymbol{x} \in \text{supp }\phi_p \cup \{\boldsymbol{x}_j : j \in J_p\},\ \boldsymbol{y} \in \text{supp }\phi_q \cup \{\boldsymbol{x}_{j'} : j' \in J_q\}\} .$$

The algorithm requires the user to choose the following parameters: real numbers $\delta, \sigma, C^* \ge 1$ and positive integers $d_{\min} \le d_{\max}$. Also $\chi := 1 - \mu - \min\{\mu, 0\}$.

Theorem 2 provides a typical convergence result for the algorithm described below. For simplicity we assume here that the mesh diameter $h < 1$ and that the mesh refinement is such that Nh^2 is bounded above and below as $h \to 0$. (Quasiuniformity is sufficient but not necessary for this.) More general versions, including extensions to anisotropically refined meshes are contained in [2, 3, 4]. The proof of Theorem 2 is obtained by extending the arguments in [3, 4], using Lemma 1 above to deal with errors in approximate weights of node-based rules.

Theorem 1 *If the stiffness matrix $\tilde{K}$ is computed by the algorithm below, and if the weights $\hat{\boldsymbol{w}}$ are computed from (10), where $\hat{\boldsymbol{b}}$ satisfies (11), then the corresponding discrete Galerkin solution $\tilde{U} \in \mathcal{S}_($ satisfies the optimal error estimate:*

$$||u - \tilde{U}||_{H^\mu} \le C(u, C^*, \sigma, \delta, \beta)h^{2-\mu} \quad \text{as} \quad h \to 0 \ . \tag{12}$$

The algorithm is called a "hybrid" Galerkin algorithm because it can be viewed as a blend of two traditional integral equation discretisations: the Galerkin and Nyström methods (for more details see [3]).

Hybrid Galerkin Algorithm

procedure generate_node_based_quadrature_rules;
begin
  for all $p \in \mathcal{N}$ **do**
    for all integers $d \in [d_{\min}, d_{\max}]$ **do**
    (i) Select at least $(d+1)(d+2)/2$ nodes $\{\boldsymbol{x}_j : j \in J_p(d)\}$ on Γ
      with the property $\max\{|\boldsymbol{x}_p - \boldsymbol{x}_j| : j \in J_p\} \leq \delta h_p$.
    (ii) Find $\{\hat{w}_j : j \in J_p(d)\}$ by solving (10) with $\hat{\boldsymbol{b}}$ satisfying (11).
    (iii) If the $\hat{w}_j$ satisfy $\sum_j |\hat{w}_j| \leq \sigma \sum_j \hat{w}_j$, then
      set $B_p(d) =$ "admissible" and store the $J_p(d)$ and $\hat{w}_j$.
      Otherwise set $B_p(d) =$ "inadmissible"
    end of loop over d
  end of loop over p
end;

The second phase of the algorithm then follows. For simplicity we give the algorithm here under the assumption that all node-based quadrature rules for $\int F\phi_p$ use nodes from suppϕ_p This is not necessary and a fully general version of the algorithm is given in [3].

procedure generate_hybrid_system_matrix;
begin
 for all $p, q \in \mathcal{N}$ **do**
 begin
  • Compute d_{pq}, the smallest positive integer satisfying

$$d_{p,q} \geq \frac{\chi + (1+\alpha)\log(C^* \rho_{p,q})/\log h_{p,q}}{1 - \log(C^* \rho_{p,q})/\log h_{p,q}} \tag{13}$$

 if $(d_{p,q} \in [d_{\min}, d_{\max}])$ and $(B_p(d_{pq}) =$ "admissible"$)$ and
  $(B_q(d_{pq}) =$ "admissible"$)$
 then
  • Compute $\tilde{K}_{pq}$ using the node-based rules obtained above.
 else
  • Compute $\tilde{K}_{pq}$ using conventional triangle-based rules,
   and ensure that

$$|K_{p,q} - \tilde{K}_{p,q}| \leq C h_{p,q}^{\chi+1} \int_\Gamma \phi_p \int_\Gamma \phi_q \ . \tag{14}$$

 end
end;

Note that (14) is always possible, with C independent of h, using triangle-based rules (see, e.g. [1, 6]). In [3] we analysed (in the case of quasi-uniform meshes) the complexity gains in using the hybrid algorithm compared with the conventional Galerkin method in the case of the three standard examples from harmonic potential theory: The single-layer, double-layer and hypersingular equations. For these three equations we computed the ratio of the number of entries of the matrix which will be computed by triangle-based rules to the number which will be computed by node-based rules. The asymptotic behaviour of this ratio is given in the following table

	$d_{\max} = 1$	$d_{\max} = 2$	$d_{\max} = 3$
single-layer	-	const	$h^{4/5}$
double-layer	const	$h^{4/5}$	$h^{4/3}$
hypersingular	$h^{2/5}$	h	$h^{10/7}$

Table 1

Thus, provided we choose $d_{\max} \geq 3$ in the hybrid algorithm, then, for all three model equations, the number of matrix entries computed by triangle-based quadrature becomes negligible compared to the number computed by node-based quadrature as $h \to 0$ and then the number of kernel evaluations required by the whole algorithm approaches N^2. Only $d_{\max} = 2$ is needed to ensure this property in the case of the double layer and $d_{\max} = 1$ in the case of the hypersingular equation. Since using triangle-based rules alone requires CN^2 kernel evaluations with moderately large C (see §4), we would expect that the hybrid algorithm would improve on conventional algorithms by a factor approaching C as $h \to 0$. The results in §4 show that the estimates Table 1 are realised in practice, even for moderately coarse meshes.

Numerical results

In this section we report on some numerical experiments for the the hybrid algorithm applied to the computation of the harmonic Dirichlet-Neumann map corresponding to a smooth bounded domain $\Omega \subset \mathbf{R}^3$ with boundary Γ. If Φ is harmonic in Ω then it is well-known that Φ satisfies *Green's third identity* on Γ:

$$\mathcal{V}\frac{\partial \Phi}{\partial n} - \mathcal{W}\Phi = \frac{1}{2}\Phi \ , \tag{15}$$

where $\partial/\partial n$ denotes differentiation in the outward normal direction from Ω and $\mathcal{V}$, $\mathcal{W}$ are, respectively, the single and double-layer potentials:

$$\mathcal{V}\phi(\boldsymbol{x}) = \frac{1}{4\pi}\int_\Gamma \frac{1}{|\boldsymbol{x}-\boldsymbol{y}|}\phi(\boldsymbol{y})dy \ , \ \ \mathcal{W}\phi(\boldsymbol{x}) = \frac{1}{4\pi}\int_\Gamma \frac{\partial}{\partial n}\left(\frac{1}{|\boldsymbol{x}-\boldsymbol{y}|}\right)\phi(\boldsymbol{y})dy \ . \tag{16}$$

If the Neumann data v of Φ is given on Γ, the Neumann-Dirichlet map can be computed by solving the integral equation

$$(\frac{1}{2}I + \mathcal{W})u = \mathcal{V}v \tag{17}$$

for the Dirichlet data u. Conversely if u is given, computation of the Dirichlet-Neumann map requires the solution of (17) for v. Both these maps are commonly used in domain decomposition procedures. These two equations constitute two different instances of problem (1), with $\lambda = 1/2$ in the first case and $\lambda = 0$ in the second. It is well known that the appropriate energy spaces for these equations is H^μ with (for the first equation) $\mu = 0$, and (for the second) $\mu = -1/2$. Since the surface Γ is here assumed smooth, the parameter α introduced in §4 is $\alpha = 1$ for both equations. Detailed experiments on using the hybrid Galerkin method for approximating and inverting the operator $\mathcal{V}$ (and hence computing, for example, the Dirichlet-Neumann map) which verify the theory above are given in [3]. In this paper we concentrate on the Neumann-Dirichlet map.

For given v, (17) has a unique solution u only up to an arbitrary additive constant. To avoid this difficulty we add the easily discretised term $\int_\Gamma u$ to the left-hand side of (17). The resulting equation has a unique solution which corresponds to one of the solutions of (17).

In our experiments we consider the specific case where Γ is the unit sphere $\{\boldsymbol{x} \in \mathbf{R}^3 : |\boldsymbol{x}| = \mathbf{1}\}$ and Φ is the harmonic function $\Phi(\boldsymbol{x}) = 1/|\boldsymbol{x} - (2,0,0)^T|$. Using the Neumann boundary data of this function as v we compute an approximation $\tilde{U}$ to the corresponding Dirichlet data u by solving (17) with Galerkin's method, using the hybrid algorithm to compute the stiffness matrix. In the tables below the L_2 norm of the error is approximated by the L_2 norm of the interpolant of the error, with the necessary integrals done using 4 point conical Gauss rules in each triangle (see also [3]).

To parametrise Γ we use 8 charts comprising the triangular faces of a double pyramid $\tilde{\Gamma}$ with nodes $(\pm 1, \pm 1, \pm 1)$, inscribed in Γ. The mapping $\eta : \tilde{\mathbf{\Gamma}} \to \mathbf{\Gamma}$ is given (globally in this case) by $\eta(\boldsymbol{x}) = \boldsymbol{x}/|\boldsymbol{x}|$. For each chart $\tilde{\Gamma}_\ell$, the extension map η_ℓ^{e} is again defined by this same formula. Meshes are constructed on $\tilde{\Gamma}$ by starting with the triangles comprising $\tilde{\Gamma}$ itself (Level 1) and performing 4 levels of hierarchical refinement, with each new level obtained from the previous by standard quadrasection of each triangle. This yields a sequence of five meshes on $\tilde{\Gamma}$ with 6, 18, 66, 258 and 1026 nodes, respectively. The finest level discretisations on $\tilde{\Gamma}$ and Γ are depicted in Fig. 1. There are many nodes on chart edges and so the overlapping procedure in §4 is very important to obtain optimal complexity here.

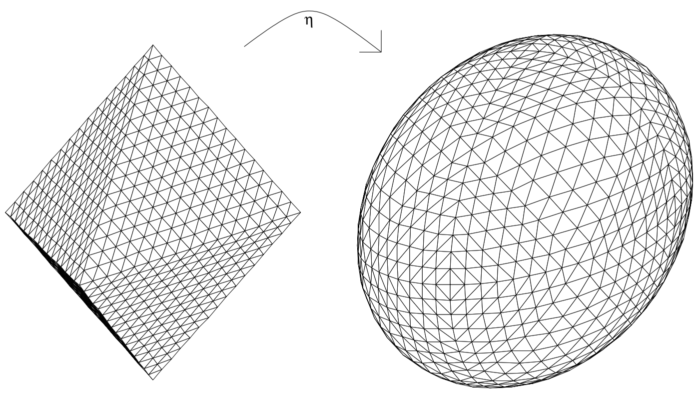


Figure 1 The unit sphere Γ and its 8 charts.

In our implementation of the hybrid algorithm we set $d_{\max} = d_{\min} = 2$ and we computed node-based rules of degree of precision 2 for $p \in \mathcal{N}$. To simplify the computation we omitted this computation for the 6 nodes which correspond to the vertices of the level 1 mesh, but we did it for all other nodes. This omission will have minimal effect on the complexity results reported below. For each p the quadrature points for the rule of precision 2 are chosen as the node $\boldsymbol{x}_p$ itself together with the 6 nodes connected to it. The resulting underdetermined systems were solved using the minimal norm algorithm in [3]. Thus $\delta = 1$ in Lemma 1. Rather than specify the stabilty constant σ in advance, we computed all quadrature rules and then recorded the observed values of σ below. In the implementation of (13) we set $h_{p,q} = h := \sqrt(2) * (2^{levels-1})$ with $levels = 1, 2, 3, 4, 5$, and $\rho_{p,q} = ||\boldsymbol{x}_p - \boldsymbol{x}_q||_2 - 2 * h$. When triangle-based rules are required we used conical Gauss rules of appropriate order and regularising transforms for the singular integrals (see [1]). For this problem $\alpha = 1$ and the energy norm is H^μ with $\mu = 0$ and so $\chi = 1$. We ran the algorithm for various choices of C^*. As C^* increases, the number of entries of the matrix computed by node-based rules increases. Nevertheless for fixed C^*, as the meshes are refined, the optimal error estimate in Theorem 2 will be realised.

In the tables below % is the percentage of the matrix actually computed with node-based rules and %max is the maximum possible percentage which could be computed in this way (i.e., the percentage of the matrix entries for which the supports of the basis functions do not intersect). In all the tables, columns headed "EOC" contain estimated orders of convergence (computed by extrapolation) for the numbers in the column immediately to its left.

			$C^* = 1/4$			$C^* = 1/2$		
N	σ	$\%_{\max}$	$\|\|u - \tilde{U}\|\|_{L_2}$	EOC	%	$\|\|u - \tilde{U}\|\|_{L_2}$	EOC	%
66	3.7	73	3.38(-3)		0	3.97(-3)		59
258	2.6	93	7.31(-4)	2.2	0	1.69(-3)	1.2	68
1026	3.3	98	1.87(-4)	2.0	0	2.25(-4)	2.9	70

Table 2

			$C^* = 1$			$C^* = 3$		
N	σ	$\%_{\max}$	$\|\|u - \tilde{U}\|\|_{L_2}$	EOC	%	$\|\|u - \tilde{U}\|\|_{L_2}$	EOC	%
66	3.7	59	3.97(-3)		59	3.97(-3)		59
258	2.6	88	1.60(-3)	1.3	87	2.00(-3)	1.0	88
1026	3.3	97	2.55(-4)	2.6	93	8.43(-4)	1.3	97

Table 3

The results clearly confirm the complexity predictions in Table 1. For small enough C^* (namely when $1/C^* > \text{diam}(\Gamma)$) no entries of the matrix are computed with node-based rules, the method is then the conventional Galerkin method and it converges with $O(h^2)$. As C^* increases, more entries are computed with the node-based rules. For each fixed C^* the order of convergence approaches $O(h^2)$ but with an asymptotic constant which increases with C^*. For the largest value of C^*, the rate of convergence has not yet achieved h^2 but 97% of the matrix is done using the cheap method. The moderate choice $C^* = 1$ achieves $O(h^2)$ but still computes 93% of the matrix cheaply. Table 4 compares the flops for the computation of the weights in the unconventional node-based rules with those needed to compute the "Nyström" matrix $k(\boldsymbol{x}_p, \boldsymbol{x}_q)$, for $p \neq q$, $p, q \in \mathcal{N}$. The first is growing with $O(N)$ and the second with $O(N^2)$. The number of flops for creating the stiffness matrix by triangle-based rules will on the other hand be typically 10-100 times the second column. Thus the cost of solving the small systems in phase 1 of the hybrid algorithm has a big pay-off in overall computation times.

N	weights	Nyström matrix
66	2.3(+5)	9.6(+4)
258	7.5(+5)	1.5(+6)
1026	2.5(+6)	2.3(+7)

Table 4: Comparison of flops for computing weights and Nyström matrix

REFERENCES

S. Erichsen and S. A. Sauter. Efficient automatic quadrature in 3D Galerkin BEM, *Comp. Meth. Appl. Mech. Engrg.*, 157:215–224 (1998).

I. G. Graham, W. Hackbusch and S. A. Sauter. Discrete boundary element methods on general meshes in 3D. Mathematics Preprint 97/19, University of Bath, to appear in *Numer. Math.*

I.G. Graham, W. Hackbusch and S.A. Sauter, Hybrid boundary elements: Theory and implementation. Mathematics Preprint 98-6, University of Kiel, to appear in *Numer. Math.*

I. G. Graham, W. Hackbusch and S. A. Sauter. The Hybrid Galerkin Method on Degenerate Meshes, in *Mathematical Aspects of Boundary Element Methods* (A.-M. Sändig and W.L. Wendland, Eds.), CRC Press, August 1999.

U. Langer, Parallel Iterative solution of symmetric coupled FE/BE equations via domain decomposition, In Contemporary Mathematics 157, AMS, Providence, 1994.

S. A. Sauter. Cubature techniques for 3D Galerkin BEM, in W. Hackbusch and G. Wittum, editors, *BEM: Implementation and Analysis of Advanced Algorithms*, Vieweg Verlag, 1996.

S. A. Sauter and A. Krapp. On the effect of numerical integration in the Galerkin boundary element method. *Numer. Math.* 74: 337-359, 1997.

A.H. Stroud. *Approximate Calculation of Multiple Integrals* Prentice Hall, Englewood Cliffs, 1973.

5

A domain decomposition method with Lagrange multipliers for linear elasticity

AXEL KLAWONN[1] & OLOF B. WIDLUND[2]

INTRODUCTION

In the last decade a lot of research has been carried out on nonoverlapping domain decomposition methods with Lagrange multipliers. In these methods the original domain is decomposed into nonoverlapping subdomains. The intersubdomain continuity is then enforced by Lagrange multipliers across the interface defined by the subdomain boundaries. A computationally very efficient member of this class of domain decomposition algorithms is the **F**inite **E**lement **T**earing and **I**nterconnecting (FETI) method introduced by Farhat and Roux [FR91]. In a variant of the FETI method introduced in Farhat, Mandel, and Roux [FMR94] a Neumann and a Dirichlet finite element problem is solved exactly on each subdomain, in each iteration.

In this paper, a new domain decomposition method with Lagrange multipliers is introduced by reformulating the preconditioned system of the FETI algorithm as a saddle point problem with both, primal and dual variables as unknowns. The resulting system is then solved using block–structured preconditioners in combination with a suitable Krylov space method. This approach avoids costly exact subdomain solves

1 Institut für Numerische und instrumentelle Mathematik, Westfälische Wilhelms-Universität Münster, Einsteinstr. 62, 48149 Münster, Germany. E-mail address: `klawonn@math.uni-muenster.de`, URL: `http://wwwmath.uni-muenster.de/math/u/klawonn`

2 Courant Institute of Mathematical Sciences, New York University, 251 Mercer Street, New York, NY 10012, USA. E-mail address: `wid lund@cs.nyu.edu`, URL: `http://www.cs.nyu.edu/cs/faculty/widlund`

Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

since it allows inexact subdomain solvers. Good features of the FETI method such as scalability and efficiency are preserved.

The remainder of this paper is organized as follows. In sect. 2, we present the equations of linear elasticity and in sect. 3 a finite element discretization thereof. In sect. 4, we review the FETI method and we introduce our new method in sect. 5. Finally, in sect. 6, we present numerical results for a cantilever problem.

THE EQUATIONS OF LINEAR ELASTICITY

The equations of linear elasticity model the displacement of a linear elastic material under the action of external and internal forces. We denote the elastic body by $\Omega \subset \mathbf{R}^d, d = 2, 3$, and its boundary by $\partial\Omega$ and we assume that one part of the boundary, Γ_0, is clamped, i.e. with homogeneous Dirichlet boundary conditions, and that the rest, $\Gamma_1 := \partial\Omega \setminus \Gamma_0$, is subject to a surface force $\mathbf{g}$, i.e. an inhomogeneous natural boundary condition. We can also introduce an internal volume force $\mathbf{f}$, e.g. gravity. The appropriate space for a variational formulation is the Sobolev space $H^1_{\Gamma_0}(\Omega) := \{\mathbf{v} \in \mathbf{H^1(\Omega)^d} : \mathbf{v}_{|\Gamma_0} = \mathbf{0}\}$. The linear elasticity problem consists in finding the displacement $\mathbf{u} \in \mathbf{H^1_{\Gamma_0}(\Omega)}$ of the elastic body Ω, such that

$$2\mu \int_\Omega \varepsilon(\mathbf{u}) : \varepsilon(\mathbf{v}) \mathbf{dx} \; + \; \lambda \int_{\mathbf{\Omega}} \operatorname{div} \mathbf{u} \operatorname{div} \mathbf{v} \, \mathbf{dx} = \langle \mathbf{F}, \mathbf{v} \rangle \quad \forall \mathbf{v} \in \mathbf{H^1_{\Gamma_0}(\Omega)}. \tag{1}$$

Here μ and λ are the Lamé constants, $\varepsilon_{ij}(\mathbf{u}) := \frac{\mathbf{1}}{\mathbf{2}}(\frac{\partial \mathbf{u_i}}{\partial \mathbf{x_j}} + \frac{\partial \mathbf{u_j}}{\partial \mathbf{x_i}})$ is the linearized strain tensor, $\varepsilon(\mathbf{u}) : \varepsilon(\mathbf{v}) = \sum_{\mathbf{i,j=1}}^{\mathbf{d}} \varepsilon_{\mathbf{ij}}(\mathbf{u})\varepsilon_{\mathbf{ij}}(\mathbf{v})$, and $\langle \mathbf{F}, \mathbf{v} \rangle := \sum_{\mathbf{i=1}}^{\mathbf{d}} \int_{\mathbf{\Omega}} \mathbf{f_i v_i \, dx} + \sum_{\mathbf{i=1}}^{\mathbf{d}} \int_{\mathbf{\Gamma_1}} \mathbf{g_i v_i \, d\sigma}$. The associated bilinear form of linear elasticity is

$$a(\mathbf{u}, \mathbf{v}) = \mathbf{2}\mu \int_{\mathbf{\Omega}} \varepsilon(\mathbf{u}) : \varepsilon(\mathbf{v}) \mathbf{dx} \; + \; \lambda \int_{\mathbf{\Omega}} \operatorname{div} \mathbf{u} \operatorname{div} \mathbf{v} \, \mathbf{dx}.$$

In this article, we only consider the case of compressible elasticity. This means that the Lamé parameter λ remains bounded.

FINITE ELEMENTS AND THE DISCRETE PROBLEM

Since we only consider compressible elastic materials, it follows from Korn's first inequality, cf. Ciarlet [Cia88], that the bilinear form $a(\cdot, \cdot)$ is uniformly elliptic. Therefore we can successfully discretize the system (1) with low–order, conforming finite elements, such as linear or bi-/trilinear elements.

We assume that a triangulation τ^h of Ω is given which is shape regular and has a typical element diameter of h. We denote by $\mathbf{H^h(\Omega)}$ the corresponding conforming space of finite element functions, e.g. piecewise linear or bi-/trilinear continuous functions. Our goal is to solve the discrete problem

$$a(\mathbf{u_h}, \mathbf{v_h}) = \langle \mathbf{F}, \mathbf{v_h} \rangle \quad \forall \mathbf{v_h} \in \mathbf{H^h(\Omega)}. \tag{2}$$

In the following sections, we work exclusively with the discrete problem. To avoid unnecessary notation, we drop the subscript h from now on.

A REVIEW OF THE FETI METHOD

In this section, we give a brief review of the original FETI method proposed by Farhat and Roux, cf. [FR91]. For more detailed descriptions and proofs, we refer to [FMR94, RF98] and the references therein.

Let the domain $\Omega \subset \mathbf{R}^d, d = 2,3$, be decomposed into N non-overlapping subdomains $\Omega_i, i = 1,\ldots,N$, and the finite element nodes of neighboring subdomains match on the interface $\Gamma := \left(\bigcup_{i=1}^N \partial\Omega_i\right) \setminus \partial\Omega$.

Let the corresponding conforming finite element spaces be $\mathbf{W_i} = \mathbf{W^h(\Omega_i)}, \mathbf{i} = \mathbf{1},\ldots,\mathbf{N}$, and let $\mathbf{W} := \prod_{\mathbf{i=1}}^{\mathbf{N}} \mathbf{W_i}$ denote the associated product space.

The finite element spaces can always be identified with $\mathbf{R}^m$ for a suitable m. We mark these spaces in the following with an underline, e.g. $\mathbf{W}$ is identified with $\underline{\mathbf{W}}$. Analogously, we denote by $\underline{\mathbf{u}}$ the vector of nodal values associated with the finite element function $\mathbf{u}$.

For each subdomain $\Omega_i, i = 1,\ldots,N$, we assemble local stiffness matrices K_i and local load vectors $\underline{f}_i$. We denote by $\underline{\mathbf{u}}_i$ the local vectors of nodal values.

We can now reformulate (2) as a minimization problem with constraints given by intersubdomain continuity conditions:

Find $\underline{\mathbf{u}} \in \underline{\mathbf{W}}$, such that

$$\left.\begin{array}{c} J(\underline{\mathbf{u}}) := \frac{1}{2}\underline{\mathbf{u}}^t K\underline{\mathbf{u}} - \underline{\mathbf{f}}^t\underline{\mathbf{u}} \to \ \min \\ B\underline{\mathbf{u}} = 0 \end{array}\right\} \tag{3}$$

where $\underline{\mathbf{u}} = [\underline{\mathbf{u}}_1,\ldots,\underline{\mathbf{u}}_N]^t$,$\underline{\mathbf{f}} = [\underline{\mathbf{f}}_1,\ldots,\underline{\mathbf{f}}_N]^t$, and $K = diag_{i=1}^N K_i$ is a block–diagonal matrix.

The matrix $B = [B_1,\ldots,B_N]$ is constructed such that the components of any vector $\underline{\mathbf{u}}$, which are associated with the same node on Γ, coincide when $B\underline{\mathbf{u}} = 0$. The local stiffness matrices K_i are positive semidefinite. We can assume that the problem (3) is uniquely solvable which is equivalent to $\ker K \cap \ker B = \{0\}$, i.e. K is invertible on the nullspace of B. This is a consequence of the original finite element model being elliptic.

By introducing Lagrange multipliers $\underline{\lambda} \in \underline{\mathbf{U}} := \operatorname{range} B$ to enforce the constraint $B\underline{\mathbf{u}} = 0$, we obtain a saddle point problem from (3):

Find $(\underline{\mathbf{u}}, \underline{\lambda}) \in \underline{\mathbf{W}} \times \underline{\mathbf{U}}$, such that

$$\left.\begin{array}{ccccc} K\underline{\mathbf{u}} & + & B^t\underline{\lambda} & = & \underline{\mathbf{f}} \\ B\underline{\mathbf{u}} & & & = & 0 \end{array}\right\} \tag{4}$$

We will also use a full rank matrix R, such that $\operatorname{range} R = \ker K$.

The solution of the first equation in (4) exists if and only if $\underline{\mathbf{f}} - B^t\underline{\lambda} \in \operatorname{range} K$.

Eliminating the primal variables $\underline{\mathbf{u}}$ leads to

$$\left.\begin{array}{ccc} PF\underline{\lambda} & = & P\underline{\mathbf{d}} \\ G^t\underline{\lambda} & = & \underline{\mathbf{e}} \end{array}\right\} \tag{5}$$

with $G := BR, F := BK^\dagger B^t, \underline{\mathbf{d}} := BK^\dagger\underline{\mathbf{f}}, P := I - G(G^tG)^{-1}G^t$, and $\underline{\mathbf{e}} := R^t\underline{\mathbf{f}}$. Here $K^\dagger$ is the pseudoinverse of K which provides a solution orthogonal to the nullspace of K. Note that P is an orthogonal projection from $\underline{\mathbf{U}}$ onto $\ker G^t$. The constraint $G^t \underline{\lambda} = \underline{\mathbf{e}}$ ensures that $\underline{\mathbf{f}} - B^t \underline{\lambda}$ is in the range of K.

We define the space of admissible increments by

$$\underline{\mathbf{V}} := \{\underline{\mu} \in \underline{\mathbf{U}} : \underline{\mu} \perp B\underline{\mathbf{w}} = 0 \; \forall \underline{\mathbf{w}} \in \ker K\} = \ker G^t.$$

The original FETI method is a conjugate gradient method in the space $\underline{\mathbf{V}}$ applied to

$$PF\underline{\lambda} = P\underline{\mathbf{d}}, \qquad \underline{\lambda} \in \underline{\lambda}_0 + \underline{\mathbf{V}} \tag{6}$$

with an initial approximation $\underline{\lambda}_0$ chosen such that $G^t \underline{\lambda}_0 = \underline{\mathbf{e}}$. Let D be a diagonal matrix. The preconditioner M^{-1} introduced in Farhat, Mandel, and Roux [FMR94] is of the form

$$M^{-1} = B \begin{bmatrix} O & O \\ O & D^{-1} S D^{-1} \end{bmatrix} B^t.$$

The matrix S is the Schur complement of K obtained by eliminating the interior degrees of freedom. This can obviously be done for all subdomains at the same time. The resulting matrix S is block–diagonal, and operates only on the degrees of freedom on the subdomain boundaries. In the simplest case, the diagonal matrix D is chosen as the identity; this is the choice in the original FETI method by Farhat and Roux. Another possibility, which leads to faster convergence, is to choose D as a diagonal matrix where the main diagonal contains the number of subdomains to which the interface node belongs. This multiplicity scaling is discussed in Rixen and Farhat [RF98].

In the application of this preconditioner, N independent Dirichlet problems have to be solved at each iteration step; it is known as the Dirichlet preconditioner.

To keep the search directions of the conjugate gradient method in the space $\underline{\mathbf{V}}$, the application of the preconditioner M^{-1} has to be followed by another application of the projection P. Hence, the Dirichlet variant of the FETI method is the conjugate gradient algorithm applied to

$$PM^{-1}PF\,\underline{\lambda} = PM^{-1}P\,\underline{\mathbf{d}}, \qquad \underline{\lambda} \in \underline{\lambda}_0 + \underline{\mathbf{V}}. \tag{7}$$

It has been shown by Mandel and Tezaur [MT96] that the condition number of the original FETI method (D=I) satisfies $\kappa(PM^{-1}PF) \leq C\,(1 + \log(H/h))^3$, where H/h is determined by the number of degrees of freedom of a subdomain and C is a constant which is independent of h, and H.

In Tezaur [Tez98], a condition number estimate on the order of $(1 + \log(H/h))^2$ is given for an algebraic FETI method developed by Park et al. [PJF97].

For a modification of the FETI preconditioner with almost no extra cost, Klawonn and Widlund [KW99b] show that the condition number is bounded asymptotically by $C\,(1 + \log(H/h)^2$.

THE NEW METHOD USING INEXACT SOLVERS

If $\lambda = \mu + \lambda_0$ with $\mu \in V$ then, we can reformulate (6) as

$$PF\underline{\mu} = P(\underline{\mathbf{d}} - F\underline{\lambda}_0)$$

which is equivalent to

$$PBK^{\dagger}B^t\underline{\mu} = PBK^{\dagger}(\underline{\mathbf{f}} - B^t\underline{\lambda}_0). \tag{8}$$

Keeping in mind that $\underline{\lambda} = \underline{\mu} + \underline{\lambda}_0$ and that $\underline{\mathbf{u}} = K^{\dagger}(\underline{\mathbf{f}} - B^t\underline{\lambda}) + R\underline{\alpha}$, we see immediately that the solution of (8) can also be obtained by solving

$$\begin{bmatrix} K & B^t \\ PB & O \end{bmatrix} \begin{bmatrix} \underline{\mathbf{u}} \\ \underline{\mu} \end{bmatrix} = \begin{bmatrix} \underline{\mathbf{f}} - B^t\underline{\lambda}_0 \\ 0 \end{bmatrix}.$$

Since $\underline{\mu} \in \mathbf{\underline{V}}$ and $P\underline{\mu} = \underline{\mu}$ for $\underline{\mu} \in \mathbf{\underline{V}}$, we can make the system matrix symmetric

$$\begin{bmatrix} K & (PB)^t \\ PB & O \end{bmatrix} \begin{bmatrix} \underline{\mathbf{u}} \\ \underline{\mu} \end{bmatrix} = \begin{bmatrix} \underline{\mathbf{f}} - B^t\underline{\lambda}_0 \\ 0 \end{bmatrix}. \tag{9}$$

Note that in this formulation we are not enforcing $B\underline{\mathbf{u}} = 0$ but only its projected version $PB\underline{\mathbf{u}} = 0$. The addition of an element of the nullspace of K does not change the solution $\underline{\mathbf{u}}$ of the first equation in (9). We use this fact to post-process $\underline{\mathbf{u}}$, such that $B\underline{\mathbf{u}} = 0$ is finally satisfied. This can be done by setting $\underline{\mathbf{u}}_{cor} := \underline{\mathbf{u}} - R(G^tG)^{-1}G^tB\underline{\mathbf{u}}$. It follows that $B\underline{\mathbf{u}}_{cor} = PB\underline{\mathbf{u}} = 0$.

For the solution of the saddle point problem (9), we propose a preconditioned conjugate residual method with a block–diagonal preconditioner; for a detailed description of this algorithm, see Hackbusch [Hac94] or Klawonn [Kla95, Kla98].

Our preconditioner has the form

$$\mathcal{B} = \begin{bmatrix} \widehat{K} & O \\ O & \widehat{M} \end{bmatrix}$$

Here, $\widehat{K}$ is symmetric and spectrally equivalent to $K + D_H\,Q$ where $Q = diag_{i=1}^{N} Q_i$ is a block–diagonal matrix with Q_i being the mass matrices associated with the discretizations on Ω_i and $D_H = (d_{ii})_{i=1,\ldots,N}$ a diagonal matrix with $d_{ii} = H_i^{-2}, i = 1,\ldots,N$. Here, $H_i, i = 1,\ldots,N$ denotes the diameters of the subdomain $\Omega_i, i = 1,\ldots,N$. We further assume that $\widehat{M}$ is symmetric and spectrally equivalent to M, i.e. we assume the existence of constants $k_0, k_1, m_0, m_1 > 0$, such that

$$\begin{array}{llcccll} k_0 & \underline{\mathbf{u}}^t(K + D_HQ)\underline{\mathbf{u}} & \leq & \underline{\mathbf{u}}^t\widehat{K}\underline{\mathbf{u}} & \leq & k_1 & \underline{\mathbf{u}}^t(K + D_HQ)\underline{\mathbf{u}} \quad \forall \underline{\mathbf{u}} \in \mathbf{\underline{W}}, \\ m_0 & \underline{\lambda}^tM\underline{\lambda} & \leq & \underline{\lambda}^t\widehat{M}\underline{\lambda} & \leq & m_1 & \underline{\lambda}^tM\underline{\lambda} \quad \forall \underline{\lambda} \in \mathbf{\underline{V}}. \end{array} \tag{10}$$

We can assume that the constants k_0, k_1, m_0, and m_1 are independent of the mesh size and the number of subdomains but also note that preconditioners with bounds that are weakly dependent on the mesh size, e.g. ILU methods, are of practical interest.

From these assumptions it is clear that our preconditioner $\mathcal{B}$ is symmetric, positive definite and thus can be used with the preconditioned conjugate residual method. In order to have a computationally efficient preconditioner, we must assume, in addition, that the application of $\widehat{K}^{-1}$ and $\widehat{M}^{-1}$ to a vector can be computed at a low cost.

To guarantee that the iterates belong to range K, we introduce the projection P_R onto range K by

$$P_R := I - R(R^tR)^{-1}R^t.$$

We recall that range $R = \ker K$ and note that P_R is a block matrix with a 3×3 block for each interior subdomain; the expense of applying P_R to a vector is therefore very modest.

The resulting domain decomposition method is the conjugate residual algorithm using the $\mathcal{B}-$inner product applied to the preconditioned system

$$\mathcal{B}^{-1}\mathcal{A}\underline{\mathbf{x}} = \mathcal{B}^{-1}\underline{\mathbf{F}}$$

with

$$\mathcal{A} = \begin{bmatrix} K & (PB)^t \\ PB & O \end{bmatrix}, \mathcal{B}^{-1} = \begin{bmatrix} P_R\widehat{K}^{-1}P_R^t & O \\ O & P\widehat{M}^{-1}P^t \end{bmatrix},$$

$$\underline{\mathbf{x}} = \begin{bmatrix} \underline{\mathbf{u}} \\ \underline{\mu} \end{bmatrix}, \underline{\mathbf{F}} = \begin{bmatrix} \underline{\mathbf{f}} - B^t\underline{\lambda}_0 \\ 0 \end{bmatrix}.$$

We note that it is easy to see that only two matrix-vector products with the projection P and one with the projection P_R are required in each step.

A proof of the following theorem is given in Theorem 2 in [KW99a].

Theorem 1 *For $D = I$, we have*

$$\kappa(\mathcal{B}^{-1}\mathcal{A}) \leq C\,(1 + \log(H/h))^2,$$

with a constant C independent of H, h.

NUMERICAL RESULTS

We have applied our domain decomposition method to a two-dimensional, linear elasticity problem. We consider a plane stress problem with a Poisson ratio $\nu = 0.3$ and an elasticity modulus $E = 2.1 \cdot 10^{11} N/m^2$ which models steel. The domain Ω is a cantilever unit square fixed on the left hand side and free on the other three edges except at the upper right corner, where we impose a point force that has equal components in the positive x and y directions equal to $10^5 N$. We discretize this problem by using bilinear finite elements with a meshsize h.

All computations were performed in MATLAB 5.0. As our Krylov space method, we use the preconditioned conjugate residual method with a zero initial guess. The stopping criterion is the relative reduction of the initial residual by 10^{-6}, i.e. $\|\underline{\mathbf{r}}_n\|_2/\|\underline{\mathbf{r}}_0\|_2 < 10^{-6}$, where $\underline{\mathbf{r}}_n, \underline{\mathbf{r}}_0$ are the $n-$th and initial residual, respectively. In our experiments, we have used a simplified implementation without the projector P_R which is needed for the theoretical analysis. For a more detailed numerical study including this projector, we refer to Klawonn and Widlund [KW99a].

We decompose our domain Ω into $N \times N$ square subdomains with $H := 1/N$. In order to analyze the numerical scalability of the method, we have carried out two different types of experiments for different combinations of preconditioners $\widehat{K}$ and $\widehat{M}$. In our first set of runs, we have kept the subdomain size H/h fixed and increased the number of subdomains and thus the overall problem size. Our second series of experiments is carried out with a fixed number of subdomains and an increasing subdomain size H/h resulting in an increased $1/h$.

Table 1 $H/h = 8$, (I) : $\widehat{K} = K + 1/H^2 M_Q$ and $\widehat{M} = M$, (II) : $\widehat{K} = K + 1/H^2 M_Q$ and $\widehat{M}$ using ILU(0), (III) : $\widehat{K}$ ILU(10^{-4}) of $K + 1/H^2 M_Q$ and $\widehat{M} = M$, (IV) : $\widehat{K}$ using ILU(10^{-4}) of $K + 1/H^2 M_Q$ and $\widehat{M}$ using ILU(0), MS : D= multiplicity scaling, I : D=Identity.

NEW METHOD		Iter (I)		Iter (II)		Iter (III)		Iter (IV)	
$1/h$	$1/H$	MS	I	MS	I	MS	I	MS	I
16	2	11	19	11	19	21	33	21	34
32	4	17	27	17	27	29	60	31	61
64	8	21	33	23	33	37	73	39	76
96	12	21	39	23	39	44	77	45	81
128	16	21	39	25	41	47	80	51	84

Table 2 $H = 1/4$; (I) : $\widehat{K} = K + 1/H^2 M_Q$ and $\widehat{M} = M$, (II) : $\widehat{K} = K + 1/H^2 M_Q$ and $\widehat{M}$ using ILU(0), (III) : $\widehat{K}$ ILU(10^{-4}) of $K + 1/H^2 M_Q$ and $\widehat{M} = M$, (IV) : $\widehat{K}$ using ILU(10^{-4}) of $K + 1/H^2 M_Q$ and $\widehat{M}$ using ILU(0), MS : D= multiplicity scaling, I : D=Identity.

NEW METHOD		Iter (I)		Iter (II)		Iter (III)		Iter (IV)	
$1/h$	H/h	MS	I	MS	I	MS	I	MS	I
16	4	15	25	15	25	27	61	27	59
32	8	17	27	17	27	31	60	32	62
64	16	19	29	21	29	40	77	45	81
128	32	19	33	23	43	98	194	116	194

In order to see how our method behaves in the best possible case, we first report on results for $\widehat{K} = K + \frac{1}{H^2}Q$ and $\widehat{M} = M$, cf. Tables 1,2, Iter (I). To gain insight into the convergence behavior in the case of inexact blocks $\widehat{K}$ and $\widehat{M}$, we have run experiments with preconditioners based on incomplete Cholesky factorizations (ILU). In the following, ILU(0) stands for incomplete Cholesky factorizations with no fill in allowed and ILU(tol) represents a threshold ILU factorization with a threshold of tol as provided in MATLAB 5.0, i.e. all entries in a column of the factor L of the Cholesky factorization are dropped if the magnitude of the entry is smaller than the drop tolerance tol times the norm of that column. Three different combinations are considered, cf. (II)–(IV) in Tables 1,2. In the experiments presented in the Tables 1,2, $\widehat{M}$ using ILU(0) means that the Schur complement is computed inexactly using ILU(0).

For all cases, we present results for $\widehat{M}$ constructed using $D = I$ as well as the multiplicity scaling $D = MS$, cf. section 5. As in the original FETI algorithm, the convergence is considerably faster for the case of multiplicity scaling. With this scaling the asymptotic convergence rate is also reached earlier than for $D = I$.

For a more detailed numerical study, we refer to [KW99a].

Acknowledgments

The first author wishes to thank Daniel Rixen for parts of his FETI MATLAB code and for interesting discussions.

REFERENCES

[Cia88] Ciarlet P. G. (1988) *Mathematical Elasticity Volume I: Three–Dimensional Elasticity*. North–Holland.

[FMR94] Farhat C., Mandel J., and Roux F.-X. (1994) Optimal Convergence Properties of the FETI Domain Decomposition Method. *Comput. Methods Appl. Mech. Engng.* 115: 367–388.

[FR91] Farhat C. and Roux F.-X. (1991) A Method of Finite Element Tearing and Interconnecting and its Parallel Solution Algorithm. *Int. J. Numer. Meth. Engng.* 32: 1205–1227.

[Hac94] Hackbusch W. (1994) *Iterative Solution of Large Sparse Systems of Equations*. Springer, New York.

[Kla95] Klawonn A. (1995) *Preconditioners for Indefinite Problems*. PhD thesis, Westfälische Wilhelms-Universität Münster. TR 716, February 1996, Courant Institute of Mathematical Sciences, New York University, USA.

[Kla98] Klawonn A. (March 1998) An optimal preconditioner for a class of saddle point problems with a penalty term. *SIAM J. Sci. Comp.* 19(2): 540–552.

[KW99a] Klawonn A. and Widlund O. B. (February 1999) A domain decomposition method with Lagrange multipliers for linear elasticity. Technical Report TR 780, Courant Insitute of Mathematical Sciences, New York University, New York, USA. `URL: file://cs.nyu.edu/pub/tech-reports/tr780.ps.gz`.

[KW99b] Klawonn A. and Widlund O. B. (1999) FETI and Neumann–Neumann Iterative Substructuring Methods: Connections and New Results. Technical report, Computer Science Department, Courant Institute of Mathematical Sciences. To appear.

[MT96] Mandel J. and Tezaur R. (1996) Convergence of a Substructuring Method with Lagrange Multipliers. *Numer. Math.* 73: 473–487.

[PJF97] Park K., Justino M., and Felippa C. (1997) An algebraically partitioned FETI method for parallel structural analysis: algorithm description. *Int. J. Num. Meth. Engrg.* 40: 2717–2737.

[RF98] Rixen D. and Farhat C. (1998) Preconditioning the FETI and balancing domain decomposition methods for problems with intra- and inter-subdomain coefficient jumps. In Bjørstad P., Espedal M., and Keyes D. (eds) *Proceedings of the Ninth International Conference on Domain Decomposition Methods in Science and Engineering, Bergen, Norway, June 1996*, pages 472–479. URL:`http://www.ddm.org/DD9/Rixen.ps.gz`.

[Tez98] Tezaur R. (1998) *Analysis of Lagrange multiplier based domain decomposition*. PhD thesis, University of Colorado at Denver.

6

Sparse Grid Spectral Methods and some Results from Approximation Theory

F. G. KUPKA[1,2]

INTRODUCTION

The sparse grid approach for the solution of partial differential equations (PDEs) was recently extended to spectral and pseudospectral methods. Here, we summarize how results from the approximation theory for Sobolev spaces of functions with bounded mixed derivative can be used to derive asymptotic error estimates for the approximation and interpolation error of functions when either the basis functions are from a *step hyperbolic cross* or when the interpolation is done on a *sparse grid*. For the special case of periodic boundary conditions we discuss how the classical convergence analysis for Galerkin and pseudospectral methods can be used for sparse grid spectral methods. Applying the new methods to simple equations has allowed a comparison with exact solutions. Some results of this comparison are summarized as well as possible applications and extensions to other variants of sparse grid methods.

Zenger [Zen91] has suggested a *sparse grid finite element method* with an approximation order of $O(h^2|\log h|)$ for a finite element subspace of size $\dim \check{X}_N = O(N \log N)$ in comparison to $O(h^2)$ obtained with a subspace of size $\dim X_N = O(N^2)$ by a traditional *full grid finite element method*. His result requires that the mixed derivative $\partial^4 u(x,y)/\partial x^2 \partial y^2$ of the solution u be bounded. The approach was extended to higher order finite element methods and more general domains. Later,

1 Institut für Mathematik, Strudlhofgasse 4, A-1090 Wien, Austria
2 also affiliated to Institut für Astronomie, Türkenschanzstraße 17, A-1180 Wien, Austria; supported by the Austrian science foundation FWF, project P11882-PHY
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

the sparse grid concept was also applied to finite difference methods (cf. Griebel and Schiekofer [GS98]).

The basic idea behind sparse grid techniques for PDEs has been applied in other branches of mathematics: while classes of functions with bounded mixed derivatives were studied for the definition of good lattice points for number theoretical cubature formulas (see Temlyakov [Tem93] for a review), hyperbolic cross points have played a role in the development of Boolean algebraic methods for interpolation on rectangular domains in two dimensions (cf. the review of Delvos and Schempp [DS89]). Beginning with Babenko [Bab60] and Smolyak [Smo63], approximation theory has seen particular efforts in the description of both approximation and interpolation of (mostly) periodic functions with bounded mixed derivatives. Many results and original references can be found in Temlyakov [Tem89] and [Tem93].

Thus, approximation theory provides a solid basis for the development of *sparse grid spectral methods*. Interpolation of — particularly bivariate — periodic functions and interpolation by Gauß–Chebyshev methods on sparse grids were studied for various Banach type Sobolev spaces with dominating mixed derivatives by Pöplau and Sprengel in [Pöp95], [PS97], [Spr97a], and [Spr97b]. Their results can be used for the numerical analysis of pseudospectral (collocation) methods. In turn, periodic approximation and interpolation on the hyperbolic cross and on sparse grids for some Hilbert type Sobolev spaces were investigated independently by Kupka [Kup97] who used the results for the convergence analysis of spectral and pseudospectral methods for the Helmholtz, linear advection, and heat equation. These equations are also used for numerical tests of the current new method. In what follows, we review some of the results.

APPROXIMATION AND INTERPOLATION ON SOBOLEV SPACES WITH DOMINANT MIXED DERIVATIVE

To motivate the results for Sobolev spaces with dominant mixed derivative, we recall that the standard Sobolev spaces of Lebesgue integrable, 2π–periodic functions $v \in L^q(\Omega)$ on the torus $\boldsymbol{T}^d = \Omega$ are defined by

$$H_p^{r,q}(\Omega) = \{v \in L^q(\Omega) \mid D^\alpha v \in L^q(\Omega) \; \forall \; \alpha : 0 \le |\alpha| \le r\}, \tag{1}$$

where $D^\alpha v = \partial^{|\alpha|} v / (\partial x_1^{\alpha_1} \cdots \partial x_d^{\alpha_d})$ with $0 \le |\alpha| \le m$, m is a non-negative integer, and $|\alpha| := \sum_{i=1}^d \alpha_i$ for $\alpha = (\alpha_1, \ldots, \alpha_d)$, is taken in the sense of periodic distributions (indicated by a subscript p). Moreover, $H_p^{r,q}(\Omega)$ is equipped with the norm

$$\|v\|_{r,q} := \|v\|_{H_p^{r,q}(\Omega)} = \left(\sum_{|\alpha| \le r} \|D^\alpha v(\mathbf{x})\|_{L^q(\Omega)}^q \right)^{1/q}. \tag{2}$$

For the $q = 2$ case, one can alternatively define a norm for functions $v \in L^2(\Omega)$ by ($\boldsymbol{Z}^d$ denotes the Cartesian product of the field of integers $\boldsymbol{Z}$)

$$|||v|||_r = (2\pi)^{d/2} \left(\sum_{\mathbf{k}\in \boldsymbol{Z}^d} (1+\mathbf{k}^2)^r |\hat{v}_{\mathbf{k}}|^2 \right)^{1/2} . \tag{3}$$

The space $\dot{H}^r(\Omega)$ consists of all $v \in L^2(\Omega)$ with finite norm $|||\cdot|||_r$,

$$\dot{H}^r(\Omega) = \left\{ v \in L^2(\Omega) \mid |||v|||_r < +\infty \right\} . \tag{4}$$

Both spaces $\dot{H}^r_p(\Omega)$ and $H^r_p(\Omega) \equiv H^{r,2}_p(\Omega)$ and their norms are equivalent.

One way to define Sobolev spaces with dominant mixed derivative on the torus $\boldsymbol{T}^d$ is (we note that permutations P_{rd} and general derivatives D^α are useful in proofs):

Definition 1 *Let P_{rd} be the set of all permutations of $D^\alpha u(\mathbf{x}) = \frac{\partial^{rd} u(\mathbf{x})}{\partial x_1^r \cdots \partial x_d^r}$, $r \geq 0$ be a natural number, $1 \leq q < \infty$, and d be the algebraic dimension of $\boldsymbol{T}^d$. Moreover, let $\alpha = \sum_{j=1}^d \alpha_j$ and $\alpha_j = r$. The periodic Sobolev spaces with dominant mixed derivative on $\boldsymbol{T}^d$ can be defined as*

$$S^{r,d}_{q,p}(\boldsymbol{T}^d) := S^{r,d}_{q,p} = \left\{ u \mid D^\alpha u \in L^q(\boldsymbol{T}^d) \right\} \tag{5}$$

equipped with the norm ($\mathbf{x} \in \boldsymbol{T}^d$)

$$||u(\mathbf{x})||_{r,d,q} = \left(||u(\mathbf{x})||_q^q + \sum_{j=1}^d || \frac{\partial^r u(\mathbf{x})}{\partial x_j^r} ||_q^q + \sum_{P_{rd}} || \frac{\partial^{rd} u(\mathbf{x})}{\partial x_1^r \cdots \partial x_d^r} ||_q^q \right)^{1/q} . \tag{6}$$

For $q = 2$, the definition of an inner product is as straightforward as for $H^r_p(\Omega)$. Likewise, the $S^{r,d}_{q,p}$ are Banach (or even Hilbert) spaces. The new spaces can also be established for the case $q = \infty$. It can be shown that $S^{r,d}_{2,p}$ and $H^r_p(\boldsymbol{T}^d)$ satisfy the embedding relations $H^{rd}_p(\boldsymbol{T}^d) \hookrightarrow S^{r,d}_{2,p} \hookrightarrow H^r_p(\boldsymbol{T}^d)$ for all $r \geq 0$ and for all $d \geq 1$. Moreover, $H^{r+1}_p(\boldsymbol{T}^d) \hookrightarrow H^r_p(\boldsymbol{T}^d)$ and $S^{r+1,d}_{2,p} \hookrightarrow S^{r,d}_{2,p}$, whence in particular $S^{r,d}_{2,p} \hookrightarrow L^2(\boldsymbol{T}^d)$. Just like the classical Sobolev spaces $H^m_p(\boldsymbol{T}^d)$, the spaces $S^{m,d}_{2,p}$ may be defined by means of the Fourier transform:

Definition 2 *Let m be a non-negative integer and let d be as in Definition 1. The periodic Hilbert spaces $\dot{S}^{m,d}$ with dominant mixed derivative on the torus $\boldsymbol{T}^d$ are defined as*

$$\dot{S}^{m,d} := \dot{S}^{m,d}(\boldsymbol{T}^d) = \left\{ u \in L^2(\boldsymbol{T}^d) \mid |||u|||_{m,d} < +\infty \right\} . \tag{7}$$

Here, we use the following norm for $\dot{S}^{m,d}$:

$$|||u|||_{m,d} = (2\pi)^{d/2} \left(\sum_{\mathbf{k}\in \boldsymbol{Z}^d} \left(1 + \sum_{j=1}^d k_j^{2m} + \sum_{P_{rd}} \prod_{j=1}^d k_j^{2m} \right) |\hat{u}_{\mathbf{k}}|^2 \right)^{1/2} . \tag{8}$$

The spaces $S^{m,d}_{2,p}$ and $\dot{S}^{m,d}$ can be identified with each other, i.e., they are equivalent for all $m \geq 0$ and $d \geq 1$ and their norms coincide. Proofs can be found in Kupka [Kup97] together with some references for the preceeding relations.

The standard projection operator for functions $v \in H^r_p(\boldsymbol{T}^d)$ is given by $P^f_N v = \sum_{\mathbf{k}\in J^f_N} \hat{v}_{\mathbf{k}} e^{i\mathbf{kx}} = \sum_{\mathbf{k}\in J^f_N} \frac{1}{(2\pi)^d}(v, e^{i\mathbf{kx}})e^{i\mathbf{kx}}$, where $J^f_N = \{-N+1 \leq k_j \leq N\}$ for all $\mathbf{k} = (k_1, \ldots, k_d)$ and $N \in \boldsymbol{N}$ ($\boldsymbol{N}$ denotes the set of natural numbers). The $e^{i\mathbf{kx}}$ span a subspace $S^f_N \subset H^r_p(\boldsymbol{T}^d)$. To recover $v \in H^r_p(\boldsymbol{T}^d)$ on a finite set of points $\mathbf{x} \in \Omega$, the *full* grid $G^f_N := \{\mathbf{x}\} = \{(x_{j_1}, \ldots, x_{j_d})\}$ with $x_{j_m} = \frac{\pi}{N} j_m$, $N \in \boldsymbol{N}$, and $j_m \in H^f_N := \{0 \leq j_m \leq 2N-1\}$ for $m = 1, \ldots, d$, is frequently used together with the interpolation operator $I^f_N v = I_{N_1}(v) \otimes \cdots \otimes I_{N_d}(v)$. Here, $N_k = N_1$ for all $2 \leq k \leq d$. I_{N_j} is the interpolation operator for one dimension. Thus, for a continuous v on $\boldsymbol{T}^d$, $I^f_N v(\mathbf{x}) = \sum_{\mathbf{k}\in J^f_N} \tilde{v}_{\mathbf{k}} e^{i\mathbf{kx}}$ with $\tilde{v}_{\mathbf{k}} = \sum_{\mathbf{j}\in H^f_N} (2N)^{-d} v(\mathbf{x_j}) e^{-i\mathbf{kx_j}}$, and $\sum_{\mathbf{j}\in H^f_N}$ ranges over all $\mathbf{x} \equiv \mathbf{x_j} \in G^f_N$. Taking $C > 0$ to be some real valued constant,[3] we have the well-known error estimate for P^f_N:

Theorem 1 *Let $0 \leq l \leq m$ with $l, m \in \boldsymbol{N}$. If $N > 0$, $\|u - P^f_N v\|_{l,2} \leq CN^{l-m}\|v\|_{m,2}$ and thus $\|v - P^f_N v\|_{L^2(\Omega)} \leq CN^{-m}\|v\|_{H^m_p(\Omega)}$ for all $v \in H^m_p(\Omega)$.*

For the interpolation operator I^f_N there is an equivalent estimate:

Theorem 2 *Let $0 \leq l \leq m$ with $l, m \in \boldsymbol{N}$ and $m > d/2$. If $N > 0$, $\|v - I^f_N v\|_{l,2} \leq CN^{l-m}\|v\|_{m,2}$ and thus $\|v - I^f_N v\|_{L^2(\Omega)} \leq CN^{-m}\|v\|_{H^m_p(\Omega)}$ for all $v \in H^m_p(\Omega)$.*

For the spaces $\dot{S}^{m,d}$, hyperbolic cross and sparse grids allow much more efficient approximation and interpolation operators. Let us define the (even) hyperbolic cross of order $N \in \boldsymbol{N}$ as $J^h_N = \left\{(\prod_{j=1}^d \max(1, |k_j|) \leq N) \wedge (k_j > -N)\right\}$ where $\mathbf{k} \in \boldsymbol{Z}^d$. For convenience, throughout this paragraph we take j and p to be integers ranging from 1 to d. We define the (even) step hyperbolic cross of size $n \in \boldsymbol{N}$ as $J^s_n = \{\mathbf{k} \in \boldsymbol{Z}^d \mid ((-2^{n_j-1}+1 \leq k_j \leq 2^{n_j-1})\ \forall\ n_j > 0)\ \wedge\ (k_j = 0 \text{ if } n_j = 0)\}$, where $\sum_{j=1}^d n_j \leq n$ for $0 \leq n_j \leq n$. Assuming $N = 2^{m-1}$ we construct a set $J^f_{\mathbf{m}} = \{\mathbf{k} \in \boldsymbol{Z}^d \mid ((-2^{m_j-1}+1 \leq k_j \leq 2^{m_j-1})\ \forall\ m_j > 0)\ \wedge\ (k_j = 0 \text{ if } m_j = 0)\}$. The $e^{i\mathbf{kx}}$ with $\mathbf{k} \in J^f_{\mathbf{m}}$ and $\mathbf{m} \in \boldsymbol{N}^d_0$ span a subspace $S^f_{\mathbf{m}} \subset \dot{S}^{m,d}$. We can use the sets $J^f_{\mathbf{m}}$ to define projections on the even step hyperbolic cross, $J^s_n = \bigcup_{\|\mathbf{m}\|_1 \leq n} J^f_{\mathbf{m}}$ and $\|\mathbf{m}\|_1 = \sum_{j=1}^d |m_j|$. Hence, the $e^{i\mathbf{kx}}$ with $\mathbf{k} \in J^s_n$ span the subspace $S^s_n \subset \dot{S}^{m,d}$ and for each $u(\mathbf{x}) \in S^s_n$ we have $u(\mathbf{x}) = \sum_{\mathbf{k}\in J^s_n} \hat{u}_{\mathbf{k}} e^{i\mathbf{kx}}$ and $\hat{u}_{\mathbf{k}} = (2\pi)^{-d}(u, e^{i\mathbf{kx}})$. In the same manner we can define the projection operator P^h_N with coefficients $\mathbf{k}$ from J^h_N. For the construction of the sparse grid interpolation operator one proceeds as follows. First, the sets $H^f_{\mathbf{m}} = \{\mathbf{j} \in \boldsymbol{N}^d_0 \mid 0 \leq j_p \leq 2^{m_p} - 1\}$ are defined which index the grid points of the *anisotropic* full grids $G^f_{\mathbf{m}} = G_{m_1} \times \cdots \times G_{m_d} = \{\mathbf{x}\}$, where $\mathbf{x} = (x_{j_1}, \ldots, x_{j_d})$

---

3 We are not interested here in the numerical value of C, nor is it the same for the various inequalities discussed here. For the case of $d = 2$ and for similar interpolation operators, numerical values can be found in the work of Pöplau and Sprengel cited above.

with $x_{j_p} = \pi j_p / N_p$ and $N_p = 2^{m_p - 1}$ for $\mathbf{j} = (j_1, \ldots, j_d) \in H^f_{\mathbf{m}}$. The construction of an interpolation operator $I^f_{\mathbf{m}} u := I_{m_1}(u) \otimes \cdots \otimes I_{m_d}(u)$ for all sufficiently well-behaved $u \in \dot{S}^{m,d}$ is straightforward. We call $G^s_n := \bigcup_{\|\mathbf{m}\|_1 \le n} G^f_{\mathbf{m}} = \{\mathbf{x}\} = \{(x_{j_1}, \ldots, x_{j_d})\}$ with $x_{j_p} = \frac{\pi}{2^{m_p - 1}} j_p$ where $\mathbf{j} \in H^s_n := \bigcup_{\|\mathbf{m}\|_1 \le n} H^f_{\mathbf{m}}$ and $0 \le j_p \le 2^{m_p} - 1$ a *Fourier sparse grid* of order n. Naturally, $n \in \boldsymbol{N}_0$. Finally, the interpolation operator I^s_n on G^s_n is defined as $I^s_n = \sum_{\|\mathbf{m}\|_1 \le n} \Delta_{\mathbf{m}}$ where $\Delta_{\mathbf{m}} = \Delta_{m_1} \otimes \cdots \otimes \Delta_{m_d}$ and $\Delta_{m_j} = \tilde{I}_{m_j} - \tilde{I}_{m_j - 1}$. Here, $\tilde{I}_m = I_{\lfloor 2^{m-1} \rfloor}$ for $m \in \boldsymbol{N}_0$ and $\tilde{I}_{-1} : f \mapsto 0$ for all f defined on G^s_n. The symbol $\lfloor r \rfloor$ means rounding to the nearest integer less than r (relevant for $m = 0$ here). The operator $I_{\lfloor 2^{m-1} \rfloor}$ is again the interpolation operator in one dimension. Sparse grid type interpolation operators have been discussed, e.g., in Smolyak [Smo63], Hallatschek [Hal92], and Temlyakov [Tem93]. For P^h_N, we can derive the following error estimate:

Theorem 3 *Let $u \in \dot{S}^{m,d}$ and let $0 \le l \le m$ with $l, m \in \boldsymbol{N}$ and $d \ge 1$. If $N > 0$, $\|P^h_N u - u\|_{l,d,2} \le C N^{l-m} \|u\|_{m,d,2}$ and in particular $\|P^h_N u - u\|_2 \le C N^{-m} \|u\|_{m,d,2}$.*

For the interpolation on G^s_n by means of I^s_n the result is only slightly worse:

Theorem 4 *Let $u \in \dot{S}^{m,d}$ and $N > 0$. Moreover, let $m \ge 1$ and $0 \le l \le m$ with $l, m \in \boldsymbol{N}$. If $d \ge 2$, then $\|I^s_n u - u\|_{l,d,2} \le C N^{l-m} (\operatorname{ld} N)^{d-1} \|u\|_{m,d,2}$ and in particular $\|I^s_n u - u\|_2 \le C N^{-m} (\operatorname{ld} N)^{d-1} \|u\|_{m,d,2}$.*

Note that the dual logarithm ld is not present in Theorem 3. However, as the total number of coefficients in J^h_N grows like $N(\log N)^{d-1}$, both types of approximation finally contain a logarithmic factor which depends on the spatial dimension d. Proofs and further details are given in Kupka [Kup97].

SPARSE GRID SPECTRAL METHODS

Fourier Galerkin spectral methods for the PDE $Lu = f$, where $f \in Y \subset L^2(\boldsymbol{T}^d)$, L is a (linear or non-linear) differential operator, and u is the solution of the problem, can be defined in the variational form: $(Lu^N - f, v) = 0$ for all $v \in S_N$. Here, $u^N \in S_N \subset X$ is the numerical approximation to u and X is some Sobolev space such that $u \in X$ (cf. Canuto et al. [CHQZ88]). The resulting equations yield a finite number of Fourier coefficients $\hat{u}_N$ which define the (numerical) approximation to u in the subspace S_N. In Fourier collocation (pseudospectral) methods both L and the inner product (e, v) are approximated via the interpolation of u and f on a grid G_N. Derivatives of u (contained in L) are approximated by derivatives of interpolation polynomials. A *sparse grid spectral method* is obtained by replacing the operators P^f_N and I^f_N used for the traditional (full grid) spectral methods by their counterparts P^h_N (resp. P^s_n) and I^s_n. As a consequence, in sparse grid Fourier Galerkin methods, the coefficients for Fourier approximations are taken from the (even) hyperbolic (resp. step) hyperbolic cross and, likewise, functions in sparse grid Fourier collocation methods are interpolated by Fourier polynomials defined through a sparse grid G^s_n. Differential operators and inner products can then be approximated accordingly.

Using the error estimates of Theorems 3 and 4 together with the standard approach for obtaining convergence estimates for spectral methods (cf. Canuto et al.[CHQZ88]), one can apply such technique to sparse grid spectral methods. Examples include the stationary Helmholtz equation, a problem $Lu = f$ with a coercive operator L, the transient Helmholtz and the heat equation, as well as the linear advection equation (see Kupka [Kup97]). Basically, each term which does not involve the operator I_n^s requires the replacement of $\|w\|_{m,2}$ with $\|w\|_{m,2,2}$, whereas otherwise a logarithmic factor $\mathrm{ld}N^{d-1}$ has to be inserted (here, N is the maximum number of grid points per coordinate direction and w must belong to some appropriate smoothness class). As an example, for the Helmholtz equation $-\triangle u + \lambda u = f$ with $\lambda > 0$ on $\boldsymbol{T}^2$, the error estimate for Fourier collocation on a full grid is $\|u - u_c^N\|_{1,2} \leq CN^{1-m}(\|u\|_{m,2} + \|f\|_{m-1,2})$ in comparison with $\|u - u_c^N\|_{1,2} \leq C'N^{1-m}(\|u\|_{m,2,2} + (\mathrm{ld}N)\|f\|_{m-1,2,2})$ for Fourier collocation on a sparse grid (f and thus u are assumed to be sufficiently smooth). Note that — as usual — $\dim S_N = O(N \log N)$ for the sparse grid case, but $O(N^2)$ for the full grid. In accordance with intuition, a von Neumann analysis of fully discrete approximations to the heat and advection equation reveals that restrictions on the time step are dominated by the most anisotropic grids "contained" in a sparse grid. This can reduce the overall efficiency of sparse grid spectral methods for some explicit solvers for time dependent problems (Kupka [Kup97]).

An efficient implementation of sparse grid spectral methods requires adequate data structures. In Fortran90 such a structure can be realized as a user defined data type containing a pointer to a one-dimensional array of another user defined data type which in turn has a pointer to an array of real values (floating point numbers) as its only component. This structure may be iterated for the case $d > 2$. Tests have shown that usual manipulations such as copying of variables, addition, and of course Fourier transformation, are slower by a factor of 3 to 5 for such a sparse grid structure with several 100 to a few 1000 points in comparison to standard arrays of rank d of the same total number of points. Although more elegant, a recursive sparse grid data structure is likely to suffer from a much lower computational efficiency due to an enhanced internal overhead. The (discrete) Fourier transformation can be done by the algorithm of Hallatschek [Hal92] which consists of a one dimensional (fast) Fourier transformation followed by a transformation into a hierarchical basis. The algorithm is performed sequentially for each spatial dimension. A final sequence of re-transformations is necessary to obtain Fourier coefficients in the standard basis $\{e^{i\mathbf{kx}}\}$. The algorithm can be run backwards to obtain function values at grid points (see Kupka [Kup97] for a discussion, also for an implementation of a fast Fourier transformation on sparse grids for the case $d = 2$).

SOME NUMERICAL EXPERIMENTS AND CONCLUSIONS

Numerical experiments presented by Kupka [Kup97] have exhibited the following properties of Fourier sparse grid spectral methods. First, for insufficient or low resolution, sparse grid Fourier collocation is subject to aliasing errors and spurious solutions in the same way as a full grid method. On the other hand, spectral convergence can easily be verified for infinitely smooth solutions, just as the finite

Table 1 The relative error in various L^p norms — approximated on a dense full grid (l_p norms) — for full grid approximations u_c^N (on G_N^f, top) and sparse grid ones (on G_n^s, bottom) to the solution u of a Helmholtz problem described in the text. The quadratic convergence as a function of N is evident. However, considering the total number of grid points the convergence is only linear for the full grid method, but still nearly quadratic in the sparse grid case.

N	grid points	$l_1(u_c^N - u)$	$l_2(u_c^N - u)$	$l_\infty(u_c^N - u)$
8	256	3.69902e-3	5.79361e-3	2.67764e-2
16	1024	9.21957e-4	1.42771e-3	7.09712e-3
32	4096	2.30752e-4	3.57583e-4	1.82264e-3
64	16384	5.78924e-5	8.98336e-5	4.61524e-4
n	grid points	$l_1(u_c^N - u)$	$l_2(u_c^N - u)$	$l_\infty(u_c^N - u)$
6	256	3.47985e-3	3.86827e-3	7.96669e-3
7	576	8.69380e-4	9.62935e-4	2.03200e-3
8	1280	2.17657e-4	2.41199e-4	5.13192e-4
9	2816	5.44621e-5	6.03750e-5	1.28952e-4

convergence order for solutions with a finite number of partial derivatives in all spatial directions. However, whereas a sparse grid method for a numerical solver with a fixed approximation order (finite differences, etc.) is more efficient than its full grid counterpart as long as the solution is sufficiently smooth, sparse grid spectral methods are more useful, if the solution has particular smoothness properties: consider the polynomial $g(z) = \pi^4/90 - \pi^2 z^2/12 + \pi z^3/12 - z^4/48$ on $[0, 2\pi]$. It is easily verified that $g(z) \in H_p^3(\boldsymbol{T})$, but $g(z) \notin H_p^4(\boldsymbol{T})$. Taking $f(x,y) = -g''(x)g(y) - g(x)g''(y) + \lambda g(x)g(y)$ it can immediately be seen that $u(x,y) = g(x)g(y)$ is the solution of $-\triangle u + \lambda u = f$ on $\boldsymbol{T}^2$. Also, $u \in H_p^3(\boldsymbol{T}^2)$ and $u \in \dot{S}^{3,2}$, but $u \notin H_p^4(\boldsymbol{T}^2)$. The resulting higher efficiency of sparse grid Fourier collocation in comparison with Fourier collocation on full grids is illustrated in Table 1.

With sparse grid Fourier collocation for the heat equation even non-smooth initial conditions can be handled reasonably well. The same does not hold true for the linear advection equation: the initial condition $u(t = 0, x, y) = u_0(x, y)$ with $u_0 = 1$ in a rectangular subdomain of $\boldsymbol{T}^2$ and $u_0 = 0$ elsewhere, causes the sparse grid method to fail while the full grid one only suffers from oscillations. Filtering, smoothing, or adaptivity thus may help to make sparse grid spectral methods more efficient building blocks for a domain decomposition solver than their full grid counterparts (which is expected from their "universality", i.e. when functions from different smoothness classes are approximated, cf. Temlyakov [Tem93]). Although due the more expensive programming model the sparse grid spectral methods have a break-even point which is lower for memory than for speed, neither is prohibitive even for moderate accuracy requirements.

The work of Pöplau and Sprengel mentioned in the introduction makes a similar study of Chebychev (and other) spectral methods both possible and worthwhile.

Likewise, the analysis of other sparse grid methods can benefit from the approximation theoretical results developed for sparse grid spectral methods. Whether turbulence simulations in fluid mechanics also benefit from the *hierarchical bases* inherent to sparse grid methods (priv. communication with M. Griebel during DD11, cf. the sparse grid FFT discussed above) is a question that requires a more careful study.

REFERENCES

[Bab60] Babenko K. (1960) Approximation by trigonometric polynomials in a certain class of periodic functions. *Soviet Math. Dokl.* 1: 672–675. Translated from *Dokl. Akad. Nauk SSSR*, 132:982–985, 1960.

[CHQZ88] Canuto C., Hussaini M., Quarteroni A., and Zang T. (1988) *Spectral Methods in Fluid Dynamics.* Springer Series in Computational Physics. Springer-Verlag, Berlin. (second printing 1990).

[DS89] Delvos F.-J. and Schempp W. (1989) *Boolean Methods in Interpolation and Approximation.* Number 230 in Pitman Research Notes in Mathematics Series. Longman Scientific & Technical, Harlow.

[GS98] Griebel M. and Schiekofer T. (1998) An adaptive sparse grid Navier-Stokes solver in 3D based on the finite difference method. In *Proceedings ENUMATH97.* World Scientific, Singapore.

[Hal92] Hallatschek K. (1992) Fouriertransformation auf dünnen Gittern mit hierarchischen Basen. *Numer. Math.* 63: 83–97.

[Kup97] Kupka F. (1997) *Sparse Grid Spectral Methods for the Numerical Solution of Partial Differential Equations with Periodic Boundary Conditions.* PhD thesis, Universität Wien.

[Pöp95] Pöplau G. (1995) *Multivariate periodische Interpolation durch Translate und deren Anwendung.* PhD thesis, Universität Rostock.

[PS97] Pöplau G. and Sprengel F. (1997) Some error estimates for periodic interpolation on full and sparse grids. In Méhauté A. L., Rabut C., and Schumaker L. (eds) *Curves and Surfaces with Applications in CAGD*, pages 355–362. Vanderbilt University Press, Nashville, TN.

[Smo63] Smolyak S. (1963) Quadrature and interpolation formulas for tensor products of certain classes of functions. *Soviet Math. Dokl.* 4: 240–243. Translated from *Dokl. Akad. Nauk SSSR*, 148:1042–1045, 1963.

[Spr97a] Sprengel F. (1997) *Interpolation und Waveletzerlegung multivariater periodischer Funktionen.* PhD thesis, Universität Rostock.

[Spr97b] Sprengel F. (1997) A unified approach to error estimates for interpolation on full and sparse Gauß-Chebyshev grids. *Rostock. Math. Kolloq.* 51: 51–64.

[Tem89] Temlyakov V. (1989) Approximations of functions with bounded mixed derivative. *Proc. Steklov Inst. Math.* 1. Translated from *Trudy Mat. Inst. Steklova*, 178, 1986.

[Tem93] Temlyakov V. (1993) *Approximation of Periodic Functions.* Nova Science, New York.

[Zen91] Zenger C. (1991) Sparse grids. In *Parallel Algorithms for PDEs*, volume 31 of *Notes on Numerical Fluid Mechanics.* Vieweg, Wiesbaden.

7

On Schwarz Alternating Methods for the Incompressible Navier-Stokes Equations in N Dimensions

S. H. LUI[1]

INTRODUCTION

This paper considers four Schwarz Alternating Methods for the N-dimensional, steady, viscous, incompressible Navier Stokes equations, $N \leq 4$. It is shown that the Schwarz sequences converge to the true solution provided the Reynolds number is sufficiently small. This appears to be the first attempt to prove convergence of Schwarz Alternating Methods for the Navier Stokes equations.

The Schwarz Alternating Method was devised by H. A. Schwarz more than one hundred years ago to solve linear boundary value problems. It has garnered interest recently because of its potential as a very efficient algorithm for parallel computers. In Tai and Espedal [TE98], Tai and Xu [TX98], and Dryja and Hackbusch [DH97], they show convergence of Schwarz methods for some nonlinear problems. In Lui [Luiar], proofs of convergence of Schwarz Alternating Methods for some 2nd-order nonlinear elliptic PDEs were given. In this sequel, we prove convergence of four Schwarz methods for the N-dimensional, steady, incompressible, viscous Navier Stokes equations, $N \leq 4$. Many authors have demonstrated numerically the effectiveness of Schwarz methods in solving fluid problems. See, for example, the proceedings of the annual domain

1 Department of Mathematics, HKUST, Clear Water Bay, Kowloon, Hong Kong (email: shlui@uxmail.ust.hk). This work was supported in part by a grant from RGC CERG HKUST726/96E.

Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

decomposition conferences, beginning with [GGMP88]. We mention in particular [DDG+92], [CGK+98] and [KKS98]. This paper follows closely the framework developed in the fundamental paper of Lions [Lio88] where the convergence of the Schwarz method for the Stokes equations is proved. While we concentrate mostly on the two-subdomain case, we show an additive Schwarz version which converges for the multiple-subdomain case.

Let Ω be a bounded domain in $\mathbb{R}^N$ with a smooth boundary. Suppose $\Omega = \Omega_1 \cup \Omega_2$, where the subdomains Ω_i have smooth boundaries and are overlapping. Let $H_0^1(\Omega)^N$ denote the Cartesian product of N Sobolev spaces $H_0^1(\Omega)$ (consisting of all functions whose first derivatives are in $L^2(\Omega)$ and whose value vanishes on $\partial\Omega$ in the trace sense) and define $L^2(\Omega)^N$ as the Cartesian product of N copies of $L^2(\Omega)$. The space $H^{1/2}(\partial\Omega)^N$ is defined similarly. Let $V = \{u \in H_0^1(\Omega)^N,\ \text{div } u = 0\}$ and $V_i = \{u \in H_0^1(\Omega_i)^N,\ \text{div } u = 0\}$, $i = 1, 2$. Let u_i denote the i-th component of the vector u. Denote the inner product in $H_0^1(\Omega)^N$ by $[u, v] = \sum_{i=1}^{N} \int_\Omega \nabla u_i \cdot \nabla v_i$ and let $\|u\|_1 = [u, u]^{1/2}$. Let $\triangle_i^{-1}$ be the inverse of the Laplacian operator considered as an operator from the dual space $H^{-1}(\Omega_i)^N$ onto $H_0^1(\Omega_i)^N$, $i = 1, 2$. We take overlapping to mean that $H_0^1(\Omega)^N = H_0^1(\Omega_1)^N + H_0^1(\Omega_2)^N$. In this paper, a function in $H_0^1(\Omega_i)^N$ is considered as a function defined on the whole domain by extension by zero.

When the two subdomains are overlapping, the work of Lions [Lio88] shows that $V = V_1 + V_2$. Let P_i denote the orthogonal (with respect to the inner product $[\cdot, \cdot]$) projection from V onto V_i, $i = 1, 2$. It is well known that

$$d \equiv \max\left(\|(I - P_2)(I - P_1)\|_2,\ \|(I - P_1)(I - P_2)\|_2\right) < 1.$$

See Lions [Lio88] and Bramble et. al. [BPWX91]. Throughout this paper, C will denote a positive constant which may not be the same at different occurrences.

In the next section, we give a statement of the problem including an estimate for the nonlinear term. Following that, we sketch a proof of convergence for a nonlinear Schwarz sequence where each subdomain problem is a nonlinear one. In the next three sections, we develop three variations of the nonlinear Schwarz sequence. These sequences are more practical in that only linear subdomain problems are encountered and that in two of these, the subdomain problems are independent so that they can be solved concurrently. In the final section, we discuss the case of many subdomains and non-homogeneous boundary conditions.

NAVIER-STOKES EQUATIONS

The N-dimensional, steady, viscous, incompressible Navier Stokes equations in non-dimensional form is

$$\begin{aligned} (u \cdot \nabla)u &= -\nabla p + \frac{1}{R}\triangle u + f \text{ on } \Omega \\ \text{div } u &= 0 \text{ on } \Omega \end{aligned}$$

with the boundary conditions

$$u = g \text{ on } \partial\Omega,$$

where $u \in H^1(\Omega)^N$, g is the given velocity on the boundary with $\|g\|_{H^{1/2}(\partial\Omega)^N} \leq 1$, f is the forcing term in $L^2(\Omega)^N$, p is the pressure and R is the Reynolds number. We assume that g satisfies the compatibility condition $\int_{\partial\Omega} g \cdot n = 0$ where n denotes the (outward) unit normal. We use u to denote the unique solution to this equation.

We shall need the following lemma. A similar version states the continuity of a certain trilinear form containing the convective term of the Navier Stokes equations. See, for instance, Temam [Tem95] on page 12.

Lemma 1 *Let $U = \{u \in H^1(\Omega)^N,\ div\ u = 0\}$. For $N \leq 4$, define the function $F : U \times U \to H_0^1(\Omega)^N$ by $F(u,v) = \triangle^{-1}(u \cdot \nabla)v$. Then $\|F(u,v)\|_1 \leq C\|u\|_1\ \|v\|_1$.*

In light of this lemma, we restrict to the case $N \leq 4$ in the remainder of this paper. Initially, *we only discuss the case $g \equiv 0$* which has a clearer exposition. The general case will be addressed in the last section. In the next section, we define a nonlinear Schwarz sequence.

NONLINEAR SCHWARZ SEQUENCE

Let $u^{(0)} \in V$. For $n = 0, 1, 2, \cdots$, define the nonlinear Schwarz sequence as

$$\begin{aligned} (u^{(n+\frac{1}{2})} \cdot \nabla)u^{(n+\frac{1}{2})} &= -\nabla p^{(n+\frac{1}{2})} + \frac{1}{R}\triangle u^{(n+\frac{1}{2})} + f \text{ on } \Omega_1 \\ \text{div } u^{(n+\frac{1}{2})} &= 0 \text{ on } \Omega_1 \\ u^{(n+\frac{1}{2})} &= u^{(n)} \text{ on } \partial\Omega_1 \end{aligned}$$

and

$$\begin{aligned} (u^{(n+1)} \cdot \nabla)u^{(n+1)} &= -\nabla p^{(n+1)} + \frac{1}{R}\triangle u^{(n+1)} + f \text{ on } \Omega_2 \\ \text{div } u^{(n+1)} &= 0 \text{ on } \Omega_2 \\ u^{(n+1)} &= u^{(n+\frac{1}{2})} \text{ on } \partial\Omega_2. \end{aligned}$$

Here, $u^{(n+\frac{1}{2})}$ is considered as a function in V by defining it to be $u^{(n)}$ on $\Omega \setminus \Omega_1$ and $u^{(n+1)}$ is defined as $u^{(n+\frac{1}{2})}$ on $\Omega \setminus \Omega_2$. The first thing to check is that the compatibility conditions

$$\int_{\partial\Omega_1} u^{(n)} \cdot n = 0 = \int_{\partial\Omega_2} u^{(n+\frac{1}{2})} \cdot n$$

are satisfied. We claim that they are satisfied for all n provided

$$\int_{\Gamma_i} u^{(0)} \cdot n = 0, \ \ i = 1, 2 \tag{1}$$

hold, where $\Gamma_i = \partial\Omega_i \cap \Omega_{3-i}$.

The claim is proved by induction. For $n = 0$,

$$\int_{\partial\Omega_1} u^{(0)} \cdot n = \int_{\Gamma_1} u^{(0)} \cdot n = 0$$

by (1). Since div $u^{(\frac{1}{2})} = 0$ on $\Omega_1 \cap \Omega_2$,

$$0 = \int_{\Gamma_1} u^{(\frac{1}{2})} \cdot n + \int_{\Gamma_2} u^{(\frac{1}{2})} \cdot n = \int_{\Gamma_1} u^{(0)} \cdot n + \int_{\Gamma_2} u^{(\frac{1}{2})} \cdot n = \int_{\Gamma_2} u^{(\frac{1}{2})} \cdot n.$$

The induction step is proved by a similar calculation.

The main result is that this Schwarz sequence converges to the true solution provided the Reynolds number is sufficiently small.

Theorem 1 *Assuming $u^{(0)}$ satisfies (1) and R is sufficiently small (depending on $\|u^{(0)} - u\|_1$), the nonlinear Schwarz sequence converges geometrically to the true solution u in the norm $\|\cdot\|_1$.*

The proof can be divided into three steps. In the first step, we derive the error equations

$$\begin{aligned} e^{(n+\frac{1}{2})} &= (I - P_1)e^{(n)} + RF_1(e^{(n+\frac{1}{2})}), \\ e^{(n+1)} &= (I - P_2)e^{(n+\frac{1}{2})} + RF_2(e^{(n+1)}), \end{aligned}$$

where $e^{(n+\frac{1}{2})} = u^{(n+\frac{1}{2})} - u$ and $e^{(n)} = u^{(n)} - u$ are the error terms and F_i are certain quadratic nonlinear terms. These nonlinear terms can be bounded using Lemma 1. In the second step, we use induction to show that provided R is smaller than some known constant, $\|e^{(n+\frac{1}{2})}\|_1$, $\|e^{(n)}\|_1 \le M$ for all n, where $M = \|e^{(0)}\|_1$. In the final step, we employ these estimates in the error equations to prove geometric convergence of the Schwarz sequence. A sufficient condition for convergence is that

$$R < \frac{\sqrt{2 - d^2} - d}{C(2\|u\|_1 + M)},$$

where C is a known constant. Full details of the proof will be reported elsewhere.

LINEAR SCHWARZ SEQUENCE

In the previous section, each Schwarz iteration requires the solution of a nonlinear PDE in each subdomain. From a practical point of view, we of course prefer to solve linear problems whenever possible. Now, we demonstrate a version of the Schwarz method where only linear PDEs need to be solved. Convergence still holds but at a possibly smaller R and that the rate of convergence may be a little slower.

Let $u^{(0)} \in V$ and assume it satisfies (1). For $n = 0, 1, 2, \cdots$, define the linear Schwarz sequence as

$$\begin{aligned} (u^{(n)} \cdot \nabla)u^{(n+\frac{1}{2})} &= -\nabla p^{(n+\frac{1}{2})} + \frac{1}{R}\triangle u^{(n+\frac{1}{2})} + f \text{ on } \Omega_1 \\ \text{div } u^{(n+\frac{1}{2})} &= 0 \text{ on } \Omega_1 \\ u^{(n+\frac{1}{2})} &= u^{(n)} \text{ on } \partial\Omega_1 \end{aligned}$$

and

$$(u^{(n+\frac{1}{2})} \cdot \nabla)u^{(n+1)} = -\nabla p^{(n+1)} + \frac{1}{R}\triangle u^{(n+1)} + f \text{ on } \Omega_2$$

$$\begin{aligned} \text{div } u^{(n+1)} &= 0 \text{ on } \Omega_2 \\ u^{(n+1)} &= u^{(n+\frac{1}{2})} \text{ on } \partial\Omega_2. \end{aligned}$$

Notice that the above equations are linear and that the compatibility conditions are satisfied.

The main result is that this Schwarz sequence converges to the true solution provided the Reynolds number is sufficiently small.

Theorem 2 *Assuming $u^{(0)}$ satisfies (1) and R is sufficiently small (depending on $\|u^{(0)} - u\|_1$), the linear Schwarz sequence converges to the true solution u in the norm $\|\cdot\|_1$.*

The proof is similar to before with geometric convergence if

$$R < \frac{1-d}{C(M+2)}.$$

PARALLEL SCHWARZ SEQUENCE

For the two previous Schwarz methods, the iterates must be computed sequentially. In this section, we suggest a Schwarz sequence where the two subdomain problems are independent and thus they can be solved simultaneously. This sequence also converges provided the Reynolds number is sufficiently small.

Let $u^{(0)} \in V$ and assume that it satisfies (1). Define $u^{(-\frac{1}{2})} = u^{(-\frac{3}{2})} = u^{(0)}$. For $n = 0, 1, 2, \cdots$, define the parallel Schwarz sequence as

$$\begin{aligned} (u^{(n)} \cdot \nabla) u^{(n+\frac{1}{2})} &= -\nabla p^{(n+\frac{1}{2})} + \frac{1}{R} \triangle u^{(n+\frac{1}{2})} + f \text{ on } \Omega_1 \\ \text{div } u^{(n+\frac{1}{2})} &= 0 \text{ on } \Omega_1 \\ u^{(n+\frac{1}{2})} &= u^{(n)} \text{ on } \partial\Omega_1 \end{aligned}$$

and

$$\begin{aligned} (u^{(n-\frac{1}{2})} \cdot \nabla) u^{(n+1)} &= -\nabla p^{(n+1)} + \frac{1}{R} \triangle u^{(n+1)} + f \text{ on } \Omega_2 \\ \text{div } u^{(n+1)} &= 0 \text{ on } \Omega_2 \\ u^{(n+1)} &= u^{(n-\frac{1}{2})} \text{ on } \partial\Omega_2. \end{aligned}$$

We define $u^{(n+1)}$ as $u^{(n-\frac{1}{2})}$ on $\Omega \setminus \Omega_2$. Notice that the above equations are linear and can be solved independently. We again can check that the compatibility conditions are satisfied.

Theorem 3 *Assuming $u^{(0)}$ satisfies (1) and R is sufficiently small (depending on $\|u^{(0)} - u\|_1$), the parallel Schwarz sequence converges to the true solution u in the norm $\|\cdot\|_1$.*

For this sequence, we obtain geometric convergence if

$$R < \frac{1-d}{C(M+2)}.$$

ADDITIVE SCHWARZ SEQUENCE

In this section, we demonstrate the convergence of the Additive Schwarz sequence, originally introduced in [DW87] for linear elliptic equations and has been used successfully in practice.

Let $u^{(0)} \in V$. For $n = 0, 1, 2, \cdots$, define

$$-\frac{1}{R}\triangle d^{(n+\frac{1}{2})} + \nabla p^{(n+\frac{1}{2})} = f - (u^{(n)} \cdot \nabla)u^{(n)} - \nabla p^{(n)} + \frac{1}{R}\triangle u^{(n)} \text{ on } \Omega_1$$

and

$$-\frac{1}{R}\triangle d^{(n+1)} + \nabla p^{(n+1)} = f - (u^{(n)} \cdot \nabla)u^{(n)} - \nabla p^{(n)} + \frac{1}{R}\triangle u^{(n)} \text{ on } \Omega_2,$$

where $d^{(n+\frac{1}{2})} \in V_1$ and $d^{(n+1)} \in V_2$. The additive Schwarz sequence is defined as $u^{(n+1)} = u^{(n)} + \omega(d^{(n+\frac{1}{2})} + d^{(n+1)})$ where ω is a relaxation parameter. Roughly speaking, $d^{(n+\frac{1}{2})}$ and $d^{(n+1)}$ are corrections to the iterate $u^{(n)}$ in the subdomains Ω_1 and Ω_2, respectively and the right-hand sides of the above equations defining the corrections are the residuals of $u^{(n)}$ in the subdomains. Ignoring the nonlinear terms, the above sequence is precisely the additive Schwarz sequence for the Stokes problem. Notice that the above Stokes equations can be solved independently and no additional assumption on $u^{(0)}$ is required.

Theorem 4 *Assuming $0 < \omega < 1/2$ and R sufficiently small, the additive Schwarz sequence converges geometrically to the true solution u in the norm $\|\cdot\|_1$.*

Proof: We only sketch the proof, recording the key equations. From the defining equations, we have

$$\begin{aligned} d^{(n+\frac{1}{2})} &= -P_1 e^{(n)} + RF_1(e^{(n)}) \\ d^{(n+1)} &= -P_2 e^{(n)} + RF_2(e^{(n)}) \end{aligned}$$

from which we obtain

$$e^{(n+1)} = (I - \omega\,(P_1 + P_2))e^{(n)} + \omega R(F_1(e^{(n)}) + F_2(e^{(n)})),$$

where F_1 and F_2 are the same nonlinearities as above. When $0 < \omega < 1/2$, $\|I - \omega\,(P_1 + P_2)\|_1 < 1$. Applying the estimates for F_i and the condition

$$R < \frac{1 - \|I - \omega\,(P_1 + P_2)\|_1}{\omega C(\|u^{(0)} - u\|_1 + 2\|u\|_1)},$$

we can show that $\|e^{(n)}\|_1 \leq \|e^{(0)}\|_1$ for all n. This condition is also sufficient to guarantee geometric convergence of the additive Schwarz sequence.

DISCUSSION AND CONCLUSION

In this paper, we show the convergence of four Schwarz Alternating Methods for the N-dimensional, steady, incompressible Navier Stokes equations, provided the Reynolds number is sufficiently small and $N \leq 4$. Our results are only valid for the two-subdomain case. The result for finitely many subdomains is again that the Schwarz sequence converges provided R is sufficiently small. Except for the additive Schwarz scheme, we cannot give an explicit bound for R. The difficulty is in obtaining an explicit expression for the spectral radius of a certain matrix. We elaborate on this point below. To simplify the analysis, assume that any three distinct subdomains have an empty intersection.

First, we need the following lemma which was shown by Lions [Lio88] for the two subdomain case. The general case is true by induction.

Lemma 2 *Suppose $H_0^1(\Omega)^N = H_0^1(\Omega_1)^N + \cdots + H_0^1(\Omega_m)^N$. Let V and V_i be subspaces of $H_0^1(\Omega)^N$ and $H_0^1(\Omega_i)^N$, respectively, consisting of divergence-free functions. Then $V = V_1 + \cdots + V_m$.*

For m subdomains, we can obtain analogous error equations. For instance,

$$e^{(n+1)} = (I - P_m) \cdots (I - P_1) e^{(n)} + \cdots,$$

where P_i is the orthogonal projection (with respect to $[\cdot, \cdot]$) of V onto V_i and $[+ \cdots]$ is a sum of m quadratic terms. Armed with Lemma 2, it is well known ([Lio88], [BPWX91]) that

$$d = \|(I - P_m) \cdots (I - P_1)\|_1 < 1$$

for overlapping subdomains. In a similar way as before, we can show that the Schwarz sequence is bounded and provided $d^{1/m} + o(\epsilon) < 1$ with $\epsilon = RC(M+1)$ sufficiently small, the Schwarz sequence converges. Because it does not seem possible to write down an explicit expression for the $o(\epsilon)$ term, we cannot give an explicit bound on R.

For the additive Schwarz method, it converges if

$$R < \frac{1 - \|I - \omega\,(P_1 + \cdots + P_m)\|_1}{m\omega C(\|u^{(0)} - u\|_1 + 2\|u\|_1)},$$

where $0 < \omega < 1/K$, where K is the minimum number of colors needed to color the subdomains in such a way that overlapping subdomains are assigned different colors.

Non-homogeneous boundary conditions can also be treated by making a change of variable. Define $w = u - g$, where g is the Dirichlet data on $\partial\Omega$ which is extended to be a divergence-free function in $H^1(\Omega)^N$. Then the Navier Stokes equations become

$$(w \cdot \nabla)w + (g \cdot \nabla)w + (w \cdot \nabla)g = -\nabla p + \frac{1}{R}\triangle w + G$$

for $w \in V$ and

$$G = \frac{1}{R}\triangle g + f - (g \cdot \nabla)g.$$

The analysis goes through as before.

One interesting question is to investigate in what sense does the sequence of pressure converge. Other future work include extending our results to the time dependent case and the consideration of Schwarz methods on non-overlapping subdomains.

ACKNOWLEDGMENT

I thank Professor Alfio Quarteroni for an extremely beneficial discussion.

REFERENCES

[BPWX91] Bramble J. H., Pasciak J. E., Wang J., and Xu J. (1991) Convergence estimates for product iterative methods with applications to domain decomposition. *Math. Comput.* 57: 1–21.

[CGK+98] Cai X. C., Gropp W. D., Keyes D. E., Melvin R. G., and Young D. P. (1998) Parallel Newton-Krylov-Schwarz algorithms for the transonic full potential equation. *SIAM J. Sci. Comput.* 19: 245–265.

[DDG+92] Dean E. J., Dinh Q. V., Glowinski R., He J., Pan T. W., and Periaux J. (1992) Least squares/domain imbedding methods for neumann problems: Applications to fluid dynamics. In Keyes D. E., Chan T. F., Meurant G. A., Scroggs J. S., and Voigt R. G. (eds) *Proc. Fifth Int. Conf. on Domain Decomposition Methods*, pages 451–475. SIAM, Philadelphia.

[DH97] Dryja M. and Hackbusch W. (1997) On the nonlinear domain decomposition method. *BIT* pages 296–311.

[DW87] Dryja M. and Widlund O. B. (1987) An additive variant of the Schwarz alternating method for the case of many subregions. Technical report, Courant Institute Technical Report 339.

[GGMP88] Glowinski R., Golub G. H., Meurant G. A., and Periaux J. (1988) *First Int. Symp. on Domain Decomposition Methods.* SIAM, Philadelphia.

[KKS98] Kaushik D. K., Keyes D. E., and Smith B. F. (1998) On the interaction of architecture and algorithm in the domain-based parallelization of an unstructured-grid incompressible flow code. In Mandel J., Farhat C., and Cai X. C. (eds) *Domain Decomposition Methods 10.* AMS, Providence.

[Lio88] Lions P. L. (1988) On the Schwarz alternating method I. In Glowinski R., Golub G. H., Meurant G. A., and Periaux J. (eds) *First Int. Symp. on Domain Decomposition Methods*, pages 1–42. SIAM, Philadelphia.

[Luiar] Lui S. H. (To Appear.) On Schwarz alternating methods for nonlinear elliptic pdes. *SIAM J. Sci. Comput.* .

[TE98] Tai X. C. and Espedal M. (1998) Rate of convergence of some space decomposition methods for linear and nonlinear problems. *SIAM J. Numer. Anal.* 35: 1558–1570.

[Tem95] Temam R. (1995) *Navier-Stokes Equations and Nonlinear Functional Analysis, Second Edition.* SIAM.

[TX98] Tai X. C. and Xu J. (1998) Global convergence of subspace correction methods for convex optimization problems. *Report 114, Department of Mathematics, University of Bergen* .

8

An Iterative Substructuring Preconditioner for Collocation with Hermite Bicubics

Gabriel Mateescu & Calvin J. Ribbens[1]

Introduction

We propose and evaluate a novel Schur complement type preconditioner for solving linear systems of equations arising from piecewise Hermite bicubic collocation discretization of elliptic partial differential equations (PDEs) with mixed boundary conditions. Although collocation is a very general and effective discretization technique for many PDE problems, relatively little is known about combining domain decomposition techniques with collocation. In this paper we define a new preconditioner in terms of two special grids—a coarse grid and a hybrid coarse/fine grid—which together provide the framework for approximating the interface subproblem. We illustrate the performance of the preconditioner by combining it with a Krylov subspace method (KSM) and reporting experimental results on three test problems.

We consider the family of boundary value problems:

$$\mathbf{Lu} = \mathbf{g} \quad \text{in } \mathbf{\Omega} \qquad \text{and} \qquad \mathbf{Bu} = \mathbf{t} \quad \text{on } \partial\mathbf{\Omega}, \tag{1}$$

where $\mathbf{L}$ is a linear second-order elliptic operator, $\mathbf{B} = \mathbf{p}\frac{\partial}{\partial \mathbf{n}} + \mathbf{q}$, $\Omega = (0, a) \times (0, a)$ and $\partial\Omega$ is the boundary of Ω. The discretization method is collocation with piecewise

1 Authors address: Computer Science Department, Virginia Tech, Blacksburg, VA 24061 USA Email: {mateescu, ribbens}@vt.edu This work was supported in part by DARPA ARO grant DAAH04-94-G-0010 and by Virginia Tech Interdisciplinary Center for Applied Mathematics.

Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

Hermite bicubic basis functions; for convenience, we refer to this method as *Hermite collocation* (HC). Hermite collocation provides $O(h^4)$ accuracy, where h is the fine grid step, for sufficiently smooth problems, and does not require the computation of any integrals for generating the coefficient matrix. Let $A\mathbf{u} = \mathbf{f}$ be the discrete problem induced by applying HC to (1), where $A \in \mathbf{R}^{\mathbf{N}\times\mathbf{N}}$. The solution method has two elements: (i) a right preconditioner M which is employed to obtain the preconditioned system $AM^{-1}\mathbf{v} = \mathbf{f}$, where $\mathbf{v} = \mathbf{Mu}$; (ii) a KSM, such as the Generalized Minimal Residual (GMRES) method [SS86], for solving the preconditioned system. The matrix A induced by HC is not symmetric even when $\mathbf{L}$ is the Laplacian; moreover, A is typically ill-conditioned and unpreconditioned GMRES takes $O(N)$ iterations.

Domain decomposition based on Schur complement methods (SCMs) can provide robust preconditioners for HC problems. SCMs work by partitioning the domain in *subdomains* and an *interface* consisting of *edges* and *vertices*, and deriving a reduced Schur complement problem for the interface subproblem (see [SBG96]). However, constructing SCM preconditioners for HC is not straightforward. Well known strategies such as *tangential preconditioning* [CK90], *Neumann-Dirichlet* [BW86],or *multilevel diagonal scaling* [BPX90] do not generalize easily to the HC problem derived from (1). Previous work on domain decomposition methods for HC problems has been limited to symmetric positive definite (SPD) problems and problems obtained by symmetrizing the initial problem. Bialecki, *et al.* have proposed several preconditioners for solving Poisson's equation with HC. One method [Bia93] is based on a decomposition in parallel strips of the domain and a SCM preconditioner. Another method [BCDF94] employs the additive Schwarz approach. Our preconditioner is remarkable in that it supports general elliptic operators, two-dimensional decomposition (important for scalability), and proposes a new method for approximately solving the Schur complement problem.

The Hermite Collocation Problem

We proposed in [MRW98] a preconditioner based on a strip decomposition of the domain. This paper generalizes that work to a preconditioner for substructuring in two dimensions, which we call the *edge-vertex preconditioner* (EVP).

Let n_0 be an integer greater than one, and $X = \{x_k\}_{k=0}^{n_0-1}$ and $Y = \{y_m\}_{m=0}^{n_0-1}$ be partitions of $[0, a]$, where $x_k = k\,h$, $y_m = m\,h$, and $h = a/(n_0 - 1)$. The set of grid points, or *nodes*, in $\bar{\Omega} = \Omega \cup \partial\Omega$ is $G^h = X \times Y$. In HC, the unknown function u is approximated by a function U in the space $\mathcal{V}$ of continuously differentiable piecewise bicubic polynomials. Let $n_G = n_0^2$ be the number of nodes in G^h. Following [HMR85], with each node n in G^h there are associated four Hermite bicubic basis functions, Φ_n^i, $1 \leq i \leq 4$, centered at n and with support $[x_k - h, x_k + h] \times [y_m - h, y_m + h]$, where (x_k, y_m) are the coordinates of n. The set $\{\Phi_n^i \mid 1 \leq i \leq 4, 1 \leq n \leq n_G\}$ is a basis for $\mathcal{V}$, and U can be written as $U(x, y) = \sum_{i=1}^{4} \sum_{n=1}^{n_G} U_n^{(i)} \Phi_n^i(x, y)$, where $U_n^{(1)}$, $U_n^{(2)}$, $U_n^{(3)}$, and $U_n^{(4)}$ are the values U, U_y, U_x, and U_{xy} at n. The HC problem is to find $U \in \mathcal{V}$ which satisfies (1) on a set $\mathcal{G} \cup \partial\mathcal{G}$ of $N = 4n_G$ *collocation points*. A good choice for $\mathcal{G}$ and $\partial\mathcal{G}$ uses the tensor product of two Gauss points per interval (see [HMR85]).

Schur Complement Preconditioning

Let $D = \{1, 2, \ldots, N\}$ denote the index set of all unknowns (equations). Substructuring performs a symmetric permutation of A such that subdomain indices (I) are first, then edge indices (E), then vertex indices (V). This induces a partition $D = I \cup E \cup V$. Let $B = E \cup V$ be the set of interface indices. The unknown vector $\mathbf{u}$ can be written as $\mathbf{u} = (\mathbf{u_I}, \mathbf{u_B})^\mathbf{T}$, where $\mathbf{u_B} = (\mathbf{u_E}, \mathbf{u_V})^\mathbf{T}$. For simplicity, assume a perfectly uniform partition, where every subdomain and edge has N_I and N_E unknowns, respectively. If N_V is the number of vertex unknowns, then $N = n_I N_I + n_E N_E + N_V$, where n_I and n_E are the number of subdomains and edges respectively.

The operator A after substructuring is

$$A = \begin{pmatrix} A_{II} & A_{IB} \\ A_{BI} & A_{BB} \end{pmatrix}, \text{with } A_{BB} = \begin{pmatrix} A_{EE} & A_{EV} \\ A_{VE} & A_{VV} \end{pmatrix}, A_{II} = \sum_{i=1}^{n_I} R_{I_i}^T A_{II}^{(i)} R_{I_i}. \quad (2)$$

The first (second) subscript of a block denotes the index set of the equations (unknowns) in that block, and $A_{BI} = (A_{EI}\, A_{VI})^T$, $A_{IB} = (A_{IE}\, A_{IV}) \neq A_{BI}^T$ even when $\mathbf{L}$ is SPD. The above structure of A is called *arrowhead*, since A_{II} is block diagonal with n_I blocks. Let $S = A_{BB} - A_{BI} A_{II}^{-1} A_{IB}$ be the Schur complement with respect to A_{BB}. A block triangular factorization of A^{-1} is

$$A^{-1} = \begin{pmatrix} I & -A_{II}^{-1} A_{IB} \\ 0 & I \end{pmatrix} \begin{pmatrix} A_{II}^{-1} & 0 \\ 0 & S^{-1} \end{pmatrix} \begin{pmatrix} I & 0 \\ -A_{BI} A_{II}^{-1} & I \end{pmatrix}. \quad (3)$$

A preconditioner for A employs a preconditioner $\hat{S}$ to S in (3), since S is dense and expensive to construct. Moreover, in general, $\hat{S}^{-1}$ is not explicitly computed, but its action on a vector $\mathbf{v_B}$ is computed instead. The seminal paper [BPS86] introduced the Bramble-Pasciak-Schatz (BPS) preconditioning strategy which uses a *coarse grid operator* A_H to provide global coupling between subdomains, and an *interpolation map* $R_H^T : V \mapsto B$ to approximate the solution on B. The BPS preconditioner is $\hat{S}_{BPS}^{-1} = \sum_{i=1}^{n_E} R_i^T S_{i,i}^{-1} R_i + R_H^T A_H^{-1} R_H$, where $R_i : B \mapsto E_i$ is the pointwise restriction map from B to the indices of the ith edge, and $S_{i,i}$ is the Schur complement for the ith edge. Note that the BPS method does not specify a scheme for approximating $S_{i,i}^{-1}$.

The Preconditioner

Substructuring applied to the HC problem induces the linear system $A\,\mathbf{u} = \mathbf{f}$, where $\mathbf{u} \in \mathbf{R^N}$, and which can be written as three subproblems:

$$\begin{aligned} A_{II}\mathbf{u_I} + \mathbf{A_{IE}u_E} + \mathbf{A_{IV}u_V} &= \mathbf{f_I} \text{ (subdomain subproblem)} && (4) \\ A_{EI}\mathbf{u_I} + \mathbf{A_{EE}u_E} + \mathbf{A_{EV}u_V} &= \mathbf{f_E} \text{ (edge subproblem)} && (5) \\ A_{VI}\mathbf{u_I} + \mathbf{A_{VE}u_E} + \mathbf{A_{VV}u_V} &= \mathbf{f_V} \text{ (vertex subproblem)} && (6) \end{aligned}$$

In (5) $A_{EE} = A_{EE}^e + A_{EE}^c$, with A_{EE}^e block diagonal, where a block comprises the degrees of freedom for an edge, and the superscripts e and c are abbreviations for "eigen" (own) and "coupled", respectively.

Our preconditioner is defined in terms of three grids: (i) the fine grid, on which the solution is sought; (ii) a hybrid fine/coarse grid, called the *edge grid*, associated with each edge, on which an approximate solution to (5) is found; and (iii) a *coarse grid* on which an approximate solution to (6) is found. The discretization on each grid is performed using the bicubics associated with that grid, e.g., the coarse grid basis functions centered at an interior vertex (x_k, y_m) have support $[x_k - H - h, x_k + H + h] \times [y_m - H - h, y_m + H + h]$, where H is the coarse grid parameter (see next paragraph). The collocation points are determined in terms of the fine grid only. Therefore, the right hand sides f_E and f_V are preserved when collocating on the edge and coarse grids, but the coefficient matrices change. Slightly abusing the notation, we denote by $(\mathbf{u_I}, \mathbf{u_E}, \mathbf{u_V})^\mathbf{T}$ the solution to the problem (4–6) as well as the solution to the approximate problem obtained using the three-grid discretization outlined above.

For simplicity, assume the domain Ω is decomposed into $n_I = (n_0 - 1)^2$ subdomains. To achieve perfect load balancing, suppose also that the fine grid is defined so that each subdomain contains exactly K^2 interior nodes. Figure 1 illustrates the case $\sqrt{n_I} = K = 3$. Let $H = Kh$ be the diameter of the corner subdomains. Subdomains away from $\partial\Omega$ have diameter $H + h$, while those along one side of $\partial\Omega$ are of size $(H + h) \times H$ or $H \times (H + h)$. Note also that nodes on $\partial\Omega$ are either vertex or interior nodes in this terminology.

We illustrate the definition of the edge grid by considering a vertical edge E_i. Assume E_i does not intersect $\partial\Omega$ (horizontal edges, and edges intersecting $\partial\Omega$ are handled similarly). Suppose E_i has endpoints (x, y) and $(x, y + H + h)$. The edge grid G_i associated with E_i is defined as $G_i = \{x - H, x, x + H\} \times \{y, y + h, y + 2h, \ldots, y + H + h\}$. Figure 1 shows the edge grid associated with E_4. Let V^i be the space of piecewise bicubic polynomials on $\mathcal{E}_i = [x - H, x + H] \times [y, y + H + h]$, where the Hermite bicubic basis functions have support $[-H, H] \times [-h, h]$ with respect to the nodes where they are centered. We approximate the restriction of u to $\mathcal{E}_i$ by an element in V^i. The collocation equation is $\hat{U}(\xi, \zeta) = \sum_{l=1}^{4} \sum_{n=1}^{4} \hat{U}_{k_n}^{(l)} \hat{\Phi}_{k_n}^{l}(\xi, \zeta)$ where $(\xi, \zeta) \in \mathcal{G} \cup \partial\mathcal{G}$ and $\hat{\Phi}_{k_n}^{l}$, $1 \le l, n \le 4$ are basis functions for V^i centered at the four nodes k_n closest to (ξ, ζ). Hence, we obtain the blocks $\hat{A}_{EE}$, $\hat{A}_{EI}$, and $\hat{A}_{EV}$ with elements $\mathbf{L}\hat{\mathbf{\Phi}}_{\mathbf{k_n}}^{\mathbf{l}}$ and $\mathbf{B}\hat{\mathbf{\Phi}}_{\mathbf{k_n}}^{\mathbf{l}}$, instead of A_{EE}, A_{EI}, and A_{EV} whose elements are $\mathbf{L}\mathbf{\Phi}_{\mathbf{k_n}}^{\mathbf{l}}$ and $\mathbf{B}\mathbf{\Phi}_{\mathbf{k_n}}^{\mathbf{l}}$. The matrix form of the edge problem is

$$\hat{A}_{EI}\mathbf{u_I} + \hat{\mathbf{A}}_{\mathbf{EE}}\mathbf{u_E} = \mathbf{f_E} - \hat{\mathbf{A}}_{\mathbf{EV}}\mathbf{u_V}, \text{ where } \hat{\mathbf{A}}_{\mathbf{EE}} = \hat{\mathbf{A}}_{\mathbf{EE}}^{\mathbf{e}} + \hat{\mathbf{A}}_{\mathbf{EE}}^{\mathbf{c}}, \tag{7}$$

with $\hat{A}_{EE}^{e}$ block diagonal. Similarly, we collocate on the coarse grid, approximate u in the function space corresponding to the coarse grid, and get the approximate vertex problem

$$\hat{A}_{VI}\mathbf{u_I} + \hat{\mathbf{A}}_{\mathbf{VV}}\mathbf{u_V} = \mathbf{f_V}. \tag{8}$$

The EVP provides an approximate solution to the system (4, 7, 8). We use implicit preconditioning, i.e., we compute the action of M^{-1} on a given vector $\mathbf{v}$ by solving the system $M\,\mathbf{u} = \mathbf{v}$. To construct the EVP we have to solve two problems: (i) decouple the coarse grid problem from the subdomains; (ii) find an approximation of (7) which removes the mutual edge coupling among the n_E edge subproblems due to $\hat{A}_{EE}^{c}$, so that the edge problems can be solved independently.

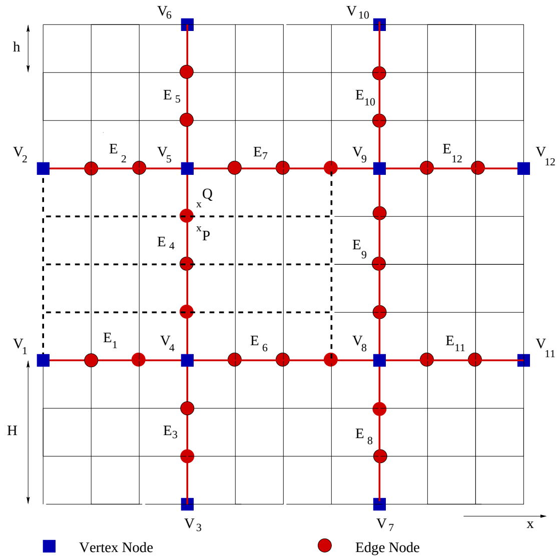


Figure 1 Three-level grid. The coarse grid is shown in thick solid lines. The edge grid for edge E_4 is shown in dashed lines. Edge nodes are shown as disks and vertex nodes as squares. P and Q are two of the collocation points associated with edge E_4.

The first problem is solved by observing that the vertex-subdomain coupling in (8) is due to the four corners of Ω which are subdomain nodes. Let $\bar{\mathbf{u}}_V = (\mathbf{u_C}, \mathbf{u_V})^{\mathbf{T}}$, where $\mathbf{u_C}$ is associated with the four corners. Let $\bar{A}_{VV}$ be the coarse grid operator obtained by expanding $\hat{A}_{VV}$ with coarse grid collocation at the collocation points associated with the corners. We get the independent vertex problem $\bar{A}_{VV}\,\bar{\mathbf{u}}_V = \bar{\mathbf{v}}_V$, where $\bar{\mathbf{v}}_V = (\mathbf{v_C}, \mathbf{v_V})^{\mathbf{T}}$.

Given $\mathbf{u_V}$, we solve the second problem by evaluating the approximate solution defined by $\mathbf{u_V}$ at those edge nodes that cause mutual edge coupling. Notice that only the edge nodes at a distance h from a vertex node cause mutual coupling. For example, the edge E_4 in Figure 1 is coupled to E_6 and E_7 by way of the nodes $(2H, H)$ and $(2H, 2H + h)$, respectively. Let $\tilde{\mathbf{u}}_E$ be the approximate value of the edge unknowns obtained by evaluating the piecewise bicubics defined by $\mathbf{u_V}$:

$\tilde{\mathbf{u}}_E = I_V^E\mathbf{u_V}$, where I_V^E denotes Hermite cubic interpolation from V to E unknowns.

We decouple the n_E edge problems by making the approximation $\hat{A}^{\mathrm{c}}_{EE}\mathbf{u_E} \approx \hat{\mathbf{A}}^{\mathbf{c}}_{\mathbf{EE}}\tilde{\mathbf{u}}_{\mathbf{E}}$ which gives $\hat{A}_{EE}\mathbf{u_E} \approx \hat{\mathbf{A}}^{\mathbf{e}}_{\mathbf{EE}}\mathbf{u_E} + \hat{\mathbf{A}}^{\mathbf{c}}_{\mathbf{EE}}\tilde{\mathbf{u}}_{\mathbf{E}}$ and (7) becomes

$$\hat{A}_{EI}\mathbf{u_I} + \hat{\mathbf{A}}^{\mathbf{e}}_{\mathbf{EE}}\mathbf{u_E} = \mathbf{v_E} - \hat{\mathbf{A}}^{\mathbf{c}}_{\mathbf{EE}}\tilde{\mathbf{u}}_{\mathbf{E}} - \hat{\mathbf{A}}_{\mathbf{EV}}\mathbf{u_V}. \tag{9}$$

The action of M^{-1} is computed in two main stages. First we solve for $\bar{\mathbf{u}}_V$ directly, $\bar{\mathbf{u}}_V = \bar{A}_{VV}^{-1}\bar{\mathbf{v}}_V$. Second we solve approximately the system consisting of (9) and the

1 $\bar{\mathbf{u}}_V = \bar{A}_{VV}^{-1}\,\bar{\mathbf{v}}_V$;	▷ Solve coarse grid problem
2 $\tilde{\mathbf{u}}_E = I_V^E \mathbf{u_V}$;	▷ Approximate edge unknowns
3 $\tilde{\mathbf{u}}_I = A_{II}^{-1}\,(\mathbf{v_I} - \mathbf{A_{IE}}\tilde{\mathbf{u}}_\mathbf{E} - \mathbf{A_{IV}}\mathbf{u_V})$;	▷ Approximately solve subdomains
4 $\mathbf{v_E} = \mathbf{v_E} - \hat{\mathbf{A}}_\mathbf{EI}\tilde{\mathbf{u}}_\mathbf{I} - \hat{\mathbf{A}}^\mathbf{c}_\mathbf{EE}\tilde{\mathbf{u}}_\mathbf{E} - \hat{\mathbf{A}}_\mathbf{EV}\mathbf{u_V}$;	▷ New right-hand side for interface
5 $\mathbf{u_E} = \hat{\mathbf{A}}^{\mathbf{e}\,-1}_{\mathbf{EE}}\,\mathbf{v_E}$;	▷ Solve interface
6 $\mathbf{u_I} = \mathbf{A_{II}^{-1}}(\mathbf{v_I} - \mathbf{A_{IE}}\,\mathbf{u_E} - \mathbf{A_{IV}}\,\mathbf{u_V})$;	▷ Solve subdomains

Figure 2 Edge-Vertex Preconditioning Algorithm: $\mathbf{u} = \mathbf{M^{-1}}\,\mathbf{v}$

subdomain problem (see (4)): $A_{II}\mathbf{u_I} + \mathbf{A_{IE}u_E} = \mathbf{v_I} - \mathbf{A_{IV}u_V}$. This is done in three steps. First, we solve approximately the subdomain problem by replacing $\mathbf{u_E}$ with $\tilde{\mathbf{u}}_E$: $A_{II}\tilde{\mathbf{u}}_I = \mathbf{v_I} - \mathbf{A_{IE}}\tilde{\mathbf{u}}_\mathbf{E} - \mathbf{A_{IV}u_V}$, for $\tilde{\mathbf{u}}_I$. Second, we plug $\tilde{\mathbf{u}}_I$ in (9) which we approximate by $\hat{A}^e_{EE}\mathbf{u_E} = \mathbf{v_E} - \hat{\mathbf{A}}_\mathbf{EI}\tilde{\mathbf{u}}_\mathbf{I} - \hat{\mathbf{A}}^\mathbf{c}_\mathbf{EE}\tilde{\mathbf{u}}_\mathbf{E} - \hat{\mathbf{A}}_\mathbf{EV}\mathbf{u_V}$. This is equivalent to approximating $\hat{A}^e_{EE}\mathbf{u_E} - \hat{\mathbf{A}}_\mathbf{EI}\mathbf{A_{II}^{-1}}\mathbf{A_{IE}}(\mathbf{u_E} - \tilde{\mathbf{u}}_\mathbf{E}) \approx \hat{\mathbf{A}}^\mathbf{e}_\mathbf{EE}\mathbf{u_E}$. This approximation is justified by the relation (see [Mat98] pages 53–54, for a proof):

$$||\hat{A}_{EI}A_{II}^{-1}A_{IE}\ \mathbf{u}||_\infty = \mathcal{O}\left(\frac{\mathbf{h}}{\mathbf{H}}\right)\ \left(\mathbf{1} + ||\mathbf{A_{EI}A_{II}^{-1}A_{IE}}||_\infty\right)\ ||\mathbf{u}||_\infty,\ \mathbf{u} \in \mathbf{R}^{\mathbf{n_E N_E}}.$$

Third, we solve for $\mathbf{u_I}$: $A_{II}\mathbf{u_I} = \mathbf{v_I} - \mathbf{A_{IE}u_E} - \mathbf{A_{IV}u_V}$. Figure 2 gives the steps to apply M^{-1} to $\mathbf{v} \in \mathbf{R^N}$.

The floating point operation cost of applying EVP is $C_P \approx 16NK + 8\frac{N^{1.5}}{K^3} + 5N + 58\frac{N}{K}$. The best choice for K is—if we account for the cost C_F of block factorizations—$K = O(N^{1/6})$, which gives $C_P = O(N^{7/6})$, $C_F = O(N^{4/3})$. The algorithm has good parallelism, except for step 1. To avoid a sequential bottleneck, the size of the coarse problem should be kept small enough.

Numerical Experiments

We employ GMRES with restart parameter 30, and the stopping rule is relative residual reduction of $\varepsilon = 10^{-5}$. The code is written in C and Fortran with parallel directives, and uses LAPACK's general band factorization and triangular solve procedures. The CPU time is given in seconds and is measured using the C-library function `times`. The tests are run on an SGI Origin 2000 machine with 16 processors running at 195-MHz. We use the following model problems defined on $\Omega = (0, 2)\times(0, 2)$ (drawn from the examples in Appendix A of [RB85]).

Problem 1. Problem 1 in [RB85] has a self-adjoint operator:

$$\left(e^{xy}\ u_x\right)_x + \left(e^{-xy}\ u_y\right)_y - (1 + x + y)^{-1}u = g_1, \tag{10}$$

where subscripts indicate partial differentiation and with g_1 such that $u_1 = \frac{3}{4}e^{xy}\sin(\pi x)\sin(\pi y)$ is the true solution. The boundary conditions are Dirichlet on one side and Neumann on the other three sides: $u = 0$ on $x = 2$ and $\frac{\partial u}{\partial \mathbf{n}} = \frac{\partial u_1}{\partial \mathbf{n}}$ on $y = 0$, $y = 2$, and $x = 0$.

Problem 2. Problem 2 in [RB85] has a general operator:

$$u_{xx} + (1 + y^2)\, u_{yy} - u_x - (1 + y^2)\, u_y = g_2, \tag{11}$$

with g_2 such that $u_2 = e^{x+y} + (x^2 - x)^2 \log(1 + y^2)$ is the solution. The boundary conditions are again Dirichlet on one side and Neumann on the other three: $u = u_2$ on $x = 2$ and $\frac{\partial u}{\partial \mathbf{n}} = \frac{\partial u_2}{\partial \mathbf{n}}$ on $y = 0$, $y = 2$, and $x = 0$.

Problem 3. Problem 12 in [RB85] has oscillatory coefficients of u_x and u:

$$u_{xx} + u_{yy} + (1 + \sin(\alpha x))\, u_x - \cos(\alpha y)\, u = g_3, \tag{12}$$

with g_3 determined by the solution $u_3 = \cos(\beta\, y) + \sin\beta\,(x - y)$, and with Dirichlet boundary conditions $u = u_3$ on $\partial\Omega$. We set $\alpha = \beta = \pi$.

Table 1 summarizes the performance of EVP on the three test problems. For constant problem size per subdomain (i.e., $K = H/h$ constant) the number of iterations is essentially constant, independent of H and h; for Problems 2 and 3 the number of iterations is reduced as N grows. For constant N, increasing the number of subdomains n_I can improve convergence as well. For example, for Problems 2 and 3 with $N = 36100$, increasing n_I from 6^2 to 8^2 reduces the number of iterations by one. This improvement is due to increasing coarse problem size; but eventually the reduction in iteration count is offset by the reduced efficiency from the sequential bottleneck of the coarse solve.

Table 2 compares EVP with PETSc's Additive Schwarz preconditioner [BGMSsc] with exact subdomain solves, for Problem 2 and constant N. Note that the ASM preconditioner is algebraic, i.e., it requires no knowledge of the PDE. We set the number of ASM blocks to p. EVP clearly outperforms ASM for this case. It is possible that ASM's performance would improve with a reordering of equations and unknowns.

Table 1 EVP performance.

K	n_I	N	Problem 1			Problem 2			Problem 3		
			Its	T_p	Mfl/sp	Its	T_p	Mfl/sp	Its	T_p	Mfl/sp
11	6^2	20164	17	1.6	29	5	0.7	35	5	0.7	37
	8^2	36100	21	2.8	27	3	0.8	32	4	0.9	34
	12^2	81796	21	5.3	21	2	1.5	25	3	1.6	26
	16^2	145924	20	11.4	13	1	2.0	25	2	2.7	19
15	6^2	36100	19	3.6	35	4	1.4	45	5	1.6	42
	8^2	64516	22	6.2	30	3	1.8	43	4	2.1	40
	10^2	101124	23	9.2	27	2	2.4	38	3	2.7	37
	12^2	145924	23	12.	24	2	3.1	36	3	3.5	35

Table 2 EVP versus Additive Schwarz. Problem 2, $N = 158404$, δ is the overlap, $S = T_1/T_p$ is the speedup, *Its* is the iteration count.

Processors	EVP, $n_I = 100, K = 19$				Additive Schwarz, $\delta = 2$			
	Its	Mflop/s per proc	T_p (sec)	S	*Its*	Mflop/s per proc	T_p (sec)	S
1	2	82	26.9	1	1	18	5053	1
2	2	80	13.8	1.9	19	16	1757	2.9
4	2	67	8.19	3.3	30	12	1257	4.0
8	2	55	5.01	5.4	67	19	490	10.3
10	2	50	4.39	6.1	92	13	606	8.3
12	2	51	3.63	7.4	117	11	642	7.9
14	2	48	3.29	8.2	120	9	736	6.9
16	2	45	3.08	8.7	150	7	831	6.1

REFERENCES

[BCDF94] Bialecki B., Cai X.-C., Dryja M., and Fairweather G. (1994) An additive Schwarz algorithm for piecewise Hermite bicubic orthogonal spline collocation. In *Domain Decomposition Methods in Science and Engineering*, volume 157 of *Contemporary Mathematics*, pages 237–244. AMS, Providence, RI.

[BGMSsc] Balay S., Gropp W., McInnes L. C., and Smith B. (1998, http://www.mcs.anl.gov/petsc) Petsc 2.0 User's Manual. Report ANL 95/11 - Revision 2.0.22, Argonne National Laboratory, Argonne, IL.

[Bia93] Bialecki B. (April 1993) A fast domain decomposition Poisson solver on a rectangle for Hermite bicubic orthogonal spline collocation. *SIAM J. Numer. Anal.* 30(2): 425–434.

[BPS86] Bramble J. H., Pasciak J. E., and Schatz A. H. (1986) The construction of preconditioners for elliptic problems by substructuring, I. *Math. Comp.* 47(175): 103–134.

[BPX90] Bramble J. H., Pasciak J. E., and Xu J. (1990) Parallel multilevel preconditioners. *Math. Comp.* 55: 1–22.

[BW86] Bjørstad P. E. and Widlund O. B. (1986) Iterative methods for the solution of elliptic problems on regions partitioned into substructures. *SIAM J. Numer. Anal.* 23(6): 1093–1120.

[CK90] Chan T. F. and Keyes D. E. (1990) Interface preconditioning for domain-decomposed convection-diffusion operators. In Chan T., Glowinski R., Périaux J., and Widlund O. (eds) *Third International Symposium on Domain Decomposition Methods for Partial Differential Equations*, pages 245–262. SIAM, Philadelphia, PA.

[HMR85] Houstis E., Mitchell W., and Rice J. (December 1985) Collocation software for second-order elliptic partial differential equations. *ACM Trans. Math. Software* 11(4): 379–412.

[Mat98] Mateescu G. (1998) *Domain Decomposition Preconditioners for Hermite Collocation Problems*. PhD thesis, Virginia Tech, Blacksburg, VA.

[MRW98] Mateescu G., Ribbens C. J., and Watson L. T. (1998) A domain decomposition algorithm for Hermite collocation problems. In *Proceedings of the Fifth Copper Mtn. Conf. on Iterative Methods*. Copper Mtn., CO.

[RB85] Rice J. R. and Boisvert R. F. (1985) *Solving Elliptic Problems Using ELLPACK*. Springer-Verlag, New York.

[SBG96] Smith B. F., Bjørstad P. E., and Gropp W. (1996) *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*. Cambridge University Press.

[SS86] Saad Y. and Schultz M. H. (1986) GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comp.* 7: 856–869.

9

Preconditioning Operators for Elliptic Problems with Bad Parameters

S.V. Nepomnyaschikh[1]

INTRODUCTION

In this paper we design preconditioning operators for the system of grid equations approximating the following boundary value problem.

$$-\sum_{i,j=1}^{2} \frac{\partial}{\partial x_i} a_{ij}(x) \frac{\partial u}{\partial x_j} + a_0(x)u = f(x), \quad x \in \Omega, \quad u(x) = 0, \quad x \in \Gamma \qquad (1)$$

We suppose that Ω is a bounded and polygonal domain, where Γ does denote its boundary. Let Ω be a union of $n+1$ nonoverlapping subdomains Ω_i, such that

$$\overline{\Omega} = \bigcup_{i=0}^{n} \overline{\Omega}_i, \quad \Omega_i \cap \Omega_j = \emptyset, \quad i \neq j,$$

holds. Here we have the polygonal subdomains Ω_i in the interior of Ω. Their boundaries are given by Γ_i, $i = 1, \ldots, n$. The domain Ω_0 is defined to be multiple connected having the boundary $\Gamma \bigcup (\bigcup_{i=1}^{n} \Gamma_i)$. We denote by $H_i = \mathrm{diam}(\Omega_i)$ the diameter of the i-th subdomain, $i = 1, \ldots, n$. We assume small parameters H_i such that

$$0 < H_i \leq 1$$

1 Russian Academy of Sciences, Computing Center, Lavrentiev Av. 6, Novosibirsk, 630090, Russia. email: svnep@oapmg.sscc.ru
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

is valid. Furthermore, for any subdomain Ω_i, if there exists a subdomain Ω_j such that

$$\operatorname{dist}(\Omega_i, \Omega_j) \leq \alpha_1 H_i$$

holds, then the conditions

$$H_j = O(H_i) \quad \text{and} \quad \alpha_2 H_i \leq \operatorname{dist}(\Omega_i, \Omega_j)$$

must be fulfilled, where α_1 and α_2 are constants wich are independent of the parameter $H_i, i = 1, \ldots, n$. This means that for any subdomain Ω_i there is no other subdomain in the neighbourhood determined by $O(H_i)$.

Let us introduce the bilinear form

$$a(u,v) = \int\limits_{\Omega} \left(\sum_{i,j=1}^{2} a_{ij}(x) \frac{\partial u}{\partial x_j} \frac{\partial v}{\partial x_i} + a_0(x) uv \right) dx$$

and the linear functional

$$l(v) = \int\limits_{\Omega} f(x) v dx.$$

We suppose that the coefficients of the problem (1) are such that $a(u, v)$ is a symmetric bilinear form in the Sobolev space $H_0^1(\Omega)$. Let the inequalities

$$\alpha_3 a(u,v) \leq \int\limits_{\Omega} \epsilon(x) |grad(u)|^2 \leq \alpha_4 a(u,v) \quad \forall u \in H_0^1(\Omega).$$

be fulfilled with positive constants α_3, α_4, which are independent of the parameter ϵ. Here we fix

$$\epsilon(x) = \text{const} = \epsilon_i, \quad \forall x \in \Omega_i,$$

where we have

$$\epsilon_0 = 1, \quad 0 < \epsilon_i \leq 1, \quad i = 1, \ldots, n \; . \tag{2}$$

The linear functional $l(v)$ is continuous in $H_0^1(\Omega)$. The weak formulation of (1) is given as follows. Find $u \in H_0^1(\Omega)$ such that the following is valid for all $v \in H_0^1(\Omega)$

$$a(u,v) = l(v) \; . \tag{3}$$

Let $\Omega^h = \bigcup_{i=0}^{n} \Omega_i^h$ be a quasiuniform triangulation of the domain Ω, which can be characterized by the parameter h.

We denote by W the space of real continuous functions being linear on the triangles of the triangulation Ω^h. Using the finite element method, see e.g. [Mar82], the variational formulation (3) can be transfered to the well known system of linear algebraic equations

$$Au = f \; . \tag{4}$$

The condition number of the matrix A depends on the parameters h, H_i and ϵ_i, and can be large. Our purpose is the design of a preconditioner B for the problem (4), such that the following inequalities are valid for all vectors $u \in R^N$

$$c_1(Bu,u) \leq (Au,u) \leq c_2(Bu,u) \; . \tag{5}$$

Here the symbol N denotes the dimension of the space W, and c_1 and c_2 are positive constants independent of the parameters h, H_i, and ϵ_i. Furthermore, the multiplication of a vector by B^{-1} should be easy to implement numerically causing low costs.
The preconditioning operator B is constructed by using the nonoverlapping and overlapping (but without "overlapping" in the coefficients) domain decomposition methods. The analysis of these methods refers to the well known Neumann-Dirichlet domain decomposition method. However, the suggested methods do not require the exact solution of subproblems with Dirichlet boundary condition.

NONOVERLAPPING DOMAIN DECOMPOSITION

The construction of the preconditioner for the system (4) is performed by means of the Additive Schwarz Method, see e.g. [Lio88],[MN85],[MN88]. To design the preconditioning operator B, we use [Nep91a],[Nep92] decomposing the space W into a sum of subspaces as follows

$$W = W_0 + W_1$$

We divide the nodes of the triangulation Ω^h into two groups, those which lie inside of $\Omega_i^h, i = 1, \ldots, n$ and those which lie in $\overline{\Omega}_0^h$. The subspace W_0 does correspond to the first set. Let us introduce the following sets

$$S = \bigcup_{i=1}^{n} \partial\Omega_i^h,$$

$$W_0 = \left\{ u^h \in W | \quad u^h(x) = 0, x \in \overline{\Omega}_0^h \right\},$$

$$W_{0,i} = \{ u^h \in W_0 | \quad u^h(x) = 0, \quad x \overline{\in} \Omega_i^h \}, \quad i = 1, 2, \ldots, n.$$

It is clear that W_0 represents the direct sum of the orthogonal subspaces $W_{0,i}$ with respect to the scalar product in $H_0^1(\Omega)$

$$W_0 = W_{0,1} \oplus \ldots \oplus W_{0,n} \ .$$

The subspace W_1 corresponds to the second group of nodes in Ω^h and can be defined as follows. Let the set V be the trace space of the functions given by W on S, i.e. we have

$$V = \{ \varphi^h | \quad \varphi^h(x) = u^h(x), \quad x \in S, \quad u^h \in W \}.$$

To define the subspace W_1, we need a norm preserving extension operator of functions given on S into Ω^h. The corresponding construction is based on the following trace lemma.

Lemma 1 *Let Ω be a bounded domain with piecewisely smooth boundary Γ satisfying the Lipschitz condition. Let*

$$\mathrm{diam}(\Omega) = H \ .$$

And let Ω^h be a quasiuniform triangulation of Ω. We denote

$$\begin{aligned} \|\varphi\|^2_{H^{\frac{1}{2}}(\Gamma)} &= H\|\varphi\|^2_{L^2(\Gamma)} + |\varphi|^2_{H^{\frac{1}{2}}(\Gamma)}, \\ \|\varphi\|^2_{L^2(\Gamma)} &= \int_\Gamma \varphi^2(x)dx, \\ |\varphi|^2_{H^{\frac{1}{2}}(\Gamma)} &= \int_\Gamma \int_\Gamma \frac{(\varphi(x)-\varphi(y))^2}{|x-y|^2}\, dxdy\,. \end{aligned}$$

Then, there exists a positive constant c_1, which is independent of the parameters h, H, such that

$$\|\varphi^h\|_{H^{\frac{1}{2}}(\Gamma)} \le c_1\|u^h\|_{H^1(\Omega)}$$

and

$$|\varphi^h|_{H^{\frac{1}{2}}(\Gamma)} \le c_1|u^h|_{H^1(\Omega)}$$

hold for any function $u^h \in W$, where $\varphi^h \in V$ is the trace of u^h on the boundary Γ. Vice versa, there exists a positive constant c_2, which is independent of h and H, such that for any function $\varphi^h \in V$ we have the function $u^h \in W$ with

$$\begin{aligned} u^h(x) &= \varphi^h(x), \quad x \in \Gamma\,, \\ \|u^h\|_{H^1} &\le c_2\|\varphi^h\|_{H^{\frac{1}{2}}(\Gamma)}, \\ |u^h|_{H^1} &\le c_2|\varphi^h|_{H^{\frac{1}{2}}(\Gamma)}. \end{aligned}$$

To define the subspace W_1, let us use the explicit extension operator

$$t^h : V \to W, \tag{6}$$

which was suggested for second order elliptic problems with smooth coefficients, such that for all $\varphi^h \in V$

$$\|u^h\|_{H^1(\Omega)} = \|t^h\varphi^h\|_{H^1(\Omega)} \le c_3\|\varphi^h\|_{H^{\frac{1}{2}}(S)}$$

holds, where the corresponding norm is given by

$$\|\varphi\|^2_{H(S)} = \sum_{i=1}^{n} \|\varphi\|^2_{H^{\frac{1}{2}}(\Gamma_i)}\,.$$

For defining and implementing the numerical algorithm see [MN93],[Nep86],[Nep91c]. Now, we can define the subspace W_1 as follows

$$\begin{aligned} W_1 = \{u^h| \quad u^h(x) &= (t^h\varphi^h)(x), \quad x \in \Omega_i, \quad i = 1,\dots,n, \quad \varphi^h(x) = v^h(x), \quad x \in S, \\ u^h(x) &= v^h(x), \quad x \in \Omega_0^h, \quad v^h \in W\}. \end{aligned}$$

Obviously we have

$$W = W_0 + W_1\,,$$

and this decomposition of the space W is stable in the following sense.

Lemma 2 *There exists a positive constant c_4, which is independent of the parameters h, H_i and ϵ_i, such that for any function $u^h \in W$ there exist functions $u_i^h \in W_i, i = 0, 1,$ such that we have*

$$u_0^h + u_1^h = u^h ,$$

$$a(u_0^h, u_0^h) + a(u_1^h, u_1^h) \leq c_4 a(u^h, u^h) .$$

Let C_i , $i = 0, 1, \dots, n$ be the preconditioning operators in the finite element subspaces $H_0^1(\Omega_i)$. Hence, we have the following inequalities for all $u^h \in W \cap H_0^1(\Omega_i)$

$$c_5 \|u^h\|_{H^1(\Omega_i)}^2 \leq (C_i u, u) \leq c_6 \|u^h\|_{H^1(\Omega_i)}^2 , \tag{7}$$

where the constants c_5, c_6 are independent of the parameters h and H_i. For example, these operators C_i can be constructed using the fictitious space lemmata in [Nep91c],[Nep91b],[Nep92],[Nep95],[Xu96]. We extend the operator C_i outside of Ω_i by zero and denote by C_i^+ the pseudo-inverse operator belonging to this extension. We introduce the following operator

$$B_{\text{nov}}^{-1} = t C_0^+ t^* + \frac{1}{\epsilon_1} C_1^+ + \dots + \frac{1}{\epsilon_n} C_n^+ .$$

Here the operator t^* is the adjoint to t. The following theorem holds.

Theorem 1 *There exist positive constants c_7, c_8, which are independent of the parameters h, H_i and ϵ_i, such that the following inequalities are fulfilled for all $u \in R^N$*

$$c_1 (B_{\text{nov}} u, u) \leq (Au, u) \leq c_2 (B_{\text{nov}} u, u) .$$

OVERLAPPING DOMAIN DECOMPOSITION

The goal of this section is the design of the preconditioning operators for the problem (4) without using the extension operator t given in (6).
Let C be the preconditioning operator in the finite element space W, such that for all functions $u^h \in W$ we have

$$c_1 \|u^h\|_{H^1(\Omega)}^2 \leq (Cu, u) \leq c_2 \|u^h\|_{H^1(\Omega)}^2 ,$$

where the constants c_1, c_2 are independent of h. We denote the preconditioner B_{ov}^{-1} as follows

$$B_{\text{ov}}^{-1} = C^{-1} + \frac{1}{\epsilon_1} C_1^+ + \dots + \frac{1}{\epsilon_n} C_n^+ .$$

Here the pseudoinverses C_i^+ are given by (7). The following theorem holds.

Theorem 2 *There exist positive constants c_3, c_4, which are independent of the parameters h, H_i and ϵ_i, such that the inequalities*

$$c_3 (B_{\text{ov}} u, u) \leq (Au, u) \leq c_4 (B_{\text{ov}} u, u)$$

are fulfilled for all $u \in R^N$.

Proof:
In the case of $\epsilon_i = 1\,, i = 1, \ldots, n$, using Theorem 1 there exist constants c_5, c_6, which are independent of h and H_i, such that

$$c_5(C^{-1}u, u) \le tC_0^+t^* + C_1^+ + \cdots + C_n^+ \le c_6(C^{-1}u, u)$$

holds for all $u \in R^N$. From (2) we get

$$0 \le (C_i^+ u, u) \le \frac{1}{\epsilon_i} \le (C_i^+ u, u) \quad \forall u \in R^N .$$

Hence, we have

$$\begin{aligned}
(B_{\text{nov}}^{-1}u, u) &= tC_0^+t^* + \frac{1}{\epsilon_1}C_1^+ + \cdots + \frac{1}{\epsilon_n}C_n^+ \\
&\le tC_0^+t^* + C_1^+ + \cdots + C_n^+ + \frac{1}{\epsilon_1}C_1^+ + \cdots + \frac{1}{\epsilon_n}C_n^+ \\
&\le \max\{c_6, 1\}((C^{-1} + \frac{1}{\epsilon_1}C_1^+ + \cdots + \frac{1}{\epsilon_n}C_n^+)u, u) = \max\{c_6, 1\}(B_{\text{ov}}^{-1}u, u) \\
&\le \max\{c_6, 1\}\max\{\frac{1}{c_5}, 1\}(tC_0^+t^* + C_1^+ + \cdots + C_n^+ + \frac{1}{\epsilon_1}C_1^+ + \cdots + \frac{1}{\epsilon_n}C_n^+)u, u) \\
&\le 2\max\{c_6, 1\}\max\{\frac{1}{c_5}, 1\}(B_{\text{nov}}^{-1}u, u) \,.
\end{aligned}$$

Remark The above Theorem 2 can be proved directly without using the extension operator t.

Acknowledgment The author wishes to thank U. Langer and J. Schoeberl from the Johannes–Kepler–University in Linz, Austria for promoting and helpfully discussing the topic of this paper.

REFERENCES

[Lio88] Lions P. (1988) On the schwarz alternating method. In *I. First International Symposium on Domain Decomposition Methods for Partial Differential Equations*. SIAM Philadelphia.

[Mar82] Marchuk G. (1982) *Methods of Numerical Mathematics*. Springer, New York.

[MN85] Matsokin A. and Nepomnyaschikh S. (1985) Schwarz alternating method in subspaces. *Soviet Mathematics* 29: 78–84.

[MN88] Matsokin A. and Nepomnyaschikh S. (1988) Norms in the space of traces of mesh functions. *Sov. J. Numer. Anal. Math. Modeling* 3: 199–216.

[MN93] Matsokin A. and Nepomnyaschikh S. (1993) Method of fictitious space and explicit extension operators. *Zh. Vychisl. Mat. Mat. Fiz.* 33: 52–68.

[Nep86] Nepomnyaschikh S. (1986) *Domain decomposition and Schwarz methods in a subspace for the approximate solution of elliptic boundary value problems.* PhD dissertation, Computing Center of the Siberian Branch of the USSR Academy of Sciences, Novosibirsk, USSR.

[Nep91a] Nepomnyaschikh S. (1991) Domain decomposition method for elliptic problems with discontinuous coefficients. In *Proc. 4th Conference on Domain Decomposition methods for Partial Differential Equations*. Philadelphia PA SIAM.

[Nep91b] Nepomnyaschikh S. (1991) Mesh theorems on traces, normalization of function traces and their inversion. *Sov. J. Numer. Anal. Math. Modelling* 6: 223–242.

[Nep91c] Nepomnyaschikh S. (1991) Method of splitting into subspaces for solving elliptic boundary value problems in complex-form domains. *Sov. J. Numer. Anal. Math. Modelling* 6: 151–168.

[Nep92] Nepomnyaschikh S. (1992) Decomposition and fictitious domain methods for elliptic boundary value problems. In *5th Conference on Domain Decomposition Methods for Partial Differential Equations*. Philadelphia PA SIAM.

[Nep95] Nepomnyaschikh S. (1995) Fictitious space method on unstructured meshes. *East-West J. Numer. Math.* 4: 71–79.

[Xu96] Xu (1996) The auxiliary space method and optimal multigrid preconditioning techniques for unstructured grids. *Computing* 56.

10

Is there a curse of dimension for integration?

ERICH NOVAK[1]

INTRODUCTION

How does the cost of optimal methods increase with the dimension? We answer this question for integration of very smooth functions that are defined on the d-dimensional unit cube. For suitable classes of smooth functions the order of convergence of optimal methods does not depend on d. Nevertheless we prove exponential lower bounds for quadrature formulas with positive weights. Therefore the optimal order of convergence does not say much about the tractability or intractability of a problem.

This paper is about lower bounds, valid for arbitrary methods. The lower bounds are proved for the problem of numerical integration. Since numerical integration is part of the problem to solve a partial differential equation, these lower bounds can be applied to the (more difficult) problem of solving PDEs.

Let $\mathcal{F}_d$ be a class of functions defined on the unit cube $\Omega_d \subset \mathbf{R}^d$. We assume that $S_d : \mathcal{F}_d \to G_d$ is a sequence of operators which we want to approximate. For example $S_d(f) = u$ could be the solution of a linear PDE $Au = f$. To approximate S_d we use methods Q_n based on n function values,

$$Q_n(f) = \varphi(f(x_1), \ldots, f(x_n)), \tag{1}$$

where $\varphi : \mathbf{R}^d \to G_d$. The (worst case) error of Q_n is given by

$$e(Q_n) = \sup_{f \in \mathcal{F}_d} \|S_d(f) - Q_n(f)\|_{G_d},$$

1 Mathematisches Institut, Universität Erlangen-Nürnberg, Bismarckstr. 1 1/2, D-91054 Erlangen, Germany, email: novak@mi.uni-erlangen.de
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

the nth minimal error by

$$e_n(\mathcal{F}_d) = \inf_{Q_n} e(Q_n).$$

The number $e_n(\mathcal{F}_d)$ is the error of the optimal method for the class $\mathcal{F}_d$, using at most n function values. The numbers

$$n(\varepsilon, d) = \min\{n \mid e_n(\mathcal{F}_d) \leq \varepsilon\}$$

are a measure for the complexity of computing the operators $S_d : \mathcal{F}_d \to G_d$.

In this paper we are mainly interested in lower bounds. We want to show that even for "simple" problems we have exponential lower bounds; hence there is an inherent "curse of dimension". Therefore we only consider the integration problem, where $G_d = \mathbf{R}$ and

$$S_d(f) = \int_{\Omega_d} f(x)\, dx.$$

Our classes $\mathcal{F}_d$ will be unit balls with respect to a norm. Then it is known that it is enough to study linear methods of the form

$$Q_n(f) = \sum_{i=1}^{n} c_i\, f(x_i) \tag{2}$$

with $c_i \in \mathbf{R}$ and $x_i \in \Omega_d$, since arbitrary nonlinear methods (1) cannot be better than linear methods (2), see [TWW88] or [Nov96]. For the classes $\mathcal{F}_d$ considered in this paper we have

$$e(Q_0) = e_0(\mathcal{F}_d) = 1, \tag{3}$$

where $Q_0(f) = 0$ is the trivial quadrature formula. Hence the problem is properly scaled for all d. We say that Q_n is positive if $c_i \geq 0$ for all $i = 1, \ldots, n$. Positive formulas are preferred due to their strong stability properties. Therefore we also define

$$e_n^+(\mathcal{F}_d) = \inf_{Q_n} e(Q_n),$$

where the infimum only runs through the set of positive quadrature formulas (2), and

$$n^+(\varepsilon, d) = \min\{n \mid e_n^+(\mathcal{F}_d) \leq \varepsilon\}.$$

The order of convergence of $e_n(\mathcal{F}_d)$ is known for many function classes $\mathcal{F}_d$. We often know that

$$e_n(\mathcal{F}_d) \asymp e_n^+(\mathcal{F}_d),$$

but we usually do not know whether $e_n(\mathcal{F}_d)$ and $e_n^+(\mathcal{F}_d)$ are equal. We only mention two specific results. Let $W_2^k([0,1]^d)$ be the classical Sobolev space with the norm

$$\|f\|_{W_2^k}^2 = \sum_{|\alpha| \leq k} \|D^\alpha f\|_{L_2}^2, \tag{4}$$

where $\alpha \in \mathbf{N}_0^d$ and $|\alpha| = \sum_{l=1}^d \alpha_l$, and let $BW_2^k([0,1]^d)$ be the unit ball of $W_2^k([0,1]^d)$. We assume the imbedding condition $2k > d$. It is well known that

$$e_n(BW_2^k([0,1]^d) \asymp e_n^+(BW_2^k([0,1]^d) \asymp n^{-k/d}.$$

Instead of BW_2^k, defined by the norm (4), we can define a set BH_2^k by the tensor product norm

$$\|f\|^2_{H_2^k} = \sum_{\alpha_l \le k} \|D^\alpha f\|^2_{L_2}, \tag{5}$$

where the sum is over all $\alpha \in \mathbf{N}_0$ with $\alpha_l \le k$ for all l. We obtain a space $H_2^k([0,1]^d)$ of functions with bounded mixed derivatives and denote its unit ball by $BH_2^k([0,1]^d)$. It is known that

$$e_n(BH_2^k([0,1]^d) \asymp e_n^+(BH_2^k([0,1]^d) \asymp n^{-k} \cdot (\log n)^{(d-1)/2}. \tag{6}$$

In [NR96] we constructed quadrature formulas using *nonuniform sparse grids* based on the Chebyshev knots. These quadrature formulas have the following key properties, see also [NR97a] and [NR97b].

- *Simplicity.* The formulas can be easily computed for arbitrary dimensions d, also for weighted integrals with a weight ρ of tensor product form. We need $\mathcal{O}(n)$ arithmetic operations to compute the knots and weights of Q_n from suitable univariate quadrature formulas.
- *High Polynomial Exactness.* A small number of knots is sufficient to integrate all d-variate polynomials of a given degree exactly. We need

$$n \approx 2^k/k! \cdot d^k$$

  knots for exactness of degree $2k+1$ if k is fixed and d is large. The well known lower bound is roughly $1/k! \cdot d^k$. Thus we have a polynomial dependence on the dimension d, and the exponent k is best possible.
- *Universality.* Small errors are guaranteed for many different smoothness classes. If f has bounded derivatives up to order k then

$$|I_d(f) - Q_n(f)| = \mathcal{O}\big(n^{-k/d} \cdot (\log n)^{(d-1)(k/d+1)}\big). \tag{7}$$

  If f has a bounded mixed derivative $f^{(k,\dots,k)}$ then

$$|I_d(f) - Q_n(f)| = \mathcal{O}\big(n^{-k} \cdot (\log n)^{(d-1)(k+1)}\big). \tag{8}$$

  These two estimates are optimal – up to logarithmic factors – in both smoothness scales. The bounds (7) and (8) hold for all $k \in \mathbf{N}$, hence the method is not only almost optimal for a specific class of integrands but universal, i.e., it is almost optimal for many different function classes.

Sparse grids with equidistant sets of knots were often studied in the literature. Some authors mainly discuss periodic functions and then equidistant knots are quite adequate, see [Tem93] and [CNR98]. We prefer to study the general (nonperiodic) case and then it is much better, already for $d = 1$, to use nonequidistant knots, such as the Chebyshev knots. These knots, or modifications of these knots, were also used by [Coo98], [FH96], [GG98], and [NRSS97]. In [NRS98] we discuss related quadrature formulas for the Wiener measure, see [Ste99] for the proof of polynomial error bounds. For the problem of *interpolation* or *optimal recovery*, see [BNR98], [Spr97a], and [Spr97b].

The order of $e_n(BH_2^k([0,1]^d)$ is between n^{-k} and $n^{-k+\delta}$, for any $\delta > 0$, independently of d. Hence we may say that the *order of convergence* does *not* depend on d. It is tempting to say that the complexity of the problem does not depend on d, there is no curse of dimension. We will show in the following that such a conclusion would be wrong.

Consider the following problem. Assume that $k \in \mathbf{N}$ is fixed and consider

$$n^+(\varepsilon, d) = \inf\{n \mid e_n^+(BH_2^k([0,1]^d) \leq \varepsilon\}.$$

The problem is to verify whether $n^+(\varepsilon, d)$ depends polynomially on d and ε^{-1}. If so then the problem is *tractable*, if not then it is *intractable*. Observe that the known order (6), with unknown constants that depend on d, does not say anything about tractability. We proved in [Nov98] that the problem is intractable since

$$e_n^+(BH_2^k([0,1]^d)^2 \geq 1 - n \cdot c_k^d,$$

where $c_k < 1$ does not depend on d. It is still open whether such a lower bound also holds for the e_n, i.e., for arbitrary quadrature formulas.

We now consider "very smooth" functions. We use the norm

$$\|f\|_{F_d}^2 = \sum_{\alpha \in \mathbf{N}_0^d} \|D^\alpha f\|_{L_2}^2 \tag{9}$$

and define

$$F_d = \{f : \Omega \to \mathbf{R} \mid \|f\|_{F_d} \leq 1\}.$$

Observe that

$$F_d \subset \bigcap_{k=1}^\infty BH_2^k([0,1]^d)$$

which yields

$$e_n^+(F_d) \leq c_{d,k} \cdot n^{-k}$$

for *all* $d, k \in \mathbf{N}$, where $c_{d,k} > 0$ depends on d and k. Moreover we have $e_1^+(F_d) = 1$ for all d, so the problem is properly scaled and the order of convergence is very high for all d.

Nevertheless we will prove the exponential (in d) lower bound

$$e_n^+(F_d)^2 \geq 1 - n \cdot c^d,$$

where $c = 0.995$, for positive quadrature formulas. For the proof it is convenient to consider even smaller classes BV_d of functions. The BV_d contain polynomials of degree at most two in each variable. For any fixed $\varepsilon < 1$ we need a number n of sample points which increases exponentially with the dimension. We do not know whether arbitrary quadrature formulas for the class F_d are better than positive quadrature formulas.

We finish this section by a bibliographic remark. Explicit *upper* bounds (without unknown constants) were recently obtained by [WW95] and [WW98]. The paper of [SW98] contains *upper and lower* bounds for certain tensor product Hilbert spaces and for quasi-Monte Carlo methods, i.e., for formulas with equal weights $c_i = 1/n$. Some lower bounds can be generalized to the class of positive quadrature formulas, see [Nov98] and [Woź98]. General results about tractability of linear problems can be found in [Woź94a], [Woź94b], and [NSW97]. For related results see also [SW97] and [HW99].

OPTIMAL ERROR BOUNDS AND OTHER EXTREMAL PROBLEMS

We begin with the definition of the classes V_d of functions on Ω_d. Here Ω_d is the unit cube in $\mathbf{R}^d$ which we define, for convenience, in a shifted way through $\Omega_d = [-1/2, 1/2]^d$. The space V_1 is generated by $f_1 = 1$, $f_2(x) = x$, and $f_3(x) = x^2$. These functions are not orthogonal, with respect to the usual Sobolev-Hilbert norms

$$\|f\|^2_{W_2^k} = \sum_{j=0}^{k} \|D^j f\|^2_{L_2}, \tag{10}$$

and hence we prefer the functions $e_1 = 1$, $e_2(x) = x$, and $e_3(x) = x^2 - 1/12$. The e_i are orthogonal for each norm (10). We obtain $\|e_1\|_{W_2^k} = 1$ for all k,

$$\|e_2\|_{W_2^k} = (1 + 1/12)^{1/2}$$

for all $k \geq 1$, and

$$\|e_3\|_{W_2^k} = (4 + 1/3 + 1/180)^{1/2}$$

for all $k \geq 2$. We define a scalar product on V_1 by $(e_i, e_j) = 0$ for $i \neq j$ and

$$(e_1, e_1) = 1, \quad (e_2, e_2) = 13/12, \quad \text{and} \quad (e_3, e_3) = (4 + 1/3 + 1/180).$$

We denote by BV_1 the unit ball of V_1 with respect to this norm. The norm on V_1 coincides with the k-norm (10) for each $k \geq 2$ and also with the norm (9).

For $d > 1$, we define V_d as the tensor product $V_d = V_1 \otimes \ldots \otimes V_1$ (d times). The dimension of V_d is 3^d. On the space V_d we define the tensor product (cross-) norm by

$$(f_1 \otimes \ldots \otimes f_d,\, g_1 \otimes \ldots \otimes g_d) := \prod_{m=1}^{d} (f_m, g_m). \tag{11}$$

Let BV_d denote the unit ball of V_d. The norm defined by (11) is the same as (5), for each $k \geq 2$, and is the same as (9). Hence we obtain $BV_d \subset F_d$ and $e_n^+(BV_d) \leq e_n^+(F_d)$. Observe that

$$I_d(f) = (f, 1) \qquad \text{and} \qquad \|1\| = 1,$$

for all $d \in \mathbf{N}$. It follows that $e(Q_0, BV_d) = e_0(BV_d) = 1$, see (3).

A Dirac functional $f \mapsto f(x)$ can be written in the form

$$f(x) = (\delta_x, f)$$

for all $f \in V_d$ since we have a kernel reproducing Hilbert space. Here δ_x is a suitable element of V_d which is a tensor product of the respective δ_{x^m}, where $x = (x^1, \ldots, x^d)$ and $\delta_{x^m} \in V_1$. To find the

$$\delta_x = \sum_{j=1}^{3} c_j(x)\, e_j$$

in the case $d = 1$, we have to solve the linear system $e_i(x) = c_i(x)\,(e_i, e_i)$, for $i = 1, 2, 3$, and get

$$c_1(x) = 1, \qquad c_2(x) = \frac{12x}{13}, \qquad c_3(x) = \frac{x^2 - 1/12}{4 + 1/3 + 1/180}.$$

Therefore we obtain the following lemma.

Lemma 1 *The kernel k_1 of V_1 is given by*

$$k_1(x,y) = (\delta_x, \delta_y) = 1 + \frac{12xy}{13} + \frac{(x^2-1/12)(y^2-1/12)}{4+1/3+1/180}.$$

For $d>1$ the kernel is given by

$$k_d(x,y) = (\delta_x, \delta_y) = \prod_{m=1}^{d} k_1(x^m, y^m),$$

where $x = (x^1, \ldots, x^d)$.

Later we need the values

$$k_{\min} = \inf_{x,y} k_1(x,y) \approx 0.94 \qquad \text{and} \qquad k_{\max} = \inf_{x} k_1(x,x) \approx 1.005.$$

For our approach it is important that the kernel is positive, $k_1(x,y) \geq 0$. It is easy to check that

$$e(Q_n, BV_d) = \left\| 1 - \sum_{i=1}^{n} c_i\, \delta_{x_i} \right\|$$

for Q_n of the form (2). The optimal error bounds $e_n(BV_d)$ for quadrature formulas are related to the solution of the following extremal problem for polynomials. For given points $x_i \in \Omega_d$ find

$$f \in V_d \quad \text{with} \quad f(x_1) = \ldots = f(x_n) = 1 \quad \text{and} \quad \|f\| \to \min!$$

We define the numbers

$$g(n,d) = \sup_{x_1,\ldots,x_n} \inf\{\|f\|^2 \,:\, f \in V_d,\ f(x_1) = \ldots = f(x_n) = 1\}.$$

Then one can prove that $g(n,d) + e_n(BV_d) = 1$, the proof is similar as in [Nov98], where a different kernel and periodic functions were studied. It follows that for $d, n \in \mathbf{N}$ and $\Phi > 0$ the following statements are equivalent:

(a) $e_n^+(BV_d)^2 \geq 1 - \Phi$;
(b) Any linear system of the form

$$\sum_{i=1}^{m} c_i\, k_d(x_i, x_j) = 1, \tag{12}$$

$j = 1, \ldots, m$, with $m \leq n$ and a solution that satisfies $c_i \geq 0$ for all i fulfills $\sum_{i=1}^{m} c_i \leq \Phi$.

Consider the linear system (12) for $c_i \geq 0$. The diagonal elements are at least $k_{\max}^d$ and the off-diagonal elements are at least $k_{\min}^d$. Summing up all the equations we get

$$m \;=\; \sum_{i,j=1}^{m} c_i\, k_d(x_i, x_j) \geq \sum_{i=1}^{m} c_i \left(k_{\max}^d + (m-1) k_{\min}^d\right).$$

This yields

$$\sum_{i=1}^{m} c_i \leq \frac{m}{k_{\max}^d + (m-1)k_{\min}^d} \leq \frac{n}{k_{\max}^d + (n-1)k_{\min}^d}.$$

Hence we obtain the following lower bound.

Theorem 1 *Let $d, n \in \mathbf{N}$. Then*

$$e_n^+(BV_d)^2 \geq 1 - n \cdot \left(k_{\max}^d + (n-1)k_{\min}^d\right)^{-1}.$$

We end this paper with a remark. We showed that high dimensional integration is intractable for positive quadrature formulas, even if the functions are very smooth. If weighted norms are considered, however, then certain high dimensional integrals can be computed efficiently, see [SW98] and [Nov98].

Acknowledgment

I am grateful to Henryk Woźniakowski for valuable comments on the paper.

REFERENCES

[BNR98] Barthelmann V., Novak E., and Ritter K. (1998) High dimensional polynomial interpolation on sparse grids. *Preprint* .

[CNR98] Cools R., Novak E., and Ritter K. (1998) Smolyak's construction of cubature formulas of arbitrary trigonometric degree. *Computing* to appear.

[Coo98] Cools R. (1998) Integration over an hypercube. Technical report.

[FH96] Frank K. and Heinrich S. (1996) Computing discrepancies of Smolyak quadrature rules. *J. Complexity* 12: 287–314.

[GG98] Gerstner T. and Griebel M. (1998) Numerical integration using sparse grids. *Num. Algorithms* 18: 209–232.

[HW99] Hickernell F. J. and Woźniakowski H. (1999) Integration and approximation in arbitrary dimension. *Preprint* .

[Nov96] Novak E. (1996) On the power of adaption. *J. Complexity* 12: 199–237.

[Nov98] Novak E. (1998) Intractability results for positive quadrature formulas and extremal problems for trigonometric polynomials. *J. Complexity* to appear.

[NR96] Novak E. and Ritter K. (1996) High dimensional integration of smooth functions over cubes. *Numer. Math.* 75: 79–97.

[NR97a] Novak E. and Ritter K. (1997) The curse of dimension and a universal method for numerical integration. In Nürnberger G., Schmidt J. W., and Walz G. (eds) *Multivariate Approximation and Splines*, pages 177–187. Birkhäuser, ISNM 125, Basel.

[NR97b] Novak E. and Ritter K. (1997) Simple cubature formulas with high polynomial exactness. *Constr. Approximation* to appear.

[NRS98] Novak E., Ritter K., and Steinbauer A. (1998) A multiscale method for the evaluation of Wiener integrals. In Chui C. K. and Schumaker L. L. (eds) *Approximation Theory IX*. Nashville, TN.

[NRSS97] Novak E., Ritter K., Schmitt R., and Steinbauer A. (1997) On a recent interpolatory method for high dimensional integration. *J. Comput. Appl. Math.* to appear.

[NSW97] Novak E., Sloan I. H., and Woźniakowski H. (1997) Tractability of tensor product linear operators. *J. Complexity* 13: 387–418.

[Spr97a] Sprengel F. (1997) Interpolation and wavelets on sparse Gauss-Chebyshev grids. In Haussmann W. (ed) *Multivariate approximation. Recent trends and results. Proceedings of the 2nd international conference, Witten-Bommerholz, Germany, September 29–October 4, 1996*, pages 269–286. Akademie Verlag, Berlin.

[Spr97b] Sprengel F. (1997) A unified approach to error estimates for interpolation on full and sparse Gauß-Chebyshev grids. *Rostocker Math. Kolloq.* 51: 51–64.

[Ste99] Steinbauer A. (1999) Quadrature formulas for the Wiener measure. *J. Complexity* to appear.

[SW97] Sloan I. H. and Woźniakowski H. (1997) An intractability result for multiple integration. *Math. Comp.* 66: 1119–1124.

[SW98] Sloan I. H. and Woźniakowski H. (1998) When are quasi-Monte Carlo algorithms efficient for high dimensional integrals? *J. Complexity* 14: 1–33.

[Tem93] Temlyakov V. N. (1993) *Approximation of Periodic Functions.* Nova Science, New York.

[TWW88] Traub J. F., Wasilkowski G. W., and Woźniakowski H. (1988) *Information-Based Complexity.* Academic Press.

[Woź94a] Woźniakowski H. (1994) Tractability and strong tractability of linear multivariate problems. *J. Complexity* 10: 96–128.

[Woź94b] Woźniakowski H. (1994) Tractability and strong tractability of multivariate tensor product problems. *J. of Computing and Information* 4: 1–19.

[Woź98] Woźniakowski H. (1998) Efficiency of quasi-Monte Carlo algorithms for high dimensions. *Preprint* to appear.

[WW95] Wasilkowski G. W. and Woźniakowski H. (1995) Explicit cost bounds of algorithms for multivariate tensor product problems. *J. Complexity* 11: 1–56.

[WW98] Wasilkowski G. W. and Woźniakowski H. (1998) Weighted tensor-product algorithms for linear multivariate problems. *J. Complexity* to appear.

11

Interface Preconditioners and Multilevel Extension Operators

P. OSWALD[1]

INTRODUCTION

Interface problems arising in nonoverlapping domain decomposition methods can often be viewed as restriction of the original elliptic problem to a discrete trace space. As has been demonstrated in many particular examples (see the references below), this connection can also be used for the construction of interface preconditioners and extension operators. We present an abstract additive Schwarz framework for this deduction, with applications to multilevel schemes.

ABSTRACT FRAMEWORK

We assume familiarity with the theory of stable space splittings and additive Schwarz methods for solving symmetric positive definite (spd) variational problems in a Hilbert space [DW90, Xu92, Osw94a, SBG96]. Let the bilinear forms $a(\cdot,\cdot)$ and $b_j(\cdot,\cdot)$ be bounded and spd on V and V_j, respectively, and $R_j : V_j \to V$ be linear operators. Then we call

$$\{V; a\} \cong \sum_j R_j \{V_j; b_j\} \tag{1}$$

1 Bell Laboratories, Lucent Technologies, Murray Hill, NJ 07974-0636, U.S.A. email: poswald@research.bell-labs.com
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

a *stable space splitting*, with stability constants $0 < \mu_1 \le \mu_2 < \infty$, if

$$\mu_1 a(v,v) \le |||v|||^2 := \inf_{v_j \in V_j : v = \sum_j R_j v_j} \sum_j b_j(v_j, v_j) \le \mu_2 a(v,v) \quad \forall\, v \in V \; . \tag{2}$$

If $V_j \subset V$ and if R_j represents the natural embeddings then R_j will be omitted. The upper estimate is equivalent to assuming that for any $v \in V$ and any $\epsilon > 0$ we can find a decomposition

$$v = \sum_j R_j v_j \; , \quad v_j \in V_j \quad : \quad \sum_j b_j(v_j, v_j) \le (1+\epsilon)\mu_2 a(v,v) \; , \tag{3}$$

while the lower bound in (2) is often reduced to the verification of so-called strengthened Cauchy-Schwarz inequalities [Xu92].

Given a continuous linear operator $T : V \to X$, its range $\hat{V} \subset X$, can be converted into a Hilbert space if we introduce the following norm on $\hat{V}$:

$$||\hat{v}||_{\hat{V}} = \inf_{v \in V : \hat{v} = Tv} ||v||_V \qquad \forall\, \hat{v} \in \hat{V} \; . \tag{4}$$

E.g., if V is a function space with domain Ω, and T represents a trace operator to a subset Γ then this is the common *implicit* definition of the trace space norm. The construction of equivalent *explicit* (or *intrinsic*) norms is one of the major problems in the theory of trace spaces. Given a $\hat{v} \in \hat{V}$, the construction of a $v \in V$ with $\hat{v} = Tv$ and $||v||_V \le C_{ext} ||\hat{v}||_{\hat{V}}$ is called *extension problem*. Clearly, one is interested in an as small as possible constant $C_{ext} < \infty$, and an efficient realization of the mapping $E : \hat{v} \to v$. Similar definitions occur in connection with *Schur complement problems* in substructuring methods.

We present the assumptions for a generic construction of a stable space splitting for $\{\hat{V}; \hat{a}\}$ if a splitting (1) is available for $\{V; a\}$. Here, $\hat{a}(\cdot,\cdot)$ is a bounded and spd bilinear form on $\hat{V}$. In particular, this implies that

$$\hat{a}(Tv, Tv) \le C_1 a(v,v) \qquad \forall\, v \in V \; , \tag{5}$$

and that for any $\hat{v} \in \hat{V}$

$$\exists\, v \in V \; : \quad \hat{v} = Tv \; , \;\; a(v,v) \le C_2 \hat{a}(\hat{v}, \hat{v}) \; . \tag{6}$$

with some absolute constants $0 < C_1, C_2 < \infty$. Suppose that T_j acts on V_j, with range $\hat{V}_j$, and that bounded spd forms $\hat{b}_j$ on $\hat{V}_j$ are introduced (in complete analogy with the introduction of $\hat{V}$ and $\hat{a}$). We assume that all appearing constants are uniform w.r.t. j. In particular,

$$\hat{b}_j(T_j v_j, T_j v_j) \le C_3 b_j(v_j, v_j) \qquad \forall\, v_j \in V_j \; , \tag{7}$$

for some C_3, independent of j. Let $E_j : \hat{V}_j \to V_j$ be right inverses of T_j, $T_j E_j = \mathrm{Id}_{\hat{V}_j}$, such that

$$b_j(E_j \hat{v}_j, E_j \hat{v}_j) \le C_4 \hat{b}_j(\hat{v}_j, \hat{v}_j) \qquad \forall\, v_j \in V_j \; , \tag{8}$$

for some C_4, independently of j. The last, and most critical condition is on the kernels of the operators T_j and TR_j: For all j,

$$\mathcal{N}(\mathcal{T}_|) \subset \mathcal{N}(\mathcal{T}\mathcal{R}_|) \; . \tag{9}$$

Theorem 1 *Under the assumptions (5), (6), (7), (8), and (9), the stability of (1) implies the stability of the space splitting*

$$\{\hat{V};\hat{a}\} \cong \sum_j \hat{R}_j\{\hat{V}_j;\hat{b}_j\} \qquad (\ \hat{R}_j = TR_jE_j\ )\ , \tag{10}$$

with stability constants $\hat{\mu}_1, \hat{\mu}_2$ satisfying

$$0 < (C_1C_4)^{-1}\mu_1 \le \hat{\mu}_1 \le \hat{\mu}_2 \le C_2C_3\mu_2 < \infty\ . \tag{11}$$

Proof. Let $\hat{v} = \sum_j \hat{R}_j\hat{v}_j$ be any representation of $\hat{v}$. Set $v_j = E_j\hat{v}_j$ and $v = \sum_j R_jv_j$. Obviously,

$$Tv = \sum_j TR_jv_j = \sum_j TR_jE_j\hat{v}_j = \sum_j \hat{R}_j\hat{v}_j = \hat{v}\ ,$$

and by (5), (8), and the lower estimate in (2), we have

$$\begin{aligned} \hat{a}(\hat{v},\hat{v}) &= \hat{a}(Tv,Tv) \le C_1a(v,v) \le C_1\mu_1^{-1}\sum_j b_j(v_j,v_j) \\ &= C_1\mu_1^{-1}\sum_j b_j(E_j\hat{v}_j,E_j\hat{v}_j) \le C_1C_4\mu_1^{-1}\sum_j \hat{b}_j(\hat{v}_j,\hat{v}_j)\ . \end{aligned}$$

This gives the lower stability estimate for (10).

In the other direction, take a v with $\hat{v} = Tv$ satisfying (6), and decompose it optimally in the sense of (3) using the upper stability estimate in (2). Set $u_j = E_jT_jv_j$, $\hat{v}_j = T_jv_j$, and observe that $T_j(u_j - v_j) = (T_jE_j - \mathrm{Id}_{\hat{V}_j})T_jv_j = 0$. Thus, by (9) $0 = TR_j(u_j - v_j) = TR_jE_jT_jv_j - TR_jv_j = \hat{R}_j\hat{v}_j - TR_jv_j$. This implies

$$\hat{v} = Tv = \sum_j TR_jv_j = \sum_j \hat{R}_j\hat{v}_j\ ,$$

and, taking into account also (7),

$$\begin{aligned} \sum_j \hat{b}_j(\hat{v}_j,\hat{v}_j) &= \sum_j \hat{b}_j(T_jv_j,T_jv_j) \le C_3\sum_j b_j(v_j,v_j) \\ &\le (1+\epsilon)C_3\mu_2a(v,v) \le (1+\epsilon)C_2C_3\mu_2\hat{a}(\hat{v},\hat{v})\ . \end{aligned}$$

Letting $\epsilon \to 0$ finishes the proof of Theorem 1. Note that we could have replaced (9) by the slightly weaker condition $\mathcal{R}(\mathrm{Id}_{V_j} - E_jT_j) \subset \mathcal{N}(\mathcal{TR}_|)$.

Corollary 2 *Under the assumptions of Theorem 1, given an arbitrary $\hat{v} \in \hat{V}$, let $\hat{v}_j \in \hat{V}_j$ be such that the analog of (3) holds:*

$$\hat{v} = \sum_j \hat{R}_j\hat{v}_j\ :\quad \sum_j \hat{b}_j(\hat{v}_j,\hat{v}_j) \le C_5\hat{a}(\hat{v},\hat{v})\ . \tag{12}$$

Then a suitable extension $u = E\hat{v} \in V$ of $\hat{v}$ satisfying $\hat{v} = Tu$ is given by

$$u = \sum_j R_jE_j\hat{v}_j\ :\quad a(u,u) \le C_4C_5\hat{a}(\hat{v},\hat{v}) \le C_1C_4C_5 \inf_{v\in V\,:\,\hat{v}=Tv} a(v,v)\ . \tag{13}$$

If $\hat{v}_j = \hat{Q}_j\hat{v}$ for some linear operators $\hat{Q}_j\ :\ \hat{V} \to \hat{V}_j$ then the extension operator $E = \sum_j R_jE_j\hat{Q}_j\ :\ \hat{V} \to V$ is linear and bounded.

A particular instance, where this abstract framework for constructing additive Schwarz preconditioners and extension operators can be applied, are *multilevel splittings* based on a hierarchy of spaces

$$V_0 \xrightarrow{I_1} V_1 \xrightarrow{I_2} \ldots \xrightarrow{I_J} V_J \xrightarrow{I_{J+1}} \ldots , \tag{14}$$

where $I_j : V_{j-1} \to V_j$ are given prolongation operators acting between subsequent spaces of the hierarchy (again, nesting of $\{V_j\}$ is not assumed). Setting $V = V_J$ and assuming that $a = a_J$ is a bounded and spd bilinear form on V_J, the associated multilevel splitting is given by

$$\{V_J; a_J\} \cong \sum_{j=0}^{J} R_j^J \{V_j; b_j\} , \qquad R_j^k = I_k \ldots I_{j+1} , \ \ 0 \le j < k . \tag{15}$$

Suppose that (15) is stable, uniformly w.r.t. J. Assume that the sequence T_j is given, and that $T = T_J$. Furthermore, let (5), (6) hold with $a, \hat{a}$ replaced by $a_J, \hat{a}_J$ and with constants independent of J. We keep the requirements (7) and (8), while (9) is replaced by the existence of another set of linear operators

$$\hat{I}_j : \hat{V}_{j-1} \to \hat{V}_j : \quad T_j I_j = \hat{I}_j T_{j-1} , \quad j \ge 1 . \tag{16}$$

This new requirement implies (9) since

$$T_J R_j^J = (T_J I_J) I_{J-1} \ldots I_{j+1} = \hat{I}_J (T_{J-1} I_{J-1}) \ldots I_{j+1} = \ldots = \hat{I}_J \ldots \hat{I}_{j+1} T_j .$$

Thus, Theorem 1 and Corollary 2 are applicable, leading to splittings and extension operators for $\{\hat{V}_J; \hat{a}_J\}$ and J-independent constants in the estimates. Moreover, by the same reasoning we obtain the recursion formula

$$\hat{R}_j^J = T_J R_j^J E_j = \hat{I}_J \ldots \hat{I}_{j+1} T_j E_j = \hat{I}_J \ldots \hat{I}_{j+1} , \quad 0 \le j < J < \infty .$$

This is important for the efficient implementation of additive and multiplicative Schwarz methods associated with $\{\hat{V}_j\}$.

APPLICATION

Applications in the spirit of Theorem 1 for preconditioning Schur complement problems in substructuring methods for finite element discretizations of second order elliptic boundary value problems are well-known (see [Osw94b], sections 2.3 and 7.1 of [XZ99], [Kho98b] for some special cases and further references). Thess [The98] has recently worked on applications to fourth order plate and shell problems. The Stokes problem has been treated in [Kho98a].

We illustrate the application of the above abstract framework by deriving an optimal complexity extension procedure for linear finite element boundary data on locally refined meshes (for similar results on preconditioning, see [KP98]). For simplicity, let Ω be a bounded polyhedral domain in $\mathbf{R^d}$, $d \ge 2$, equipped with a nested sequence of regular, quasi-uniform simplicial partitions $\mathcal{T}_j$ of element diameter $\approx 2^{-j}$, $j \ge 0$. Let

$\tilde{V}_j$ be the space of linear finite elements w.r.t. $\mathcal{T}_|$, the standard basis function in $\tilde{V}_j$ associated with a vertex $P_{j,i}$ of $\mathcal{T}_|$ is denoted by $N_{j,i}$. *Nested refinement* is modelled by selecting a finite increasing sequence

$$\emptyset = \Omega_0' \subset \Omega_1' \subset \ldots \Omega_J' \subset \Omega_{J+1}' = \Omega \, , \tag{17}$$

where each Ω_j', $1 \le j \le J$, is or empty or the closure of the union of some simplices from $\mathcal{T}_{|-\infty}$. The sequence $\Omega_j = \mathrm{clos}(\Omega \backslash \Omega_j')$ is decreasing and, roughly speaking, represents the simplices of $\mathcal{T}_{|-\infty}$ which are refined at level j. The spaces V_j are spanned by all basis functions $N_{j,i}$ for which $P_{j,i}$ belongs to the interior of Ω_j, and as the finite element space V corresponding to this refinement process we take $V = \sum_{j=0}^{J} V_j$ (this construction is discussed in [Osw94a]section 4.2.3, be aware of some index errors there!). It is easy to see that an algebraic basis $\mathcal{N}$ for V is given by all $N_{j,i} \in V_j \backslash V_{j+1}$, $j \le J$. Although V is not a traditional finite element space associated with an triangulation, its approximation power on each domain Ω_j is almost as good as that of $\tilde{V}_j$ (and less good in Ω_j'). Along the lines of [Osw94a]section 4.2.3 it follows that

$$\{V; (\cdot,\cdot)_{H^1(\Omega)}\} \cong \sum_{j=0}^{J} \sum_{i\,:\,N_{j,i} \in V_j} \{V_{j,i}, 2^{2j}(\cdot,\cdot)_{L_2(\Omega)}\} \, , \tag{18}$$

where $V_{j,i}$ are the one-dimensional spaces spanned by individual $N_{j,i}$, and with bounds for the stability constants that are independent on J and $\{\Omega_j'\}$. The stability of the splitting (18) expresses nothing but the optimality of BPX multilevel preconditioning in the nested refinement case, and can be derived from the corresponding result in the uniform refinement case (see [Osw94a]Theorem 19) by quasi-interpolant techniques.

Let us now consider the construction of a discrete harmonic extension operator $E : \hat{V} \equiv V|_\Gamma \to V$, where Γ is the boundary of Ω (note that everything extends to polyhedral manifolds Γ consisting of $(d-1)$-dimensional faces of simplices in $\mathcal{T}_\prime$). Obviously, $\hat{V} = \sum_{j=0}^{J} \hat{V}_j$, where $\hat{V}_j \equiv V_j|_\Gamma \subset \hat{\tilde{V}}_j \equiv \tilde{V}_j|_\Gamma$. The sequence $\{\hat{V}_j\}$ could have been derived by a nested refinement procedure on Γ similar to the one described above (consider the triangulations $\hat{\mathcal{T}}_j$ induced from $\mathcal{T}_|$ on Γ, and use $\hat{\Omega}_j' = \Omega_j' \cap \Gamma$ for the increasing sequence (17)). A basis in $\hat{V}_j$ is given by all non-vanishing $\hat{N}_{j,i} = N_{j,i}|_\Gamma$, where $N_{j,i} \in V_j$. Accordingly, a basis $\hat{\mathcal{N}}$ in $\hat{V}$ is given by all $\hat{N}_{j,i} \in \hat{V}_j \backslash \hat{V}_{j+1}$, $j \le J$.

Let $T : V \to \hat{V}$, $T_{j,i} : V_{j,i} \to \hat{V}_{j,i}$ coincide with the restriction of the trace operator $|_\Gamma$ to the corresponding spaces (if $P_{j,i} \notin \Gamma$ then $\hat{V}_{j,i}$ is the null space and $T_{j,i}$ the null operator). Since the operators $R_{j,i}$ represent the natural embeddings for $V_{j,i} \subset V$, condition (9) is automatically fulfilled. Define extension operators $E_{j,i} : \hat{V}_{j,i} \to V_{j,i}$ by simply setting $E_{j,i}(c\hat{N}_{j,i}) = cN_{j,i}$. Combining the $E_{j,i}$ with j fixed leads to the usual level-j extension-by-zero operator $E_j : \hat{V}_j \to V_j$. Obviously, the assumptions (7), (8) are satisfied if we set

$$b_{j,i}(u,v) = 2^{2j}(u,v)_{L_2(\Omega)} \, , \qquad \hat{b}_{j,i}(\hat{u},\hat{v}) = 2^j(\hat{u},\hat{v})_{L_2(\Gamma)} \, .$$

From the first part of Theorem 1 it follows that

$$\{\hat{V}; (\cdot,\cdot)_{\hat{V}}\} \cong \sum_{j=0}^{J} \sum_{i\,:\,\hat{N}_{j,i} \in \hat{V}_j} \{\hat{V}_{j,i}, 2^j(\cdot,\cdot)_{L_2(\Gamma)}\} \tag{19}$$

is also a stable splitting.

The next steps are as follows. First, it can be shown that

$$\|\hat{v}\|_{\hat{V}} \approx \|\hat{v}\|_{H^{1/2}(\Gamma)} \equiv \inf_{u\in H^1(\Omega)\,:\,v=Tu} \|u\|_{H^1(\Omega)} \qquad \forall \hat{v} \in \hat{V} \ . \tag{20}$$

This relates the $\|\cdot\|_{\hat{V}}$ norm defined in (4) to the norm in the trace space $H^{1/2}(\Gamma)$ for $H^1(\Omega)$ functions (intrinsic $H^{1/2}(\Gamma)$ norms are described, e.g., in [Kho98b, Xu92]). By definition of $\|\cdot\|_{\hat{V}}$, we obviously have $\|\hat{v}\|_{H^{1/2}(\Gamma)} \le \|\hat{v}\|_{\hat{V}}$. The proof of the opposite estimate uses the same techniques as were applied in [Osw94a]section 4.2.3 to derive (18) from the standard BPX-splitting for the whole space $\{H^1(\Omega);(\cdot,\cdot)_{H^1}\}$. The necessary quasi-interpolant operators will be described below, the details are omitted.

Secondly, in order to define an extension operator $E\ :\ \hat{V} \to V$ along the lines of Corollary 2, we need a realization of (12). Using the L_2-stability of the finite element bases (this fact will be used below without further mentioning), the desired representation follows if we construct a sequence of linear, uniformly L_2-bounded projections $\hat{P}_j\ :\ L_2(\Gamma) \to \hat{\hat{V}}_j$ such that $\hat{Q}_j = \hat{P}_j - \hat{P}_{j-1}$ maps $\hat{V}$ into $\hat{V}_j$, $0 \le j \le J$ (set $\hat{P}_{-1} = 0$). We use specific quasi-interpolants: Set

$$\hat{P}_j \hat{v}(x) = \sum_{i\,:\,P_{j,i}\in\Gamma} (\hat{v}, \hat{\lambda}_{j,i})_{L_2(\Gamma)} \hat{N}_{j,i}(x) \ , \tag{21}$$

where the functions $\hat{\lambda}_{j,i}$ are piecewise linear (but not continuous!) on the $(d-1)$-dimensional simplices of $\hat{\mathcal{T}}_j$ and have support $\hat{\Lambda}_{j,i}$ consisting of a few simplices attached to $P_{j,i}$. More precisely, we define

$$\hat{\Lambda}_{j,i} = \begin{cases} \mathrm{supp}\,\hat{\mathrm{N}}_{\mathrm{j,i}} & \text{if } \mathrm{supp}\,\hat{\mathrm{N}}_{\mathrm{j,i}} \subset \hat{\Omega}_{\mathrm{j+1}} \\ \mathrm{supp}\,\hat{\mathrm{N}}_{\mathrm{j,i}} \cap \Omega'_{\mathrm{j+1}} & \text{otherwise} \end{cases} \ . \tag{22}$$

To ensure the projection property w.r.t. $\hat{V}_j$, we define

$$\hat{\lambda}_{j,i}(x) = (\int_{\hat{\Lambda}_{j,i}} \hat{N}_{j,i}\, dy)^{-1} \hat{\psi}_{j,i}(x)\,, \quad x \in \hat{\Lambda}_{j,i} \ , \tag{23}$$

where the piecewise linear function $\hat{\psi}_{j,i}$ has value d at $P_{j,i}$, and -1 at all other vertices of simplices attached to $P_{j,i}$. The reader can easily verify that

$$(\hat{N}_{j,l}, \hat{\lambda}_{j,i})_{L_2(\Gamma)} = \delta_{i,l} \ , \qquad |(\hat{v}, \hat{\lambda}_{j,i})_{L_2(\Gamma)}|^2 \le C 2^{j(d-1)} \|\hat{v}\|^2_{L_2(\hat{\Lambda}_{j,i})} \ ,$$

which ensures the projection property w.r.t. $\hat{\hat{V}}_j$ and the uniform L_2-boundedness of the $\hat{P}_j$, respectively. Finally, if $\hat{v} \in \hat{V}$ then $\hat{v}$ is a linear finite element function w.r.t. $\mathcal{T}_{|-\infty}$ outside $\hat{\Omega}_j$. Thus, by our choice of the $\hat{\Lambda}_{j,i}$, both $\hat{P}_j$ and $\hat{P}_{j-1}$ reproduce $\hat{v}$ on $\hat{\Omega}'_j$ exactly. This gives $\hat{Q}_j \hat{v} = 0$ on this set and since $\hat{Q}_j \hat{v} \in \hat{\hat{V}}_j$ we arrive at $\hat{Q}_j \hat{v} \in \hat{V}_j$ for any $\hat{v} \in \hat{V}$.

Let $\hat{Q}_{j,i}$ ($\hat{Q}_j = \sum_i \hat{Q}_{j,i}$) be the resulting linear mappings from $\hat{V}$ into the one-dimensional $\hat{V}_{j,i}$ corresponding to the basis functions $\hat{N}_{j,i}$. Using the properties of $\hat{P}_j, \hat{Q}_j$ in conjunction with (19), (20), from Corollary 2 we deduce

Corollary 3 *Under the above assumptions on the refinement process, the multilevel extension operator* $E = \sum_{j=0}^{J} E_j \hat{Q}_j = \sum_{j=0}^{J} \sum_i E_{J,i} \hat{Q}_{j,i} \ : \ \hat{V} \to V$ *satisfies*

$$\|E\hat{v}\|_{H^1(\Omega)} \leq C \|\hat{v}\|_{H^{1/2}(\Gamma)} \qquad \forall\, \hat{v} \in \hat{V}\ , \tag{24}$$

where the constant C *is independent of the particular refinement process.*

We briefly describe how to efficiently implement this extension operator. First of all, we need to fix vector representations for finite element functions in V and $\hat{V}$. In both cases, we could use the bases $\mathcal{N}$, $\hat{\mathcal{N}}$ mentioned above or (allowing for some non-uniqueness in the representation) slightly larger generating systems

$$\mathcal{R} = \{\mathcal{N}_{|,\rangle}\ :\ \mathcal{N}_{|,\rangle} \in \mathcal{V}_|\ ,\ | \leq \mathcal{J}\}\ , \quad \hat{\mathcal{R}} = \{\hat{\mathcal{N}}_{|,\rangle}\ :\ \hat{\mathcal{N}}_{|,\rangle} \in \hat{\mathcal{V}}_|\ ,\ | \leq \mathcal{J}\}\ .$$

It can be shown that $\#\mathcal{R} \leq \mathcal{C}\#\mathcal{N}$, $\#\hat{\mathcal{R}} \leq \#\hat{\mathcal{N}}$, with C independent on the refinement process. The operator E naturally maps any $\hat{v}$ into a coefficient vector w.r.t. a subset of $\mathcal{R}$ of the same size as $\hat{\mathcal{R}}$, and can be implemented in $\mathrm{O}(\#\hat{\mathcal{R}})$ operations. Thus, except for the conversion of $E\hat{v}$ into a representation w.r.t. $\mathcal{N}$, storage and amount of work associated with E are proportional to $\dim \hat{V}$.

To substantiate this claim, we outline a computational scheme for $\hat{Q}_j \hat{v}$. The basic idea is to switch to yet another representation of finite element functions, this time w.r.t. the standard basis of discontinuous piecewise linear functions w.r.t. the given nested refinement structure. Set $\hat{\mathcal{B}} = \cup_{j=0}^{J} \hat{\mathcal{B}}_j$, where $\hat{\mathcal{B}}_j$ consists of all piecewise linear functions $\hat{\psi}_{\Delta,P}$ with support in a simplex $\Delta \in \hat{\mathcal{T}}_j$ belonging to $\hat{\Omega}_j^*$ and having value 1 at the vertex P of Δ and vanishing at all other vertices. Minimally, a suitable $\hat{\Omega}_j^*$ should contain $\hat{\Omega}_j$ and possibly an additional layer of $\hat{\mathcal{T}}_{j-1}$ simplices containing the supports of all $\hat{N}_{j,i}$ whose supports intersect with $\hat{\Omega}_{j+1}$ (the reason for this will become clear below). Obviously, the size of $\hat{\mathcal{B}}$ is still within a constant multiple of $\dim \hat{V}$. We also need the subsets $\hat{\mathcal{B}}_j' \subset \hat{\mathcal{B}}_j$ of all those functions $\hat{\psi}_{\Delta,P}$ for which $\Delta \in \hat{\Omega}_j^* \backslash \hat{\Omega}_{j+1}^*$ (again, $\hat{\Omega}_{J+1}^* = \emptyset$, and $\Omega_0^* = \Omega$).

The unique coefficients of $\hat{v} \in \hat{V}$ w.r.t. $\hat{\mathcal{B}}' = \cup_{j=0}^{J} \hat{\mathcal{B}}_j'$ can be computed fast, from either the $\hat{\mathcal{R}}$ or the $\hat{\mathcal{N}}$ representation, by recursion in $j = 0, 1, \ldots, J$. After this, by recursion in $j = J, J-1, \ldots, 0$, all scalar products $(\hat{v}, \hat{\psi}_{\Delta,P})_{L_2(\Gamma)}$ associated with functions $\hat{\psi}_{\Delta,P} \in \hat{\mathcal{B}}_j$ can be computed in an amount of operations which stays proportional to $\#\hat{\mathcal{B}}$. Indeed, for a level j simplex $\Delta \in \hat{\Omega}_j^* \backslash \hat{\Omega}_{j+1}^*$ this reduces to a multiplication with a local Gram matrix, while for $\Delta \subset \hat{\Omega}_{j+1}^*$ the refinability of the basis functions $\hat{\psi}_\Delta$ is used: each such function can be expressed as a linear combination of a fixed finite number of functions $\hat{\psi}_{\Delta',P'}$ associated with $\Delta' \in \hat{\mathcal{T}}_{j+1}$ in $\hat{\Omega}_{j+1}^*$ for which the corresponding integrals have been computed before. Finally, by formula (21) this gives the coefficient vector of $\hat{Q}_j \hat{v} = \hat{P}_j \hat{v} - \hat{P}_{j-1} \hat{v}$ in the standard basis of $\hat{V}$ if we express the $\hat{N}_{j,i}$ as the corresponding finite linear combination of $\hat{\psi}_\Delta$ and recall that due to $\hat{P}_j \hat{v} = \hat{P}_{j-1} \hat{v}$ outside $\hat{\Omega}_j$ only terms with $\mathrm{supp}\, \hat{\mathrm{N}}_{\mathrm{j,i}} \subset \hat{\Omega}_{\mathrm{j}}$ and $\hat{N}_{j-1,i} \not\equiv 0$ on $\hat{\Omega}_j$ are to be considered (here, the special definition of $\hat{\Omega}_j^*$ is used). Concatenating all these vectors results in the coefficients of $E\hat{v}$ w.r.t. $\mathcal{R}$ (coefficients corresponding to functions $N_{j,i} \in \mathcal{R}$ with $N_{j,i}|_\Gamma \equiv 0$ are set to 0). More details can be found in an extended version of this note at `http://cm.bell-labs.com/who/poswald`.

CONCLUSION

We have presented an abstract framework for deriving multilevel preconditioners and extension operators, such as needed for interface problems and other Schur complement problems arising in non-overlapping domain decomposition methods for symmetric elliptic boundary value problems. The basic assumption is the availibility of a suitable multilevel decomposition for the global variational problem. Details are given for linear finite element discretizations on partitions arising from nested local refinement.

REFERENCES

[DW90] Dryja M. and Widlund O. (1990) Towards a unified theory of domain decomposition for elliptic problems. In Chan T., Glowinski R., Periaux J., and Widlund O. (eds) *3rd International Symposium on Domain Decomposition Methods*. SIAM, Philadelphia.

[Kho98a] Khoromskij B. N. (June 1998) Direct and mixed Schur complement methods for the Stokes equation. Technical Report 98/5, ICA3, University Stuttgart.

[Kho98b] Khoromskij B. N. (June 1998) Lectures on multilevel Schur complement methods for elliptic differential equations. Technical Report 98/4, ICA3, University Stuttgart.

[KP98] Khoromskij B. N. and Prössdorf S. (1998) Fast computations with harmonic Poincaré-Steklov operators on nested refined meshes. *Advances in Comp. Math.* 8: 1–25.

[Osw94a] Oswald P. (1994) *Multilevel Finite Element Approximation: Theory & Applications*. Teubner Skripten zur Numerik. Teubner, Stuttgart.

[Osw94b] Oswald P. (1994) Stable subspace splittings for Sobolev spaces and some domain decomposition algorithms. In D.Keyes and J.Xu (eds) *7th International Symposium on Domain Decomposition Methods*, volume 180 of *Contemporary Mathematics*, pages 87–98. AMS, Providence.

[SBG96] Smith B. F., Bjørstad P. E., and Gropp W. D. (1996) *Domain Decomposition - Parallel Multilevel Methods for Elliptic Partial Differential Equations*. Cambridge Univ. Press, Cambridge.

[The98] Thess M. (1998) *Parallel multilevel preconditioners for problems of thin shells*. PhD thesis, TU Chemnitz.

[Xu92] Xu J. (1992) Iterative methods by space decomposition and subspace correction. *SIAM Review* 34: 581–613.

[XZ99] Xu J. and Zou J. (1999) Some non-overlapping domain decomposition methods. *SIAM Review* to appear.

12

Robust Subspace Correction Methods for Thin Elastic Shells

E. E. OVTCHINNIKOV & L. S. XANTHIS[1]

Introduction

In this paper we address the issue of developing subspace correction methods for thin elastic shells which are *robust* with respect to the thickness, i.e. they exhibit convergence which is not adversely affected by the small thickness of the shell. This kind of robustness has been tackled so far only in some specific cases, such as zero curvature and linear approximation in the transverse variable (see e.g. [Bre96, Kla96, AFW97, BS98]). In this paper we invoke the so-called *Korn's type inequality in subspaces* [OX96, OX97b, OX97a, OX98a] to present a technique for developing robust iterative subspace correction methods applicable to thin elastic shells of arbitrary geometry discretized by finite elements of any order.

Boundary value problem of elastostatics.

Consider a boundary value problem of linear elastostatics in variational form:

$$\mathbf{u} \in \mathbf{V}: \quad E(\mathbf{u}, \mathbf{v}) = F(\mathbf{v}) \quad \forall \mathbf{v} \in \mathbf{V} \tag{1}$$

where the space of admissible displacements $\mathbf{V}$ is given by

$$\mathbf{V} = \{\mathbf{v} \in H^1(\Omega)^3 : \mathbf{v} = 0 \text{ on } \Gamma \subset \partial\Omega\}, \quad \Omega \subset \mathbf{R}^3$$

1 Centre for Techno-Mathematics & Scientific Computing Laboratory, University of Westminster (Harrow Campus), London HA1 3TP, U.K. email: eeo@wmin.ac.uk, lsx@wmin.ac.uk

Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

and the elastic energy form E is given by

$$E(\mathbf{u},\mathbf{v}) = \int_\Omega \varepsilon(\mathbf{u})^T H\, \varepsilon(\mathbf{v})\, d\Omega,$$

where the vector ε is formed by the six components of the strain tensor:

$$\varepsilon(\mathbf{v}) = (\varepsilon_{11}, \varepsilon_{22}, \varepsilon_{33}, \varepsilon_{12}, \varepsilon_{23}, \varepsilon_{31})^T, \quad \varepsilon_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)$$

and H is symmetric positive definite matrix whose elements are piecewise smooth functions. The right-hand side of (1) is given by

$$F(\mathbf{v}) = \int_\Omega \mathbf{f}\cdot\mathbf{v}\, d\Omega + \int_{\Gamma_0} \mathbf{T}\cdot\mathbf{v}\, d\Gamma_0, \quad \Gamma_0 = \partial\Omega \setminus \Gamma$$

Here and throughout the paper we use boldface for 3-vectors, i.e. $\mathbf{u}= (u_1, u_2, u_3)$ etc.

Thin shell.

The domain of a shell is defined as an image $\Omega = \Phi(\hat{\Omega})$ of a plate-like *reference* domain $\hat{\Omega}$:

$$\hat{\Omega} = \{\ (\xi_1, \xi_2, \eta) : (\xi_1, \xi_2) \in \Omega_\xi \subset R^2, -\frac{t}{2} < \eta < \frac{t}{2}\ \}$$

We assume that the mapping $(x, y, z) = \Phi(\xi_1, \xi_2, \eta)$ is

- continuous and piecewise-C^2
- non-degenerate on $\overline{\hat{\Omega}}$
- linear in η

Numerical solution.

Let us approximate the space of admissible displacements $\mathbf{V}$ by e.g. the space of piecewise-polynomial functions:

$$\mathbf{V} \approx \mathbf{V}^{pq} = \{\ \mathbf{v} \in \mathbf{V} : \mathbf{v}\circ\Phi \in \hat{\mathbf{V}}^{pq}\ \}$$

where $\hat{\mathbf{V}}^{pq}$ is the space of piecewise-polynomial functions $\hat{\mathbf{v}}(\xi_1, \xi_2, \eta)$ which are of degree p in ξ_1, ξ_2 and of degree q in η. By Galerkin projection onto $\mathbf{V}^{pq}$ we obtain the following discretization of (1):

$$\mathbf{u} \approx \mathbf{u}^{pq} \in \mathbf{V}^{pq} : \quad E(\mathbf{u}^{pq}, \mathbf{v}^{pq}) = F(\mathbf{v}^{pq}) \quad \forall \mathbf{v}^{pq} \in \mathbf{V}^{pq} \tag{2}$$

In this paper we consider the solution of the problem (2) by iterative subspace correction methods of additive type (see e.g. [Xu92]).

Additive subspace correction (ASC).

In this section we omit for simplicity the superscripts p and q.

Assume that the space $\mathbf{V}$ is decomposed into the sum of subspaces:

$$\mathbf{V} = \mathbf{V}_1 + \ldots + \mathbf{V}_n$$

Following the subspace decomposition technique (see e.g. [Xu92, Osw94]) we can transform the problem (2) into an abstract operator equation

$$T\mathbf{u} = \mathbf{g} \tag{3}$$

where

$$T = T_1 + \ldots + T_n, \quad \mathbf{g} = \mathbf{g}_1 + \ldots + \mathbf{g}_n$$

$$T_i : \mathbf{V} \to \mathbf{V}_i : \quad E(T_i\mathbf{v}, \mathbf{w}) = E(\mathbf{v}, \mathbf{w}) \quad \forall \mathbf{w} \in \mathbf{V}_i$$

$$\mathbf{g}_i \in \mathbf{V}_i : \quad E(\mathbf{g}_i, \mathbf{w}) = F(\mathbf{w}) \quad \forall \mathbf{w} \in \mathbf{V}_i$$

The equation (3) is then solved using a suitable iterative algorithm:

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \tau_k(T\mathbf{u}^k - \mathbf{g})$$

Each iteration involves the calculation of $T\mathbf{u}^k$ which, according to the definition of the operator T, amounts to solving problems in subspaces. Thus, the problem in the whole space $\mathbf{V}$ is reduced to solving problems in the subspaces $\mathbf{V}_i$.

Subspace splitting and convergence of ASC.

There are several approaches to analysing the convergence of the above iterative scheme of which we opt for the one suggested by Oswald which is based on the concept of a stable subspace splitting [Osw94].

If a norm $\|\cdot\|_{(i)}$ is introduced in each subspace $\mathbf{V}_i$ then the set of pairs $\{\mathbf{V}_i, \|\cdot\|_{(i)}\}_{i=1,n}$ is called a splitting of the pair $\{\mathbf{V}, \|\cdot\|\}$, which is formally expressed as

$$\{\mathbf{V}, \|\cdot\|\} = \sum_{i=1}^{n} \{\mathbf{V}_i, \|\cdot\|_{(i)}\} \tag{4}$$

The so-called additive Schwarz norm $[\cdot]$ for (4) is defined as

$$[\mathbf{v}] = \inf \left\{ \left(\sum_{i=1}^{n} \|\mathbf{v}_i\|_{(i)}^2 \right)^{\frac{1}{2}} : \ \mathbf{v}_i \in \mathbf{V}_i, \ \sum_{i=1}^{n} \mathbf{v}_i = \mathbf{v} \right\}$$

and the following two values are called characteristic numbers of (4):

$$a_0 = \inf_{\mathrm{v} \in \mathrm{V}} \frac{[\mathbf{v}]}{\|\mathbf{v}\|}, \quad a^0 = \sup_{\mathrm{v} \in \mathrm{V}} \frac{[\mathbf{v}]}{\|\mathbf{v}\|}$$

and the ratio a^0/a_0 is called the condition number of this splitting.

If the energy norm is taken for the norms in $\mathbf{V}$ and $\mathbf{V}_i$, i.e. $\|\cdot\| = \|\cdot\|_{(i)} = \|\cdot\|_E \equiv E(\cdot,\cdot)^{1/2}$ then the condition number of the operator T is equal to the condition number of the corresponding splitting (see e.g. [Osw94]).

Stable splitting and robust convergence.

The splitting of the space $\mathbf{V}^{pq}$ is called stable if its condition number is uniformly bound with respect to the number of subspaces, the discretization parameters p and q and the parameters of the problem we are concerned with (in the case in hand – thickness of the shell).

If the splitting is stable, then the condition number of the operator T is also uniformly bound with respect to the parameters mentioned above, and accordingly the convergence of the subspace correction method is robust with respect to those parameters.

A well-known example of the technique for developing subspace correction methods which are robust w.r.t discretization parameters is the overlapping domain decomposition technique. A less known example is the *effective dimensional reduction algorithm* (EDRA) [OX95]. This algorithm is based on the following *semi-discretization* of the space $\mathbf{V}$:

$$\mathbf{V}^q = \{\ \mathbf{v} \in \mathbf{V} : \mathbf{v} \circ \Phi = \sum_{i=0}^{q} \mathbf{v}_i(\xi_1, \xi_2) Q_i(2\eta/t)\ \}$$

where Q_i are the eigenfunction of the eigenvalue problem

$$\int_{-1}^{1} \frac{dQ_i}{d\eta}\frac{dQ}{d\eta}\, d\eta = \lambda_i \int_{-1}^{1} Q_i Q\, d\eta \quad \forall Q: \text{ polyn. of deg. } q$$

The semi-discretized problem is solved using the additive subspace correction method with the subspaces given by

$$\mathbf{V}^q_{3i+j} = \{\ (v_1, v_2, v_3) \in \mathbf{V}^q : v_j \circ \Phi = w(\xi_1, \xi_2) Q_i(2\eta/t),\ v_k = 0,\ k \neq j\ \},$$
$$j = 1, 2, 3;\ i = 0, \ldots, q$$

The EDRA is robust with respect to to the semi-discretization parameter q, but, just as the overlapping DDM, it is not robust with respect to the thickness *in general*, i.e. unless special decompositions are used, such as those presented later on in this paper.

Classical Korn's inequality: diagnosis

The convergence of iterative methods for thin elastic structures deteriorates because of the poor coerciveness of the elastic energy form E as diagnosed by the classical Korn's inequality (see e.g. [Fic72, Hor95]):

$$Ct^2 \|\mathbf{v}\|_1^2 \leq E(\mathbf{v}, \mathbf{v}) \quad \forall \mathbf{v} \in \mathbf{V}$$

where $\|\cdot\|_1$ is the Sobolev norm, i.e.

$$\|\mathbf{v}\|_1^2 = \int_\Omega (|\mathbf{v}|^2 + \sum_{i=1}^{3} \left|\frac{\partial \mathbf{v}}{\partial x_i}\right|^2) d\Omega$$

Korn's inequality in subspaces: treatment

In the *Korn's type inequality in subspaces* recently introduced by the authors (see [OX96, OX97b, OX97a, OX98a]) the constant in front of the Sobolev norm is expressed in terms of the angle between a given subspace and the subspace $\mathbf{V}_0$ of the displacements which are linear in the transverse direction:

$$\forall \mathbf{U} \subset \mathbf{V} \quad C\beta^2_{\mathrm{U},\mathrm{V}_0} \|\mathbf{v}\|_1^2 \le E(\mathbf{v},\mathbf{v}) \quad \forall \mathbf{v} \in \mathbf{U} \tag{5}$$

where

$$\mathbf{V}_0 = \{\ \mathbf{v} \in \mathbf{V} : \mathbf{v} \circ \Phi = \mathbf{v}_0(\xi_1,\xi_2) + \eta \mathbf{v}_1(\xi_1,\xi_2)\ \}$$

and

$$\beta_{\mathrm{U},\mathrm{W}} = \min_{\mathrm{u}\in\mathrm{U}} \min_{\mathrm{w}\in\mathrm{W}} \frac{\|\mathbf{u}-\mathbf{w}\|_1}{\|\mathbf{u}\|_1}$$

This paves the way to overcome the difficulty: taking proper care of the displacements which are linear in the transverse direction.

More economic alternatives for $\mathbf{V}_0$

Assume for simplicity $\Phi = I$ and $\mathbf{V} = H_0^1(\Omega)^3$. The subspace $\mathbf{V}_0$ in (5) can be reduced to

$$\mathbf{V}_0 = \{\ (\eta u(\xi_1,\xi_2), \eta v(\xi_1,\xi_2), w(\xi_1,\xi_2)),\ u,v,w \in H_0^1(\Omega_\xi)\ \} \tag{6}$$

or to

$$\mathbf{V}_0 = \{\ (\eta w(\xi_1,\xi_2), \eta w(\xi_1,\xi_2), -w(\xi_1,\xi_2)),\ w \in H_0^2(\Omega_\xi)\ \}$$

Robust ASC for thin shells.

First approach.

Theorem 1 *[OX98b] Denote $\mathbf{V}_0^p = \mathbf{V}_0 \cap \mathbf{V}^{pq}$. If the constant a_0 for*

$$\{\mathbf{V}^{pq}, \|\cdot\|_1\} = \sum_{i=1}^{n} \{\mathbf{V}_i^{pq}, \|\cdot\|_1\}$$

is independent of n, p, q, t and

$$E(\sum_{i=1}^{n} \mathbf{v}_i) \leq C \sum_{i=1}^{n} E(\mathbf{v}_i)$$

then

$$\{\mathbf{V}^{pq}, \|\cdot\|_E\} = \{\mathbf{V}_0^p, \|\cdot\|_E\} + \sum_{i=1}^{n} \{\mathbf{V}_i^{pq}, \|\cdot\|_E\}$$

is stable and, hence, the corresponding ASC is robust with respect to n, p, q and t.

The conditions of this theorem are satisfied by most overlapping DDMs with $\mathbf{V}_0^p$ added to the usual subspace decomposition.

Second approach.

Theorem 2 *[OX98b] If the space $\mathbf{V}^{pq}$ can be decoupled into the sum*

$$\mathbf{V}^{pq} = \mathbf{V}_0^p + \mathbf{V}_0^{pq} : \quad \beta(\mathbf{V}_0^p, \mathbf{V}_0^{pq}) \geq \beta_0 > 0$$

and the splitting

$$\{\mathbf{V}_0^{pq}, \|\cdot\|_1\} = \sum_{i=1}^{n} \{\mathbf{V}_i^{pq}, \|\cdot\|_1\}$$

is stable then

$$\{\mathbf{V}^{pq}, \|\cdot\|_E\} = \{\mathbf{V}_0^p, \|\cdot\|_E\} + \sum_{i=1}^{n} \{\mathbf{V}_i^{pq}, \|\cdot\|_E\}$$

is stable and, hence, the corresponding ASC is robust with respect to n, p, q and t.

The conditions of this theorem are satisfied by EDRA with $\mathbf{V}_1, \ldots, \mathbf{V}_6$ replaced by $\mathbf{V}_0^p$ or, in the case of a plate, with $\mathbf{V}_3, \mathbf{V}_4, \mathbf{V}_5$ replaced by the subspace $\mathbf{V}_0^p$ given by (6).

Numerical example.

Here we present numerical results for the shell shown in Fig. 1. The elastic moduli are: $E = 4 \cdot 10^{10}$ and $\nu = 0.2$ for the outer layer and $E = 10^{10}$ and $\nu = 0.4$ for the inner. The left horizontal edge is clamped (i.e $\mathbf{u} = 0$) and the right edge is simply supported (i.e. $u_3 = 0$) and subject to a unit traction force in the direction of the x-axis (i.e. $T_1 = 1$ and $T_2 = 0$). The front edge is simply supported (i.e. $u_2 = 0$). The discretised problem is solved using the modified EDRA described briefly in the previous section.

Tables 1 and 2 show the number of the iterations of the subspace correction method executed until the relative residual falls below 10^{-4} for $t/R = 0.01$ and various values of

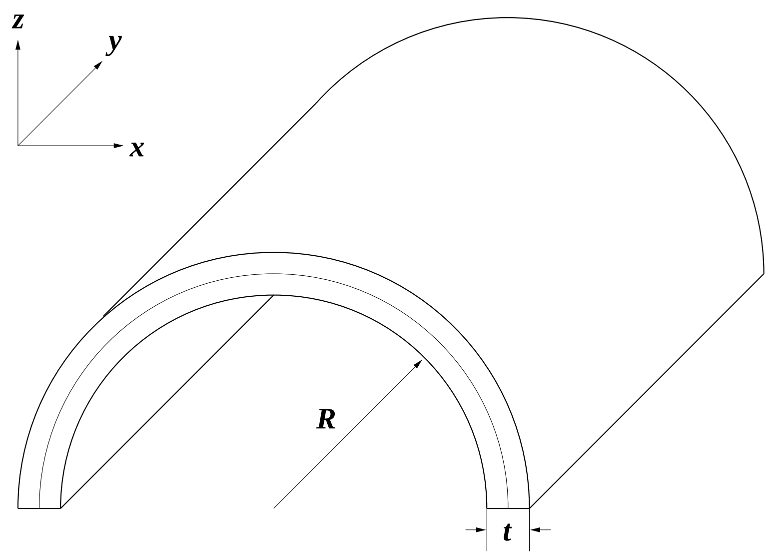


Figure 1 Two-layered thin elastic shell.

Table 1 Uniform convergence with respect to p ($q = 20$).

Polynomial degree p	1	2	3	4
Number of iterations	13	16	16	16

p and q. Table 3 shows the number of the iterations of the subspace correction method under the same stopping criterion for $p = 2$, $q = 20$ and various values of t/R. These results clearly demonstrate that the convergence of the subspace correction method based on the splitting given in the previous section is *uniform* with respect to the thickness parameter t and the discretisation parameters p and q. Thus, the subspace correction method presented here is robust with respect to all these parameters.

REFERENCES

[AFW97] Arnold D. N., Falk R. S., and Winther R. (1997) Preconditioning discrete approximations of the Reissner-Mindlin plate model. *Math. Model. Numer. Analysis* 31: 517–557.

[Bre96] Brenner S. (1996) Multigrid methods for parameter dependent problems. *Math. Model. Numer. Analysis* 30: 265–297.

[BS98] Bramble J. H. and Sun T. (1998) A negative-norm least squares method for Reissner-Mindlin plates. *Math. comp.* 67: 901–916.

[Fic72] Fichera G. (1972) Existence theorems in elasticity. In Flügge S. and Truesdell C. (eds) *Handbuch der Physik, Vol 6a/2*, pages 347–389. Springer Verlag.

[Hor95] Horgan C. O. (1995) Korn's inequalities and their applications in continuum mechanics. *SIAM Review* 37(4): 491–511.

Table 2 Uniform convergence with respect to q ($p = 2$).

Polynomial degree q	15	20	25	30
Number of iterations	16	16	16	16

Table 3 Uniform convergence with respect to the thickness parameter d

Thickness ratio t/R	0.08	0.04	0.02	0.01
Number of iterations	18	17	16	16

[Kla96] Klawonn A. (1996) Preconditioners for indefinite problems. Technical Report 8/96-N, Universität Münster.

[Osw94] Oswald P. (1994) *Multilevel finite element approximation*. Teubner, Stuttgart.

[OX95] Ovtchinnikov E. E. and Xanthis L. S. (1995) Effective dimensional reduction for elliptic problems. *C. R. Acad. Sci. Paris, Série I* 320: 879–884.

[OX96] Ovtchinnikov E. E. and Xanthis L. S. (1996) A new Korn's type inequality for thin domains and its application to iterative methods. *Comput. Meths. Appl. Mech. Engrg.* 138: 299–315.

[OX97a] Ovtchinnikov E. E. and Xanthis L. S. (1997) The Korn's type inequality in subspaces and thin elastic structures. *Proc. R. Soc. Lond. A* 453: 2003–2016.

[OX97b] Ovtchinnikov E. E. and Xanthis L. S. (1997) A new Korn's type inequality for thin elastic structures. *C. R. Acad. Sci. Paris, Série I* 324: 577–583.

[OX98a] Ovtchinnikov E. E. and Xanthis L. S. (1998) The discrete Korn's type inequality in subspaces and thin elastic structures. *Comput. Meths. Appl. Mech. Engrg.* 160: 23–37.

[OX98b] Ovtchinnikov E. E. and Xanthis L. S. (1998) Iterative subspace correction methods for thin elastic structures and Korn's type inequality in subspaces. *Proc. R. Soc. Lond. A* 454: 2023–2039.

[Xu92] Xu J. (1992) Iterative methods by subspace decomposition and subspace correction. *SIAM Rev.* 34: 581–613.

13

V-cycle Multigrid Methods for Wilson Nonconforming Element

ZHONG-CI SHI[1], XUEJUN XU[2]

INTRODUCTION

In this paper, we present some optimal V-cycle multigrid algorithms for Wilson's element. Two types of multigrid methods are considered.

The first one uses the bilinear conforming element space as the coarse grid correction space. Similar to [10], here we also consider the effect of numerical integrations. Both the standard and a weak quadrature formula are discussed. As in [2], the quadrature formula on the finest level is also used for all coarse levels. We show that the V-cycle multigrid method for Wilson's element with the standard or the weak quadrature formula needs only one smoothing iteration on each level. Moreover, the result holds for problems without full elliptic regularity (e.g., an L-shaped domain or a domain with a crack boundary). We mention here that in [2] only the standard quadrature formula for the conforming case is studied.

The second type of V-cycle multigrid method for Wilson's element is the so-called nonconforming multigrid, i.e., the both fine mesh and coarse mesh space are the same nonconforming finite element space. Little work has been done in this direction. Many existing nonconforming multigrid methods have been shown to converge only for W-cycle with sufficiently many smoothing steps on each level (cf. [4] for details). One interesting exception is the so-called rotated Q1 nonconforming finite element; see [7], where it is shown that the W-cycle multigrid with any number of smoothing steps converges for Laplace equation. In [6], a V-cycle nonconforming multigrid is

1 Institute of Computational Mathematics, Chinese Academy of Sciences, P.O.Box 2719, Beijing 100080, China Email: shi@lsec.cc.ac.cn
2 Address same as above.
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

proposed for the rotated Q1 element and P1 nonconforming element through a modified quadratic form on the coarse mesh space. However, the convergence rate is dependent on the mesh level J. Meanwhile, the numerical experiment [6] shows that the nonconforming multigrid still converges without changing the quadratic form. We do hope that a good choice of an intergrid operator may result in a convergent V-cycle nonconforming multigrid method even without changing the quadratic form on the coarse meshes. It has been done with Wilson's element. Here we define a simple intergrid operator with good stability properties for Wilson's element. Using this operator, an optimal V-cycle nonconforming multigrid with one smoothing step on each level converges for Laplace's equation, and the convergence rate is independent of the mesh level J. It seems to be the first optimal V-cycle nonconforming multigrid method. Similar to the rotated Q1 nonconforming element, the W-cycle nonconforming multigrid, with any number of smoothing steps, is also shown to be convergent for Wilson's element.

A MODEL PROBLEM AND WILSON'S ELEMENT

We consider the second order problem

$$\begin{cases} -\sum_{i,j=1}^{2} \frac{\partial}{\partial x_i}(a_{ij}\frac{\partial u}{\partial x_j}) & = f \quad in \quad \Omega, \\ \hfill u & = 0 \quad on \quad \partial\Omega, \end{cases} \tag{1}$$

where $\Omega \subset R^2$ is a bounded polygonal domain, $f \in L^2(\Omega)$, and all function a_{ij} and f are smooth enough. We assume that the differential operator is uniformly elliptic, i.e., there exist positive numbers c, C such that

$$c\xi^t\xi \le \sum_{i,j=1}^{2} a_{ij}\xi_i\xi_j \le C\xi^t\xi \quad \forall (x,y) \in \Omega, \xi \in R^2. \tag{2}$$

If Ω is a convex polygon, then the solution u satisfies the so-called full regularity assumption

$$||u||_2 \le C||f||_0. \tag{3}$$

However, the results of next section are independent of (3).

Here and throughout this paper, c and C (with or without subscript) denote generic positive constants, independent of the mesh parameter J and h_J which will be defined below.

The variational form of (1) is to find $u \in H_0^1(\Omega)$ such that

$$a(u,v) = (f,v) \quad \forall v \in H_0^1(\Omega), \tag{4}$$

where the bilinear form

$$a(u,v) = \int_\Omega \sum_{i,j=1}^{2} a_{ij}\frac{\partial u}{\partial x_j}\frac{\partial v}{\partial x_i}dx \quad \forall u, v \in H^1(\Omega).$$

Let h_0 and $\Gamma_{h_0} := \Gamma_0$ be given, where Γ_0 is a partition of Ω into rectangles and h_0 is the maximum diameter of all the rectangles. For each integer, $1 \leq k \leq J$, let $h_k = 2^{-k} h_0$ and $\Gamma_{h_k} := \Gamma_k$ be constructed by connecting the midpoints of two opposite edges of each rectangle in Γ_{k-1}, $k = 1, ..., J$. The finest grid is Γ_J.

On each level k, we define Wilson's nonconforming element space as follows:

First, on the reference rectangle $\hat{K} = [-1,1] \times [-1,1]$, the shape functions of Wilson's element is a quadratic polynomial $\hat{p}$, defined by the values of $\hat{p}$ at four vertices of $\hat{K}$ and the mean values of $\frac{\partial^2 \hat{p}}{\partial \xi^2}$ and $\frac{\partial^2 \hat{p}}{\partial \eta^2}$ on $\hat{K}$. We have

$$
\begin{aligned}
\hat{p} \;&= \frac{(1+\xi)(1+\eta)}{4} p_1 + \frac{(1-\xi)(1+\eta)}{4} p_2 + \frac{(1-\xi)(1-\eta)}{4} p_3 + \frac{(1+\xi)(1-\eta)}{4} p_4 \\
&+ \frac{1}{2}(\xi^2 - 1)\frac{\partial^2 \hat{p}}{\partial \xi^2} + \frac{1}{2}(\eta^2 - 1)\frac{\partial^2 \hat{p}}{\partial \eta^2},
\end{aligned}
\tag{5}
$$

where p_i is the value of $\hat{p}$ at the vertex $\hat{A}_i$.

Then, for each rectangle $\tau_K \in \Gamma_k$, using an affine transformation, we can define a Wilson's element. We denote Wilson's finite element space on Γ_k by V_k. It is known that this element is not continuous on interelement boundaries, so it is nonconforming for a second order problem.

In the following, we also need the bilinear element space on Γ_k, which is denoted by U_k. we note that U_k is a conforming element.

Define the discrete bilinear form over the space V_k by

$$
a_k^E(u,v) = \sum_{\tau_K \in \Gamma_k} \int_{\tau_K} \sum_{i,j=1}^{2} a_{ij} \frac{\partial u}{\partial x_j} \frac{\partial v}{\partial x_i} \quad \forall u, v \in V_k, \tag{6}
$$

where superscript E indicates that all integrals are computed exactly.

Next we define a discrete semi-norm by

$$
|u|_{i,k}^2 = \sum_{\tau_k \in \Gamma_k} |u|_{i,\tau_k}^2 \quad \forall u \in V_k, \quad i = 1, 2. \tag{7}
$$

It is known that this semi-norm is also a norm over V_k. Then, the Wilson's nonconforming approximation of (1) on Γ_J is to find $u_J^* \in V_J$ such that

$$
a_J^E(u_J^*, v_J) = (f, v_J) \quad \forall v_J \in V_J. \tag{8}
$$

It is known that there exists a unique solution of (8); see [5].

We now approximate the integral in $a_J^E(\cdot,\cdot)$ by a quadrature scheme Q_K over each $K \in \Gamma_J$. We first consider the reference rectangle $\hat{K}$ and approximate the integral $\int_{\hat{K}} \hat{\phi}(\hat{x}) d\hat{x}$ as follows:

$$
\int_{\hat{K}} \hat{\phi}(\hat{x}) d\hat{x} \approx \sum_{l=1}^{L} w_l \hat{\phi}(b_l),
$$

where w_l are positive weights and $b_l \in \hat{K}$ are quadrature points. We then define the quadrature scheme on $K \in \Gamma_J$ by

$$
\int_K \phi(x) dx \approx \sum_{l=1}^{L} w_{K,l} \phi(b_{K,l}) \equiv Q_K[\phi], \tag{9}
$$

where $\phi(x) = \hat{\phi}(\hat{x})$, the weights $w_{K,l}$ and quadrature points $b_{K,l}$ are defined in terms of the w_l and b_l by means of the affine mapping from $\hat{K}$ onto K that takes each x in K into $\hat{x}$ in $\hat{K}$.

The quadrature error functional is

$$E_K(\phi) = \int_K \phi(x)dx - Q_K[\phi]. \tag{10}$$

Using the quadrature scheme, we approximate $a_J^E(\cdot,\cdot)$, $(f,\cdot)$ by $a_J(\cdot,\cdot)$, $(f,\cdot)_J$ as follows

$$a_J(u,v) = \sum_{K\in\Gamma_J}\sum_{i,j=1}^{2} Q_K[a_{ij}\frac{\partial u}{\partial x_i}\frac{\partial v}{\partial x_j}] \tag{11}$$

and

$$(f,v)_J = \sum_{K\in\Gamma_J} Q_K[fv]. \tag{12}$$

Now the Wilson's and the bilinear element approximation u_J and w_J with the quadrature scheme Q_K are defined, respectively by

$$a_J(u_J,v) = (f,v)_J \quad \forall v\in V_J \tag{13}$$

and

$$a_J(w_J,v) = (f,v)_J \quad \forall v\in U_J. \tag{14}$$

Following [5], a standard quadrature scheme Q_K satisfies the following two assumptions: (**H1**) The quadrature error funtional

$$E_K(\phi) = 0 \quad \forall\phi\in P_2(K),$$

where $P_2(K)$ is the quadratic polynomial space on K.

(**H2**)

$$\textit{All} \quad \textit{weights} \quad w_{K,l} > 0 \quad \textit{and} \quad \frac{1}{4}\sum_l w_{K,l} = 1.$$

Meanwhile, instead of (H1), there exist other weak assumptions, like

(**H3**) The union of all quadrature points b_l on $\hat{K}$ contains a $P_1(\hat{K})$ unisolvent subset.

(**H4**) The quadrature scheme Q_K satisfies:

$$E_K(\phi) = 0 \quad \forall\phi\in Q_1(K),$$

where $Q_1(K)$ is the bilinear polynomial space on K.

Here are two examples.

Scheme 1 (the standard formula with (H1) and (H2).)

$$\int_{\hat{K}} \hat{\phi}(\hat{x})d\hat{x} \approx \sum_{l=1}^{4}\hat{\phi}(B_l),$$

where $B_1 = (\frac{1}{\sqrt{3}},\frac{1}{\sqrt{3}})$, $B_2 = (\frac{1}{\sqrt{3}},-\frac{1}{\sqrt{3}})$, $B_3 = (-\frac{1}{\sqrt{3}},\frac{1}{\sqrt{3}})$, $B_4 = (-\frac{1}{\sqrt{3}},-\frac{1}{\sqrt{3}})$.

Scheme 2 (the weak formula with (H2), (H3) and (H4).)

$$\int_{\hat{K}} \hat{\phi}(\hat{x})d\hat{x} \approx \sum_{l=1}^{4}\hat{\phi}(A_l),$$

where $A_1 = (1,1)$, $A_2 = (1,-1)$, $A_3 = (-1,1)$, $A_4 = (-1,-1)$.

V-CYCLE MULTIGRID WITH NUMERICAL INTEGRATION FOR WILSON'S ELEMENT

In this section, we describe a V-cycle multigrid method with numerical integrations for Wilson's nonconforming element. An optimal V-cycle multigrid with only one smoothing step on each level is derived. In the following, we make use of the bilinear conforming element space U_k, $k = 1, ..., J-1$ as the coarse grid correction space.

It is obvious that

$$U_1 \subset U_2 \subset \cdots \subset U_{J-1} \subset V_J.$$

On each level $k = 1, ..., J$, we introduce the operator $A_{U_k} : U_k \to U_k$, $k = 1, ..., J-1$, by

$$(A_{U_k} v, w) = a_J(v, w) \quad \forall v, w \in U_k \tag{15}$$

and $A_J : V_J \to V_J$ by

$$(A_J v, w) = a_J(v, w) \quad \forall v, w \in V_J. \tag{16}$$

We note that we here apply the quadrature formula on the level J to all coarse levels as in [2].

For $k = 1, ..., J-1$, we define the projection operator $P_k : V_J \to U_k$ by

$$a_J(P_k v, w) = a_J(v, w) \quad \forall v \in V_J, w \in U_k, \tag{17}$$

and $P_k^0 : L^2(\Omega) \to U_k$ by

$$(P_k^0 v, w) = (v, w) \quad \forall v \in V_J, w \in U_k. \tag{18}$$

In order to define a V-cycle multigrid algorithm, we must choose on each level k a smoothing operator $R_{U_k} : U_k \to U_k$, $k = 2, ..., J-1$. We set

$$K_k = I - R_{U_k} A_{U_k}$$

and

$$\bar{R}_k = (I - K_k^* K_k) A_k^{-1},$$

where $R_{U_k}^t$ denotes the adjoint of R_{U_k} with respect to $(\cdot, \cdot)$, and $K_k^* = I - R_{U_k}^t A_{U_k}$ denotes the adjoint of K_k with respect to the $a_J(\cdot, \cdot)$. We set $R_{U_1} = A_{U_1}^{-1}$, i.e., we solve (3.1) on the coarsest space. Finally, set $T_k = R_{U_k} P_k^0 A_J$. Then by (17), (18), it is easy to check that

$$P_k^0 A_J = A_{U_k} P_k. \tag{19}$$

So we have $T_k = R_{U_k} A_{U_k} P_k$. Note that $T_1 = P_1$.

Now we define the following multigrid algorithm.

Multigrid Algorithm 1.

If $J = 1$, set $B_1 = A_1^{-1}$ and if $J > 1$, define $B_J g$ for $g \in V_J$ by
(1) Set $x_J = R_J g$.
(2) Define $B_J g = x_J + q$, where $q \in U_{J-1}$ is given by

$$q = N_{J-1} P_{J-1}^0 (g - A_J x_J).$$

Here N_{J-1} is defined as follows: set $N_1 = A_{U_1}^{-1}$. For $2 < k \leq J-1$, assume that N_{k-1} has been defined and define $N_k g$ for $g \in U_k$ by

(i) Set $x_k = R_{U_k} g$.

(ii) Define $N_k g = x_k + q$, where $q \in U_{k-1}$ is given by

$$q = N_{k-1} P_{k-1}^0 (g - A_{U_k} x_k).$$

We can prove that

$$E := I - B_J A_J = (I - T_1)(I - T_2) \cdots (I - T_J). \tag{20}$$

Then, we have

Theorem 1. *For the Wilson's element, there exists a constant $C_R > 1$ dependent of the constants C_0 and M, but independent of h_J and J such that*

$$a_J(Ev, Ev) \leq (1 - \frac{1}{C_R}) a_J(v, v) \quad \forall v \in V_J.$$

It is seen that the contraction factor $1 - \frac{1}{C_R} < 1$, is independent of h_J and J, so we have an optimal multigrid algorithm.

Remark 1. From Multigrid Algorithm 1, we see that our V-cycle multigrid method for Wilson's element with the standard or the weak quadrature formula needs only one smoothing iteration on each level. Moreover, the result holds for problems without full elliptic regularity (e.g., an L-shaped domain or a domain with a crack boundary).

V-CYCLE NONCONFORMING MULTIGRID FOR WILSON'S ELEMENT

In this section, we consider a nonconforming multigrid method, where both the coarse and the fine mesh space use the same nonconforming element space. In the following, we construct a suitable intergrid transfer operator I_k for Wilson's element. Using this operator, an optimal V-cycle nonconforming multigrid method can be obtained. For convenience, we do not consider numerical intergration, and simply write the bilinear form $a_k^E(\cdot,\cdot)$ in (6) as $a_k(\cdot,\cdot)$.

We define the operator $A_k : V_k \to V_k$ on each level k by

$$(A_k v, w) = a_k(v, w) \quad \forall v, w \in V_k, k = 1, ..., J.$$

In the nonconforming multigrid method, the intergrid transfer operator plays an important role. For Wilson's element, let π_{k-1} be the bilinear interpolation operator; it is easy to check that for any $v \in V_{k-1}$,

$$\pi_{k-1} v \in U_{k-1} \subset U_k \subset V_k. \tag{21}$$

So we define the intergrid transfer operator $I_k : V_{k-1} \to V_k$ as follows

$$I_k := \pi_{k-1}. \tag{22}$$

We can prove that the operator $I_k : V_{k-1} \to V_k$ satisfies the inequality

$$a_k(I_k v, I_k v) \leq a_{k-1}(v, v) \quad \forall v \in V_{k-1}. \tag{23}$$

Now as in above section, we define the projection operator $P'_{k-1} : V_k \to V_{k-1}$ and $Q^0_{k-1} : V_k \to V_{k-1}$ by

$$a_{k-1}(P'_{k-1}v, w) = a_k(v, I_k w) \quad \forall v \in V_k, w \in V_{k-1}, k = 1, ..., J, \tag{24}$$

and

$$(Q^0_{k-1}v, w)_{k-1} = (v, I_k w) \quad \forall v \in V_k, w \in V_{k-1}, k = 1, ..., J. \tag{25}$$

Finally, let $R_k : V_k \to V_k$, for $k = 1, ..., J$, be a linear smoothing operator and let R^t_k be the adjoint of R_k with respect to $(\cdot, \cdot)$. Define

$$R^{(l)}_k = \begin{cases} R_k & l \quad odd, \\ R^t_k & l \quad even. \end{cases}$$

A general multigrid operator $B_k : V_k \to V_k$ can be defined recursively as follows:

Multigrid Algorithm 2

Let $1 \le k \le J$ and let p be a positive interger. Set $B_0 = A_0^{-1}$. Assume that B_{k-1} has been defined and define $B_k g$ for $g \in V_k$ by

(1) Set $x_0 = 0$ and $q^0 = 0$.

(2) Define x^l for $l = 1, ..., m(k)$ by

$$x^l = x^{l-1} + R_k^{(l+m(k))}(g - A_k x^{l-1}).$$

(3) Define $y^{m(k)} = x^{m(k)} + I_k q^p$, where q^i for $i = 1, ..., p$ is determined by

$$q^i = q^{i-1} + B_{k-1}(Q^0_{k-1}(g - A_k x^{m(k)}) - A_{k-1}q^{i-1}).$$

(4) Define y^l for $l = m(k) + 1, ..., 2m(k)$ by

$$y^l = y^{l-1} + R_k^{(l+m(k))}(g - A_k y^{l-1}).$$

(5) Set $B_k g = y^{2m(k)}$.

In the Multigrid Algorithm 2, $m(k)$ gives the number of pre- and post-smoothing steps and can vary as a function of k. If $p = 1$, we have a V-cycle method. If $p = 2$, we have a W-cycle method.

The convergence rate for the multigrid algorithm 2 on the kth level is measured by a convergence factor δ_k which satisfies

$$|a_k((I - B_k A_k)v, v)| \le \delta_k a_k(v, v). \tag{26}$$

We then have

Theorem 2. *For the Wilson's nonconforming element, define B_k by $p = 1$ and $m(k) = m$ for all k in the algorithm 2. Then if $a_{ii} = 1$, $a_{ij} = 0$, $i \ne j$, there exists $C > 0$, independent of k, such that*

$$\delta_k \le \delta = \frac{C}{C + m}.$$

It is seen that the convergence factor δ_k in (26) is independent of the level l, we then obtain an optimal V-cycle multigrid method with one smoothing step for the Wilson's nonconforming element for Laplace equation.

Remark 2. We can also obtain an optimal convergence order of the W-cycle multigrid method with any number of smoothing steps for Wilson's element.

REFERENCES

J.H. Bramble, Multigrid Methods, Pitman, (1993).

J.H. Bramble, C.L. Goldstein, J.E. Pasciak, Analysis of V-cycle multigrid algorithms for forms defined by numerical quadrature, SIAM J. Sci. Comput., **15**(1994), 566-576.

J.H. Bramble, J.E. Pasciak, J. Xu, The analysis of multigrid algorithms with nonnested spaces and noninherited quadratic forms, Math. Comp., **56**(1991), 1-34.

S.C. Brenner, An optimal-order multigrid method for P1 nonconforming finite elements, Math. Comp., **52**(1989), 1-15.

P.G. Ciarlet, The Finite Element Method for Elliptic Problem, North-Holland, Amsterdam, (1978)

Z. Chen, On the convergence of Galerkin-multigrid methods for nonconforming finite elements, Technical Report, Southern Methodist University, Texas, 1996.

Z. Chen, P. Oswald, Multigrid and multilevel methods for rotated Q1 nonconforming finite element, Math. Comp., **67**(1998), 695-714.

P. Lesaint, On the convergence of Wilson's nonconforming element for solving the elastic problem, Comput. Methods. Appl. Meth. Engrg., **12**(1976), 1-16.

Z. Shi, A convergence condition for the quadrilateral Wilson element, Numer.Math., **44**(1984), 349-361.

Z. Shi, B. Jiang, Multigrid method for Wilson nonconforming finite element with numerical integration, Domain Decomposition Methods and Related Topics, Symposium of Research Institute for Mathematical Sciences, Kyoto University, 103-121, 1996.

J. Xu, Theory of Multilevel Methods, Ph.D. Thesis, Dept. of Math. Cornell, Univ(1989).

J. Xu, Iterative methods by space decomposition and subspace correction, SIAM Review, **34**(1992), 581-613.

14

The FETI Method for Mortar Finite Elements

DAN STEFANICA[1] & AXEL KLAWONN[2]

INTRODUCTION

The Finite Element Tearing and Interconnecting (FETI) method is a Lagrange multiplier based iterative substructuring method. It was introduced by Farhat and Roux [FR91]; a detailed presentation is given in [FR94], a monograph by the same authors. Originally used to solve second order, self-adjoint elliptic equations, it has later been extended to many other problems, e.g. time-dependent problems, cf. Farhat, Chen, and Mandel [FCM95], plate bending problems, cf. Farhat et al [FCR98, FM98, FMT96], and heterogeneous elasticity problems with composite materials, cf. Farhat and Rixen [RF97, RF99].

In this paper, we present a numerical study of the FETI method for two dimensional self-adjoint elliptic equations, when the underlying finite elements defined on the subdomains of Ω are low order mortar finite elements. We use geometrically nonconforming mortar finite elements of the second generation, for which no continuity conditions are imposed at the vertices.

We have tested three different preconditioners: the Dirichlet preconditioner, which has been used successfully for conforming finite elements (see Farhat, Mandel, and Roux [FMR94]), a block diagonal preconditioner used by Lacour [Lac97a], and

1 Courant Institute of Mathematical Sciences, 251 Mercer Street, New York, N.Y. 10012. Email address: stefanic@cims.nyu.edu URL: http://www.math.nyu.edu/˜stefanic
2 Institut für Numerische und instrumentelle Mathematik, Westfälische Wilhelms–Universität Münster, Einsteinstr. 62, 48149 Münster, Germany. Email address: klawonn@math.uni-muenster.de URL: http://wwwmath.uni-muenster.de/math/u/klawonn/
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

a new preconditioner of Klawonn and Widlund [KW99]. The last preconditioner performs best, and has scalability properties similar to those of the classical FETI algorithm with the Dirichlet preconditioner in the case of a conforming finite element discretization.

For other work on preconditioners for mortar finite element discretizations, see Achdou, Kuznetsov, and Pironneau [AKP95], Kuznetsov [Kuz95], Achdou, Maday, and Widlund [AMW99], and the references therein.

The rest of the paper is structured as follows. In the next section, we describe the mortar finite element method. In section 3, we present the classical FETI method, and in section 4, our new FETI algorithm for mortars together with the preconditioner introduced in Lacour [Lac97a]. In the last section, we present numerical results for all the different preconditioners.

MORTAR FINITE ELEMENTS

The mortar finite element methods were first introduced by Bernardi, Maday, and Patera in [BMP94], and a three dimensional version was developed by Ben Belgacem and Maday in [BBM97]. They are nonconforming finite elements that allow for a nonconforming decomposition of the computational domain and for the optimal coupling of different variational approximations in different subregions. It has been shown that the global error is bounded by the sum of the local best approximation errors on each subregion. We are working with geometrically nonconforming mortars, i.e. we do not require that the intersection of the boundaries of two different subregions is empty, a vertex, or an entire edge.

Using mortars instead of conforming finite elements has some significant advantages. The mesh generation is more flexible and can be made quite simple on the individual subregions. It is also possible to move different parts of the mesh relative to each other, which is useful for time dependent problems and in design optimization. The mortar methods also allow for local refinement of finite element models in only certain subdomains of the computational domain, and they are well suited for parallel computing.

Let us briefly describe the mortar finite element space V^h, restricting our discussion to the two dimensional case. The computational domain Ω is decomposed into a nonoverlapping polygonal partition $\{\Omega_i\}_{i=\overline{1:N}}$. The restriction of the mortar space to any subregion Ω_i is a conforming P_1 or Q_1 finite element space. Across the interface Γ, i.e. the set of points that belong to the boundaries of at least two subregions, pointwise continuity is not required. We partition Γ into a union of nonoverlapping edges of the subregions $\{\Omega_i\}_{i=\overline{1:N}}$, called nonmortars. The edges not chosen to be nonmortars, are called mortars. Note that the mortars also form a partition of the interface. On the two sides of an edge which coincides with a nonmortar, there are two distinct traces of the mortar functions. We only require that the difference of these two traces is L^2-orthogonal to a space of test functions.

More formally, if γ is a nonmortar side, let $V^h(\gamma)$ be the continuous piecewise polynomial space which is the restriction of V^h to γ. Let $\widetilde{\gamma}$ be the union of the parts of the mortars opposite γ. Then $w_h \in V^h$ is a mortar function if its restrictions, $w_\gamma = w_{|_\gamma}$ and $w_{\tilde{\gamma}} = w_{|_{\tilde{\gamma}}}$, satisfy the following L^2-orthogonality condition for every nonmortar side γ:

$$\int_\gamma (w_\gamma - w_{\bar\gamma})\psi ds = 0, \quad \forall \psi \in \Psi^h(\gamma). \tag{1}$$

Here, $\Psi^h(\gamma)$ is the space of the test functions consisting of piecewise linear functions from $V^h(\gamma)$ which are constant in the first and last mesh intervals.

THE FETI METHOD

In this section, we review the original FETI method of Farhat and Roux for elliptic problems discretized by conforming finite elements. We consider P_1 or Q_1 finite elements with a typical mesh size h.

We first partition the finite element mesh along mesh lines into N non–overlapping subdomains $\Omega_i \subset \Omega, i = 1, \ldots, N$, such that the subdomain boundary nodes match across the interface. For all subdomains $\Omega_i, i = 1, \ldots, N$, we construct local stiffness matrices K_i and local load vectors f_i. Denote by K the block–diagonal stiffness matrix with the K_i on the diagonal and by f the vector $[f_1, \ldots, f_N]$. Analogously, we denote by u_i the vector of nodal values on Ω_i and by u the vector $[u_1, \ldots, u_N]$.

Let $B = [B_1, B_2, \ldots, B_N]$ be a matrix which measures the jump of a given vector $u = [u_1, \ldots, u_N]$; $Bu = 0$ means that the values of the degrees of freedom, at nodes which belong to at least two subdomains, coincide.

Let us consider the following minimization problem with constraints:

$$J(u) := \frac{1}{2} u^t K u - f^t u \to \min \quad \text{subject to } Bu = 0. \tag{2}$$

By introducing Lagrange multipliers λ for the constraint $Bu = 0$, we obtain the saddle point problem

$$\begin{array}{lclcl} Ku & + & B^t\lambda & = & f, \\ Bu & & & = & 0. \end{array} \tag{3}$$

Let R be a given matrix that spans the nullspace of K, i.e. range$R$ = ker$K$. The solution of the first equation in (3) exists if and only if $f - B^t\lambda \in$ range$K$. Then, we have

$$u = K^\dagger(f - B^t\lambda) + R\alpha,$$

where $K^\dagger$ is the pseudoinverse of K which provides a solution orthogonal to the nullspace of K and α has to be determined.

To formulate the FETI method, we need the following notation:

$$G := BR, F := BK^\dagger B^t, \; d := BK^\dagger f, \; P := I - G(G^tG)^{-1}G^t, \; e := R^t f.$$

Note that P is an l_2–orthogonal projector onto ker$G^t =: V$. Elimination of the primal variables u in (3) gives

$$BK^\dagger B^t\lambda = BK^\dagger f + BR\alpha, \tag{4}$$

which leads to

$$\begin{array}{rcl} PF\lambda & = & Pd, \\ G^t\lambda & = & e. \end{array}$$

The FETI algorithm is the solution of this dual problem with a preconditioned projected conjugate gradient (PCG) method, where all the increments $\lambda_k - \lambda_0$ are in V.

Given an initial approximation λ_0 with $G^t\lambda_0 = e$, we have to solve

$$PF\lambda = Pd, \quad \lambda \in \lambda_0 + V. \tag{5}$$

One possible preconditioner is of the form

$$M := B \begin{bmatrix} O & O \\ O & S \end{bmatrix} B^t, \tag{6}$$

where S is the Schur complement of K obtained by eliminating the interior degrees of freedom. In the application of M, at each iteration step, N independent Dirichlet problems have to be solved; M is known as the Dirichlet preconditioner. Note that the Schur complement never has to be computed explicitly, since only the action of S on a vector is needed.

It has been shown by Mandel and Tezaur [MT96] that the condition number κ of $PMPF$ satisfies

$$\kappa(PMPF) \leq C \left(1 + \log \frac{H}{h}\right)^3,$$

where C is a positive constant independent of h, H.

This result is similar to estimates for other non–overlapping domain decomposition methods; see Dryja, Smith, and Widlund [DSW94] for iterative substructuring methods, Dryja and Widlund [DW95] for Neumann-Neumann algorithms, and Mandel [Man93] and Mandel and Brezina [MB96] for balancing algorithms.

THE FETI METHOD FOR MORTARS

The FETI algorithm can also be applied when mortar finite elements are considered on Ω. The price we have to pay for the inherent flexibility of the mortar finite elements is related to the fact that the Lagrange multiplier matrix B is more complicated in this case compared to that arising in the classical FETI method with conforming finite elements. This is due to the fact that we no longer have matching nodes across the interface.

Let us briefly describe the construction of the matrix B. From the mortar conditions (1), we see that the interior nodes of the nonmortar sides are not associated with genuine degrees of freedom in the finite element space V^h. Let w be a mortar finite element function and γ a nonmortar. Let w_γ^1 be the vector constructed from the values of w at the interior nodes of γ, and w_γ^2 be the vector constructed from the values of w at all the nodes on the edges on the interface opposite to γ. We assume that the intersection of γ and the support of the corresponding nodal basis functions is not empty. Then w_γ^1 is uniquely determined by w_γ^2, and the mortar conditions (1) for γ can be written in matrix form as

$$M_\gamma w_\gamma^1 - N_\gamma w_\gamma^2 \;=\; 0,$$

where M_γ is a tridiagonal matrix and N_γ is a banded matrix. The matrix B will have one block, B_γ, for each nonmortar side, which consists of the columns of the corresponding matrices M_γ and N_γ and zeros in the other columns. The matrix K is again a block–diagonal matrix $diag_{i=1}^N K_i$, where the local stiffness matrices $K_i, i = 1, \ldots, N$, are obtained from the finite element discretizations on each subdomain Ω_i. As in the case of conforming finite elements, we now have to solve the problem

$$\begin{array}{lcccl} Ku & + & B^t\lambda & = & f, \\ Bu & & & = & 0. \end{array} \tag{7}$$

Note that, in contrast to the conforming finite element case, the K_i can now be build from different discretizations on each subdomain Ω_i. For a more detailed discussion of mortar finite elements with Lagrange multipliers, see Ben Belgacem [BB94] or Braess, Dahmen, and Wieners [BDW98]. To obtain a dual problem as in (5), we can now proceed completely analogously to sect. 14. Due to space limitations, we will not repeat these steps here.

For the dual problem (5) obtained from the mortar system (7), we use the preconditioner

$$\widehat{M} \;:=\; (BB^t)^{-1}B \begin{bmatrix} O & O \\ O & S \end{bmatrix} B^t(BB^t)^{-1}, \tag{8}$$

introduced by Klawonn and Widlund in [KW99]. It will also be shown in [KW99] that $\widehat{M}$ is an almost optimal preconditioner in the conforming finite element case, in the sense that

$$\kappa(P\widehat{M}PF) \;\leq\; C\left(1 + \log\frac{H}{h}\right)^2.$$

Another preconditioner $\overline{M}$ has been suggested by Lacour in [Lac97a, Lac97b]. It can be obtained from $\widehat{M}$ by taking only the block diagonal part of BB^t, instead of the whole matrix BB^t, i.e.

$$\overline{M} := (diagB_\gamma B_\gamma^t)^{-1}B \begin{bmatrix} O & O \\ O & S \end{bmatrix} B^t(diagB_\gamma B_\gamma^t)^{-1}, \tag{9}$$

where $diagB_\gamma B_\gamma^t$ has an entry on the block–diagonal for each nonmortar γ.

NUMERICAL RESULTS

In this section, we first present computational results for the preconditioners discussed in the previous sections in the case of mortar finite elements. As a model problem we have considered the Poisson equation on the unit square $\Omega = [0,1]^2$ with zero Dirichlet boundary conditions.

In the geometrically nonconforming case, cf. sect. 14, the domain Ω has been partitioned into $N \in \{16, 32, 64, 128\}$ rectangles of average diameter H, cf. Figure 14.

We use Q_1 elements of mesh size h. The local meshes on the subdomains do not match across the interface and the mortar conditions are enforced by using Lagrange

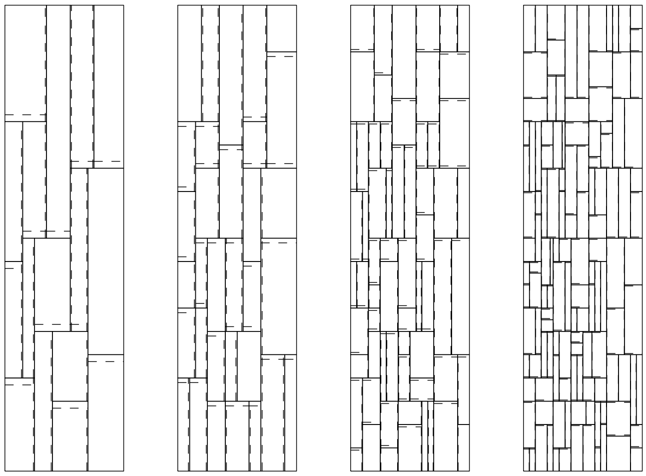


Figure 1 Geometrically non-conforming decompositions of the unit square into $16, 32, 64$, and 128 subdomains.

Table 1 Geometrically non–conforming domain decomposition; Non-matching grids across the interface.

Number of Subdomains	H/h	Precond $\widehat{M}$	Precond $\overline{M}$	Precond M
16/32/64/128	4	11/12/15/16	20/24/32/33	108/233/487/1107
16/32/64/128	8	13/14/16/18	22/25/33/37	290/438/1071/1413
16/32/64/128	16	14/15/18/20	23/27/35/40	406/620/1725/1761
16/32/64/128	32	15/16/20/22	24/27/39/42	486/692/2130/–

multipliers. The stopping criterion used in the PCG routine is the relative reduction of the initial dual residual by 10^{-6}.

In Table 1, we report the iteration counts for the new preconditioner $\widehat{M}$, cf. (8), the preconditioner $\overline{M}$, cf. (9), and the Dirichlet preconditioner M, cf. (6).

As a comparison, we also present iteration counts for the geometrically conforming case using the preconditioners $\widehat{M}$ and M, cf. Table 2. Here, Ω is partitioned in a conforming fashion into 16, 36, 64, and 121 squares.

Table 2 Geometrically conforming domain decomposition; Matching grids across the interface.

Number Subdomains	H/h	Precond $\widehat{M}$	Precond M
16/36/64/121	4	7/9/9/9	18/23/25/25
16/36/64/121	8	9/10/10/10	19/24/25/25
16/36/64/121	16	10/11/11/12	20/26/27/27
16/36/64/121	32	11/13/13/13	21/28/28/28

Note that in order to ensure matching grids across the interface, in the geometrically conforming case the number of nodes per edge of subdomain (i.e. H/h) is the same for every edge, while in the nonconforming case it can be much more general.

In both cases, geometrically conforming and nonconforming, the new preconditioner $\widehat{M}$ yields a lower number of iterations than the other two preconditioners. In the geometrically nonconforming case, the Dirichlet preconditioner M does not seem to yield a numerically scalable method.

In general, $\widehat{M}$ needs some extra work in comparison to the other two preconditioners. In each iteration step, when we multiply a vector by the preconditioner, we have to solve two systems with the matrix BB^t. In the conforming case, BB^t is very close to twice the identity matrix, and therefore little extra work is required. In the mortar case, BB^t is an almost block diagonal matrix, with each block corresponding to a nonmortar γ, and of size equal the number of interior nodes on γ. A more detailed numerical study, which also takes the complexity of the different preconditioners into account is given in [SK98].

ACKNOWLEDGEMENTS

The authors would like to thank Olof Widlund for suggesting this problem, and for many helpful and interesting discussions.

REFERENCES

[AKP95] Achdou Y., Kuznetsov Y. A., and Pironneau O. (1995) Substructuring preconditioners for the Q_1 mortar element method. *Numer. Math.* 71(4): 419–449.

[AMW99] Achdou Y., Maday Y., and Widlund O. B. (1999) Iterative substructuring preconditioners for mortar element methods in two dimensions. *SIAM J. Numer. Anal.* 36: 551–580.

[BB94] Ben Belgacem F. (1994) The mortar element method with Lagrange multipliers. Technical report, Université Paul Sabatier, Toulouse, France. To appear in Numer. Math.

[BBM97] Ben Belgacem F. and Maday Y. (1997) The mortar element method for three dimensional finite elements. *RAIRO Modél. Math. Anal. Numér.* 31: 289–302.

[BDW98] Braess D., Dahmen W., and Wieners C. (March 1998) A multigrid algorithm for the mortar finite element method. Technical Report 153, Institut für Geometrie und Praktische Mathematik, Aachen. To appear in SIAM J. Numer. Anal.

[BMP94] Bernardi C., Maday Y., and Patera A. T. (1994) A new non conforming approach to domain decomposition: The mortar element method. In Brezis H. and Lions J.-L. (eds) *Collège de France Seminar.* Pitman. This paper appeared as a technical report about five years earlier.

[DSW94] Dryja M., Smith B. F., and Widlund O. B. (December 1994) Schwarz analysis of iterative substructuring algorithms for elliptic problems in three dimensions. *SIAM J. Numer. Anal.* 31(6): 1662–1694.

[DW95] Dryja M. and Widlund O. B. (February 1995) Schwarz methods of Neumann-Neumann type for three-dimensional elliptic finite element problems. *Comm. Pure Appl. Math.* 48(2): 121–155.

[FCM95] Farhat C., Chen P.-S., and Mandel J. (1995) A scalable Lagrange multiplier based domain decomposition method for time-dependent problems. *Int. J. Numer. Meth. Eng.* 38: 3831–3853.

[FCR98] Farhat C., Chen P.-S., and Roux F.-X. (1998) The two-level FETI method - part II: Extensions to shell problems, parallel implementation, and performance results. *Comput. Methods Appl. Mech. Eng.* 155: 153–180.

[FM98] Farhat C. and Mandel J. (1998) The two-level FETI method for static and dynamic plate problems - part I: An optimal iterative solver for biharmonic systems. *Comput. Methods Appl. Mech. Eng.* 155: 129–152.

[FMR94] Farhat C., Mandel J., and Roux F.-X. (1994) Optimal convergence properties of the FETI domain decomposition method. *Comput. Methods Appl. Mech. Eng.* 115: 367–388.

[FMT96] Farhat C., Mandel J., and Tezaur R. (1996) A scalable substructuring method by Lagrange multipliers for plate bending problems. Technical report, Center for Computational Mathematics, University of Colorado at Denver. To appear in SIAM J. Num. Anal.

[FR91] Farhat C. and Roux F.-X. (1991) A method of finite element tearing and interconnecting and its parallel solution algorithm. *Internat. J. Numer. Meth. Eng.* 32: 1205–1227.

[FR94] Farhat C. and Roux F.-X. (1994) Implicit parallel processing in structural mechanics. In Oden J. T. (ed) *Computational Mechanics Advances*, volume 2 (1), pages 1–124. North-Holland.

[Kuz95] Kuznetsov Y. A. (1995) Efficient iterative solvers for elliptic finite element problems on nonmatching grids. *Russ. J. Numer. Anal. Math. Modelling* 10: 187–211.

[KW99] Klawonn A. and Widlund O. B. (1999) FETI and Neumann–Neumann iterative substructuring methods: Connections and new results. Technical report, Department of Computer Science, Courant Institute. In preparation.

[Lac97a] Lacour C. (1997) *Analyse et Resolution Numérique de Méthodes de Sous-Domaines Non Conformes pour des Problèmes de Plaques.* PhD thesis, Université Pierre et Marie Curie, Paris.

[Lac97b] Lacour C. (1997) Iterative substructuring preconditioners for the mortar finite element method. In Bjørstad P., Espedal M., and Keyes D. (eds) *Ninth International Conference of Domain Decomposition Methods.* URL = `http://www.ddm.org/DD9/`.

[Man93] Mandel J. (1993) Balancing domain decomposition. *Comm. Numer. Meth. Engrg.* 9: 233–241.

[MB96] Mandel J. and Brezina M. (1996) Balancing domain decomposition for problems with large jumps in coefficients. *Math. Comp.* 65: 1387–1401.

[MT96] Mandel J. and Tezaur R. (1996) Convergence of a substructuring method with Lagrange multipliers. *Numer. Math.* 73: 473–487.

[RF97] Rixen D. and Farhat C. (1997) Preconditioning the FETI method for problems with intra- and inter-subdomain coefficient jumps. In Bjørstad P., Espedal M., and Keyes D. (eds) *Ninth International Conference of Domain Decomposition Methods.* URL = `http://www.ddm.org/DD9/`.

[RF99] Rixen D. and Farhat C. (1999) A simple and efficient extension of a class of substructure based preconditioners to heterogeneous structural mechanics problems. *Int. J. Numer. Meth. Engng.* 44: 489–516.

[SK98] Stefanica D. and Klawonn A. (1998) A numerical study of a class of FETI preconditioners for mortar finite elements in two dimensions. Technical Report 773, Department of Computer Science, Courant Institute. URL = `http://www.cs.nyu.edu/publications/reports.html`.

15

Subspace Correction Methods for Convex Optimization Problems

Xue-Cheng Tai,[1] Jinchao Xu [2]

Introduction

Domain decomposition and multigrid methods have been intensively studied for linear partial differential equations. Recent research, see for example [Xu92], reveals that domain decomposition and multigrid methods can be analysed using a same framework, see also [BPWX91], [SBG96], [GO95]. The present work uses this framework to analyse the convergence of two algorithms for convex optimization problems. Our emphasis is on nonlinear problems instead of linear problems. The algorithms reduce to the standard additive and multiplicative Schwarz methods when used for linear partial differential equations.

Researches for domain decomposition and multigrid methods have been mostly concentrating on linear elliptic and parabolic partial differential equations. Extension to more difficult problems have been considered by some recent works. In this work, a general nonlinear convex minimization problems is considered. The proposed algorithms can be used for nonlinear partial differential equations, optimal control problems related to partial differential equations and eigenvalue problems [CSar] [Sha97]. The space decomposition can be a domain decomposition method, a multigrid

1 Department of Mathematics, University of Bergen, Johannes Brunsgate 12, 5007, Bergen, Norway. Email: tai@mi.uib.no and URL: http://www.mi.uib.no/˜tai.
2 Center for Computational Mathematics and Applications and Department of Mathematics, Pennsylvania State University, University Park, Pennsylvania 16802. Email: xu@math.psu.edu.
The work of the first author was partially supported by the Norwegian Research Council by Project SEP-115837/431 and by ELF Petroleum Norge AS. The work of the second author was partially supported by NSF DMS-9706949 through Penn State.
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

method or some other decomposition techniques.

Domain decomposition methods and multigrid methods have been studied for nonlinear partial differential equations by some earlier works, see [AL96], [Ban82], [DH97], [FG97], [HR89], [TE98], [Tai92], [Tai94], [Tai95a], [Tai95b], [Tai95c], [Xu94], [Xu96], etc. In comparison with the existing works, our approach has several features. For example, the proposed algorithms can be used for certain degenerated or singular nonlinear diffusion problems, i.e the nonlinear diffusion coefficient can be zero or infinity and our approach do not need extra assumption on the smoothness of the solutions. The methods work for natural domain decomposition and multigrid meshes. Moreover, only small size nonlinear problems need to be solved on the decomposed subspaces. We also emphasis that our approach is valid for general space decomposition techniques. So the applications is not restricted to domain decomposition and multigrid methods. Other space decomposition techniques can also be considered, see [SBG96], [XZ98].

The two algorithms given in this work were first proposed in [Tai92], see also [Tai94], [Tai95a], [Tai95c] and [TE98], where the qualitative convergence of the algorithms was proved, but the uniform rate of convergence was not given there.

Optimization problems and subspace correction methods

Consider the nonlinear optimization problem

$$\min_{v\in V} F(v) \ . \tag{1}$$

Here V is a reflexive Banach space and $F : V \to R$ is a convex functional. This problem has different applications, see §1

We shall use a space decomposition method to solve (1). A space decomposition method refers to a method that decomposes the space V into a sum of subspaces, i.e. there are subspaces $V_i,\ i = 1, 2, \cdots, m$, such that

$$V = V_1 + V_2 + \cdots + V_m \ . \tag{2}$$

Following the framework of [Xu92] for linear problems, we consider two types of subspace correction methods based on (2), namely the parallel subspace correction (PSC) method and the successive subspace correction (SSC) method.

Algorithm 1 *[A parallel subspace correction method].*

1. *Choose initial value $u^0 \in V$ and relaxation parameters $\alpha_i > 0$ such that $\sum_{i=1}^m \alpha_i \le 1$.*
2. *For $n \ge 0$, if $u^n \in V$ is defined, then find $e_i^n \in V_i$ in parallel for $i = 1, 2, \cdots, m$ such that*

$$F\left(u^n + e_i^n\right) \le F\left(u^n + v_i\right) \ , \quad \forall v_i \in V_i \ . \tag{3}$$

3. *Set u^{n+1} as in (4) and go to the next iteration.*

$$u^{n+1} = u^n + \sum_{i=1}^m \alpha_i e_i^n \ , \tag{4}$$

Algorithm 2 *[A successive subspace correction method].*

1. *Choose initial values* $u_i^0 = u^0 \in V$.
2. *For* $n \geq 0$, *if* $u^n \in V$ *is defined, find* $u^{n+i/m} = u^{n+(i-1)/m} + e_i^n$ *with* $e_i^n \in V_i$ *sequentially for* $i = 1, 2, \cdots, m$ *such that*

$$F\left(u^{n+(i-1)/m} + e_i^n\right) \leq F\left(u^{n+(i-1)/m} + v_i\right), \quad \forall v_i \in V_i . \tag{5}$$

3. *Go to the next iteration.*

When $p = q = 2$, some asynchronous algorithms are proposed in [TTed] and [TT98] for nonlinear variational inequalities and corresponding rate of convergence was also analysed in [TT98].

Global convergence of the algorithms

In the following, the notation $\langle \cdot, \cdot \rangle$ is used to denote the duality pairing between V and V', here V' is the dual space of V. The functional F is assumed to be Gateaux differentiable (see [ET76]) and there are constants $K,\ L > 0$, $p \geq q > 1$ such that

$$\begin{aligned} \langle F'(w) - F'(v), w - v\rangle &\geq K\|w - v\|_V^p, \quad \forall w, v \in V , \\ \|F'(w) - F'(v)\|_{V'} &\leq L\|w - v\|_V^{q-1}, \quad \forall w, v \in V , \end{aligned} \tag{6}$$

and from which it is easy to deduce that

$$F(w) - F(v) \geq \langle F'(v), w - v\rangle + \frac{K}{p}\|w - v\|_V^p , \quad \forall v, w \in V , \tag{7}$$

$$F(w) - F(v) \leq \langle F'(v), w - v\rangle + \frac{L}{q}\|w - v\|_V^q , \quad \forall v, w \in V . \tag{8}$$

Under assumption (6), problem (1) and subproblems (3) and (5) have unique solutions, see [ET76]. For some nonlinear problems, the constants K and L depend on v and w. However, just under the condition that F is strictly convex, it has been proved in [Tai92] and [Tai95c] that the iterative solutions of Algorithm 1 and Algorithm 2 converge to the true solution. Thus, one can assume that the computed solutions are in a neighbourhood of the true solution and so the constants K and L can be assumed to be independent of v and w. In case that the functional F is only locally convex in a neighbourhood of the true solution, by choosing the initial value close enough to the true solution, it can be proved that the computed solutions stay always inside the neighbourhood that the functional F is convex (the essential techniques of the proof is contained in the proof of Lemma 4.2 and 4.3 of [KT97]), and so the results given in this work are also applicable to this kind of problems.

For simplicity, we define

$$\sigma = \frac{p}{p - q + 1}, \quad \sigma' = \frac{p}{q - 1}, \quad \text{which satisfy} \quad \frac{1}{\sigma} + \frac{1}{\sigma'} = 1.$$

Note that $\sigma \leq p$. We shall use u to denote the unique solution of (1) which satisfies $\langle F'(u), v\rangle = 0, \quad \forall v \in V$. It is an easy consequence of (7) and (8) that

$$\frac{K}{p}\|v - u\|_V^p \leq F(v) - F(u) \leq \frac{L}{q}\|v - u\|_V^q, \quad \forall v \in V . \tag{9}$$

Therefore, in the following, we shall use

$$d_n = F(u^n) - F(u), \ \forall n \geq 0 \ , \tag{10}$$

as a measure of the error between u^n and the true solution u. For the decomposed spaces, we assume that there exits a constant $C_1 > 0$ such that for any $v \in V$, we can find $v_i \in V_i$ to satisfy:

$$v = \sum_{i=1}^{m} v_i \ , \qquad \text{and} \qquad \left(\sum_{i=1}^{m} \|v_i\|_V^\sigma \right)^{\frac{1}{\sigma}} \leq C_1 \|v\|_V \ . \tag{11}$$

Moreover, assume that there is a $C_2 > 0$ such that there holds

$$\begin{aligned} &\sum_{i=1}^{m} \sum_{j=1}^{m} \langle F'(w_{ij} + u_i) - F'(w_{ij}), v_j \rangle \\ \leq \quad & C_2 \left(\sum_{i=1}^{m} \|u_i\|_V^p \right)^{\frac{q-1}{p}} \left(\sum_{j=1}^{m} \|v_j\|_V^\sigma \right)^{\frac{1}{\sigma}} , \forall \ w_{ij} \in V, u_i \in V_i \text{ and } v_j \in V_j \ . \end{aligned}$$

The rate of convergence for Algorithm 1 can be estimated as in the following:

Theorem 1 *Assume that the space decomposition satisfies (11), (12) and the functional F satisfies (6). Set*

$$r = \frac{p(p-1)}{q(q-1)}, \quad C^* = \left[C_1 C_2 \left(\alpha_{max}^{\frac{(p-1)(q-1)}{p}} + \alpha_{min}^{-\frac{q-1}{p}} \right) K^{-1} \right]^{\frac{p}{q-1}} pK^{-1} \left(Lq^{-1} \right)^r .$$

Then for Algorithm 1 and d_n given by (10), we have:

1. If $r = 1$ (namely $p = q$),

$$d_{n+1} \leq \frac{C^*}{1 + C^*} d_n, \quad \forall n \geq 1 \ . \tag{12}$$

2. If $r > 1$, then there exists an $\xi_0 = \xi_0(d_0, C^*, r) \in [0,1]$ such that

$$d_{n+1} \leq \left(\frac{r-1}{C^*} \xi_0 + d_n{}^{1-r} \right)^{\frac{1}{1-r}} \leq \left(\frac{r-1}{C^*} (n+1) \xi_0 + d_0^{1-r} \right)^{\frac{1}{1-r}} , \quad \forall n \geq 1 \ . \tag{13}$$

See [TX98] for the details of the proofs. The estimate implies that when $r = 1$, the convergence is uniform. In case that $r > 1$, the convergence can be slow, i.e. $d_n = O\left((rn)^{-\frac{1}{r-1}} \right)$. Especially, when r is very big, $\frac{1}{1-r} \approx 0$ and the convergence can be very slow. Using that fact that $\sigma \leq p$, we see that it is impossible to have $r < 1$. In order to have $r = 1$, we must require $p = q$. The analysis given in [TE98] and [AL96] was done for $p = q = 2$.

The convergence of Algorithm 2 is similar to Algorithm 1.

Theorem 2 *Let the space decomposition satisfies (11), (12) and the functional F satisfies (6). Define*

$$r = \frac{p(p-1)}{q(q-1)}, \quad C^* = \left[\frac{C_1 C_2}{K}\right]^{\frac{p}{q-1}} \frac{p}{K}\left(\frac{L}{q}\right)^r . \tag{14}$$

1. If $r = 1$, we have

$$d_{n+1} \le \frac{C^*}{1+C^*} d_n, \quad \forall n \ge 1 . \tag{15}$$

2. If $r > 1$, then there exists an $\xi_0 = \xi_0(d_0, C^*, r) \in [0,1]$ such that

$$d_{n+1} \le \left(\frac{r-1}{C^*}\xi_0 + {d_n}^{1-r}\right)^{\frac{1}{1-r}} \le \left(\frac{r-1}{C^*}(n+1)\xi_0 + d_0^{1-r}\right)^{\frac{1}{1-r}}, \quad \forall n \ge 1 . \tag{16}$$

Overlapping domain decomposition for $W^{1,p}(\Omega)$

Let $\{\Omega_i\}_{i=1}^M$ be a shape-regular finite element division, or a coarse mesh, of Ω and Ω_i has diameter of order H. For each Ω_i, we further divide it into smaller simplices with diameter of order h. In case that Ω has a curved boundary, we shall also fill the area between $\partial\Omega$ and $\partial\Omega_H$, here $\bar{\Omega}_H = \cup_{i=1}^M \bar{\Omega}_i$, with finite elements with diameters of order h. We assume that the resulting elements form a shape regular finite element subdivision of Ω, see Ciarlet [Cia78]. We call this the fine mesh or the h-level subdivision of Ω with mesh parameter h. We denote $\Omega_h = \cup\{\mathcal{T} \in \mathcal{T}_h\}$ as the fine mesh subdivision. Let $S_0^H \subset W_0^{1,p}(\Omega_H)$ and $S_0^h \subset W_0^{1,p}(\Omega_h)$ be the continuous, piecewise r^{th} order polynomial finite element spaces, with zero trace on $\partial\Omega_H$ and $\partial\Omega_h$, over the H-level and h-level subdivisions of Ω respectively. More specifically,

$$S_0^H = \left\{ v \in W_0^{1,p}(\Omega_H) \middle| \quad v|_{\Omega_i} \in P_r(\Omega_i), \forall i \right\},$$

$$S_0^h = \left\{ v \in W_0^{1,p}(\Omega_h) \middle| \quad v|_{\mathcal{T}} \in P_r(\mathcal{T}), \forall \mathcal{T} \in \mathcal{T}_h \right\}.$$

For each Ω_i, we consider an enlarged subdomain $\Omega_i^\delta = \{\mathcal{T} \in \mathcal{T}_h, dist(\mathcal{T}, \Omega_i) \le \delta\}$. The union of Ω_i^δ covers $\bar{\Omega}_h$ with overlaps of size δ. Let us denote the piecewise r^{th} order polynomial finite element space with zero traces on the boundaries $\partial\Omega_i^\delta$ as $S_0^h(\Omega_i^\delta)$. Then one can show that

$$S_0^h = S_0^H + \sum S_0^h(\Omega_i^\delta) . \tag{17}$$

For the overlapping subdomains, assume that there exist m colors such that each subdomain Ω_i^δ can be marked with one color, and the subdomains with the same color will not intersect with each other. For suitable overlaps, one can always choose $m = 2$ if $d = 1$; $m \le 4$ if $d = 2$; $m \le 8$ if $d = 3$.Let Ω_i' be the union of the subdomains with the $i^{\rm th}$ color, and $V_i = \{v \in S_0^h | \quad v(x) = 0, \quad x \notin \Omega_i'\}$. By denoting subspaces $V_0 = S_0^H$, $V = S_0^h$, we find that decomposition (17) means

$$V = V_0 + \sum_{i=1}^m V_i, \tag{18}$$

and so the two-level method is a way to decompose the finite element space. Similar as in [Wid92], let $\{\theta_i\}_{i=1}^m$ be a partition of unity with respect to $\{\Omega_i'\}_{i=1}^m$, i.e. $\theta_i \in C_0^\infty(\Omega_i' \cap \Omega)$, $\theta_i \geq 0$, $\sum_{i=1}^m \theta_i = 1$ and $|\nabla \theta_i| \leq C/\delta$. Let I_h be an interpolation operator which uses the function values at the h-level nodes. For any $v \in V$, let $v_0 \in V_0$ be the solution of $(v_0, \phi_H) = (v, \phi_H), \forall \phi_H \in V_0$, and $v_i = I_h(\theta_i(v - v_0))$. They satisfy $v = \sum_{i=0}^m v_i$, and

$$\left(\|v_0\|_{1,p}^s + \sum_{i=1}^m \|v_i\|_{1,p}^s \right)^{\frac{1}{s}} \leq C(m+1)^{\frac{1}{s}} \left(1 + \left(\frac{H}{\delta}\right)^{\frac{p-1}{p}} \right) \|v\|_{1,p}, \quad \forall s > 1. \tag{19}$$

See [TX98] for the details of proofs. Using the Cauchy-Schwarz inequality, it is easy to prove by using (6) that:

$$\begin{aligned} \sum_{i=1}^m \sum_{j=1}^m \langle F'(w_{ij} + u_i) - F'(w_{ij}), v_j \rangle &\leq \sum_{i=1}^m \sum_{j=1}^m L \|u_i\|_V^{q-1} \|v_i\|_V \\ &\leq L m^{\frac{s-1}{s}} \left(\sum_{i=1}^m \|v_i\|_V^s \right)^{\frac{1}{s}} m^{\frac{1}{\sigma}} \left(\sum_{i=1}^m \|u_i\|_V^p \right)^{\frac{q-1}{p}} \\ &\forall u_i \in V_i,\ w_{ij} \in V \text{ and } v_j \in V_j,\ \forall s > 1\ . \end{aligned} \tag{20}$$

Estimates (19) and (20) show that for the constants in (11) and (12) only depend on the number of colors m and the ratio H/δ, i.e.

$$C_1 = C(m) \left(1 + \left(\frac{H}{\delta}\right)^{\frac{p-1}{p}} \right), \qquad C_2 = C(m).$$

Multilevel decomposition for $W^{1,p}(\Omega)$

From the space decomposition point of view, a multigrid algorithm is built upon the subspaces that are defined on a nested sequence of finite element partitions. We assume that the finite element partition $\mathcal{T}$ is constructed by a successive refinement process. More precisely, $\mathcal{T} = \mathcal{T}_J$ for some $J > 1$, and $\mathcal{T}_j$ for $j \leq J$ are a nested sequence of quasi-uniform finite element partitions, i.e. $\mathcal{T}_j$ consist of finite elements $\mathcal{T}_j = \{\tau_j^i\}$ of size h_j such that $\Omega = \cup_i \tau_j^i$ for which the quasi-uniformity constants are independent of j (cf. [Cia78]) and τ_{j-1}^l is a union of elements of $\{\tau_j^i\}$. We further assume that there is a constant $\gamma < 1$, independent of j, such that h_j is proportional to γ^{2j}.

Corresponding to each finite element partition $\mathcal{T}_j$, a finite element space $\mathcal{M}_j$ can be defined by

$$\mathcal{M}_j = \{v \in W^{1,p}(\Omega) : v|_\tau \in \mathcal{P}_1(\tau), \quad \forall\ \tau \in \mathcal{T}_j\}.$$

Each finite element space $\mathcal{M}_j$ is associated with a nodal basis, denoted by $\{\phi_j^i\}_{i=1}^{n_j}$ satisfying $\phi_j^i(x_j^k) = \delta_{ik}$, where $\{x_j^k\}_{k=1}^{n_j}$ is the set of all nodes of the elements of $\mathcal{T}_j$. Associated with each such a nodal basis function, we define a one dimensional subspace as follows

$$\mathcal{M}_j^i = \text{span } (\phi_j^i).$$

On each level, the nodes can be colored so that the neighboring nodes are always of different colors. The number of colors needed for a regular mesh is always a bounded constant; call it m_c. Let V_j^k, $k = 1, 2, \cdots m_c$ be the sum of the subspaces $\mathcal{M}_j^i$ associated with nodes of the k^{th} color on level j. Letting $V = \mathcal{M}_J$, we have the following trivial space decomposition:

$$V = \sum_{j=1}^{J} \sum_{k=1}^{m_c} V_j^k. \tag{21}$$

Each subspace V_j^k contains some orthogonal one dimensional subspaces $\mathcal{M}_j^i$ and so the minimization problems (3) and (5) for each V_j^k can be done in parallel over the one dimensional subspaces $\mathcal{M}_j^i$. Under the assumption that

$$\langle F'(w+u) - F'(w), v \rangle \leq L \|u\|_{1,p,supp(u)\cap supp(v)}^{q-1} \|v\|_{1,p,supp(u)\cap supp(v)}, \tag{22}$$

it was proved in [TX98] that

$$C_1 \cong J^{\frac{1}{\sigma}} \cong |\log h|^{\frac{1}{\sigma}}, \qquad C_2 = C.$$

1 Some Applications

The algorithms can be used for linear second order equation

$$-\nabla \cdot (a \nabla u) = f \text{ in } \Omega \subset R^d , \qquad u = 0 \text{ on } \partial\Omega ,$$

and linear fourth order equation

$$\Delta(a \Delta u) = f \text{ in } \Omega \subset R^d , \qquad u = 0, \ \frac{\partial u}{\partial n} = 0, \text{ on } \partial\Omega .$$

If we use Algorithm 2 for a general symmetric positive define linear problem

$$a(u, v) = (f, v), \qquad \forall v \in V.$$

then the implementation can be divided into the following steps:

Algorithm 3 *(Application to linear problems)*

1. *Choose initial values* $u^0 \in V$ *and compute the initial residual* r^0 *such that* $(r^0, v) = (f, v) - a(u^0, v), \forall v \in V$.
2. *For* $i = 1, 2, \cdots, m$, *if* $r^{n+\frac{(i-1)}{m}}$ *is known, compute* $e_i^n \in V_i$ *such that*

$$a(e_i^n, v_i) = (r^{n+\frac{(i-1)}{m}}, v_i) , \quad \forall v_i \in V_i . \tag{23}$$

3. *Update the residual* $r^{n+\frac{i}{m}}$ *such that*

$$(r^{n+\frac{i}{m}}, v) = (r^{n+\frac{(i-1)}{m}}, v) - a(e_i^n, v), \quad \forall v \in V . \tag{24}$$

4. Update the solution as in (25) and go to the next iteration.

$$u^{n+\frac{i}{m}} = u^{n+\frac{(i-1)}{m}} + e_i^n \ . \tag{25}$$

The implementation for Algorithm 1 is similar. If the subspaces V_i are associated with the overlapping domain decomposition, then equation (23) is the solving of the subdomain problems. Equations (24) and (25) are just the simple updatings of the residual and the solution in the subdomains. If the subspaces V_i are associated with the multigrid method, then equation (23) is to compute the correction value for the nodal bases at different levels. Equations (24) and (25) are the updatings for the residual and solution corresponding to the nodal bases.

For applications to nonlinear problems, consider

$$-\nabla \cdot (|\nabla u|^{s-2}\nabla u) = f \text{ in } \Omega \subset R^d \ (1 < s < \infty) \ , \qquad u = 0 \text{ on } \partial\Omega \ . \tag{26}$$

We assume $f \in W^{-1,s'}(\Omega)$, $\frac{1}{s} + \frac{1}{s'} = 1$. By standard techniques, it can be shown, see [ET76], that (26) possesses a unique solution which is the minimizer of

$$\min_{v \in W_0^{1,s}(\Omega)} \left[\frac{1}{s} \int_\Omega |\nabla v|^s - \langle f, v \rangle \right] \ .$$

For the functional F associated with (26), it is true that conditions (6) and (22) are valid with

$$\begin{aligned} p &= s, \quad q = 2 \qquad \text{if } s \geq 2; \\ p &= 2, \quad q = s \qquad \text{if } 1 < s \leq 2. \end{aligned}$$

See Ciarlet [Cia78], Glowinski and Marrocco [GM75] and [TX98] for the details. The full potential equation considered in [CGKT94] is of a similar type to equation (26).

For more general problem

$$\min_{v \in W_0^{1,s}(\Omega)} \int_\Omega \frac{1}{2} a(|\nabla v|^2) + f(v) \ , \tag{27}$$

we assume that a is strictly convex and f is convex and both are differentiable. If we use Algorithm 2 for (27), then we obtain

Algorithm 4 *(Application to nonlinear problems)*

1. *Choose initial values* $u^0 \in V$.
2. *For* $i = 1, 2, \cdots, m$, *if* $u^{n+\frac{(i-1)}{m}}$ *is known, compute* $e_i^n \in V_i$ *such that*

$$\int_\Omega \Bigg[a'(|\nabla(u^{n+\frac{(i-1)}{m}} + e_i^n)|^2)\nabla(u^{n+\frac{(i-1)}{m}} + e_i^n) \cdot \nabla v_i \\ + f'(u^{n+\frac{(i-1)}{m}} + e_i^n) v_i \Bigg] dx = 0 \ , \forall v_i \in V_i \ . \tag{28}$$

3. *Update the solution as in (29) and go to the next iteration.*

$$u^{n+\frac{i}{m}} = u^{n+\frac{(i-1)}{m}} + e_i^n \ . \tag{29}$$

If V_i are the domain decomposition subspaces, then problem (28) is a nonlinear problem in each subdomain, which has a smaller size than the original problem. For some minimization methods, the convergence and the computing time depend on the size of the problem. Thus by first reducing the problem into smaller size problems and then minimize, we may gain efficiency. If V_i are the multigrid nodal basis subspaces, then (28) is equivalent to some one dimensional nonlinear problems and we can use efficient minimization routines to solve the one dimensional problems.

REFERENCES

[AL96] Axelsson A. and Layton W. (1996) A two-level method for the discretization of nonlinear boundary value problems. *SIAM J. Numer. Anal.* 33: 2359–2374.

[Ban82] Bank R. (1982) Analysis of a multilevel iterative method for nonlinear finite element equations. *Math. Comp.* . 39: 453–465.

[BPWX91] Bramble J. H., Pasciak J. E., Wang J., and Xu J. (1991) Convergence estimates for product iterative methods with applications to domain decomposition. *Math. Comp.* 57: 1–21.

[CGKT94] Cai X.-C., Groop W. D., Keyes D. E., and Tidriri M. D. (1994) Parallel implicit methods for aerodynamics. In Keyes D. E. and Xu (eds) *Proceeding of the seventh international conference on domain decomposition methods in Science and scientific computing (Penn. state Univ.)*, pages 465–470. AMS, Providence.

[Cia78] Ciarlet P. G. (1978) *The Finite Element Method for Elliptic Problems.* North-Holland, Amsterdam.

[CSar] Chan T. F. and Sharapov I. (To appear) Subspace correction multilevel methods for elliptic eigenvalue problems. In Bjørstad P., Espedal M., and Keyes D. (eds) *Proceedings of the 9th international domain decomposition methods.* John Wiley and Sons.

[DH97] Dryja M. and Hackbusch W. (1997) On the nonlinear domain decomposition method. *BIT* 37: 296–311.

[ET76] Ekeland I. and Temam R. (1976) *Convex analysis and variational problems.* North–Holland, Amsterdam.

[FG97] Ferguson R. C. and Graham I. G. (1997) Multilevel adaptive methods for semilinear equations with applications to device modelling. In Bjøstard P., Espedal M., and Keyes D. (eds) *Proceedings of the 9th international domain decomposition conference, Norway.* John Wiley and Sons.

[GM75] Glowinski R. and Marrocco A. (1975) Sur l'approximation parééléments finis d'ordre un, et lan résolution par pénalisation-dualité, d'une classe de problémes de Dirichlet non linéaires. *Rev. Fr. Autom. Inf. Rech. Oper. Anal. Numér. R-2* pages 41–76.

[GO95] Griebel M. and Oswald P. (1995) On the abstract theory of additive and multiplicative schwartz algorithms. *Numer. Math.* 70: 163–180.

[HR89] Hackbusch W. and Reusken A. (1989) Analysis of a damped nonlinear multilevel method. *Numer. Math.* 55: 225–246.

[KT97] Kunisch K. and Tai X.-C. (1997) Sequential and parallel splitting methods for bilinear control problems in Hilbert spaces. *SIAM J. Numer. Anal.* 34: 91–118.

[SBG96] Smith B. F., Bjørstad P. E., and Gropp W. D. (1996) *Domain decomposition: Parallel multilevel algorithms for elliptic partial differential equations.* Cambridge Univ. Press, Cambridge.

[Sha97] Sharapov I. (1997) Multilevel subspace correction for large scale optimization problems. Technical Report cam-97-31, University of California at Los Angeles.

[Tai92] Tai X.-C. (1992) Parallel function decomposition and space decomposition methods with applications to optimisation, splitting and domain decomposition. *Preprint No. 231–1992, Institut für Mathematik, Technische Universität Graz* See also http://www.mi.uib.no/˜tai.

[Tai94] Tai X.-C. (1994) Parallel function and space decomposition methods. In Neittaanmäki P. (ed) *Finite element methods, fifty years of the Courant element, Lecture notes in pure and applied mathematics*, volume 164, pages 421–432. Marcel Dekker inc. See also http://www.mi.uib.no/˜tai.

[Tai95a] Tai X.-C. (1995) Domain decomposition for linear and nonlinear elliptic problems via function or space decomposition. In Keyes D. and Xu J. (eds) *Domain decomposition methods in scientific and engineering computing (Proc. of the 7th international conference on domain decomposition, Penn. State University, 1993)*, pages 355–360. American Mathematical Society.

[Tai95b] Tai X.-C. (1995) Parallel function and space decomposition methods – part I. function decomposition. *Beijing Mathematics* 1, part 2: 104–134. See also http://www.mi.uib.no/˜tai.

[Tai95c] Tai X.-C. (1995) Parallel function decomposition and space decomposition methods: Part II. Space decomposition. *Beijing Mathematics* 1, part 2: 135–152. See also http://www.mi.uib.no/˜tai.

[TE98] Tai X.-C. and Espedal M. (1998) Rate of convergence of some space decomposition method for linear and nonlinear elliptic problems. *SIAM J. Numer. Anal.* 35: 1558–1570.

[TT98] Tai X.-C. and Tseng P. (August, 1998) Convergence rate analysis of an asynchronous space decomposition method for convex minimization. Technical Report 120, Department of Mathematics, University of Bergen. See also http://www.mi.uib.no/˜tai.

[TTed] Tai X.-C. and Tseng P. (Submitted) An asynchronous space decomposition method. In *Proceedings of the 11th international conference on domain decomposition method, London, 1998.*

[TX98] Tai X.-C. and Xu J. (1998) Global convergence of subspace correction methods for convex optimization problems. Report no. 114, Department of Mathematics, University of Bergen, Norway. See also http://www.mi.uib.no/˜tai.

[Wid92] Widlund O. (1992) Some Schwarz methods for symmetric and nonsymmetric elliptic problems. In *Proceedings of the fifth international sympsium on domain decomposition methods for partial differential equations, Norfolk, May, 1991.* SIAM, Philadelphia.

[Xu92] Xu J. (1992) Iteration methods by space decomposition and subspace correction. *SIAM Rev.* 34: 581–613.

[Xu94] Xu J. (1994) A novel two-grid method for semilinear elliptic equations. *SIAM J. Sci. Comp.* . 15: 231–237.

[Xu96] Xu J. (1996) Two-grid discretization techniques for linear and nonlinear pde. *SIAM J. Numer. Anal.* 27: 1759–1777.

[XZ98] Xu J. and Zou J. (1998) Nonoverlapping domain decomposition methods. *SIAM Rev.* pages 857–914.

16

Some Local and Parallel Properties of Finite Element Discretizations

Jinchao Xu[1], Aihui Zhou[2]

Introduction

This paper is devoted to the study of some local and parallel properties of finite elements for elliptic boundary value problems of second order. Several local and parallel algorithms are proposed and analyzed by means of two-grid discretizations. The algorithms are motivated from the observation that, for a solution to some elliptic problems, low frequency components can be approximated well by a relatively coarse grid and high frequency components can be computed on a fine grid by some local and parallel procedure. One major technical tool for the analysis is some sharp local a priori estimates for finite element solutions on general shape-regular grids.

This paper can be considered as a sequel of our earlier paper [XZ98] on a similar topic. While in [XZ98] we studied our methods for a rather general class of model partial differential equations, in this paper we focus our attention on a rather specific case, namely, Neumann boundary value problems on a smooth domain and as a result we obtain some better error estimates for higher order elements.

---

1 Center for Computational Mathematics and Applications and Department of Mathematics, The Pennsylvania State University, University Park, PA 16802. Email: xu@math.psu.edu. This work was partially supported by NSF DMS-9706949, NSF ACI-9800244 and NASA NAG2-1236 through Penn State and Center for Computational Mathematics and Applications, The Pennsylvania State University, and by NSF ASC 9720257 through UCLA.

2 Institute of Systems Science, Academia Sinica, Beijing 100080, China. Email: azhou@bamboo.iss.ac.cn. This work was partially supported by NSF DMS-9706949, NSF ACI-9800244 and NASA NAG2-1236 through Penn State and the Center for Computational Mathematics and Applications, The Pennsylvania State University, and by National Science Foundation of China.

Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

Preliminaries

Let Ω be a bounded smooth domain in $R^d (d \geq 1)$. We will use the standard notation for Sobolev spaces $W^{s,p}(\Omega)$ and their associated norms and seminorms, see e.g. [Ada75, CL91]. For $p = 2$, we denote $H^s(\Omega) = W^{s,2}(\Omega)$, $\|\cdot\|_{s,\Omega} = \|\cdot\|_{s,2,\Omega}$ and $\|\cdot\|_{\Omega} = \|\cdot\|_{0,2,\Omega}$. (In some places of this paper, $\|\cdot\|_{s,2,\Omega}$ should be viewed as defined piecewise if it is necessary.) For $D \subset G \subset \Omega$, we use the notation $D \subset\subset G$ to mean that $\text{dist}(\partial D \backslash \partial\Omega, \partial G \backslash \partial\Omega) > 0$. In this paper, we will use the letter C (with or without subscripts) to denote a generic positive constant which may stand for different values at its different occurrences. Following [Xu92a], $A \lesssim B$ means that $A \leq CB$ for some constant C is independent of mesh parameters.

Finite element spaces

Assume that $T^h(\Omega) = \{\tau\}$ is a mesh of Ω with mesh-size function $h(x)$ whose value is the diameter h_τ of the element τ containing x. Let $h_\Omega = \max_{x\in\Omega} h(x)$ be the (largest) mesh size of $T^h(\Omega)$. Sometimes, when it is clear from the context, we will drop the subscript in h_Ω and use h for the mesh size on a domain. One basic assumption on the mesh is that it is not exceedingly over-refined locally:

A.0. There exists $\gamma \geq 1$ such that

$$h_\Omega^\gamma \lesssim h(x), \quad x \in \Omega.$$

Associated with a mesh $T^h(\Omega)$, let $S^h(\Omega) \subset H^1(\Omega)$ be a finite dimensional subspace on Ω. Given $G \subset \Omega$, we define $S^h(G)$ and $T^h(G)$ to be the restriction of $S^h(\Omega)$ and $T^h(\Omega)$ to G, respectively, and

$$S_h^0(G) = \{v \in S^h(\Omega) : \text{supp } v \subset\subset G\}.$$

For any $G \subset \Omega$ mentioned here, we assume that it aligns with $T^h(\Omega)$ when it is necessary. We now state our basic assumptions on the finite element spaces.

A.1. *Approximation.* There exists $r \geq 1$ such that for $w \in H^1(\Omega)$,

$$\inf_{v\in S^h(\Omega)} (\|h^{-1}(w-v)\|_{0,\Omega} + \|w-v\|_{1,\Omega}) \lesssim \|h^s w\|_{1+s,\Omega}, \quad 0 \leq s \leq r.$$

A.2. *Inverse Estimate.* For any $v \in S^h(\Omega)$,

$$\|v\|_{1,\Omega} \lesssim \|h^{-1}v\|_{0,\Omega} \text{ and } \|v\|_{0,\Omega} \lesssim \|h^{-s}v\|_{-s,\Omega}.$$

A.3. *Superapproximation.* For $G \subset \Omega_0 \subset \Omega$, let $\omega \in C_0^\infty(\Omega)$ with supp $\omega \subset\subset G$. Then for any $w \in S^h(G)$, there is $v \in S_h^0(G)$ such that

$$\|h^{-1}(\omega w - v)\|_{1,G} \lesssim \|w\|_{1,G}.$$

The assumptions mentioned above are satisfied by most of the finite element spaces used in practice when the elements that meet $\partial\Omega$ are curved to fit $\partial\Omega$ exactly. We refer to [SW77, SW95, Wah91, Wah95, XZ98] for details.

A model problem

We consider a Neumann boundary value problem

$$\begin{cases} Lu \equiv -\sum_{i,j=1}^{d} \frac{\partial}{\partial x_j}(a_{ij}\frac{\partial u}{\partial x_i}) + cu & = f, \quad \text{in} \quad \Omega, \\ \frac{\partial u}{\partial n_L} & = 0, \quad \text{on} \quad \partial\Omega. \end{cases} \tag{1}$$

Here $\frac{\partial u}{\partial n_L}$ denotes the co-normal derivative on $\partial\Omega$, a_{ij} and c are smooth, and (a_{ij}) is uniformly positive definite on $\bar{\Omega}$.

The weak form of (1) is as follows: Find $u \equiv L^{-1}f \in H^1(\Omega)$ such that

$$a(u, v) = (f, v), \qquad \forall v \in H^1(\Omega), \tag{2}$$

where $(\cdot,\cdot) \equiv (\cdot,\cdot)_\Omega$ is the standard inner-product of $L^2(\Omega)$ and

$$a(u, v) \equiv a_\Omega(u, v) = \int_\Omega \sum_{i,j=1}^{d} a_{ij}\frac{\partial u}{\partial x_i}\frac{\partial v}{\partial x_j} + cuv.$$

Our basic assumption is that (2) is well-posed, namely (2) is uniquely solvable for any $f \in H^{-1}(\Omega)$. Thus, since $\partial\Omega$ is smooth, if $f \in H^{r-1}(\Omega)$, then $L^{-1}f \in H^{r+1}(\Omega)$.

For $u \in H^1(\Omega)$, we define a finite element solution $u_h : H^1(\Omega) \mapsto S^h(\Omega)$ by

$$a(u_h, v) = (f, v), \quad \forall v \in S^h(\Omega) \tag{3}$$

and it is well-known that

$$\|u_h\|_{1,\Omega} \lesssim \|u\|_{1,\Omega} \quad \text{and} \quad \|u - u_h\|_{-s,\Omega} \lesssim h_\Omega^{s+1}|h^r u|_{r+1,\Omega}, \; -1 \le s \le r-1. \tag{4}$$

Locality of finite element approximations

In this section, we shall study some local properties for finite element discretizations on general shape regular grids. The results presented here generalize local a priori error estimates known in literature (cf. [NS74, SW77, SW95, Wah91, Wah95, ZLSL98]) to more general finite element meshes, which will play a crucial role in our analysis.

We assume that $D \subset\subset \Omega_0 \subset \Omega$ and that $\partial\Omega_0$ is smooth. The proof of the following lemma is similar to a corresponding result in [XZ98] (see also [NS74, SW77, SW95, Wah91, Wah95]).

Lemma 1. *If Assumptions A.0, A.2 and A.3 hold and* $w \in S^h(\Omega_0)$ *satisfies*

$$a(w, v) = 0, \qquad \forall v \in S_h^0(\Omega_0), \tag{5}$$

then

$$\|w\|_{1,D} \lesssim \|w\|_{-l,\Omega_0}, \quad l = 0, 1, 2, \cdots, r-1. \tag{6}$$

Proof. Let p be an integer such that $p \geq \gamma - 1$ and Ω_j $(j = 1, 2, \cdots, p)$ be subdomains with smooth boundaries which satisfy

$$D \subset\subset \Omega_{2p+1} \subset\subset \Omega_{2p} \subset\subset \cdots \subset\subset \Omega_1 \subset\subset \Omega_0.$$

Choose $D_1 \subset \Omega$ satisfying $D \subset\subset D_1 \subset\subset \Omega_{2p+1}$ and $\omega \in C_0^\infty(\Omega)$ such that $\omega \equiv 1$ on $\bar{D}_1$ and supp $\omega \subset\subset \Omega_{2p+1}$. For any $\phi \in H^l(D)(l = 0, 1, \cdots, r-1)$, let $E\phi$ be the universal extension of ϕ on Ω and $\Phi = L^{-1}\phi$. Then $\Phi \in H^{l+2}(\Omega)$ and

$$(w, \phi)_D = (\omega w, \phi) = a(\omega w, \Phi).$$

Note that $\|\Phi\|_{l+2,\Omega_{2p+1}} \lesssim \|\phi\|_{l,D}$ and, for any $v \in S_h^0(\Omega_{2p+1})$,

$$\begin{aligned} & a(\omega w, \Phi) \lesssim a(w, \omega\Phi) + \|w\|_{-l-1,\Omega_{2p+1}} \|\Phi\|_{l+2,\Omega_{2p+1}} \\ = \quad & a(w, \omega\Phi - v) + \|w\|_{-l-1,\Omega_{2p+1}} \|\Phi\|_{l+2,\Omega_{2p+1}}, \end{aligned}$$

we get

$$\|w\|_{-l,D} \lesssim \|w\|_{-l-1,\Omega_{2p+1}} + h_{\Omega_{2p+1}}^{l+1} \|w\|_{1,\Omega_{2p+1}}. \tag{7}$$

Using $\|w\|_{1,\Omega_{2p+1}} \lesssim \|w\|_{0,\Omega_{2p}}$(see [XZ98]) and taking $l = 0$, we obtain

$$\|w\|_{0,D} \lesssim \|w\|_{-1,\Omega_0} + h_{\Omega_0} \|w\|_{0,\Omega_{2p}}.$$

The argument may be repeated for $\|w\|_{\Omega_{2p}}$ on the right to yield

$$\|w\|_{0,\Omega_{2p-2j+2}} \lesssim \|w\|_{-1,\Omega_0} + h_{\Omega_0} \|w\|_{0,\Omega_{2p-2j}}, \quad j = 1, 2, \cdots, p.$$

Thus, we obtain from Assumption A.2 that

$$\begin{aligned} & \|w\|_{1,D} \lesssim \|w\|_{-1,\Omega_0} + h_{\Omega_0}^{p+1} \|w\|_{0,\Omega_0} \\ \lesssim \quad & \|w\|_{-1,\Omega_0} + h_{\Omega_0}^{p+1} \|h^{-1} w\|_{-1,\Omega_0} \lesssim \|w\|_{-1,\Omega_0}. \end{aligned}$$

This proves (5) for $l = 1$ (see [XZ98] for $l = 0$). The argument then proceeds via induction using (7) to complete the proof.

Theorem 2. *If Assumptions A.0, A.1, A.2 and A.3 hold, then*

$$\|u - u_h\|_{1,D} \lesssim \inf_{v \in S^h(\Omega)} (\|u - v\|_{1,\Omega_0} + \|u - u_h\|_{1-r,\Omega}). \tag{8}$$

Proof. Let $R_h : H^1(\Omega_0) \longrightarrow S^h(\Omega_0)$ be the Galerkin-projection defined by

$$a_{\Omega_0}(w - R_h w, v) = 0, \quad \forall w \in H^1(\Omega_0), \ \forall v \in S^h(\Omega_0). \tag{9}$$

Choose $D_1 \subset \Omega$ satisfying $D \subset\subset D_1 \subset\subset \Omega_0$ and $\omega \in C_0^\infty(\Omega_0)$ such that $\omega \equiv 1$ on $\bar{D}_1$, then for $\tilde{u} = \omega u$,

$$a(R_h \tilde{u} - u_h, v) = 0, \quad \forall v \in S_h^0(D_1).$$

Thus, Lemma 1 yields

$$\|R_h \tilde{u} - u_h\|_{1,D} \lesssim \|R_h \tilde{u} - u_h\|_{1-r,D_1}.$$

Therefore, estimates similar to (4) lead to

$$
\begin{aligned}
& \|u-u_h\|_{1,D} \le \|\tilde u - R_h\tilde u\|_{1,D} + \|R_h\tilde u - u_h\|_{1,D} \\
\lesssim\ & \|\tilde u - R_h\tilde u\|_{1,D} + \|R_h\tilde u - u_h\|_{1-r,D_1} \lesssim \|\tilde u - R_h\tilde u\|_{1,D_1} + \|u-u_h\|_{1-r,D_1} \\
\lesssim\ & \|\tilde u - R_h\tilde u\|_{1,\Omega_0} + \|u-u_h\|_{1-r,D_1} \lesssim \|u\|_{1,\Omega_0} + \|u-u_h\|_{1-r,\Omega},
\end{aligned}
$$

which together with a standard argument produce (8).

Theorem 3. *Under the assumption of Theorem 2, there holds*

$$
\|u-u_h\|_{1,D} \lesssim (h^r_{\Omega_0} + h^{2r}_{\Omega})|u|_{r+1,\Omega}. \tag{10}
$$

Local and parallel algorithms

In this section, we shall develop some local and parallel finite element algorithms. These algorithms are motivated by the local properties of finite element studied in the previous section. We shall first discuss the local algorithms. The parallelization of the local algorithms is straightforward.

Let $S^h(\Omega) \subset H^1(\Omega)$ be a finite element subspace satisfying Assumptions A.1, A.2 and A.3 associated with a grid $T^h(\Omega)$ which satisfies Assumption A.0. Let $u_h \in S^h(\Omega)$ be the solution of the standard finite element scheme for solving (2). Either locally or globally, with proper regularity assumption, we have the following error estimate:

$$
\|u-u_h\|_1 \lesssim h^r.
$$

With this type of error estimates in mind, we will, in the rest of this section, only compare the approximate solutions of our algorithms with u_h instead of the exact solution u.

Local algorithms

The main idea for the local algorithm is that the more global component of a finite element solution may be obtained by a relatively coarser grid and, the rest of the computation can then be localized.

Roughly speaking, the algorithms will sometimes be based on a coarse grid of size H and a fine grid of size $h \ll H$, and sometimes on a grid that is fine in a subdomain and coarse on the rest of the domain. The fine grid may be defined locally only. In our analysis, we shall use an auxiliary fine grid, denoted by $T^h(\Omega)$, that is globally defined and coincide with the local fine grid in the subdomain of interest.

Let $T^H(\Omega)$ be a shape-regular coarse grid, of size $H \gg h$, and $T^h(\Omega_0)$ an highly locally refined mesh, where Ω_0 is a slightly larger subdomain with smooth boundary, containing a subdomain $D \subset \Omega$ (namely $D \subset\subset \Omega_0$). More precisely, we let $T^h_H(\Omega)$ denote a locally refined shape-regular mesh that may be viewed as being obtained by refining $T^H(\Omega)$ locally around the subdomain D in such a way that $T^h_H(\Omega_0) = T^h(\Omega_0)$. We are interested in obtaining the approximate solution in the given subdomain D with an accuracy comparable to that from $T^h(\Omega)$. We shall propose two different gridding strategies for obtaining finite element approximations on the subdomain D.

We denote the corresponding finite element space by $S^{H,h}(\Omega) \subset H^1(\Omega)$, which satisfies Assumptions A.1, A.2 and A.3, too.

The first strategy is simply to solve a standard finite element solution in $S^{H,h}(\Omega)$.

Algorithm A0. *Find $u_H^h \in S^{H,h}(\Omega)$ such that*

$$a(u_H^h, v) = (f, v), \quad \forall v \in S^{H,h}(\Omega).$$

Although this algorithm is still a global algorithm since a global problem is solved, it is designed to obtain a local approximation in the subdomain D and it makes use of a mesh that is much coarser away from D.

We have the following theorem

Theorem 4. *If $u_H^h \in S^{H,h}(\Omega)$ is obtained by Algorithm A0, then*

$$\|u_h - u_H^h\|_{1,D} \lesssim H^{2r}|u|_{r+1,\Omega}.$$

Proof. By the definition of Algorithm A0 and our assumption on the auxiliary grid $T^h(\Omega)$ that coincide with $T_H^h(\Omega)$ on Ω_0, we have

$$a(u_H^h - u_h, v) = 0, \quad \forall v \in S_h^0(\Omega_0).$$

By Lemma 1, we get

$$\|u_h - u_H^h\|_{1,D} \lesssim \|u_h - u_H^h\|_{1-r,\Omega_0}$$

and can then finish the proof.

Our second strategy is an improvement of the first strategy by using a residual-correction technique as in [Xu92b, Xu94, Xu96]. In this strategy, we first solve a global problem only on the given coarse grid $T^H(\Omega)$ and then correct the residual locally on the fine mesh $T^h(\Omega_0)(= T_H^h(\Omega_0))$.

A prototype of the local algorithms is as follows.

Algorithm B0.

1. *Find a global coarse grid solution $u_H \in S^H(\Omega)$:*

$$a(u_H, v) = (f, v), \quad \forall v \in S^H(\Omega).$$

2. *Find a local fine grid correction $e_h \in S^h(\Omega_0)$:*

$$a_{\Omega_0}(e_h, v) = (f, v)_{\Omega_0} - a_{\Omega_0}(u_H, v), \quad \forall v \in S^h(\Omega_0).$$

3. *Set $u^h = u_H + e_h$, in Ω_0.*

Theorem 5. *If $u^h \in S^h(\Omega_0)$ is obtained by Algorithm B0, then*

$$\|u_h - u^h\|_{1,D} \lesssim H^{2r}|u|_{r+1,\Omega}.$$

Proof. By the definition of Algorithm B0,

$$a(u^h - u_h, v) = 0, \quad \forall v \in S_h^0(\Omega_0).$$

By Lemma 1, we get

$$\|u_h - u^h\|_{1,D} \lesssim \|u^h - u_h\|_{1-r,\Omega_0} \lesssim \|u_h - u_H\|_{1-r,\Omega_0} + \|e_h\|_{1-r,\Omega_0}.$$

To estimate $\|e_h\|_{1-r,\Omega_0}$, we use the Aubin-Nitsche duality argument. Given any $\phi \in H^{r-1}(\Omega_0)$, there exist $w \in H^{r+1}(\Omega_0)$ and $\tilde{w} \in H^{r+1}(\Omega)$ such that

$$a_{\Omega_0}(v, w) = (\phi, v)_{\Omega_0}, \ \forall v \in H^1(\Omega_0) \quad \text{and} \quad a(v, \tilde{w}) = (\phi, v)_{\Omega_0}, \ \forall v \in H^1(\Omega).$$

One sees that there exist $\tilde{w}_H \in S^H(\Omega)$ and $w_h^0 \in S^h(\Omega_0)$ satisfying

$$a(v, \tilde{w}_H) = (\phi, v)_{\Omega_0}, \ \forall v \in S^H(\Omega) \quad \text{and} \quad \|\tilde{w} - \tilde{w}_H\|_{1,\Omega} \lesssim H^r \|\phi\|_{r-1,\Omega_0},$$

$$a_{\Omega_0}(v, w_h^0) = (\phi, v)_{\Omega_0}, \ \forall v \in S^h(\Omega_0) \quad \text{and} \quad \|w - w_h^0\|_{1,\Omega_0} \lesssim h^r \|\phi\|_{r-1,\Omega_0}.$$

It follows that

$$\begin{aligned} & (e_h, \phi)_{\Omega_0} = a_{\Omega_0}(e_h, w) = a_{\Omega_0}(e_h, w_h^0) = a_{\Omega_0}(u_h - u_H, w_h^0) \\ = \ & a_{\Omega_0}(u_h - u_H, w_h^0 - w) + a(u_h - u_H, \tilde{w} - \tilde{w}_H) \lesssim H^r \|u_h - u_H\|_{1,\Omega} \|\phi\|_{r-1,\Omega_0}, \end{aligned}$$

which implies $\|e_h\|_{1-r,\Omega_0} \lesssim H^r \|u_h - u_H\|_{1,\Omega}$. The desired result then follows.

Parallel algorithms

Given an initial coarse triangulation $T^H(\Omega)$, let us divide Ω into a number of disjoint subdomains $D_1, \ldots, D_m$ and we then enlarge each D_j to obtain Ω_j that align with $T^H(\Omega)$ and have smooth boundaries. The basic idea of our parallel algorithm is very simple: we just apply the local algorithms in parallel in all Ω_j's.

Let us first discuss the parallel version of Algorithm A0. For each j, we use some adaptive process to obtain a shape-regular mesh $T_j(\Omega)$ and the corresponding finite element solution denoted by u_j. We note that each $T_j(\Omega)$ has a substantially finer mesh inside Ω_j. We note that the $T_j(\Omega)$ provide different triangulations for Ω and that they can be very arbitrary; for simplicity of exposition, we assume that each $T_j(\Omega)$ has the same size h in Ω_j (more precisely, $T_j(\Omega_j) = T^h(\Omega_j)$) and has the size H away from Ω_j. Let $S^{h_j}(\Omega) \subset H^1(\Omega)$ be the corresponding finite element spaces.

Algorithm A1.

1. *Find* $u_j \in S^{h_j}(\Omega)$ $(j = 1, 2, \cdots, m)$ *in parallel:*

$$a(u_j, v) = (f, v), \quad \forall v \in S^{h_j}(\Omega).$$

2. *Set* $u^h = u_j$, *in* D_j $(j = 1, 2, \cdots, m)$.

Define a piecewise norm by $|||u_h - u^h|||_{1,\Omega} = (\sum_{j=1}^m \|u_h - u^h\|_{1,D_j}^2)^{1/2}$. By Theorem 4, we have

$$|||u_h - u^h|||_{1,\Omega} \lesssim H^{2r} |u|_{r+1,\Omega} \text{ and } |||u - u^h|||_{1,\Omega} \lesssim (h^r + H^{2r}) |u|_{r+1,\Omega}.$$

We now discuss the parallel version of Algorithm B0. For clarity of exposition, it appears to be most convenient to discuss this method using two globally defined grids: an initial coarse grid $T^H(\Omega)$ and a refinement of $T^H(\Omega)$, $T^h(\Omega)$ with $h \ll H$.

Algorithm B1.

1. Find a global coarse grid solution $u_H \in S^H(\Omega)$*:*

$$a(u_H, v) = (f, v), \quad \forall v \in S^H(\Omega).$$

2. Find local fine grid corrections $e_h^j \in S^h(\Omega_j)$ $(j = 1, 2, \cdots, m)$ *in parallel:*

$$a_{\Omega_j}(e_h^j, v) = (f, v)_{\Omega_j} - a_{\Omega_j}(u_H, v). \quad \forall v \in S^h(\Omega_j),$$

3. Set $u^h = u_H + e_h^j$, *in* D_j $(j = 1, 2, \cdots, m)$.

By Theorem 5, we have the following result for this algorithm.

Theorem 6. *If* u^h *is the solution obtained by Algorithm B1, then*

$$|||u_h - u^h|||_{1,\Omega} \lesssim H^{2r}|u|_{r+1,\Omega} \text{ and } |||u - u^h|||_{1,\Omega} \lesssim (h^r + H^{2r})|u|_{r+1,\Omega}.$$

We note that the approximations obtained by Algorithms A1 and B1 are defined piecewise and they are in general discontinuous and, that $\|u_h - u^h\|_{0,\Omega}$ does not in general have higher order than $|||u_h - u^h|||_{1,\Omega}$. We point out that some further modifications of these algorithms can achieve: (1) smoothing u^h to obtain a global $H^1(\Omega)$ approximation; (2) improving $\|u_h - u^h\|_{0,\Omega}$ to be of higher order, cf. [XZ98].

REFERENCES

[Ada75] Adams R. (1975) *Sobolev Spaces*. Academic Press Inc., New York.

[CL91] Ciarlet P. and Lions J. (1991) *Handbook of Numerical Analysis, Vol.II, Finite Element Methods (Part I)*. Elsevier Science Publisher, North-Holland.

[NS74] Nitsche J. and Schatz A. (1974) Interior estimates for Ritz-Galerkin methods. *Math. Comp.* 28: 937–955.

[SW77] Schatz A. and Wahlbin L. (1977) Interior maximum-norm estimates for finite element methods. *Math. Comp.* 31: 414–442.

[SW95] Schatz A. and Wahlbin L. (1995) Interior maximum-norm estimates for finite element methods, Part II. *Math. Comp.* 64: 907–928.

[Wah91] Wahlbin L. (1991) Local behavior in finite element methods. In *Handbook of Numerical Analysis, Vol. II, Finite Element Methods (Part 1) (P.G. Ciarlet and J.L. Lions, eds.)*, pages 355–522. North-Holland.

[Wah95] Wahlbin L. (1995) *Superconvergence in Galerkin Finite Element Methods*. Springer.

[Xu92a] Xu J. (1992) Iterative methods by space decomposition and subspace correction. *SIAM Review* 34: 581–613.

[Xu92b] Xu J. (1992) A new class of iterative methods for nonselfadjoint or indefinite problems. *SIAM J. Numer. Anal.* 29: 303–319.

[Xu94] Xu J. (1994) A novel two-grid method for semilinear equations. *SIAM J. Sci. Comp.* 15: 231–237.

[Xu96] Xu J. (1996) Two-grid discretization techniques for linear and nonlinear pdes. *SIAM J. Numer. Anal.* 33: 1759–1777.

[XZ98] Xu J. and Zhou A. (1998) Local and parallel finite element algorithms based on two-grid discretizations. *Math. Comp.* 68.

[ZLSL98] Zhou A., Liem C., Shih T., and Lu T. (1998) Error analysis on biparameter finite elements. *Comput. Methods Appl. Mech. Engrg.* 158: 329–339.

Part II

Algorithms

Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

17

A Nonoverlapping Characteristic Domain Decomposition Method for Unsteady State Advection-Diffusion Equations

Mohamed Al-Lawatia[1], Hong Wang [2]

Introduction

Advection-diffusion partial differential equations (PDEs) arise in petroleum reservoir simulation, subsurface contaminant transport and remediation, and many other applications. These problems typically exhibit solutions with moving steep fronts within some relatively small regions, where important chemistry and physics take place. Furthermore, an identifying feature of these applications is the presence of extremely large scale fluid flows coupled with transient transport of physical quantities such as pollutants, chemical species, radionuclides, and temperature. Consequently, an extremely refined global mesh is not feasible due to the excessive computational and storage cost. Domain decomposition techniques prove to be a feasible and powerful approach for the solution of these problems, because they allow a significant reduction of the size of the problems and the use of different physical or numerical models on different subdomains to model fluid flows more accurately. Furthermore, they are easily parallelizable.

However, many domain decomposition methods that work well for elliptic and

1 Department of Mathematics and Statistics, Sultan Qaboos University, P.O. Box 36, Al-Khod Postal Code 123, Muscat, Sultanate of Oman.
2 Department of Mathematics, University of South Carolina, Columbia, South Carolina 29208, USA.
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

parabolic PDEs [SBG96] can perform very poorly for advection-diffusion PDEs. For example, the well known Dirichlet-Neumann algorithm [BW86, MQ89], which assigns a Dirichlet condition to one subdomain and a Neumann boundary condition to its adjacent subdomain, and has been successfully applied to solve diffusion dominated problems, could generate nonphysical layers at each iteration. The fundamental reason is that these methods do not necessarily take into account the advection dominance or the hyperbolic nature of advection-diffusion PDEs. In particular, even though the Dirichlet-Neumann matching conditions are mathematically correct, they might not respect the hyperbolic limit of advection-diffusion problems. The errors generated at subdomain interfaces are then propagated into the interior domain and could destroy the accuracy of the solutions on the entire domain.

Extensive research has been carried out on developing domain decomposition methods for advection-diffusion problems. Cai [Cai91, Cai94] developed multilevel additive and multiplicative Schwarz preconditioners for parabolic and unsteady state advection-diffusion PDEs. On the other hand, the Adaptive Dirichlet Neumann (ADN) and Adaptive Robin Neumann (ARN) nonoverlapping domain decomposition methods introduced in [GGQ96, Cic96, Tro96] choose interface conditions to be adapted to the local flow direction. These methods prevent the rise of artificial layers at subdomain interfaces as the advection becomes dominant. While these methods could solve the discrete algebraic systems with strongly nonsymmetric coefficient matrices fairly efficiently at each time step, the underlying numerical methods used in these domain decomposition algorithms are standard finite element/difference/volume methods or upwinding methods with temporal discretization in time. It is well known that standard finite element/difference/volume methods tend to generate solutions with serious non-physical oscillations while upwinding methods often produce solutions with excessive numerical dispersion and grid orientation effect when applied for solving time-dependent advection-diffusion PDEs, unless the spatial grids and time steps are chosen small enough such that the mesh Peclet number is around one and the Courant number is less than or equal to one [CRHE90, WALT97, WDEESM].

Characteristic methods (e.g. the modified method of characteristics by Douglas and Russell [DR82]) effectively solve the advective component by a characteristic tracking algorithm and treat the diffusive term separately. These methods symmetrize the governing PDEs and generate accurate numerical solutions even if large time steps are used. Unfortunately, many characteristic methods fail to conserve mass and have difficulty in treating boundary fluxes when characteristics intersect the boundary of the domain. This is one of the reasons why the few characteristic domain decomposition methods developed so far are overlapping domain decomposition methods [TJDE97]. The Eulerian-Lagrangian localized adjoint method (ELLAM), which was first presented in [CRHE90] by Celia, Russell, Herrera, and Ewing, provides a general characteristic solution procedure for the solution of advection-diffusion PDEs with general boundary conditions in a mass conservative manner. Thus, ELLAM overcomes the principle shortcomings of previous characteristic methods while maintaining their numerical advantages. Our previous work [WALT97, WDEESM] also show that ELLAM schemes generate accurate solutions even if very large time steps and spatial grids are used, and often outperform many well received and widely used numerical methods.

In this paper we present an ELLAM-based, nonoverlapping characteristic domain decomposition method for unsteady state advection-diffusion equations. With the standard Dirichlet-Neumann interface condition, with which many domain decomposition methods are known to generate poor solutions for advection-diffusion PDEs, our domain decomposition method produces accurate and stable solutions that are free of artifacts even if large time steps are used in the simulation. Numerical experiments are presented to show the strong potential of the method.

An Underlying Numerical Method

In this section we briefly outline an underlying Runge-Kutta Eulerian-Lagrangian localized adjoint method (RKELLAM) [WALT97, WDEESM] for the model problem

$$u_t + (V(x,t)u - D(x,t)u_x)_x = f(x,t) \qquad x \in (a,b),\ t \in (0,T], \tag{1}$$

with an initial condition and any combination of Dirichlet, Neumann, or flux boundary conditions at the two boundary points $x = a$ and $x = b$. $V(x,t)$ is the velocity field and $D(x,t)$ is the diffusion coefficient, both of which are assumed positive.

Let I and N be two positive integers, we define a space-time partition: $x_i = a + i\Delta x$ $(0 \le i \le I)$ with $\Delta x = (b-a)/I$ and $t_n = n\Delta t$ $(0 \le n \le N)$ with $\Delta t = T/N$. Multiplying Eq. (1) by a test function w that vanishes outside $[a,b] \times (t^n, t^{n+1}]$ and integrating the resulting equation by parts we obtain

$$\begin{aligned}
&\int_a^b u(x,t^{n+1})w(x,t^{n+1})\,dx + \int_{t^n}^{t^{n+1}}\int_a^b Du_x w_x\,dxdt \\
&\quad + \int_{t^n}^{t^{n+1}} (Vu - Du_x)w\Big|_a^b\,dt - \int_{t^n}^{t^{n+1}}\int_a^b u(w_t + Vw_x)\,dxdt \\
&= \int_a^b u(x,t^n)w(x,t^n_+)\,dx + \int_{t^n}^{t^{n+1}}\int_a^b fw\,dxdt,
\end{aligned} \tag{2}$$

where $w(x,t^n_+) = \lim_{t\to t^n_+} w(x,t)$, which takes into account the fact that $w(x,t)$ is discontinuous in time at time t^n.

Based on the weak form (2), we derive a RKELLAM scheme through the following steps: (i) We approximate the characteristic curves of Eq. (1) by a second-order Runge-Kutta approximation. (ii) We define the test functions w to be the standard hat functions on $[a,b]$ at time t^{n+1} and at the outflow boundary $\{b\}\times[t^n,t^{n+1}]$, and extend w constant along the characteristics into the space-time strip $[a,b]\times[t^n,t^{n+1}]$. (iii) We incorporate these test functions into the weak form (2) and evaluate the temporal integrals in the second terms on both sides of Eq. (2) by the trapezoidal quadrature along the characteristics, leading to terms defined on $[a,b]$ at the time levels t^n and t^{n+1} as well as on the outflow boundary $\{b\}\times[t^n,t^{n+1}]$. (iv) We incorporate the inflow and outflow boundary conditions into the third term on the left-hand side of weak form (2). (v) We approximate the solution u in Eq. (2) by a piecewise linear trial function on $[a,b]$ at time t^{n+1} and at the outflow boundary $\{b\}\times[t^n,t^{n+1}]$. (vi) The last (adjoint) term on the left-hand side of Eq. (2) is dropped since it only measures the error of characteristic tracking which is within the desired order.

With the prescribed inflow and outflow boundary conditions as well as the solution $u(x,t^n)$ which is known from the computations at the previous time level, the derived

RKELLAM scheme solves for $u(x, t^{n+1})$ and $u(b, t)$ for $t \in [t^n, t^{n+1}]$. The RKELLAM scheme also symmetrizes the governing equation (1), generates accurate numerical solutions even if large time steps are used, and conserves mass. Due to the length constraint, we omit the detailed derivation of the RKELLAM scheme. We refer readers to [WALT97, WDEESM] for details of RKELLAM and for its comparison with many well received and widely used methods that shows the strength of this method.

A Nonoverlapping Characteristic Domain Decomposition Method

In this section we derive a nonoverlapping characteristic domain decomposition method for the model problem (1), based on the RKELLAM scheme in the previous section. We choose the standard Dirichlet-Neumann matching condition at subdomain interfaces, despite that many domain decomposition methods with the Dirichlet-Neumann matching condition generate poor solutions for advection-diffusion PDEs. The numerical experiments in the next section show that our method generates accurate and stable solutions that are free of artifacts. This is due to the fact that the underlying RKELLAM scheme symmetrizes the governing equation (1). In other words, the governing equation (1) can be written as a parabolic equation without any advective term along the characteristics. This is why our method is free of the problems other methods suffer from. The detailed method is described below

Partition of the Domain

Decompose the spatial domain $\Omega = [a, b]$ into a union of $2M$ subdomains

$$\Omega = \bigcup_{i=1}^{2M} \Omega^{(i)}, \qquad \Omega^{(i)} = [d_{i-1}, d_i], \quad i = 1, 2, \ldots, 2M \tag{3}$$

with

$$a = d_0 < d_1 < d_2 < \ldots < d_{2M} = b. \tag{4}$$

for $n = 0, 1, \ldots, N-1$ **do**

Initialize the iteration parameter $l = 0$.
if ERROR $>$ TOLERANCE **then**
$l = l + 1$.

L1. **for** $i = 1, 3, \ldots, 2M-1$ **do**

A. Use the prescribed inflow boundary condition at $x = a$ for $\Omega^{(1)}$ and the following relation

$$u^{(l)}(d_{i-1}^{+}, t) = u^{(l-1)}(d_{i-1}^{-}, t), \qquad i = 3, 5, \ldots, 2M-1 \tag{5}$$

to define an artificial inflow Dirichlet boundary condition for the subdomain $\Omega^{(i)}$ $(i = 3, 5, \ldots, 2M-1)$. Here $u(d_{i-1}^{-}, t)$ and

$u(d_{i-1}^{+},t)$ represent the left- and right-limits of u at the point d_{i-1}, respectively, and $u^{(0)}(d_{i-1}^{-},t)$ is defined by

$$u^{(0)}(d_{i-1}^{-},t) = u(d_{i-1}^{*}(t),t^{n}), \qquad i = 3,5,\ldots,2M-1, \quad (6)$$

where $d_{i-1}^{*}(t) \in \Omega^{(i-1)}$ is the point at time level t^n which the point (d_{i-1},t) backtracks to.

B. Use the relation

$$u_x^{(l)}(d_i^{-},t) = u_x^{(l-1)}(d_i^{+},t), \qquad i = 1,3,\ldots,2M-1 \quad (7)$$

to define an artificial outflow Neumann boundary condition for the subdomain $\Omega^{(i)}$ $(i = 1,3,\ldots,2M-1)$. Here $u_x^{(0)}(d_i^{+},t)$ is defined by

$$u_x^{(0)}(d_i^{+},t) = u_x(d_i^{*}(t),t^{n}), \qquad i = 1,3,\ldots,2M-1, \quad (8)$$

where $d_i^{*}(t) \in \Omega^{(i)}$ is the point at time level t^n which the point (d_i,t) backtracks to.

C. With the inflow and outflow boundary conditions introduced in Steps L1.A and L1.B, use the RKELLAM scheme presented in Section 17 to solve equation (1) on the *odd numbered* subdomains $\Omega^{(i)}$ $(i = 1,3,\ldots,2M-1)$ in parallel, yielding the l-th iterative solution $u^{(l)}(x,t^{n+1})$ on $\Omega^{(i)}$ and $u^{(l)}(d_i^{-},t)$ on the outflow boundary of $\Omega^{(i)}$ for all the *odd numbered* subdomains $\Omega^{(i)}$ $(i = 1,3,\ldots,2M-1)$.

end

L2. **for** $i = 2,4,\ldots,2M$ **do**

A. Use the following relation

$$u^{(l)}(d_{i-1}^{+},t) = u^{(l)}(d_{i-1}^{-},t), \qquad i = 2,4,\ldots,2M \quad (9)$$

to define an artificial inflow Dirichlet boundary condition for the subdomain $\Omega^{(i)}$ $(i = 2,4,\ldots,2M)$.

B. Use the prescribed outflow boundary condition at $x = b$ for the subdomain $\Omega^{(2M)}$ and the relation

$$u_x^{(l)}(d_i^{-},t) = u_x^{(l)}(d_i^{+},t), \qquad i = 2,4,\ldots,2M-2 \quad (10)$$

to define an artificial outflow Neumann boundary condition for the subdomain $\Omega^{(i)}$ $(i = 2,4,\ldots,2M-2)$. Here

$$\begin{aligned} u_x^{(l)}(d_i^{+},t) &= u_x(d_i^{*}(t),t^{n})\frac{t^{n+1}-t}{\Delta t} \\ &\quad + u_x^{(l)}(d_i^{**}(t),t^{n+1})\frac{t-t^{n}}{\Delta t}, \end{aligned} \quad (11)$$

where $d_i^{**}(t) \in \Omega^{(i+1)}$ is the point at time level t^{n+1} which the point (d_i,t) tracks forward to.

C. With the inflow and outflow boundary conditions introduced in Steps L2.A and L2.B, use the RKELLAM scheme presented in Section 17 to solve equation (1) on the *even numbered* subdomains $\Omega^{(i)}$ $(i = 2, 4, \ldots, 2M)$ in parallel, yielding the l-th iterative solution $u^{(l)}(x, t^{n+1})$ on $\Omega^{(i)}$ and $u^{(l)}(d_i^-, t)$ on the outflow boundary of $\Omega^{(i)}$ for all the *even numbered* subdomains $\Omega^{(i)}$ $(i = 2, 4, \ldots, 2M)$.

end

else

Define $u(x, t^{n+1}) = u^{(l)}(x, t^{n+1})$ on $\Omega^{(i)}$ for $i = 1, 2, \ldots, 2M$.

endif

end

Numerical Experiments

In this section we apply the domain decomposition method to two standard test problems to observe the performance of the method developed.

Example 1. This example considers the transport of a Gaussian pulse in a variable velocity field given by $V(x,t) = 1 + 0.01x$, the diffusion coefficient is $D(x,t) = 10^{-4}$. The spatial domain is $[a, b] = [0, 3.2]$. The initial Gaussian pulse is located at the inflow boundary $x = 0$, with a unit height and a spread or standard deviation of $\sigma = 0.0316$. In the experiments, six subdomains are used (i.e. $M = 3$)

$$[0, 3.2] = [0, 0.5] \cup [0.5, 1] \cup [1, 1.5] \cup [1.5, 2] \cup [2, 2.5] \cup [2.5, 3.2]. \tag{12}$$

$\Delta x = 0.01$ and $\Delta t = 0.02$. In the numerical experiments, four iterations are used. In Figure 1(A), the numerical solution is presented against the analytical solution at time instant $t = 0.5, 1, 1.5, 2$, and 2.5, when the numerical solution is located at the interfaces. The L_2 and L_1 norms of the absolute error at $t = 2.5$ are 4.53×10^{-4} and 1.92×10^{-4}, respectively.

Example 2. This example considers the transport of a diffused step function. All the parameters are the same as those in Example 1 with the only exception that the velocity field is $V(x,t) = 1$. The initial configuration is a box function of a unit height supported on the interval $[0.1, 0.4]$. The numerical solution is presented in Figure 1(B). The L_2 and L_1 norms of the absolute error at $t = 2.5$ are 1.33×10^{-3} and 5.80×10^{-4}, respectively.

Conclusion

In this paper we develop a nonoverlapping characteristic domain decomposition method for the solution of advection-diffusion equations, using the standard Dirichlet-Neumann matching conditions at subdomain interfaces. The numerical results show that the developed method generate accurate and stable solutions without noticeable artifacts even if large time steps are used, leading to greatly improved efficiency.

REFERENCES

[BW86] Bjørstad P. and Widlund O. (1986) Iterative methods for the solution of elliptic problems on regions partitioned into substructures. *SIAM J. Numer. Anal.* 23(6): 1093–1120.

[Cai91] Cai X.-C. (1991) Additive Schwarz algorithms for parabolic convection-diffusion equations. *Numer. Math.* 60(1): 41–61.

[Cai94] Cai X.-C. (1994) Multiplicative Schwarz methods for parabolic problems. *SIAM J. Sci Comput.* 15(3): 587–603.

[Cic96] Ciccoli M. (1996) Adaptive domain decomposition algorithms and finite volume/finite element approximation for advection-diffusion equations. *J. Sci. Comput.* 11: 299–341.

[CRHE90] Celia M., Russell T., Herrera I., and Ewing R. (1990) An Eulerian-Lagrangian localized adjoint method for the advection-diffusion equation. *Advances in Water Resources* 13: 187–206.

[DR82] Douglas J. and Russell T. (1982) Numerical methods for convection-dominated diffusion problems based on combining the method of characteristics with finite element or finite difference procedures. *SIAM J. Numer. Anal.* 19: 871–885.

[GGQ96] Gastaldi F., Gastaldi L., and Quarteroni A. (1996) Adaptive domain decomposition methods for advection dominated equations. *East-West J. Numer. Math.* 4: 165–206.

[MQ89] Marini L. and Quarteroni A. (1989) A relaxation procedure for domain decomposition methods using finite elements. *Numer. Math.* 55: 575–598.

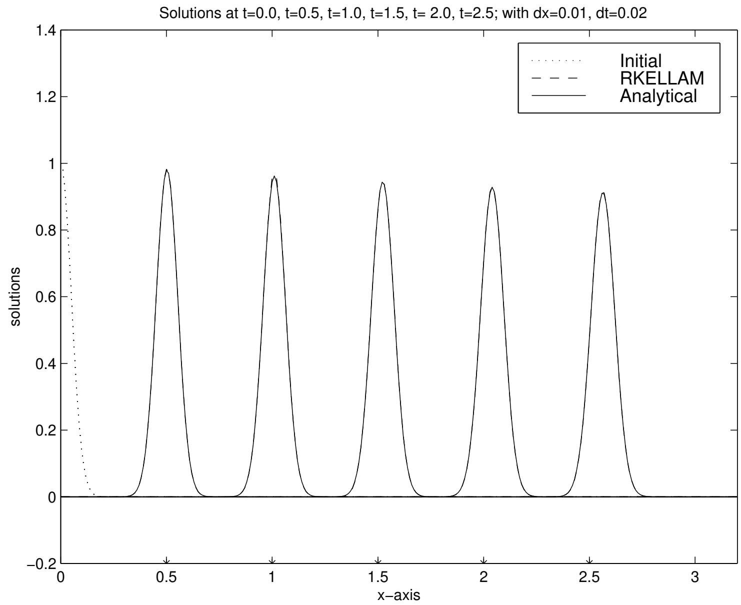
[SBG96] Smith B., Bjørstad P., and Gropp W. (1996) *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations.* Cambridge University Press.

[TJDE97] Tai X.-C., Johansen T., Dalhe H., and Espedal M. (1997) A characteristic domain splitting method. In Glowinski R., Périaux J., Shi Z.-C., and Widlund O. (eds) *Domain Decomposition Methods in Science and Engineering*, pages 317–323. John Wiley, New York. Proceedings of the Eighth International Conference on Domain Decomposition Methods.

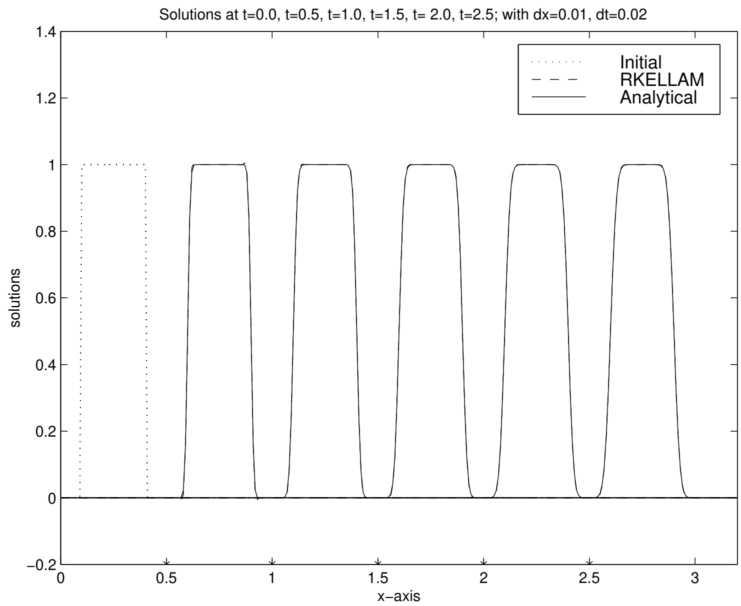
[Tro96] Trotta R. (1996) Multidomain finite elements for advection-diffusion equations. *Appl. Numer. Math.* 21: 91–118.

[WALT97] Wang H., Al-Lawatia M., and Telyakovskiy A. (1997) A Runge-Kutta characteristic method for first-order linear hyperbolic equations. *Numer. Meth. for PDEs* 13: 617–661.

[WDEESM] Wang H., Dahle H., Ewing R., Espedal M., Sharpley R., and Man S. (in press) An Eulerian-Lagrangian localized adjoint method for advection-dispersion equations in two dimensions and its comparison to other schemes. *SIAM J. Sci. Comput.* .



(A) Solutions of Example 1



(B) Solutions of Example 2

Figure 1 Solutions at time instance t= 0 (initial), 0.5, 1, 1.5, 2, 2.5

18

An Explicit Multi-Model Compressible Flow Formulation Based on the Full Potential Equation and the Euler Equations on 3D Unstructured Meshes

Xiao-Chuan Cai[1], Marius Paraschivoiu[2], & Marcus Sarkis[3]

Introduction

The development of a multi-model formulation to simulate three dimensional compressible flows on parallel computers is presented. The goal is to reduce the overall time and memory required to simulate the flow by using locally selected cheaper and more computational efficient physical models without sacrificing the global fidelity of the simulation. Our approach involves splitting the computational domain into different fluid flow regions and using the full potential model instead of the Euler or Navier-Stokes equations in regions where this approximation is valid. We show numerically that solving the full potential equation in regions of irrotational flow is not only more efficient but also improves the accuracy; avoiding any numerical generation of entropy. The main considerations addressed in this paper are the full potential and the Euler coupling and the discretization of the interface conditions between these

1 Dept. of Comp. Sci., Univ. of Colorado at Boulder, Boulder, CO 80309. cai@cs.colorado.edu
2 Dept. of Mech. and Ind. Eng., Univ. of Toronto, Toronto, Canada, M5S 3G8. *marius@mie.utoronto.ca*
3 Math. Sci. Dept., Worcester Poly. Inst., Worcester, MA 01609. msarkis@wpi.edu
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

domains. We use a fully unstructured finite volume discretization for both the full potential and the Euler equations, and the interface condition is derived by imposing the discrete conservation laws in the control volumes shared by both flow regions. 3D transonic flow simulations around a NACA0012 airfoil are investigated.

Numerical simulations of fluid flow have sufficiently matured to be considered accurate for engineering design and analysis. However, for large scale simulations, the response time remains too large for the software to be used as an interactive tool even on the lastest supercomputers. While parallel computing reduces computation time proportionally to additional computational resource, new algorithms should be constructed to perform faster on new and existing resources. In this paper we describe the initial steps for the development of a multi-model formulation to decrease the computation time of three dimensional compressible flow simulations on unstructured grids.

Compressible fluid flow simulations needed for aerodynamic applications can be modeled with different degree of sophistication. The simplest model is the full potential equation which assumes inviscid, irrotational and isentropic flows. This equation is a single second-order nonlinear differential equation that is inexpensive with respect to the execution time and the memory requirement. Validity of the full potential equation is, however, restricted. The isentropic assumption of the potential flow model leads to inaccurate physics for transonic flows with strong shocks. The next level of approximation is the Euler equations which describe the complete behavior of inviscid compressible flows. The Euler equations are a coupled system of five nonlinear differential equations of first order. Note that this set of equations involves five field variables. Finally, the Navier-Stokes equations include the viscous effects needed for accurate modeling of the boundary layer. However, these equations are not only more time consuming to solve but also require an associated mesh that is stretched and very fine in viscous regions. Nevertheless, for complex flows with separation of the boundary layer, the Navier-Stokes equations are mandatory to provide an accurate simulation. Furthermore, for high Reynolds number flows, turbulence appears and needs to be modeled.

When considering transonic flows over a wing, three regions can be identified: the boundary layer, the region around the shock, and the farfield. A multi–model formulation can be used to combine the strength of each model described above. Indeed, a multi-model formulation will take advantage of the quick computational time associated with solving the full potential equation while capturing all the important features of the flow such as boundary layers and shocks using the Navier-Stokes equations and/or the Euler equations, respectively. Furthermore, we can benefit from the extensive experience of numerical methods and software developed over the years to solve these equations separately.

Numerical techniques for the solution of the full potential equation and the Euler/Navier-Stokes equations were developed respectively in the 1970s and in the 1980s [BGP+85, HSM79, HSYB82, JBW86, PJ83]. Indeed, compressible flow around entire aircrafts have been simulated. For example, the full potential equation has been solved for a 747-200 transport configuration with wing, body, struts, and nacelles [YMB+91]. For the Euler model, calculations over a complete aircraft have been performed as early as in 1986 [JBW86]. On the other hand, accurate viscous simulations at high Reynolds number over such complex geometries require enormous computational resources. Approximate solutions, i.e., with less than adequate number of mesh points, have been performed. A Navier-Stokes prediction for the F–18 wing and fuselage is presented in [CRSC92]. A discussion of the drastic difference in computational cost related to the choice of models can be found in [Jam97].

However only recently have there been interests to couple these solvers to reduce the computational cost, to reduce the memory requirement and to improve the accuracy of the solution. Certainly, boundary layer coupling or thin layer Navier-Stokes coupling have been widely used but such approaches do not quite include all the physics we intend to incorporate [YMB+90]. For some mathematical description of coupling heterogeneous models for compressible flows we refer the readers to [QV99]. For three dimensional flows, it is shown in [BSBT97, SBT93] that the computation cost can be reduced by a factor of two for a Navier-Stokes/full potential coupling. Their formulation is based on a structured grid discretization where the full potential equation is solved using a finite difference method and the Navier-Stokes equations are solved either with a finite difference or a finite volume discretization. The saving is justified by the fact that two third to one half of the cells are outside the Navier-Stokes region. In general, the cost of the full potential solver can be considered negligible compared to the Navier-Stokes solver. Note that each region is solved alternatively, similar to a subdomain iterative method.

Our formulation differs from [BSBT97, SBT93] by providing a general finite volume approach and therefore ensure that the mass will also remain conserved at the discrete level. This approach also has the advantage of being readily extended to a coupled *implicit* scheme. While in [BSBT97, SBT93] each region is solved separately, we are expecting to improve convergence by solving the coupled system simultaneously. In addition, an unstructured discretization of the computational domain provides more flexibility to mesh complex geometry and for adaptive control of the numerical error. Lastly, a parallel version is implemented to obtain the reasonable execution time.

In this paper we address the initial step of this research. We first investigate the coupling between the full potential equation and the Euler equations. In addition to the computational savings, solving the full potential equation in the vicinity of the stagnation point is more accurate by avoiding the numerical entropy generated by Euler solvers at low Mach numbers. Furthermore, the full potential solver is less sensitive to the quality of the elements. Both the Euler and the Navier-Stokes solvers calculate convective fluxes through edges. When the surface of the control volume is different from the perpendicular surface of the edges, a numerical error is created. This error does not appear in the full potential discretization because fluxes are constants in each element. As mentioned above, a finite volume formulation is adopted to adequately interface these different solvers. An explicit approach is first considered to validate the spatial discretization but also as a precursor to the implicit implementation [PCS+99].

For simplicity, we address steady flows. For unsteady flows, it is required to have the temporal derivative of the density which is difficult to include in our explicit scheme because this derivation involves potential values of the neighboring control volumes. However, in an implicit solver such discretization can be easily included. As already mentioned, to obtain the most accurate flow simulation, the full Navier-Stokes equations need to be included in our multi-model formulation.

A more urgent objective, however, is to address some algorithmic extensions. First, we intend to develop an implicit scheme which is essential for large scale simulations [PCS+99]. The convergence rates reported with implicit schemes are much faster, in particular when using methods such as the overlapping Schwarz preconditioned GMRES methods [CFS98]. In our formulation, the Jacobian matrix used in the implicit approach will include the spatial discretization of two or more equations. It is expected that such system is not positive definite and thus it is not clear how it can be solved efficiently or which preconditioner should perform well. Second, we plan to develop a procedure to automatically position the interface between the different computational domains based on the existing field variables (i.e., dynamic zonal configuration). Third, we need to consider load balancing for parallel computations, in particular when dynamic zonal configuration procedures will be used. Recall that different partial differential equations are solved in different regions but each equation does not require the same number of operations. One palliative is to decompose each region into subregions equal to the number of processors. By using such an approach, we can allocated one subregion of each type to each processor. Finally, we expect that the outcome of this research will lead to a dramatic reduction in computational time and memory resources to allow faster simulation of compressible flows including turbulent viscous effect, in particular for external aerodynamics applications.

In this paper we focus on the description of the two-model formulation. Section 2 describes the explicit full potential solver and the Euler solver with more emphasis on the full potential solver. In Section 3, we briefly introduce the coupled solver and compare two types of interface conditions with overlap and without overlap. To demonstrate the feasibility of our approach we solve a transonic flow over a NACA0012 airfoil at zero angle of attack which is analyzed in Section 4. We conclude with remarks and extensions in Section 4.

Simulation of compressible flows

Our interest lies in the numerical simulation of three dimensional compressible invisid flows. We assume that there is no external force or heat transfer. As described above, these flows can be modeled with the Euler equations or with the full potential equation for the particular case when the irrotational and isentropic flows assumption is satisfied. For simplicity of presentation, all the descriptions given in the paper are based on the first order finite volume discretization, and the extension to the $2nd$ order discretization is easy. All the numerical results presented in Section 4 are for, however, the $2nd$ order discretization.

The governing equations

Let $\Omega \subset \Re^3$ be the computational flow domain and Γ its boundary. The conservative form of the Euler equations is given by

$$\frac{\partial U}{\partial t} + \nabla \cdot F(U) = 0. \qquad (1)$$

Here U contains the conservative variables, i.e., $U = (\rho, \rho u, \rho v, \rho w, \rho E)^T$. The explicit definitions of $F()$ can be found on page 87 of [Hir90]. When the flow is irrotational, there exists a potential variable Φ satisfying the full potential equation

$$\frac{\partial \rho(\Phi)}{\partial t} + \nabla \cdot G(\Phi) = 0, \tag{2}$$

where $G(\Phi) = \rho \nabla \Phi$ and

$$\nabla \Phi = (u, v, w)^T. \tag{3}$$

In the rest of the paper, we shall refer to U as the Euler variable, which is a vector, and Φ as the full potential variable, which is a scalar.

By appealing to the isentropic flow assumption we can write the density ρ as a nonlinear function of the potential, such as

$$\rho(\Phi) = \rho_\infty \left(1 + \frac{\gamma - 1}{2} M_\infty^2 \left(1 - \frac{||\nabla \Phi||_2^2}{q_\infty^2}\right)\right)^{1/(\gamma-1)}. \tag{4}$$

There are two types of boundaries that bound the computational domain for external flows past bodies or obstacles: the farfield boundary and the solid wall boundary. On the solid wall boundary, Γ_w, the normal velocity, $\mathbf{v}_n$, is zero, since no mass crosses the boundary. On the farfield, Γ_∞, we impose an uniform free-stream state defined by the following parameters: the density, ρ_∞, the velocity vector, $\mathbf{v}_\infty$, the pressure, p_∞, and the Mach number M_∞. These conditions are given by

$$\mathbf{v}_n = 0, \quad \text{on} \quad \Gamma_w \tag{5}$$

and

$$\rho_\infty = 1, \quad \mathbf{v}_\infty = \begin{pmatrix} \cos(\alpha) \times \cos(\theta) \\ \sin(\theta) \\ \sin(\alpha) \times \cos(\theta) \end{pmatrix}, \quad p_\infty = \frac{1}{\gamma M_\infty^2} \quad \text{on} \quad \Gamma_\infty, \tag{6}$$

where α and θ are the angles of the flow direction (the angle of attack and the yaw angle, respectively).

The Euler solver

To solve the Euler equations, we take advantage of an existing code based on an unstructured finite volume discretization of the convective fluxes [FL94, FLCL95]. The computational flow domain is divided into tetrahedrons to provide maximum flexibility for tessellating complex geometries. Euler variables are located at the vertices of the elements. This code uses a second order flux discretization based on the MUSCL (Monotonic Upwind Scheme for Conservative Laws) scheme [Lee79]. A classical forward Euler method for time integration with a local time step size is chosen. Recall that we are only interested in solving the steady state and a simple time integration scheme offers more flexibility to be coupled with the full potential solver. The local time step size $\Delta t^n_{\tau_i^c}$ is defined for each control volume τ_i^c (with characteristic size $||h_i^c||$) by

$$\Delta t^n_{\tau_i^c} = ||h_i^c|| \frac{\text{CFL}}{C^n_{\tau_i^c} + ||\mathbf{U}^n_{\tau_i^c}||_2}, \tag{7}$$

where CFL is a preselected positive number, $C^n_{\tau_i^c}$ is the sound speed and $\mathbf{U}^n_{\tau_i^c}$ is the velocity vector at the nth time step.

The spatial discretization of the boundary condition, (5)-(6), is obtained using a non-reflecting version of the flux–splitting scheme [FL94].

The full potential solver

A new finite volume full potential solver is developed to adequately interface with the existing Euler solver. Therefore, the same control volume is used and only the flux calculations are different. We describe now the spatial discretization of the mass flux required in this scheme.

The spatial discretization

The integral form of the full potential equation for the discrete volume τ_i^c is simply

$$\int_{\tau_i^c} \nabla \cdot G(\Phi) dA = 0. \tag{8}$$

Note that the sum of all τ_i^c covers the whole domain Ω, i.e., $\bar{\Omega} = \bigcup \bar{\tau_i^c}$. By analogy to the discretization of the Euler equations, the discretization here is accomplished by dividing the domain into tetrahedron elements, $\tau_{i,j}^h$. The potential variable is stored at the vertices. This choice is illustrated in Fig. 1 for two space dimensions. By using this discretization, the space of the potential solution is taken to be piecewise linear continuous functions in each element determined from the vertices values, Φ_i.

For the control volume τ_i^c associated with the dual mesh, we can write the discrete form of (8) as

$$\int_{\tau_i^c} \nabla(\rho \nabla \Phi) dA = \int_{\partial \tau_i^c} \rho \nabla \Phi \cdot \mathbf{n} \, dS = \sum_{\tau_{i,j}^h} \rho_{i,j} (\nabla \Phi)_{i,j} \cdot \mathbf{S}_{i,j}^c, \tag{9}$$

where $\tau_{i,j}^h$ is the "triangulation" associated with the control volume τ_i^c and $\mathbf{S}_{i,j}^c = \int_{\partial \tau_i^c \cap \tau_{ij}^h} \mathbf{n} \, dA$. Here $\mathbf{n}$ is the unit outward normal vector of the surface $\partial \tau_i^c \cap \tau_{ij}^h$. Note that $\rho_{i,j}$, the discrete density, is a function of $(\nabla \Phi)_{i,j}$ which is a constant for each element $\tau_{i,j}^h$.

An explicit approach

To solve (8) we add a time dependent term which vanishes at steady state. Hence, we rewrite (8) as

$$\frac{d}{dt} \int_{\tau_i^c} \Phi dA + \int_{\tau_i^c} \nabla \cdot G(\Phi) dA = 0. \tag{10}$$

The resulting semi-discrete form of (10) is

$$\|\tau_i^c\| \frac{d\Phi_i}{dt} + G_i(\Phi) = 0, \tag{11}$$

where G_i is the discretized mass flux associated with τ_i^c as in (9).

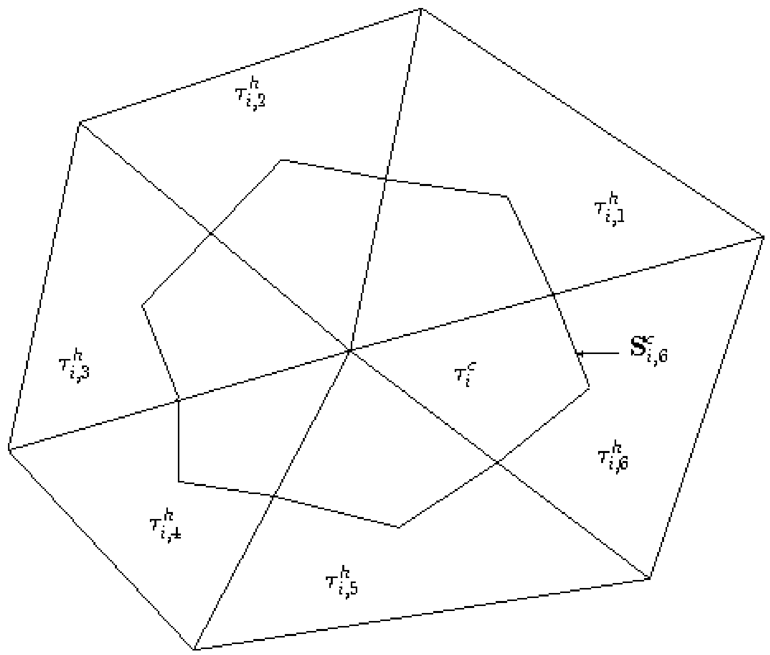


Figure 1 Two space dimensions representation of the control volume.

The system of ODE's associated with all control volumes is integrated in "time" using the forward Euler discretization. For a control volume τ_i^c, the equation is

$$\frac{\Phi_i^{n+1} - \Phi_i^n}{\Delta t^n_{\tau_i^c}} = -\frac{1}{||\tau_i^c||} G_i(\Phi^n). \tag{12}$$

The time step size $\Delta t^n_{\tau_i^c}$ is determined in the following way,

$$\Delta t^n_{\tau_i^c} = \min_{\tau^h_{i,j}} \left(C||h_j^c||^2 \right), \tag{13}$$

where $||h_j^c||$ is the characteristic length of element $\tau^h_{i,j}$ and C is a global constant. Because of the nonlinearity of this equation, it is difficult to determine C analytically. However, based on our numerical experiments, C equal to 0.4 is a good approximation.

The density upwinding scheme

For transonic flows, upwinding is required; therefore, the density is modified to add artificial compressibility. The upwinding is introduced prior to the flux calculation after which the same subsonic procedure is used. For simplicity, we describe our upwinding method for two space dimensions. Following [HB79, YMB+91], we write

$$\tilde{\rho} = \rho - \mu \mathbf{v} \cdot \nabla_{-} \rho, \tag{14}$$

where $\mathbf{v}$ is the normalized element velocity and $\nabla_{-}\rho$ is an upwind difference. In two space dimensions there are two cases to consider. Either the mass flux enter on one side (Fig. 2 a) or the mass flux enter through two sides (Fig. 2 b). It follows that the density for each case becomes

$$\tilde{\rho_j} = \rho_j + \mu \mathbf{v} \cdot \mathbf{n}_k (\rho_j - \rho_k) \tag{15}$$

$$\tilde{\rho_j} = \rho_j + \mu \mathbf{v} \cdot \mathbf{n}_k (\rho_j - \rho_k) + \mu \mathbf{v} \cdot \mathbf{n_l} (\rho_\mathbf{j} - \rho_\mathbf{l}). \tag{16}$$

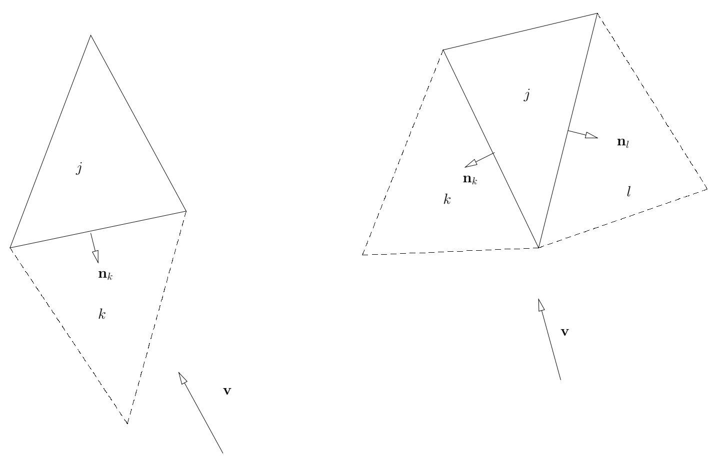


Figure 2 Upwind configurations.

The switching function, μ, is defined for each element as

$$\mu = \nu_o \max\{0, 1 - M_c^2/M^2\}, \tag{17}$$

where M is the element Mach number, M_c is a pre-selected cutoff Mach number chosen to introduce dissipation in the transonic regime. The parameter ν_o is used to increase the amount of dissipation in the supersonic elements. These parameters M_c and ν_o are selected by hand; M_c is just smaller than 1 and ν_o is usually set between 1 and 3. Additional viscosity is added by taking the switching function in each element to be the maximum value of all its immediate neighbors. We refer the readers to [CGK+98, HH82] for more details.

The boundary conditions

The full potential spatial discretization of the boundary condition is now described. On solid boundaries, Γ_w, we apply the surface flow tangency condition. We write,

$$\rho \frac{\partial \Phi}{\partial n} = 0, \tag{18}$$

for solid wall at rest. In our control volume approach, this boundary condition is identical to no flux across the solid boundary. It is, therefore, straightforward to implement; we just sum the flux across the boundary of the control volume which are in the interior of the computational domain.

For the farfield boundary, Γ_∞, we normalize the farfield flow speed, q_∞, to equal unity and we define the farfield potential as

$$\Phi_\infty = \int_\Omega q_\infty dx, \tag{19}$$

where x is the direction of the farfield flow. To enforce this Dirichlet boundary condition we impose the value of Φ_∞ in the control volumes which lie on the farfield boundary. It is also possible to replace the above condition with a more transparent condition such as imposing a mass flux associated with the free stream state. This condition is implemented on parts of Γ_∞ adjacent to the Euler domain. We recognize that these boundary conditions are simplistic and we recommend a boundary condition based on Riemann invariants [SIGO87] or on linearized Euler equations [CW98]. Furthermore, a farfield correction is required for lifting wings.

The coupled solver

The spatial computational domain, Ω, is split into two subdomains, Ω_E and Ω_Φ, wherein the Euler equations and the full potential equation are solved respectively. We denote by Γ_I the interface between Ω_E and Ω_Φ.

The formulation presented herein for the full potential is similar to the unsteady Euler formulation for finite volume. In fact, we can define W as the *simulation variable*, which represents either U or Φ. A general formulation can thus be constructed. In the future version of our software implementation, W will be a pointer and its true value and size are determined while the flow is being calculated. We assume that W is the solution of the equation

$$\frac{\partial W}{\partial t} + \nabla \cdot P(W) = 0, \tag{20}$$

where the flux function P is called the *model function* that equals to either F or G. The decision to choose a specific model will be made for each subdomain. However, the main part of this coupled solver is the treatment of the conservation law at the interface boundary.

The interface boundary conditions

There are several issues related to the interface boundary such as location, formulation and discretization. In this paper we mainly describe the discretization. We report on two different domain partitioning approaches: the overlapping and non-overlapping partitioning.

An overlapping partition

Consider the interface between the full potential domain and the Euler domain presented in Fig. 3. This interface is located between tetrahedra. Therefore the control volume associated with the nodes on this interface are shared between both the full potential domain and the Euler domain. These control volumes are considered as the overlapping region. Conservation laws for the Euler equations as well as the conservation laws for the full potential equation are forced on this control volume. First, we describe discretization of the conservation laws associated with the Euler equations, i.e., the conservation of mass, momentum and energy. The fluxes across the surface of the control volume that lies in the Euler domain (Ω_E) can be readily calculated. However, we require to convert the full potential variable to the Euler variable to calculate the fluxes across the surface that lies in the potential domain (Ω_Φ). To this end we use the potential to Euler variable transfer function (**Appendix A**) to calculate, at vertex, k and l for example, the momentum and the energy. These vertex values are obtained by the volume weighted averaging of the density and the velocity in the elements surrounding this vertex. Note that, the density and the velocity, $\nabla\Phi$, are constant in the elements laying in the full potential domain. Therefore, these Euler fluxes calculations depend on the Euler variable on one side and on the potential converted to mass, momentum and energy on the other side. We introduce an operator Q related to the transfer of the potential variable to the Euler variable defined as

$$U = Q(\Phi). \tag{21}$$

The interface condition for the Euler solver becomes

$$\int_{\Gamma_E} F(U)\cdot \mathbf{n}_E dS + \int_{\Gamma_\Phi} F(Q(\Phi))\cdot \mathbf{n}_\Phi dS = 0, \tag{22}$$

where the subscript E and Φ refer to the Euler and the potential segments of the control volume, respectively.

Second, we present the conservation of mass for the same control volume required for the full potential solver. To be more precise the conservation of mass is written as

$$\int_{\Gamma_E} \rho V\cdot \mathbf{n}_E dS + \int_{\Gamma_\Phi} \rho\nabla\Phi\cdot \mathbf{n}_\Phi dS = 0, \tag{23}$$

where ρV is the first component of the Euler flux vector. These two integrals can be discretized into sums over the edges of the control volume, such as,

$$\sum_{\tau^h_{i,j}} \rho V\cdot \mathbf{S}^E_{i,j} + \sum_{\tau^h_{i,j}} \rho_{i,j}(\nabla\Phi)_{i,j}\cdot \mathbf{S}^\Phi_{i,j} = 0, \tag{24}$$

where $\mathbf{S}_{i,j}$ is the surface intergal of the outward normal vector along the j surface of the control volume τ^c_j associated with each element $\tau^h_{i,j}$. Note that in this approach we have over-determined the conservations laws. Indeed, the same control volume will satisfy the conservation of mass for both the Euler equations and for the full potential equation.

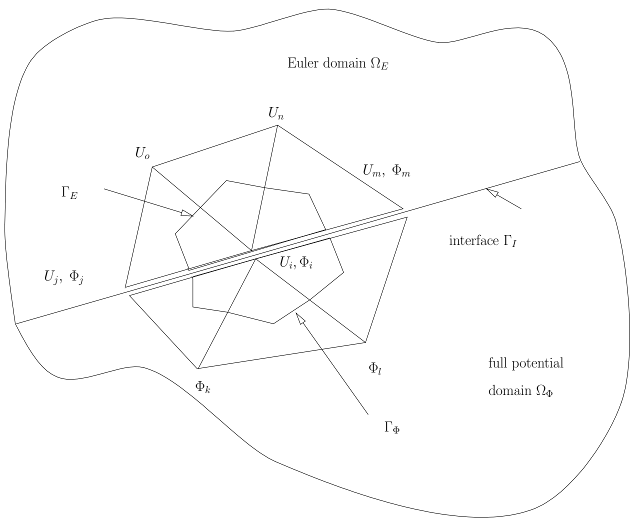


Figure 3 Overlapping interface of the Euler and full potential domains.

A non-overlapping partition

The non-overlapping domain permits a proper discretization of the equations without over-determining the conservation laws. In this approach, control volumes are flagged either for solving the full potential equation or for solving the Euler equations. The location of the interface, therefore, lies between the control volumes as in the two dimensional illustration in Fig. 4. The interface condition for the Euler solver is given by (22) where Γ_Φ is the portion of the control volume that lies on the interface Γ_I. The Euler fluxes through this control surface are calculated using the same procedure as in the interior of the Euler domain which is based on solving a Riemann problem defined on an edge using the two end–point nodal values. Because this edge crosses the interface, it contains the potential variable at one end and the Euler variable at the other. To use the same procedure, we convert the potential variable to the Euler variable. To convert Φ to U at one vertex, we use $\nabla\Phi$ in the tetrahedra surrounding that vertex. When the tetrahedron intersects the interface then an approximation of the velocity is used instead of $\nabla\Phi$.

The mass flux balance for the full potential control volume that lies on the interface is given by (23) where Γ_E is now the portion of the control volume that lies on the interface. The term ρV is the type first term in the Euler flux vector which is related to the conservation of mass. Clearly, this approach guarantees that the mass fluxes at

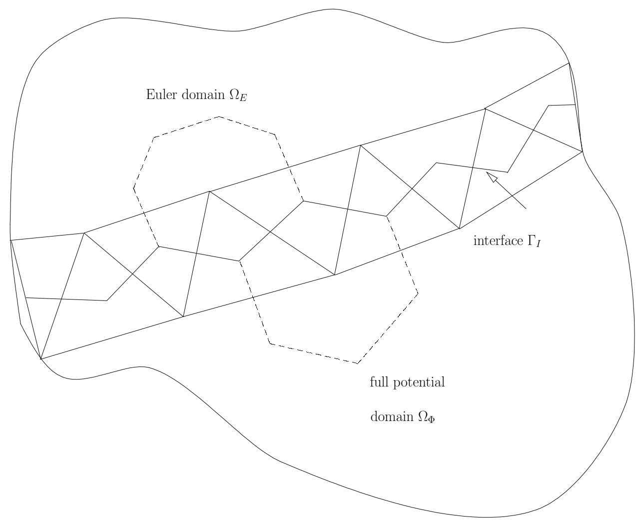


Figure 4 Non-overlapping interface of the Euler and full potential domains.

the interface are conserved.

Computational results

Transonic flow passing a NACA0012 airfoil

In this section we present heterogeneous full potential and Euler solutions in three space dimensions. We test our scheme for a two dimensional flow over a NACA0012 airfoil at $M_\infty = 0.8$ in a three dimensional computational domain. However, only half of the geometry is required for this symmetric flow. The computational domain is such that Ω is a rectangle domain where an upper surface of a NACA0012 is located on the bottom face as presented in Fig. 5. The boundary conditions of this problem are as follows: on Γ_1, Γ_2, and Γ_3 we impose farfield conditions; on Γ_4, Γ_6, Γ_7, and Γ_8 we impose the non-penetration condition for symmetry and on Γ_5 we impose the solid wall condition. Note that, a non-penetration condition or a solid condition are identical. For the farfield boundary Γ_1, Γ_2, and Γ_3, the discretization of this condition differs for the full potential solver between Γ_2 where we impose Φ_∞ as a Dirichlet condition and Γ_1 and Γ_3 where we only specify a flux at the boundary. Concerning the switching function μ, the cutoff Mach number is set to $M_\infty^2 = 0.95$ and the viscous parameter is set to $\nu_o = 1.5$.

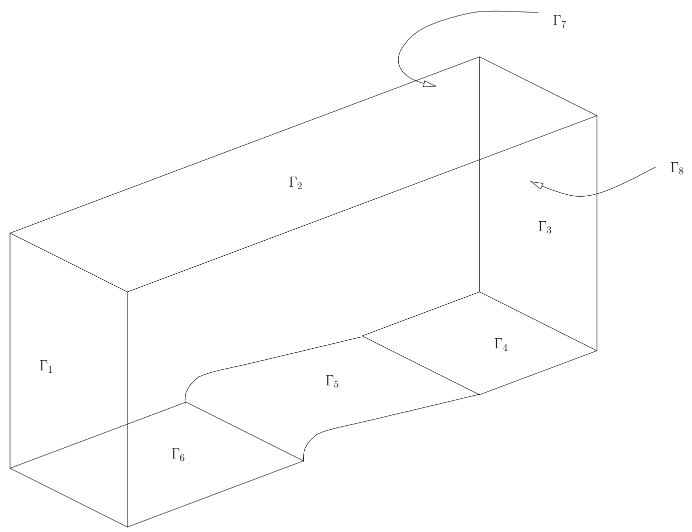


Figure 5 Computational domain for the NACA0012 geometry.

We present in Fig. 6 the mesh associated with the computational domain and the domain partitioning of this mesh into the Euler domain (bottom) and the full potential domain (top). The gap between the domains is artificially added for visual purpose. In our current implementation, the interface is hand-picked, and with such a partition the shock is contained completely in the Euler domain. Note that this mesh is also partitioned into eight subdomains for parallel processing. Each of the subdomains have more or less the same number of mesh points even thought ultimately the node distribution per processor should take into account the type of solver used. We show the Mach number contours for the overlapping and the non-overlapping partitions in Fig. 7 and Fig. 8, respectively. Note the smooth transition of the iso–contours between the Euler and full potential domains. The interface of the Euler and the full potential domains lies at the intersection of different processors. This interface is presented in Fig. 6. The flow domain is discretized into 16,200 control volumes and the solution was obtained after 80,000 explicit iterations.

We also report on the pressure coefficient distribution over the airfoil, Fig. 9. The solid line represents the Euler solution, the dash lines represent the full potential solution and the stars represent the coupled solution. Based on our numerical experience, we report that the full potential solution is very sensitive to the farfield boundary location. Clearly, the farfield boundary in our computational domain is still very close to the airfoil. On the other hand, the Euler solution is far less sensitive to the farfield boundary location. This hidden cost of the full potential should be accounted for when evaluating the cost reduction. The shock location in the full potential solution is therefore sensitive to the boundary location. However, in all the runs we have performed, we note that the coupled solution obtains a shock which is located upwind of the potential shock. The shock location also depends on the location of the interface between the two domains. In this case, we report that the shock in the coupled solution is at the same location as the Euler shock on the surface of the airfoil but differs slightly in the interior.

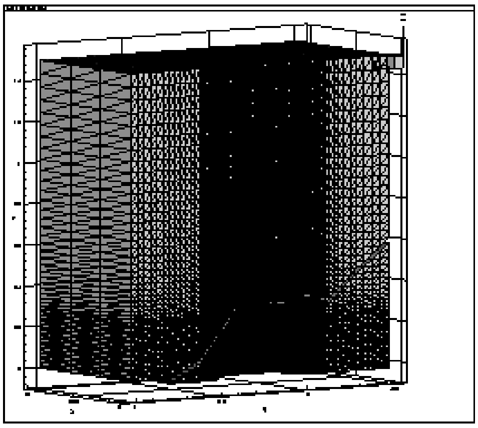


Figure 6 Computational mesh and domain decomposition for the NACA0012 airfoil.

Remarks

Before concluding, we want to report on our numerical experience of the full potential and the Euler models coupling.

First, we observe that solutions are similar for the overlapping or the non-overlapping partitioning. The solution is smooth across the interface and the convergence rate is not significantly affected by the choice of overlapping or non-overlapping approaches. The main difference is the added computational cost of calculating the Euler fluxes for the overlapping control volume and the storage associated with the existence of the overlapping control volume which has both the Euler variable and the potential variable. On the other hand the overlapping partitioning is required in implicit solutions using an overlapping Schwarz algorithm.

Second, for simplicity of presentation, the discretization presented herein has focused on the first order scheme for the Euler equations and for the interface discretization. For higher order schemes, we convert from potential to the Euler variable not only the

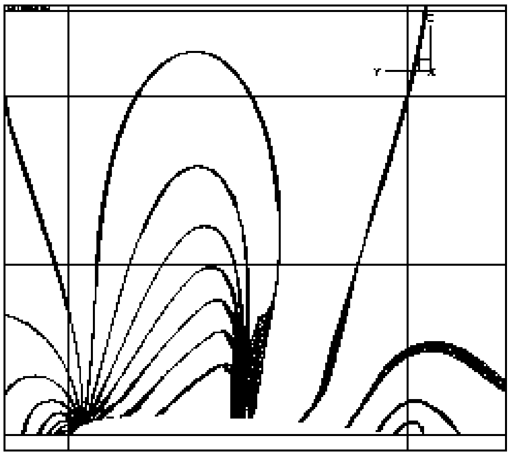


Figure 7 Mach number contours for the NACA0012 airfoil at $M_\infty = 0.8$ for overlapping domains.

vertex values of the control volume located on the interface but also the vertices of the neighbors such that the evaluation of the gradient of the solution at each vertex can be calculated. Indeed, the results presented in Section 18 are obtained using a second order scheme. A similar approach will be required when attempting the coupling with the Navier-Stokes equations.

Lastly, physical solutions for the full potential flow over a wing are obtained by imposing the Kutta condition; that the flow leaves the trailing edge smoothly. For a full potential solver such a condition is enforced by adding a jump in the potential equal to the circulation. Note that for non-lifting airfoils, as in our model problem, we do not need to enforce the Kutta condition. However, it is fortunate that the Euler solution of such flows intrinsically respects this condition. For future considerations such as lifting wings we will define the Euler domain to cover the wing and the wake region. With this partition we avoid any special treatment in the full potential domain because the full potential region does not cross the trailing edge vortex sheet.

Conclusions

In this paper we have showed the feasibility of coupling the Euler equations and the full potential equation in the simulations of three dimensional steady compressible flows. An explicit formulation was presented based on a forward Euler time integration scheme and a fully unstructured finite volume scheme for the spatial variables. Numerical results obtained on a distributed memory parallel computer were reported for a transonic flow passing a NACA0012 wing. We have also laid down the background to fully extend this formulation for multiple flow models and for different numerical

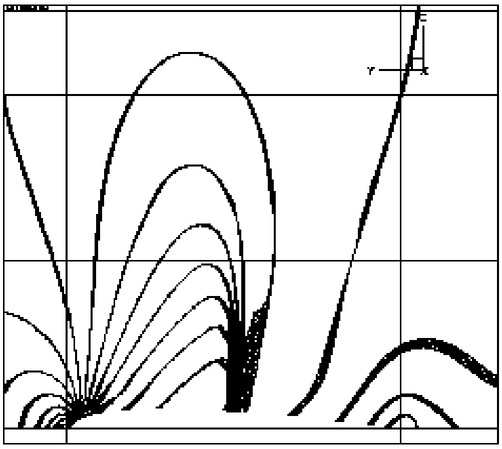


Figure 8 Mach number contours for the NACA0012 airfoil at $M_\infty = 0.8$ for non-overlapping domains.

approaches. We have not investigated the reduction in computation time because the fastest solvers are based on implicit approaches. The next step is to expand this formulation to the implicit approach and address the evaluation of computation time reduction as well as other parallel implementation issues.

Acknowledgments

The work was supported in part by the NSF grants ASC-9457534, ECS-9527169, and ECS-9725504.

We thank D. Keyes and D. P. Young for many stimulating discussions.

Appendix A. Transfer operators

To carry out our multi–model formulation, we do not require the transfer operator from Euler to potential $R : U \longrightarrow \Phi$. Only the mass flux, given by $G(R(U))$, is required to update the potential in each control volume. We define the full potential to Euler transfer operator as

$$Q : \Phi \longrightarrow U. \tag{25}$$

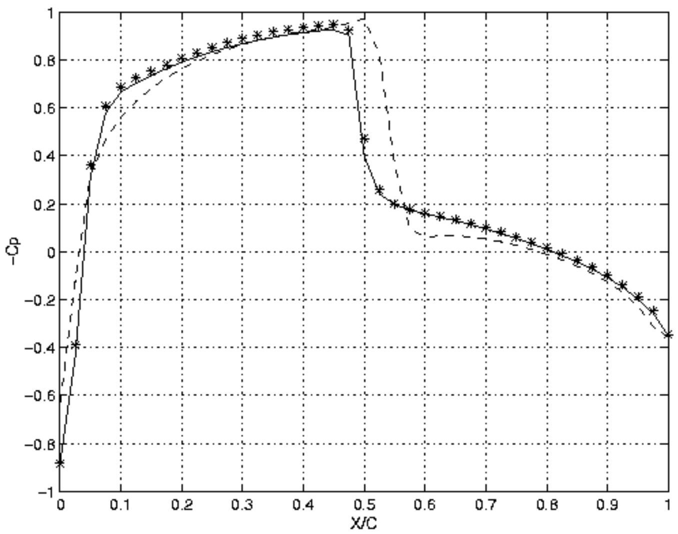


Figure 9 Pressure coefficient distribution on a NACA0012 airfoil at $M_\infty = 0.8$. — Euler solution; $--$ full potential solution; $*$ mixed solution.

Recall that U has five components. To obtain its first component, we appeal to

$$\rho(\Phi) = \rho_\infty \left(1 + \frac{\gamma - 1}{2} M_\infty^2 \left(1 - \frac{||\nabla\Phi||_2^2}{q_\infty^2}\right)\right)^{1/(\gamma-1)}. \tag{26}$$

The next three components can be computed with relation (3). The last component ρE

$$\rho E = \rho \left(e + \frac{u^2+v^2+w^2}{2}\right) = \frac{p}{\gamma - 1} + \rho \frac{u^2+v^2+w^2}{2},$$

where the pressure

$$p = p_\infty \left(\frac{\rho}{\rho_\infty}\right)^\gamma.$$

REFERENCES

[BGP+85] Bristeau M., Glowinski R., Periaux J., Perrier P., Pironneau O., and Poirier C. (1985) On the numerical solution of nonlinear problems in fluid dynamics by least squares and finite element methods (ii), application to transonic flow simulations. *Comput. Methods. Appl. Mech. Engrg.* 51: 363–394.

[BSBT97] Berkman M. E., Sankar L. N., Berezin C. R., and Torok M. S. (1997) Navier–Stokes/full potential/free–wake method for rotor flows. *J. of Aircraft* 34: 635–640.

[CFS98] Cai X.-C., Farhat C., and Sarkis M. (1998) A minimum overlap restricted additive Schwarz preconditioner and applications in 3D flow simulations. In J. Mandel C. F. and Cai X.-C. (eds) *The Tenth International Conference on Domain Decomposition Methods for Partial Differential Equations*. AMS.

[CGK+98] Cai X.-C., Gropp W. D., Keyes D. E., Melvin R. G., and Young D. P. (1998) Parallel Newton–Krylov–Schwarz algorithms for the transonic full potential equation. *SIAM J. Sci. Comput.* 19: 246–265.

[CRSC92] Cummings R., Rizk Y., Schiff L., and Chaderjian N. (1992) Navier–Stokes predictions for the F–18 wing and fuselage at large incidence. *J. of Aircraft* 29: 565–574.

[CW98] Coclici C. and Wendland W. L. (1998) Domain decomposition methods and farfield boundary conditions for 2D compressible viscous flows. In *Proc. of ECCOMAS 98*.

[FL94] Farhat C. and Lanteri S. (1994) Simulation of compressible viscous flows on a variety of MPPs: Computational algorithms for unstructured dynamic meshes and performance results. *Comput. Methods Appl. Mech. Engrg* 119: 35–60.

[FLCL95] Farhat C., Lesoinne M., Chen P. S., and Lantéri S. (1995) Parallel heterogeneous algorithms for the solution of the three–dimensional transient coupled aeroelastic problems. *AIAA Paper 95-1290* .

[HB79] Holst T. L. and Ballhaus W. F. (1979) Fast, conservative schemes for the full potential equation applied to transonic flows. *AIAA J.* 17: 145–152.

[HH82] Habashi W. G. and Hafez M. M. (1982) Finite element solutions of transonic flow problems. *AIAA J.* 20: 1368–1375.

[Hir90] Hirsch C. (1990) *Numerical Computation of Internal and External Flows.* John Wiley and Sons, New York.

[HSM79] Hafez M., South J., and Murman E. (1979) Artificial compressibility method for numerical solution of the transonic full potential equation. *AIAA J.* 17: 838–844.

[HSYB82] Holst T. L., Slooff J. W., Yoshihara H., and Ballhaus W. F. (1982) Applied computational transonic aerodynamics. Technical Report AG-266, AGARD.

[Jam97] Jameson A. (1997) Essential elements of computational algorithms for aerodynamics. Technical Report 97-68, ICASE.

[JBW86] Jameson A., Baker T., and Weatherill N. (1986) Calculation of inviscid transonic flow over a complete aircraft. *AIAA Paper 86-0103* .

[Lee79] Leer B. V. (1979) Towards the ultimate conservative difference scheme. V. A second order sequel to Godunov's method. *J. Comput. Phys.* 32: 361–370.

[PCS+99] Paraschivoiu M., Cai X.-C., Sarkis M., Young D. P., and Keyes D. (1999) Multi-domain multi-model formulation for compressible flows: Conservative interface coupling and parallel implicit solvers for 3D unstructured meshes. *AIAA Paper 99-0784* .

[PJ83] Pelz R. B. and Jameson A. (1983) Transonic flow calculations using triangular finite elements. *AIAA Paper 83-1922* .

[QV99] Quarteroni A. and Valli A. (1999) *Domain Decomposition Methods for Partial Differential Equations.* Oxford University Press, Oxford.

[SBT93] Sankar L. N., Bharadvaj B. K., and Tsung F. (1993) Three-dimensional Navier–Stokes/full potential coupled analysis for transonic viscous flow. *AIAA J.* 31: 1857–1862.

[SIGO87] Shankar V., Ide H., Gorski J., and Osher S. (1987) A fast, time–accurate, unsteady full potential scheme. *AIAA J.* 25: 230–238.

[YMB+90] Young D. P., Melvin R. G., Bieterman M. B., Johnson F. T., and Samant S. S. (1990) Global convergence of inexact Newton methods for transonic flow. *Int. J. Numer. Methods. Fluids* 11: 1075–1095.

[YMB+91] Young D. P., Mervin R. G., Bieterman M. B., Johnson F. T., Samant S. S., and Bussoletti J. E. (1991) A locally refined rectangular grid finite element methods: Application to computational fluid dynamics and computational physics. *J. Comput. Phys.* 92: 1–66.

19

Overlapping Schwarz Methods for Helmholtz's Equation

M. A. Casarin [1], O. B. Widlund [2]

Introduction

In this paper, we report on our experience in solving the scalar indefinite Helmholtz equation iteratively. We propose a modification of the standard Overlapping Schwarz Methods, allowing discontinuities across the interfaces between the subdomains, and using a Sommerfeld-type, quasi-transparent boundary condition for the local solvers; this results in a very efficient iterative method. We concentrate on the numerical performance of the accelerated version of our method and study the dependence of the rate of convergence on several important parameters, e.g. the number of subdomains, the geometry of the subdomains, the wave number, the mesh size, the size of overlap, coefficient inhomogeneities and the shape of the obstacle. We can conclude that our method, not yet fully explained theoretically, is very fast and efficient. Previous work on this project was performed in collaboration with F. W. Elliott and X-C. Cai [CCFWEW97].

We consider a Helmholtz model problem given by

1 IMECC-UNICAMP, Caixa Postal 6065, 13081 - 970 - Campinas - SP, Brazil. URL: http://www.ime.unicamp.br/~casarin. Electronic mail address: casarin@ime.unicamp.br . This work was supported in part by the National Science Foundation under Grants NSF-ECS-9527169 and NSF-CCR-9732208, in part by CNPq Brazil, and in part by FAPESP - Brazil, under project 97/12389-5.

2 Courant Institute of Mathematical Sciences, 251 Mercer Street, New York, N.Y. 10012. URL: http://cs.nyu.edu/cs/faculty/widlund/index.html. Electronic mail address: widlund@cs.nyu.edu. This work was supported in part by the National Science Foundation under Grants NSF-CCR-9732208 and NSF-ECS-9527169, and in part by the U.S. Department of Energy under Contract DE-FG02-92ER25127.

Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

$$\begin{cases} -\Delta u - (k(x))^2 u &= f \quad \text{in } \Omega \\ \frac{\partial u}{\partial n} + iku &= g_S \quad \text{on } \partial\Omega_S \\ \frac{\partial u}{\partial n} &= g_N \quad \text{on } \partial\Omega_N \\ u &= h \quad \text{on } \partial\Omega_D \end{cases}, \tag{1}$$

where Ω is a bounded two or three-dimensional region. We assume that $\partial\Omega_S$ is non empty. This equation is then uniquely solvable, and we note that the boundary condition, said to be of Sommerfeld type, is essential in the proof of this fact.

We use Green's formula, and complex conjugation of the test functions, to convert (1) into variational form: Find $u \in V$ such that,

$$\begin{aligned} b(u,v) &= \int_\Omega (\nabla u \cdot \nabla \bar{v} - k^2 u\bar{v})dx + ik\int_{\partial\Omega_S} u\bar{v}ds \\ &= \int_\Omega f\bar{v}dx + \int_{\partial\Omega_S} g_S\bar{v}ds + \int_{\partial\Omega_N} g_N\bar{v}ds \ = \ F(v) \ \ \forall v \in V, \end{aligned}$$

where V is an appropriate subspace of $H^1(\Omega)$. A finite element discretization can now be defined straightforwardly by replacing V by a suitable conforming finite element space. We concentrate here on low order finite elements.

Our interest in the application of the direct (non-mixed) finite element method to this equation has been inspired by the work of Ihlenburg and Babuška [IB95, IB97]. They have considered the well-posedness of the original problem and different finite element discretizations and proven, for a model problem in one dimension, that the basic estimate

$$|u|_{H^1} \le Ck|F|_{H^{-1}}$$

holds. In the finite element case, an assumption of $hk < 1$ is used. The constant C is independent of p, the degree of the finite elements. Ihlenburg has also conducted extensive numerical experiments which suggest that this bound holds for problems in two or three dimensions. Error bounds of the following form are also given for $p = 1$ and kh small enough:

$$|error|_{H^1} \le C_1\theta + C_2k\theta^2 \text{ where } \theta = \text{best } H^1\text{−error}.$$

With oscillatory solutions typical, we can expect θ to be on the order of kh. The second term, which is due to the phase error, will dominate unless k^2h is on the order of 1. In our experiments we have used a mesh fine enough to guarantee that the phase error is of the same order as the other component of the error.

Overlapping Method – Continuous and Discontinuous

In the past few years, much work has been done on the iterative solution of the Helmholtz's equation that governs the time-harmonic scattering of sound waves by an object. Among many others, see, e.g., [FML98, Kim98] we mention, in particular, the thesis of Després [Des91]. We consider an overlapping version of his method (which was briefly considered in his work in a particular case). Our basic multiplicative, one-level overlapping Schwarz method can be described as follows: let $\{\Omega_j\}$ be a set of open subregions that covers the given open region Ω. Each subregion Ω_j can have many disconnected components; it is often profitable to color the subregions of an original overlapping decomposition of Ω using different colors for any pair of subregions that intersect. The original set of subregions can then be partitioned into sets of subregions, one of each color, effectively reducing the number of subregions. This decreases the number of fractional steps of our Schwarz methods and helps make the algorithm parallel. The number of colors is denoted by J.

For $j = 1, \ldots, J$, given a finite element function $u^{n+\frac{j-1}{J}}$, defined on Ω, we compute $u_{\Omega_j}^{n+\frac{j}{J}}$, the restriction of $u^{n+\frac{j}{J}}$ to Ω_j, by solving the variational form of the following local problem:

$$\left\{\begin{array}{lcll}
-\Delta u_{\Omega_j}^{n+\frac{j}{J}} - (k(x))^2\, u_{\Omega_j}^{n+\frac{j}{J}} & = & f & \text{in } \Omega \\
\dfrac{\partial u_{\Omega_j}^{n+\frac{j}{J}}}{\partial n_{\Omega_j}} + iku_{\Omega_j}^{n+\frac{j}{J}} & = & -\dfrac{\partial u_{\Omega_j^c}^{n+\frac{j-1}{J}}}{\partial n_{\Omega_j^c}} + iku_{\Omega_j^c}^{n+\frac{j-1}{J}} & \text{on } \partial\Omega_j \cap \Omega \\
\dfrac{\partial u_{\Omega_j}^{n+\frac{j}{J}}}{\partial n} + iku_{\Omega_j}^{n+\frac{j}{J}} & = & g_S & \text{on } \partial\Omega_j \cap \partial\Omega_S \\
\dfrac{\partial u_{\Omega_j}^{n+\frac{j}{J}}}{\partial n} & = & g_N & \text{on } \partial\Omega_j \cap \partial\Omega_N \\
u_{\Omega_j}^{n+\frac{j}{J}} & = & h & \text{on } \partial\Omega_j \cap \partial\Omega_D
\end{array}\right. , \quad (2)$$

Here, Ω_j^c refers to the complement of Ω, and n is the outward normal. The boundary of the original region Ω has been partitioned into three parts, two of which are possibly empty: $\partial\Omega = \partial\Omega_S \cup \partial\Omega_N \cup \partial\Omega_D$. The Sommerfeld condition is imposed on $\partial\Omega_j \cap \Omega$ to guarantee solvability and stability of this problem; it also has a dramatic impact on the convergence rate, if compared with the use of standard Dirichlet conditions for this problem, even in the case when the latter is enough to guarantee stability of the problems defined on the Ω_j; see [CCFWEW97].

An important detail, somewhat hidden in the above formulation of the local problems is that at step j, represented by (2), the values computed for u^{n+1}, at the mesh points on $\partial\Omega_{j,}$ might differ from those of u^n at the same points. This forces us to consider two alternatives:

1. the new values replace the old ones, and the resulting finite element function is continuous, with possibly large gradients just outside $\partial\Omega_{j;}$
2. the new *and* the old values are kept.

The second alternative produces an algorithm which proceeds to the converged, continuous solution through a sequence of discontinuous iterates. It is hereafter called the Discontinuous Overlapping Schwarz Method (OSM-D). In [CCFWEW97], we established, through a variety of numerical experiments, that the discontinuous version is actually superior in most cases considered; many details about the implementation of the method are also given in that paper in which the discontinuous method is referred to as ALG3. We only mention here that in the finite element context the computation of normal derivatives of the form $\dfrac{\partial u^n_{\Omega^c_j}}{\partial n_{\Omega^c_j}}$ is made with the consistent use of Green's formula, and that the discontinuities force us to perform a subassembly involving elements outside Ω_j, which are next to $\partial\Omega_j$. Similarly, the computation of terms originating from the term $iku^n_{\Omega^c_j}$ involve a subassembly of the mass matrix corresponding to different parts of $\partial\Omega_j$. It is easy to see that in the absence of crosspoints interior to Ω, and formed by the boundaries of the internal overlapping subdomains, the continuous and discontinuous methods produce the same iterates.

In this paper, we report on an experimental study of the discontinuous method which is more detailed than the previous ones, and show that the method is indeed very fast and efficient, even in the absence of a coarse space, which is very important for the efficiency of some competing alternatives; see [FML98]. We consider several different geometries, unstructured meshes and different partitions of the domain into subdomains. The experiments were all conducted with MATLAB, and the geometry, mesh generation, and matrix assemblies were done using the PDE toolbox.

The theoretical results that we have obtained have already been outlined in [CCFWEW97]. They are satisfactory only for the continuous version of the method.

Convergence Study

We now describe the geometry and discretization used in our numerical experiments. We work with three different geometries and boundary conditions: a) a Sommerfeld square, which is a unit square with homogeneous Sommerfeld conditions on the four sides. The right hand side is chosen so that the solution is the combination of two plane waves incoming at angles 36^o and 178^o; b) a waveguide problem, which is again a unit square, with homogeneous Sommerfeld conditions on the horizontal sides, homogeneous Neumann conditions on the right vertical side, and a constant Dirichlet condition identically equal to 1 on the left vertical side. The right hand side f is zero; and c) the scattering geometry, which corresponds to a cavity of diameter approximately 1 inside a circle of radius approximately 3.7; the right hand side is again zero, and Dirichlet data is imposed on the surface of the scatterer, corresponding to an incoming plane wave of wavenumber k (given in the following) and angle 36^o. For this last case, the geometry of Ω and the 14 non-overlapping subdomains Ω_j is given in Figure 1 under the name GSC14-1. The scatterer is a cavity with vertices at: $(-0.52,-0.52),(-0.52,0.52),(0.5,0.5),(-0.5,0.5),(-0.5,-0.5),(0.5,-0.5)$. The overlap is produced by a layer of δ elements added in all directions which are interior to Ω. In all experiments the mesh is unstructured, and except for the very small test cases, the number of mesh points is certainly enough to resolve the oscillations in the solution.

Apart from the geometry, the relevant parameters for describing an experiment are:

- the total number of grid points N. We define $1/h \simeq \sqrt{N}$ when Ω is a unit square;
- k the spatial frequency, which can depend on $x \in \Omega$;
- $nsub$ the number of subregions;

- δ the number of elements across half of the overlap;

Our main goal is to study the dependence of the number of iterations of our method on the parameters listed above, as well as to evaluate how this iteration count will depend on the original boundary conditions, the variation of the coefficients, the geometrical placement of the subdomains, and the global geometry. The linear, sequential method of (2) is accelerated by the GMRES method with restart equal to 30, and the algorithm is stopped as soon as the norm of the residual is reduced by 5 orders of magnitude (10^{-5}).

When possible, we have compared our method with two competing alternatives: a) a FETI type algorithm as reported by Farhat, Macedo, and Lesoinne [FML98]; and b) the ADOP method proposed by Kim [Kim98]. This is possible because our implementation is extremely flexible with respect to the geometry and all other parameters; this comparison is also made in order to provide a fair evaluation of our scheme. We note that the comparisons are made using exactly the same problems as reported in the references given above.

Dependence on the mesh size

Our first set of experiments involves a study of the optimality with respect to the mesh size. We compare our method first with the FETI method, applied to a waveguide problem [FML98]Table 1, and then with the ADOP method, for a Sommerfeld square [Kim98]Table 6. We can see, from Tables 1 and 2, that the OSM-D method is very competitive, and that when compared with the ADOP method it has the added advantage of requiring no parameter estimation. The data shows that the iteration growth is mild with respect to the number of points. This observation is enforced by the data in the other tables; see, e.g. Tables 10 and 11. All our experiments used unstructured meshes, and that therefore we cannot exactly reproduce the discrete problem being solved by an alternative algorithm on a structured mesh.

Table 1 Waveguide problem with $k = 20$, variable mesh size: comparison between the FETI method with 5×5 subdomains, and OSM-D with 24 subdomains and $\delta = 6$.

FETI		OSM-D	
$1/h$	itcount	$1/h$	itcount
100	71	83	26
200	70	163	32
250	73		

Dependence on the wavenumber k

We also study the behavior of the method as the wavenumber k varies. The waveguide problem is solved with both the FETI and the OSM-D methods, and the data is

Table 2 Sommerfeld square problem, $k = 25$, and 16×4 subdomains: comparison between the ADOP method and OSM-D with $\delta = 5$

ADOP		OSM-D	
$1/h$	itcount	$1/h$	itcount
128	60	114	37
256	68	173	43

reported in Table 3.

Table 3 Waveguide Problem: comparison between the FETI method with $1/h = 384$, no coarse space, $nsub = 5 \times 5$, and the OSM-D method with 27649 points, 24 subdomains, and $\delta = 6$

FETI	Waveguide	OSM-D	Waveguide
k	itcount	k	itcount
20	70	20	58
40	136	40	64
60	164	60	70

We have also solved the scattering problem; the data is in Table 4, and the geometry (GSC14-1) is given in Figure 1.

Table 4 OSM-D method applied to the Scattering problem. The domain is partitioned into 14 subdomains (GSC14-1), $\delta = 6$, and there are a total of 26596 points

OSM-D	
k	itcount
12	11
24	15
36	19 ($\delta = 9$)

We can see from Tables 3 and 4 that the increase of the wavenumber does lead to a growth in the iteration count, although this effect is not as pronounced as for the FETI method without a coarse space. It is also clear that even in the scattering geometry the dependence on the wave number is quite weak. This is in contrast with a plausible intuitive interpretation of the method as producing a system of waves propagating across the subdomains; the larger the wavenumber the more complex would be the reflection and diffraction phenomena, and one would therefore expect the performance of the algorithm to deteriorate quite rapidly.

In actual applications, the coefficients of the time-harmonic wave equation often vary, and a natural question is therefore whether the performance of the method deteriorates if the coefficients are inhomogeneous. We borrow the following coefficient functions from Kim [Kim98]equation (69):

$$\begin{aligned} k_1 &= 12/\left((2+\sin \pi x)\,(2-\sin 3\pi y)\right) \\ k_2 &= 12/\left((1.1+\sin \pi x)\,1\,(1.1-\sin \pi y)\right). \end{aligned}$$

The Discontinuous Overlapping Method with $\delta = 4$ is then applied to the scattering geometry partitioned into 14 subdomains, and the results appear in Table 5, where we have also included a set of results for the case of constant k. It can be seen from this Table that the OSM-D appears to be very robust with respect to coefficient inhomogeneities in the wavenumber function; the actual iteration count is also very modest. Inhomogeneities in the other coefficients have also been studied, but since the behavior is very similar, we omit that particular data.

Table 5 Application of the OSM-D method to a Scattering problem with variable wavenumber. The domain is partitioned into 14 subdomains (GSC14-1), and $\delta = 4$.

	$k = 12$	k_1	k_2
N	itcount	itcount	itcount
1735	7	7	10
6746	7	10	14
26596	11	13	20

Dependence on the number of subdomains

The OSM-D method proposed here relies only on local solves. The performance of similar methods, in the positive definite case, deteriorates with an increasing number of subdomains. The same behavior occurs here; our goal is now to explore this dependence quantitatively, and to compare that with the FETI method without a coarse space. The results for the Waveguide problem are given in Table 6. The value of k is chosen so that we have approximately 3 wavelengths across the domain, and the number of mesh points is of the same order in both cases. The iteration count clearly grows with the number of subdomains but less so for the OSM-D method. At this time, we do not have a good explanation for this fact.

We also provide the iteration count for a scattering problem, with the domain partitioned into $5, 14$, and 98 subdomains, respectively, all with good aspect ratios. The value of k is set to 20, and $\delta = 4$. The same trend appears here, although the actual number of iterations is low. The development and implementation of computationally

Table 6 Waveguide problem, $k = 20$, 3 wavelengths across the domain. Comparison between the FETI method with approximately 40,000 points in the mesh, and $\delta = 6$.

FETI		OSM-D	
$nsub$	itcount	$nsub$	itcount
2×2	15	2×2	12
4×4	46	4×4	26
5×5	70	24	32

efficient coarse spaces is currently being considered, in order to treat cases with possibly many more subdomains.

Table 7 OSM-D, Scattering $k = 20$, $\delta = 4$

N	$nsub$	itcount
27649	5	8
26596	14	14
15392	98	27

Dependence on the original boundary conditions

We next compare two sets of runs of the OSM-D on the same square region. The main difference from one run to the other is the boundary condition: the first is a Waveguide square, partitioned into 24 subdomains, and the second is a Sommerfeld square, partitioned into 5×5 subdomains. Again, the subdomain partition is not regular, and was made by hand, in order to produce non-square subregions with relatively good aspect ratios. We use $k = 20$ and $\delta = 4$ in both cases and the results are reported in Table 8. It is immediately apparent that the boundary condition imposed on $\partial\Omega$ has a considerable impact on the iteration count. This is surprising if compared with similar experiments for the standard OSM method in the positive definite case, with Dirichlet local problems. It appears that the waveguide boundary condition excites modes that are not damped as efficiently by the algorithm, and therefore the iteration count is larger for this case.

Dependence on the subdomain geometry

As previously mentioned, our method could be viewed intuitively in terms of waves propagating across the subdomains: the Sommerfeld-like condition for the local problems could be considered a better choice than the standard Dirichlet condition in this respect, and that would be a possible explanation for its much better performance;

Table 8 Study of the dependence on the original boundary conditions for the OSM-D method. Comparison between the Waveguide square partitioned into 24 subdomains, and the Sommerfeld square partitioned into 25 subdomains. In both cases, $k = 20$ and $\delta = 4$.

Waveguide		Sommerfeld	
$1/h$	itcount	$1/h$	
83	26	121	9
163	32	161	13

see [CCFWEW97]. We note that such heuristics is at the core of some methods for the Helmholtz's equation which look for more transparent boundary conditions for the local problems; see [Gha96]. Inspired by this heuristics, we explore, in this section, how the algorithm depends on the particular choice and configuration of the subdomains.

Our first test case involves the scattering geometry GSC14-1. We use the OSM-D method, accelerated with GMRES, for various values of k, δ, and N (the total number of mesh points). The results are reported in Table 9. The geometry of this and the next cases are represented in Figure 1.

Table 9 OSM-D, Scattering problem, geometry GSC14-1, full opening, 14 subdomains, and $\delta = 6$, unless otherwise noted.

N	k = 12	k = 24	k = 36, $\delta = 9$	k = 12, $\delta = 4$
4438	46	78	50	>60
17404	72	>200	> 100	> 60

The tips at the entrance of the cavity are possibly points where our method would have difficulty converging to the exact solution. In order to check that, we change the subdomain partition from the geometry GSC14-1 to GSC14-2 (see Figure 1). The number of subdomains and the other parameters are kept the same, and the results are reported in Table 10. The iteration count decreases in all cases considered, at times by a factor of 6, showing that the configuration of the subdomains (and not only their number and aspect ratio) is indeed of a considerable importance for the iterative solution of this problem.

Continuing the exploration of the heuristics mentioned above, we now question whether the aperture of the cavity has a deleterious impact on iteration count. We keep roughly the same subdomain configuration as in GSC14-2, except that we change the cavity into one with a tighter aperture, respectively 1/2 and 1/3 of the opening, for Tables 11 and 12, corresponding to the geometries GSC14-3 and GSC14-4. The iteration count increases from GSC14-3 to GSC14-4, but the changes are not very dramatic.

Table 10 OSM-D, Scattering problem, geometry GSC14-2, full opening, 14 subdomains, $\delta = 6$ unless otherwise noted. The subdomain configuration is slightly different from that of the previous table.

N	k = 12	k = 24	k = 36, $\delta = 9$	k = 12, $\delta = 4$
6746	7	14	29	11
26596	11	15	19	13
105608				13

Table 11 OSM-D, Scattering problem, geometry GSC14-3, 1/2 the opening, 14 subdomains, $\delta = 6$ unless otherwise noted. The subdomain configuration is slightly different from that of the previous table.

N	k = 12	k = 24	k = 36, $\delta = 9$	k = 12, $\delta = 4$
7389	11	58	39	15
28966	10	20	44	13

Table 12 OSM-D, Scattering problem, geometry GSC14-4, 14 subdomains, $\delta = 6$ unless otherwise noted. The subdomain configuration is slightly different from that of the previous table.

N	k = 12	k = 24	k = 36, $\delta = 9$	k = 12, $\delta = 4$
9552	13	89	54	24
37576	15	33	74	17

In these last four tables, one can also observe the effect of increasing the total number of points, and/or increasing the wavenumber k. The behavior follows the trends identified in our previous tables. In Table 11, for example, we can also observe that increasing the size of the overlap δ does improve the method, both for $k = 12$ and $k = 36$. So far, we have not been able to explain the extraneous behavior showed by the method for $k = 24$, namely that increasing the number of points improves the method, and also that the iteration count is relatively large.

Conclusion

We have developed and tested the OSM-D method in a variety of situations, and showed that it compares favorably with some alternative algorithms. While efforts to develop coarse spaces appropriate to our method are under way, we have here used only a method without a global component even for cases with relatively many subdomains. The iteration count does deteriorate with the number of subdomains, but the actual numbers are small. The algorithm is robust in regards to the wavenumber; this is a salient feature of our method. An interesting new issue considered is the effect of the configuration of the subdomains on the performance of the scheme; this has been studied in subsection 19, where we have established a basic trend; a more refined study is also in progress. At last we want to point out that for a non-symmetric, indefinite problem like the Helmholtz's equation considered here, the boundary conditions on $\partial\Omega$ can have a significant impact on the iterative method; see subsection 19.

REFERENCES

[CCFWEW97] Cai X.-C., Casarin M. A., Frank W. Elliott J., and Widlund O. B. (1997) Overlapping Schwarz methods for solving Helmholtz's equation. In Mandel J., Farhat C., and Cai X.-C. (eds) *Domain Decomposition Methods 10*, volume 218 of *Contemporary Mathematics*, pages 391–399. AMS.

[Des91] Després B. (October 1991) *Méthodes de Décomposition de Domaine pour les Problémes de Propagation d'Ondes en Régime Harmonique*. PhD thesis, Paris IX Dauphine.

[FML98] Farhat C., Macedo A., and Lesoinne M. (March 1998) A two-level domain decomposition method for the iterative solution of high frequency Helmholtz problems. Technical Report CU-CAS-98-06, Center for Aerospace Structures, University of Colorado at Boulder. Submitted to Numerische Mathematik.

[Gha96] Ghanemi S. (January 1996) *Méthode de Décomposition de Domaine avec Conditions de Transmissions Non Locales pour des Problémes de Propagation d'Ondes*. PhD thesis, Paris IX Dauphine.

[IB95] Ihlenburg F. and Babuška I. (1995) Finite element solution of the Helmholtz equation with high wave number, Part I: The h-version of the FEM. *Computers Math. Applic.* 30(9): 9–37.

[IB97] Ihlenburg F. and Babuška I. (1997) Finite element solution of the Helmholtz equation with high wave number, Part II: The h-p-version of the FEM. *SIAM J. Numer. Anal.* 34(1): 315–358.

[Kim98] Kim S. (1998) Domain decomposition iterative procedures for solving scalar waves in the frequency domain. *Numer. Math.* 79: 231–259.

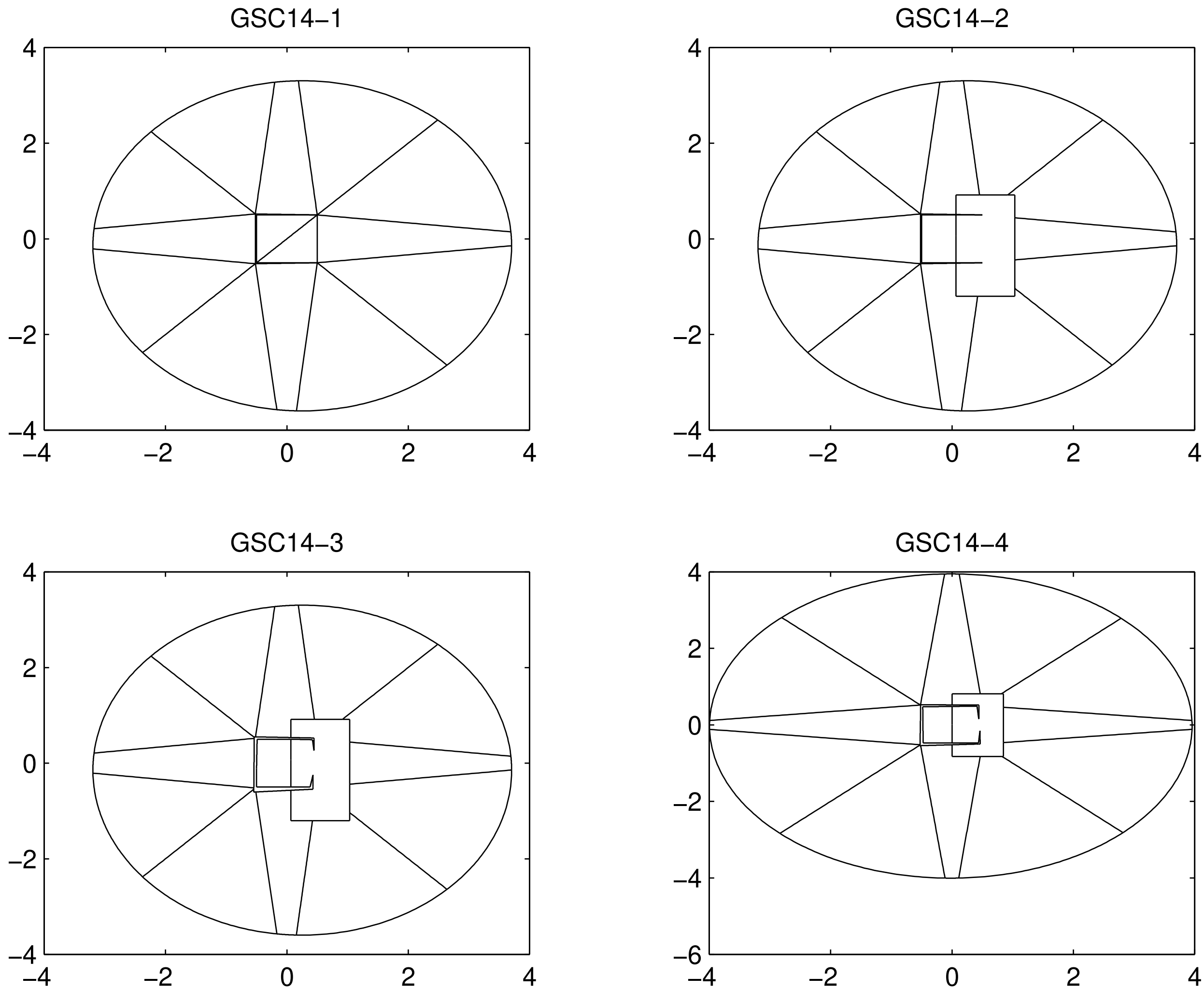


Figure 1 Scattering geometries, with several configurations of the subdomains

20

Multilevel Spectral Partitioning of Unstructured Grids

TONY F. CHAN[1], SUSIE GO[2], & JUN ZOU[3]

INTRODUCTION

The graph partitioning problem is an important component of parallel computing (e.g. for constructing subdomains in domain decomposition methods) and as a result, many partitioning methods and associated sophisticated software packages have been developed recently. The goal in partitioning a graph is to find a separator which bisects the the graph while minimizing the number of edges cut. Such partitioners include algorithms based on greedy, coordinate, inertial, multilevel spectral and graph bisection. However, there is usually an unavoidable tradeoff between quality and speed. Of these, spectral partitioning has the reputation of being the most time consuming, but it also consistently produces high-quality partitions which for some applications can offset its costliness with better convergence rates of iterative solvers. We will show that when the graph is a standard finite element mesh, the graph Laplacian is spectrally equivalent (up to a diagonal scaling) to the mesh Laplacian. This equivalence can be exploited to adapt recently developed multilevel elliptic algorithms

1 Department of Mathematics, University of California at Los Angeles, Los Angeles, CA 90095-1555, USA. E-mail: chan@math.ucla.edu. The work of this author was partially supported by grants from NSF under grant number ASC-9720257 and from Sandia National Laboratory under LG 4440.
2 Research Institute of Advanced Computer Science (RIACS), Mail Stop 19-39, NASA Ames Research Center, Moffett Field, CA 94035-1000, USA. E-mail: susie@riacs.edu. Support from RIACS under contract number NCC2-1006.
3 Department of Mathematics, The Chinese University of Hong Kong, Shatin, Hong Kong. E-mail: zou@math.cuhk.edu.hk. The work of this author was partially supported by Hong Kong RGC Grant No. CUHK 338/96E and CUHK 4004/98P.
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

for unstructured grids to solving the graph partitioning problem with a true multigrid (MG) convergence rate. We also give some preliminary results of such an approach.

In Section 20, the spectral partitioning algorithm is briefly discussed. The main cost of the spectral method is in solving for the Fiedler vector (the eigenvector corresponding to the lowest non-trivial eigenvalue of the associated graph Laplacian). Attempts at accelerating the computation of the Fiedler vector by multilevel methods encounters the difficulty of adapting standard elliptic multilevel algorithms to solving the discrete graph Laplacian eigenvalue problem. Many approaches rely on piecewise constant interpolations to develop multilevel algorithms. Thus, it is difficult to construct algorithms which are optimal for calculating solutions involving the Laplacian operator.

If the graph is a grid, we can use the spectral equivalence of the mesh and graph Laplacian to develop efficient ways to find the Fiedler vector. There exist techniques for dealing with the mesh Laplacian on unstructured grids; we can now apply them to the case of the graph Laplacian. One possibility is described in Section 20. A multigrid approach for solving elliptic eigenvalue problems is adapted for use on unstructured grids and on discrete problems. We give some numerical results which show that the algorithm, when appropriately defined, yields optimal convergence rates for finding the Fiedler vector.

SPECTRAL EQUIVALENCE OF THE GRAPH AND MESH LAPLACIAN

We briefly define the spectral partitioning method for bisecting a graph. For a more complete description of partitioning algorithms, we refer the reader to Pothen's survey [Pot96]. Let $G = (V, E)$ be an undirected graph, where $V = \{v_i\}_{i=1}^m$ is the set of vertices of the graph, and $E = \{(v_i, v_j)\}$ is the set of edges. An associated matrix called the graph Laplacian, L, is defined as:

$$L_{ij} = \begin{cases} \mathrm{deg(v_i)} & \mathrm{if\ i = j} \\ -1 & \mathrm{if\ (v_i, v_j) \in E} \\ 0 & \mathrm{otherwise,} \end{cases} \tag{1}$$

where $\mathrm{deg(v_i)}$ is the number of edges in E containing vertex v_i.

The spectral bisection method works by solving the discrete Laplacian eigenvalue problem, $Lu = \lambda u$, for the eigenvector corresponding to the second smallest eigenvalue, known as the Fiedler vector. One way of defining a bisection of the graph is found by sorting the components of the resulting Fielder vector and defining the median value of the components as the so-called splitting value. This median cut partitioning was shown to be optimal in [CCS97]. A bisection of the graph is found by assigning those vertices whose value is less than the splitting value to one subgraph and assigning the rest to the other subgraph.

We now show that for standard finite element meshes, the graph Laplacian is spectrally equivalent, up to a diagonal scaling, to the mesh Laplacian. This equivalence can then be used to show a that multilevel elliptic eigenvalue solver can be used to solve the graph partitioning problem with true multigrid convergence rates.

Assume $\mathcal{T}^h$ is a shape regular finite element triangulation of a domain Ω in R^n ($n = 2$ or 3) and $G = (V, E)$ is the graph induced from $\mathcal{T}^h$. Let L be the Laplacian of the graph G and A be the stiffness matrix arising from the finite element discretization of the continuous Laplacian operator Δ on the triangulation $\mathcal{T}^h$.

Theorem 1 *The graph Laplacian L is spectrally equivalent to the stiffness matrix defined by the quadratic form:*

$$\sum_{K\in\mathcal{T}^h} h_K^{2-n}\int_K |\nabla u|^2\, dx. \tag{2}$$

Proof. Let K be any triangle ($n = 2$) or tetrahedron ($n = 3$) in $\mathcal{T}^h$, h_K be the diameter of K, and $\hat{K}$ be the reference element with $n+1$ vertices $\hat{x}_i$, for $i = 1, \ldots, n+1$.

Let $F : \hat{K} \to K$ be the affine mapping defined by $F(\hat{x}) = B\hat{x} + b$ such that the vertices $\hat{x}_i, i = 1, \ldots, n+1$ of the element $\hat{K}$ are mapped to the vertices $x_i, i = 1, \ldots, n+1$ of the element K. Let u be any linear function on K, with nodal values u_i at x_i. Clearly, $\hat{u}$ has the same values at its vertices as u. Moreover, we have (cf. [Cia78])

$$\|\nabla u\|^2_{L^2(K)} \le C|\det(\mathrm{B})|\|\mathrm{B}^{-1}\|^2\|\nabla\hat{\mathrm{u}}\|^2_{\mathrm{L}^2(\hat{\mathrm{K}})} \tag{3}$$

where $\|\cdot\|$ is the spectral norm of a matrix.

By the definition of the matrix norm and the shape regularity assumption, we can derive the following bounds [Cia78]

$$\|B^{-1}\| \le h_K^{-1}, \quad \|B\| \le Ch_K, \tag{4}$$

and

$$|\det(\mathrm{B})| \le \mathrm{meas(K)} \le \mathrm{Ch_K^n}, \quad |\det(\mathrm{B}^{-1})| \le \mathrm{Ch_K^{-n}}. \tag{5}$$

Then from (3), it follows that

$$\|\nabla u\|^2_{L^2(K)} \le Ch_K^{n-2}\|\nabla\hat{u}\|^2_{L^2(\hat{K})}. \tag{6}$$

Now consider the finite dimensional quotient space $P_1(\hat{K})/R$. It is easy to check that

$$|||\hat{u}||| \stackrel{\text{def}}{=} \left(\sum_{\hat{x}_i,\hat{x}_j\in\hat{K}} (\hat{u}(\hat{x}_i) - \hat{u}(\hat{x}_j))^2\right)^{1/2}$$

is a norm on $P_1(\hat{K})/R$. We also know that $\|\nabla\hat{u}\|^2_{L^2(\hat{K})}$ is a norm in $P_1(\hat{K})/R$. Therefore, they are equivalent and from (6), this implies

$$h_K^{2-n}\|\nabla u\|^2_{L^2(K)} \le C\|\nabla\hat{u}\|^2_{L^2(\hat{K})} \le C\sum_{x_i,x_j\in K}(u(x_i) - u(x_j))^2 \tag{7}$$

since $\hat{u}(\hat{x}_i) = u(x_i)$.

Similarly as (3), we have

$$\|\nabla\hat{u}\|^2_{L^2(\hat{K})} \le C|\det(\mathrm{B}^{-1})|\|\mathrm{B}\|^2\|\nabla\mathrm{u}\|^2_{\mathrm{L}^2(\mathrm{K})} \le \mathrm{Ch_K^{2-n}}\|\nabla\mathrm{u}\|^2_{\mathrm{L}^2(\mathrm{K})}, \tag{8}$$

combining this with (7) gives

$$h_K^{2-n}\|\nabla u\|_{L^2(K)}^2 \le \sum_{x_i,x_j\in K}(u(x_i)-u(x_j))^2 \le h_K^{2-n}\|\nabla u\|_{L^2(K)}^2. \qquad (9)$$

From (9), we obtain

$$\sum_{K\in\mathcal{T}^h} h_K^{2-n}\int_K |\nabla u|^2\,dx \simeq \sum_{i,j\in E}(u_i-u_j)^2 \qquad (10)$$

where each edge is just counted once.

But note that $u^T Lu = \sum_{(i,j)\in E}(u_i-u_j)^2$, hence we have proved the equivalence.

FULL APPROXIMATION SCHEME MULTIGRID ALGORITHM FOR ELLIPTIC EIGENVALUE PROBLEMS

The spectral equivalence of the graph Laplacian and the mesh Laplacian motivates us to use the same tools for solving the discrete eigenvalue problem of spectral partitioning as we use for a continuous elliptic eigenproblem. Two possible alternatives for using the mesh Laplacian to solve for Fiedler vector would be to:

1. Use the mesh Laplacian A to precondition the graph Laplacian L in some iterative method (e.g. inverse iteration),
2. Use a multigrid method directly by applying the same interpolants of the mesh problem to define a hierarchy of coarse problems for the associated discrete problem.

We adapt an eigensolver which uses the full approximation scheme (FAS) multigrid method [BMR83] to develop a multigrid partitioner. The multigrid eigensolver treats the eigenvalue problem as a non-linear problem, thus FAS is used. The two-level algorithm for solving for the smallest eigenvalue and corresponding eigenvector of $Au_h - \lambda u_h = 0, \|u_h\| = 1$ is given by:

Algorithm 1 *Two-level FAS multigrid:*

1. *Given an initial guess, λ^0 and u^0,*
2. *Do one V-cycle FAS multigrid step:*
   (a) *Presmooths.*
   (b) *Restrict solution and calculate f_H.*
   (c) *Solve coarse problem $A_H u_H - \lambda u_H = f_H$.*
   (d) *Update λ using the Rayleigh-quotient modified for the extended eigenvalue problem and normalize vector.*
   (e) *FAS interpolate solution to fine level.*
   (f) *Postsmooths.*

The method can be extended recursively to yield a multigrid algorithm. To solve for multiple eigenvalues/eigenvectors, Ritz acceleration can be added to the method after all the values and vectors desired have been found. The algorithm for finding q vector/value pairs:

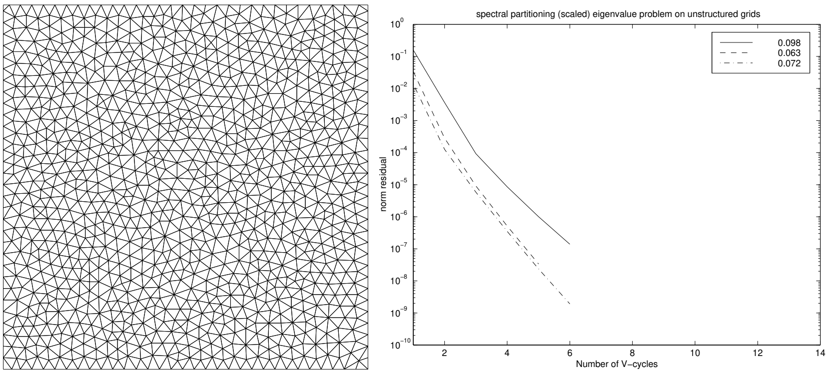


Figure 1 FAS MG for the spectral bisection problem on the unit square with unstructured grids of varying mesh sizes. Solid line (310 nodes), dashed line (1064 nodes), and dash-dotted line (4164 nodes). The grid with 1064 nodes is shown on the left.

Algorithm 2 *Multiple eigenvector/value solver:*

1. *Given initial guesses,* $(\vec{u}_j, \lambda_j), j = 1, \ldots q,$
2. *While not converged,*
   (a) *For* $i = 1,$ *to* q
      i. *Call FAS multigrid to solve for* $(\vec{u}_i, \lambda_i)$.
   (b) *Ritz orthogonalize the set of* q *vector/value pairs.*

We can use all the same machinery as the continuous elliptic eigenvalue solver: the same FAS interpolation and the same coarse grid solution process. Since spectral partitioning requires computing the Fiedler vector, the FAS MG method is used to solve the discrete problem, but modified so as to seek only the second eigenvector, since the first eigenvector/value pair is known to be $(\vec{1}, 0)$.

We use a grid hierarchy which is generated via maximal independent sets of the fine grid and then retriangulated to create coarser grids. Interpolation operators are defined to be piecewise linear in the regions where both fine and coarse grids overlap. In the case of non-matching boundaries, they can be defined to extend by zero where the coarse grid exists but the fine grid does not. Where the coarse grid does not cover the fine grid, they must provide a non-zero extension, for example, by linear interpolation with respect to a nearby edge or element, or by modifying the coarse grid so that it covers the fine grid (we refer the reader to [CGZ96] for more details).

Figure 1 shows the FAS MG convergence histories for solving the discrete Laplacian eigenvalue problem $Lu = \lambda u$ on unstructured grids of varying mesh sizes. It can be observed that the method is independent of mesh size.

Next, we show a comparison for solving the spectral bisection partitioning problem using the FAS MG scheme with various interpolants. Since the spectral bisection problem has Neumann boundary conditions, care must be taken in the definition of interpolation operators at non-matching boundaries (see [CGZ96, CSZ96] for details). We observe that using a standard linear interpolant with zero extension cannot

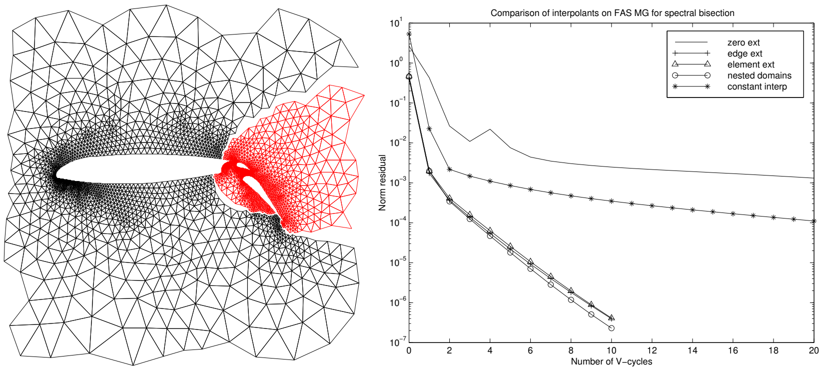


Figure 2 4-level FAS MG convergence histories for the spectral bisection problem on an unstructured airfoil grid (left) using different interpolants.

be accurate enough to define Neumann boundary conditions when non-matching boundaries exist. In addition, note that piecewise constant interpolants also cannot be used.

The edges cut and times required for bisecting the airfoil using various popular partitioners [BS93, HL95, KK97] are compared in Table 1. Options for the partitioners were set to generate a spectral bisection to an eigentolerance of 10^{-6}, with local refinement turned off. The bisection found by Metis has a different number of cut edges because spectral bisection is used to generate a coarse bisection which is then "uncoarsened" to yield a bisection of the original graph. Depending on the type of maximal matching chosen for the coarsening step, the number of cut edges ranged from 114 to 153 when the original airfoil grid was coarsened down to 100 nodes.

We should comment that the current FAS MG implementation was not optimized in any way and that the main objective was in achieving provable optimal multigrid performance, that is, obtaining a solution process which was grid-size independent. It was also assumed that the interpolation operators are immediately available and the timings do not include the time to generate the Galerkin coarse operators.

Finally, Table 2 shows the relationship between the tolerance and the number of V-cycles used, as well as the number of edges cut. It should be noted that there is little or no advantage to solving the partitioning problem to extremely low tolerance. This was also observed previously, by the authors of Chaco [HL95], where they recommend an eigentolerance of between 10^{-3} and 10^{-6} in most cases, for generating sufficiently accurate approximations for partitions using spectral methods. Although the edges cut dropped to 106 for some intermediate tolerances, the algorithm had not yet converged to the Fiedler vector.

REFERENCES

[BMR83] Brandt A., McCormick S., and Ruge J. (1983) Multigrid methods for differential eigenproblems. *SIAM J. Sci. Stat. Comput.* 4(2): 244–260.
[BS93] Barnard S. and Simon H. (1993) A fast multilevel implementation of

Table 1 Performance of various partitioners implementing spectral bisection on an airfoil grid (4253 nodes).

Partitioner	# of Edges cut	Time (sec)
Chaco	132	12.34
FAS MG	132	7.98
Inverse iteration	132	5.60
Metis	122	0.59

Table 2 Spectral bisection using 4-level FAS MG on an airfoil grid.

Reduct. residual	# V-cycles	# edges cut
10^{-0}	0	191
10^{-1}	1	106
10^{-2}	1	106
10^{-3}	2	138
10^{-4}	5	136
10^{-5}	7	132
10^{-6}	10	132

recursive spectral bisection. *Proc. 6th SIAM Conf. Parallel Proc. for Sci. Comp.* pages 711 –718.

[CCS97] Chan T. F., Ciarlet, Jr. P., and Szeto W. K. (1997) On the optimality of the median cut spectral bisection graph partitioning method. *SIAM J. Sci. Comput.* 18(3): 943–948.

[CGZ96] Chan T. F., Go S., and Zou J. (September 1996) Boundary treatments of multilevel methods on unstructured meshes. Technical Report CAM 96–30, Department of Mathematics, University of California at Los Angeles.

[Cia78] Ciarlet P. (1978) *The Finite Element Method for Elliptic Problems.* North-Holland.

[CSZ96] Chan T. F., Smith B., and Zou J. (April 1996) Overlapping Schwarz methods on unstructured meshes using non-matching coarse grids. *Numer. Math.* 73(2): 149–167.

[HL95] Hendrickson B. and Leland R. (July. 1995) The Chaco User's Guide Version 2.0. Technical Report SAND94-2692, Sandia Natl. Lab., Albuquerque, N.M.

[KK97] Karypis G. and Kumar V. (1997) Metis: A software package for partitioning unstructured graphs, partitioning meshes, and computing fill-reducing orderings of sparse matrices. available at http://www-users.cs.umn.edu/ karypis/metis/metis.html.

[Pot96] Pothen A. (1996) Graph partitioning algorithms with applications to scientific computing. In Keyes D. E., Sameh A. H., and Venkatakrishnan V. (eds) *In Parallel Numerical Algorithms.* Kluwer Academic Press.

21

Quasi-Simultaneous Coupling for Wing and Aerofoil Flow

E.G.M. COENEN[1]

INTRODUCTION

For the design and analysis of aerofoils and wings, a fast robust and accurate computer code is an essential tool. These days the prediction of aerodynamic characteristics is generally accomplished using Navier-Stokes simulation. Whilst this approach potentially offers generality, the computational cost involved currently limits their use for practical application. A possible alternative, is to use viscous-inviscid interaction (VII), which is an application of Schwarz non-overlapping domain-decomposition with Neumann-Dirichlet boundary conditions. VII methods have been shown to be very computationally economical and for many cases of aerodynamic design VII methods match the experimental data as well as Navier-Stokes simulation. However, despite the likelihood of there being similar advantages for three-dimensional problems, only a limited amount of research has been done in this area. The development of fully three-dimensional VII methods proves difficult [LW87].

The research described in this paper is preliminary work on two-dimensional quasi-simultaneous VII, carried out as a first stage in the development of a general three-dimensional viscous-inviscid interaction code. A Dirichlet interaction law has been derived for the coupling of the boundary layer to a panel method for two-dimensional incompressible flow that is more efficient than the well-known thin-aerofoil theory interaction law.

Quasi-simultaneous coupling [Vel81] has also been implemented in the DERA

1 Department of Aerospace Engineering, University of Bristol, Queen's Building, University Walk, Bristol BS8 1TR, U.K. email: E.G.M.Coenen@Bristol.ac.uk
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

Viscous Full Potential (VFP) program for the calculation of transonic flow over swept tapered wings [AS85]. This is a case where the three-dimensional boundary layer equations reduce to a system of ODE's and the calculations are therefore performed section-wise, similar to the 2D problem.

QUASI-SIMULTANEOUS COUPLING

Various coupling techniques have been proposed over the years and a comprehensive review of the various viscous-inviscid interaction schemes can be found in Lock and Williams [LW87]. The technique used here to couple the boundary layer to the inviscid region is the quasi-simultaneous (QS) method [Vel81]. QS has a very fast convergence rate and is able to calculate in separated regions. The QS method is described in more detail and put into a domain-decomposition context in the paper by Veldman and Lai in these proceedings [VL99]. The QS coupling resembles the direct method, where the velocity (or pressure) distribution is calculated in the inviscid region and then used to determine the displacement effect from the boundary layer equations. There is however a significant difference between direct and QS. The QS method performs the boundary layer calculations simultaneously with a local representation of the external flow, termed the interaction law.

This interaction law functions as a boundary condition to the boundary layer equations. In this way, the problems the direct method encounters at separation are avoided. The choice of interaction law is important since convergence speed depends on how well this law describes the interaction with the outer flow.

In most papers [CHR95, VLDBS90] in which the coupling is described between the boundary layer and the external flow, the interaction law used is based on thin-aerofoil theory. This gives a symmetric and diagonally dominant interaction law matrix. However, for most cases this interaction law is not close to the inviscid model, which results in a loss of convergence speed.

This dilemma is also described in Arnold and Thiede [AT93]. They instead derive a Laplace interaction law for the coupling of their Smith and Hess panel method [Kat91] and boundary layer. This gives them a more accurate representation. However, because of the panel method used, the calculations are performed in the panel midpoints causing the interaction law matrix to become less well behaved.

Here a new interaction law is derived from the influence matrix of the Dirichlet panel method [Kat91]. It allows the calculations to be performed in the panel endpoints, which leads again to a well behaved interaction law matrix.

The other application of quasi-simultaneous coupling that is to be discussed in this paper is of a swept tapered wing in transonic flow. For this case it is not easy to find an approximation to the external flow, as it is governed by the transonic small perturbation equation which is essentially non-linear. An interaction law based on thin-aerofoil theory has therefore initially been applied, to show the use of QS for this quasi-three-dimensional case. It is hoped to derive a more accurate representation in the near future.

DERIVATION OF THE DIRICHLET INTERACTION LAW

For the 2D steady incompressible case the interaction law will be derived from the inviscid formulation. This will be explained here in more detail. The inviscid flow solution is found by using a two-dimensional potential method [Kat91]. Along the aerofoils surface a source sheet of strength σ is distributed together with a doublet sheet of strength μ. The latter is also extended into the wake. With free stream potential ϕ_∞, the total potential ϕ^* can be written as:

$$\phi^* \quad = \quad \phi_\infty + \frac{1}{2\pi}\int_{S_B} \sigma \ln r \, ds - \frac{1}{2\pi}\int_{S_B+S_W} \mu \frac{\partial}{\partial n}(\ln r)\, ds, \tag{1}$$

in which n is the vector normal to the surface in the direction of μ, s a coordinate along the surface and r the distance between a point at s and a field point (x, y). Furthermore, S_B and S_W are the contours of the aerofoil and the wake respectively.

For an enclosed region, equation (1) has a zero normal velocity boundary condition. If there is however a boundary layer present, the outer flow does not see the real aerofoil but sees instead a thickened one. To take this displacement effect of the boundary layer into account the normal velocity boundary condition has to be changed. The normal velocity at the boundary should be taken to be equal to the transpiration velocity V_n:

$$\frac{\partial \phi^*}{\partial n} \quad = \quad V_n \; = \; -\frac{\partial u_e \delta^*}{\partial s}, \tag{2}$$

in which δ^* is the displacement thickness and u_e the velocity at the edge of the boundary layer. The minus sign is a result of n being the inward normal on the body and pointing down in the wake. Physically, the above transpiration velocity represents the stream wise rate of change of mass defect, $m = u_e\delta^*$.

In the wake, the displacement effect is a jump in normal velocity, ΔV, across a convenient line that divides the flows coming from the upper and lower surfaces of the aerofoil:

$$\Delta V \quad = \quad \frac{\partial}{\partial s}(u_{e_{upper}}\delta^*_{upper}) + \frac{\partial}{\partial s}(u_{e_{lower}}\delta^*_{lower}). \tag{3}$$

However, for simplicity the wake is modelled as just one layer, with thickness $\delta^* = \delta^*_{upper} + \delta^*_{lower}$ and tangential velocity $u_e = u_{e_{upper}} = u_{e_{lower}}$.

Instead of using the Neumann boundary condition (2), the Dirichlet boundary condition is used, by specifying the potential inside the body surface ϕ^*_{int} on the boundary. For collocation points inside the body, the internal potential is:

$$\phi^*_{int} \quad = \quad \phi_\infty + \frac{1}{2\pi}\int_{S_B+S_W} \sigma \ln r \, ds - \frac{1}{2\pi}\int_{S_B+S_W} \mu \frac{\partial}{\partial n}(\ln r)\, ds. \tag{4}$$

With the Dirichlet boundary condition $\phi^*_{int} = \phi_{int} + \phi_\infty = \phi_\infty$, where ϕ_{int}, the perturbation potential is set to zero, equation (4) simply becomes:

$$\frac{1}{2\pi}\int_{S_B+S_W} \mu \frac{\partial}{\partial n}(\ln r)\, ds \quad = \quad \frac{1}{2\pi}\int_{S_B+S_W} \sigma \ln r \, ds. \tag{5}$$

Consequently the source distribution σ is found which is, along the aerofoil's surface:

$$\sigma \quad = \quad \frac{\partial \phi^*_{int}}{\partial n} - \frac{\partial \phi^*}{\partial n} = U_\infty \cdot n + \frac{\partial u_e \delta^*}{\partial s}. \tag{6}$$

The term $\partial u_e \delta^* / \partial s$ on the right hand side is an effect of the presence of the boundary layer. It's value is to be determined from the boundary layer equations. In the wake (6) holds with $U_\infty \cdot n$ omitted.

To determine the problem uniquely, an implicit Kutta condition is used to define the doublet strength in the wake. The final result is a system of algebraic equations for the unknown doublet strengths on the surface of the aerofoil.

Discretisation

To determine the unknown doublet strength, the aerofoil's surface and wake are discretised into a number of straight line panels, with N nodes on the aerofoil and N_w nodes in the wake. Each panel will have a constant doublet strength, μ, and a constant source strength, σ, as defined previously. The collocation points are taken in the midpoints of the panels since defining them in the endpoints would lead to singularities.

From equation (5), after discretisation and manipulation, a relation for the doublet strength can be determined in the midpoints of the panels. However, as the tangential velocity of the outer flow (OF) is the gradient of the doublet strength, a relation for u_{eOF} can be found in the endpoints:

$$u_{eOF_i} \quad = \quad u_{e0_i} + \sum_{j=1}^{N+N_w-1} D(i,j) u_{e_i} \delta^*_i. \tag{7}$$

In the above expression, u_{e0} is the real inviscid velocity and the second part on the right hand side represents the disturbance caused by the presence of the boundary layer. Matrix D also contains the central discretisation over a panel of the gradient $\partial u_e \delta^* / \partial s$. The values of u_{eOF} at the trailing edge points are obtained by extrapolation.

In the wake, where the dividing streamline is taken parallel to the free stream, the velocity is obtained with the total potential:

$$u_{eOF_i} \quad = \quad \nabla \phi^* \cdot t|_i,$$

with t the vector tangential to the surface. As ϕ^* is defined in the midpoints of the panels, u_{eOF} in the wake is calculated there as well. By simple averaging, the value of u_{eOF} can be found in the endpoints. After discretisation and manipulation, an equation similar to (7) is found for u_{eOF} in the wake region.

The now determined influence matrix D has a suitable structure, being diagonally dominant and with off-diagonals having equal and opposite sign.

A simple interaction law, $u_e \approx I[\delta^*]$, containing only the local influence, can now be derived from the above determined equations for u_{eOF} on the surface and wake.

$$
\begin{aligned}
u_{e_i} &\approx u_{e0_i} + D(i,i-1)u_{e_{i-1}}\delta^*_{i-1} + D(i,i)u_{e_i}\delta^*_i + D(i,i+1)u_{e_{i+1}}\delta^*_{i+1}, \\
&\approx u_{e0_i} + \sum_{j=i-1}^{i+1} I(i,j)u_{e_j}\delta^*_j,
\end{aligned}
$$

where $I(i,j)$ is the tridiagonal interaction law matrix, containing the main and off-diagonal values of matrix D. The above interaction law equation together with the integral boundary layer equations are to be solved with a Newton method to give a new displacement thickness.

Results NACA4412 aerofoil

Calculations with the Dirichlet QS program Viscous-Inviscid Boundary Layer Analysis (VIBLA) have been performed for several aerofoils. Here results will be presented for the NACA 4412 section at $Re = 4.17 \times 10^6$. The results shown in figure 1 are compared with experiment and calculations from Hastings and Williams [HW87].

In the first figure the lift coefficient is shown. Up to an incidence of 11^o the results correspond very well with experiment and with the results of Hastings and Williams. However, near stall the prediction is slightly less accurate as Cl_{max} is obtained too early. After an incidence of 12.6^o no calculations could be performed.

In the figure top right the pressure distribution is compared with experimental results at an incidence of 12.5^o. The results are in good agreement. Only in the trailing edge region, where there is separation, do the results differ. In the figure bottom left the displacement thickness is shown and compares very well with experiment. Near the leading edge the displacement thickness is slightly under predicted, which is possibly a result of not having modelled the presence of the transition strip. The shape factor is shown in the figure bottom right. In the region of separation, i.e. $H > 4$, the shape factor is over predicted. The poor shape factor predictions are due to the empirical closure relation for H_1. Separation took place at $x/c = 0.8$, which was also predicted by experiment.

DERA-VFP CODE FOR SWEPT, TAPERED WINGS IN TRANSONIC FLOW

The DERA Viscous Full Potential (VFP) code [Smi89] is based on a viscous-coupled method for computing the compressible flow over wing-body combinations. It combines a full-potential flow solver and an integral boundary layer method. The boundary layer equations are derived in a local co-ordinate system for each chord wise strip on the wing. This derivation assumes that each section of the wing behaves as if it were part of a larger simply-tapered wing with the same leading and trailing edge sweeps as the local section. Ashill and Smith [AS85] show that, for conventional wings, this gives results for the boundary layer development that are very similar to those obtained using a fully three-dimensional version of the boundary layer equations.

In the original code, the interaction between the external inviscid flow and the viscous regions was governed by the boundary-layer equations through the semi-inverse

(SI) method. The aim of the work described here was to replace the semi-inverse method with a quasi-simultaneous (QS) coupling method. The motivation for this study was the, already observed, improved performance (in terms of convergence rate and robustness) of the QS scheme when compared with the SI method (*e.g.*, King and Williams [KW88]) for two-dimensional flows.

The interaction law VFP-QS

As previously explained, the QS method requires a suitable locally linearised representation of the external flow to be defined. For incompressible flow the most commonly used interaction law is derived from thin-aerofoil theory. For subsonic flow ($M < 0.7$, say) the thin-aerofoil representation is still valid if a compressibility correction term, such as the Prandtl-Glauert rule, is used [VLDBS90]. The linear relation between u_e and δ^* can be found as follows,

$$u_{e_i} = u_{e0_i} + u'_i,$$

where u_{e0} is the inviscid velocity and u' is the velocity perturbation induced by the boundary layer. Now, by the Prandtl-Glauert law:

$$u'_i = \frac{u'_{i incompr}}{\sqrt{1 - M_\infty^2}},$$

and $u'_{i incompr}$ may be found locally with the thin-aerofoil theory solution,

$$u'_{i incompr} \approx \frac{1}{\pi} \int_{x_{i-1}}^{x_{i+1}} \frac{V_n d\xi}{(x_i - \xi)},$$

in which V_n is the transpiration velocity.

It would be more computationally efficient to use an interaction law that takes account of the local flow conditions, including the local Mach number of the external flow. For transonic flow it is not easy to find a more accurate interaction law based on a linear representation for the external flow from the transonic small perturbation (TSP) equation. Not least because the TSP equation is essentially nonlinear. However, the function of the interaction law within a QS procedure is not to define the solution, but to 'point' the solution in the right direction. The more accurate the interaction law is in representing the outer flow, the better it can point the solution in the right direction and the faster the method will converge. As the interaction law described above for transonic flow is not representative, a penalty is probably introduced in terms of slower convergence.

Runge-Kutta-4 versus Newton in VFP

A four-stage Runge-Kutta (RK4) method is used in the original VFP program to solve the set of boundary layer equations which is not suitable for use with a QS scheme. As described above, an extra equation, the interaction law, has to be solved simultaneously with the boundary layer equations if QS coupling is applied. This would have to be applied at each stage of the RK4 scheme (i.e. at each of the four stages between grid points). To avoid the calculation of an interaction law at all the intermediate stages between grid points, a Newton iterative scheme has been used.

The use of a Newton scheme however has the disadvantage that the scheme is first-order accurate (using upwinded differences), whereas the RK4 scheme is fourth-order accurate. This consequently leads to a loss of formal accuracy.

Results W4 wing

Results are shown in figures 2, 3 and 4 for $M = 0.78$, $Re = 13.3 \times 10^6$ at an incidence $\alpha = 1^o$. Comparing solutions from the SI and QS versions of the VFP code, the agreement is generally good, particularly in view of the quite different integration procedures being used to compute the boundary layer in each case. In particular the shock is predicted by the QS method to be at the same location as predicted by the SI method. The solution of the QS scheme differs only slightly at the trailing edge and in the wake.

The calculations for both the SI scheme and the QS scheme took similar computer time, with the number of iterations being fixed. However, little experimentation has been carried out with the numerical parameters of the QS scheme, and there is scope for optimisation of these to reduce the run time.

CONCLUSIONS

Quasi-simultaneous coupling has been applied successfully for two-dimensional incompressible flow and quasi-three-dimensional transonic flow. For the 2D incompressible case a new Dirichlet interaction law has been derived and for the transonic case a thin-aerofoil theory interaction law with compressibility correction term has been used. For the latter case it is hoped to improve the interaction law by finding a more suitable approximation, to speed up convergence. The methods used are to be extended to the development of a fully three-dimensional interaction method.

ACKNOWLEDGEMENTS

The research was funded by the University of Bristol, the Engineering and Physical Sciences Research Council and the Defence Evaluation and Research Agency. The author would like to thank S.P. Fiddes, A.E.P. Veldman and J. Doherty for their support during the research.

REFERENCES

[AS85] Ashill P. and Smith P. (1985) An integral method for calculating the effects on turbulent boundary-layer development of sweep and taper. *Aeron. J.* 89: 43–54.

[AT93] Arnold F. and Thiede F. (1993) A laplace interaction law for the computation of viscous airfoil flow in low and high speed aerodynamics. *AIAA-93-3462-CP* pages 490–498.

[CHR95] Cebeci T., Hefazi H., and Roknaldin F. (1995) Predicting stall and post-stall behaviour of airfoils at low mach numbers. *AIAA J.* 33: 595–602.

[HW87] Hastings R. and Williams B. (1987) Studies of the flow field near a naca 4412 aerofoil at nearly maximum lift. *Aeron. J.* 91: 29–44.

[Kat91] Katz J. & Plotkin A. (1991) *Low-speed aerodynamics; from wing theory to panel method.* McGraw-Hill.

[KW88] King D. and Williams B. (1988) Developments in computational methods for high-lift aerodynamics. *Aeron. J.* pages 265–288.

[LW87] Lock R. and Williams B. (1987) Viscous-inviscid interactions in external aerodynamics. In *Progress in Aerospace Sciences*, number 24, pages 51–171.

[Smi89] Smith P. (1989) A viscous package for attached and separated flows on swept and tapered wings. TR 89027, RAE.

[Vel81] Veldman A. (1981) New, quasi-simultaneous method to calculate interacting boundary layers. *AIAA J.* 19: 79–85.

[VL99] Veldman A. and Lai C.-H. (1999) Viscous-inviscid interaction: domain decomposition avant la lettre. In Lai C.-H., Bjø rstad P., Cross M., and Widlund O. (eds) *Eleventh International Conference on Domain Decomposition Methods.*

[VLDBS90] Veldman A., Lindhout J., De Boer E., and Somers M. (1990) Vistrafs: A simulation method for strongly interacting viscous transonic flow. In Cebeci T. (ed) *Numerical and Physical Aspects of Aerodynamic Flows IV*, pages 37–51.

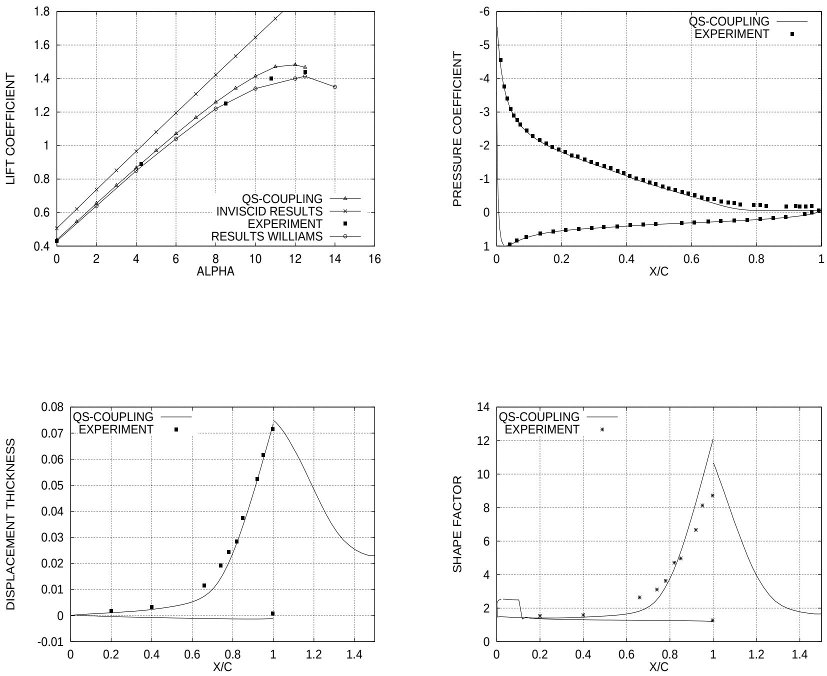


Figure 1 *Results for NACA 4412 at Re* $= 4.17 \times 10^6$

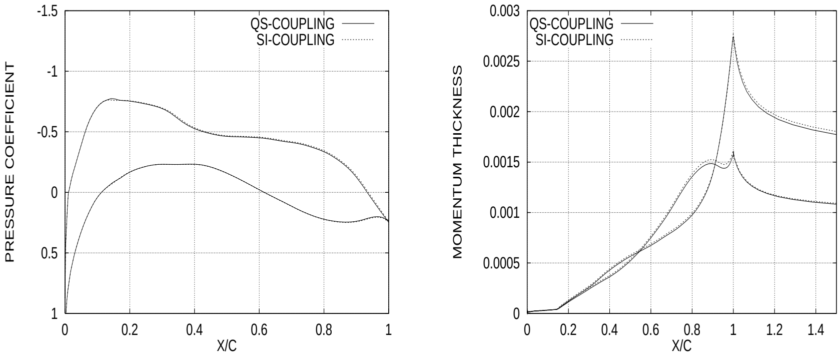


Figure 2 *Results W4 wing at* $\eta = 0.132$, $Re = 13.3 \times 10^6$, $M = 0.78$

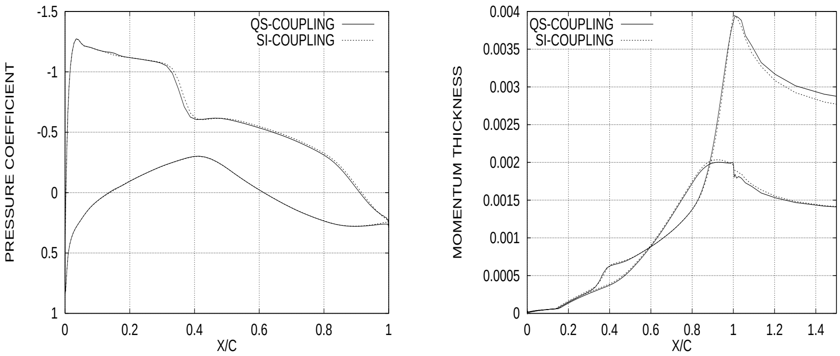


Figure 3 *Results W4 wing at* $\eta = 0.534$, $Re = 13.3 \times 10^6$, $M = 0.78$

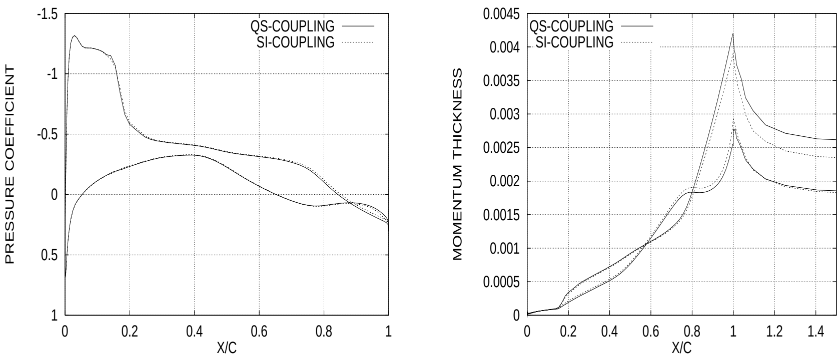


Figure 4 *Results W4 wing at* $\eta = 0.988$, $Re = 13.3 \times 10^6$, $M = 0.78$

22

Domain Decomposition Methods for a System of Coupled Acoustic and Elastic Helmholtz Equations

Peter Cummings[1], & Xiaobing Feng[2]

Introduction

This paper develops some parallelizable non–overlapping domain decomposition iterative methods for a system of coupled acoustic and elastic Helmholtz equations which describes the interaction of an inviscid fluid and an elastic solid in the frequency domain. Two classes of iterative methods are proposed for decoupling the whole domain problem into fluid and solid subdomain problems. The crux of the methods is to replace the physical interface conditions with equivalent relaxation conditions as the transmission conditions. The utility of these methods is established by showing their strong convergence in the energy norm of the underlying fluid–solid interaction problem. Numerical experiments are provided to validate the analysis and to show the effectiveness of the methods.

The problems of wave propagation in composite media have long been subjects of both theoretical and practical studies. Important applications of such problems are found in inverse scattering, elastoacoustics, geosciences, and oceanography. For some recent developments on modeling, mathematical and numerical analysis, and computational simulations, we refer to [Bou87, DOAG91, FLW97a, FLW97b] and the references therein.

Because of existence of the physical interface, it is natural to use non–overlapping

1 Department of Mathematics, The University of Tennessee, Knoxville, TN 37996. cummings@math.utk.edu
2 Address same as above. xfeng@math.utk.edu.
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

domain decompositions method to solve the fluid–solid interaction problem. In fact, non–overlapping domain decomposition methods have been successfully used to solve several coupled boundary value problems from scientific applications. See [QPV92] and the references therein. The non–overlapping domain decomposition methods developed in this paper are based on the idea of using convex combinations of the original physical interface conditions to transmit information between subdomains. See [BF97, Des91, Fen97, Lio90, SBG96] for expositions and discussions on this approach for homogeneous problems. It is more delicate to apply the idea to the heterogeneous fluid–solid interaction problem because using straightforward combinations of the original interface conditions as transmission conditions may lead to divergent iterative procedures. An implementation issue is also addressed in the paper (cf. [BF97]); we show that the difficulty of explicitly computing fluxes on the interface can be avoided through a simple modification.

The organization of this paper is as follows. In Section 2, the system of coupled acoustic Helmholtz equation and elastic Helmholtz equations is introduced as a special form of the fluid–solid interaction model proposed in [FLW97b]. In Section 3, two classes of non–overlapping domain decomposition algorithms are proposed for solving the coupled system. Strong convergence in the energy norm of the underlying fluid–solid interaction problem is established for the iterative methods. Finally, some numerical test results on the methods are presented in Section 4.

Description of the problem

Wave propagation in a composite medium consisting of a pure (acoustic) fluid part and a pure (elastic) solid part is described by the following system of coupled acoustic and elastic wave equations (cf. [FLW97b])

$$
\begin{aligned}
&\tfrac{1}{c^2}P_{tt} - \Delta P = G_f, && \text{in } \Omega_f, && (1)\\
&\rho_s \mathbf{U}_{tt} - \operatorname{div}(\sigma(\mathbf{U})) = \mathbf{G}_s, && \text{in } \Omega_s, && (2)\\
&\tfrac{\partial P}{\partial n_f} - \rho_f \mathbf{U}_{tt}\cdot n_s = 0, && \text{on } \Gamma, && (3)\\
&\sigma(\mathbf{U})n_s - \tfrac{\partial P}{\partial n_f} = 0, && \text{on } \Gamma, && (4)\\
&\tfrac{1}{c}P_t + \tfrac{\partial P}{\partial n_f} = 0, && \text{on } \Gamma_f, && (5)\\
&\mathcal{A}_s \mathbf{U}_t + \sigma(\mathbf{U})n_s = 0, && \text{on } \Gamma_s, && (6)\\
&P(x,0) = P_0(x), \quad P_t(x,0) = P_1(x), && \text{in } \Omega_f, && (7)\\
&\mathbf{U}(x,0) = \mathbf{U}_0(x), \quad \mathbf{U}_t(x,0) = \mathbf{U}_1(x), && \text{in } \Omega_s, && (8)
\end{aligned}
$$

where

$$\sigma(\mathbf{U}) = \lambda_s(\operatorname{div}\mathbf{U})I + 2\mu_s\epsilon(\mathbf{U}), \qquad \varepsilon(\mathbf{U}) = \frac{1}{2}[\nabla\mathbf{U} + (\nabla\mathbf{U})^T]. \tag{9}$$

In the system Ω_f and Ω_s denote the fluid and solid domains respectively, and $\Omega = \Omega_f \cup \Omega_s \subset \mathbf{R}^N$ $(N = 2, 3)$ denotes the domain of the whole composite medium. $\Gamma_f = \partial\Omega_f \setminus \Gamma$, $\Gamma_s = \partial\Omega_s \setminus \Gamma$ and $\Gamma = \partial\Omega_f \cap \partial\Omega_s$ is the interface between two media. P is the pressure in Ω_f, $\mathbf{U}$ is the displacement vector in Ω_s, and c is the wave speed in the fluid medium. ρ_i $(i = f, s)$ denotes the density of Ω_i, n_i $(i = f, s)$ denotes the unit outward normal to $\partial\Omega_i$. $\lambda_s > 0$ and $\mu_s \geq 0$ are the Lamé constants of Ω_s. Equation (9) is the constitutive relation for Ω_s. I stands for the $N \times N$ identity matrix. The boundary conditions in (5) and (6) are the first order absorbing boundary conditions for the acoustic and elastic waves respectively. These boundary conditions are transparent to waves arriving normally at the boundary (cf. [EM79]). Finally, equations (3) and (4) are the interface conditions which describe the interaction between the fluid and the solid.

For many application problems, one is asked to find solutions of (1)–(9) of the form $\mathbf{U}(t, x) = \mathbf{u}(x)e^{i\omega t}$, $P(t, x) = p(x)e^{i\omega t}$ (i.e. time harmonic solutions) for some given time harmonic sources $G_f = g_f e^{i\omega t}$ and $\mathbf{G}_s = \mathbf{g}_s e^{i\omega t}$. Substituting $\mathbf{U}(t, x) = \mathbf{u}(x)e^{i\omega t}$ and $P(t, x) = p(x)e^{i\omega t}$ into (1)–(9) yields the following system of equations

$$
\begin{aligned}
-\tfrac{\omega^2}{c^2}p - \Delta p &= g_f, && \text{in } \Omega_f, && (10)\\
\tfrac{i\omega}{c}p + \tfrac{\partial p}{\partial n_f} &= 0, && \text{on } \Gamma_f, && (11)\\
-\omega^2\rho_s\mathbf{u} - \operatorname{div}(\sigma(\mathbf{u})) &= \mathbf{g}_s, && \text{in } \Omega_s, && (12)\\
i\omega\mathcal{A}_s\mathbf{u} + \sigma(\mathbf{u})n_s &= 0, && \text{on } \Gamma_s, && (13)\\
\tfrac{\partial p}{\partial n_f} + \omega^2\rho_f\mathbf{u}\cdot n_s &= 0, && \text{on } \Gamma, && (14)\\
\sigma(\mathbf{u})n_s - pn_f &= 0, && \text{on } \Gamma. && (15)
\end{aligned}
$$

Clearly, the above is the system of coupled scalar Helmholtz equation in the fluid subdomain and the (elastic) vector Helmholtz equations in the solid subdomain. The main objective of this paper is to develop some domain decomposition iterative methods for solving the system. Before introducing our domain decomposition methods, we first state the following existence and uniqueness results for the system.

Theorem 1 *The boundary value problem (10)–(15) has a unique solution* $(p, \mathbf{u}) \in H^1(\Omega_f) \times \mathbf{H}^1(\Omega_s)$ *provided that* $g_f \in L^2(\Omega_f)$, $\mathbf{g}_s \in \mathbf{L}^2(\Omega_s)$.

The proof of Theorem 1 is based the Fredholm Alternative Principle. Notice that the coupled system is elliptic but not coercive. Due to space limitation, we will not give the proof here. We refer readers to ([MIB95]) to see a proof of similar type.

Non–overlapping domain decomposition methods

Due to the existence of the physical interface and the heterogeneous nature of the problem, it is natural to solve the coupled system using a non–overlapping domain decomposition method. The goal of this section is to present some parallelizable iterative procedures for the problem based on non–overlapping domain decomposition. We show the utility of these iterative algorithms by establishing their convergence in the energy space of the underlying fluid–solid interaction problem.

As in the comparable methods of ([BF97, Des91, Fen97]) , the main idea here is to replace the original physical interface conditions (14)–(15) with the following equivalent Robin type interface conditions

$$\frac{\partial p}{\partial n_f} + \alpha p = -\omega^2 \rho_f \mathbf{u} \cdot n_s - \alpha \sigma(\mathbf{u}) n_s \cdot n_s, \quad \text{on } \Gamma, \tag{16}$$

$$\sigma(\mathbf{u}) n_s \cdot n_s + \beta \mathbf{u} \cdot n_s = -p - \frac{\beta}{\omega^2 \rho_f} \frac{\partial p}{\partial n_f}, \quad \text{on } \Gamma, \tag{17}$$

$$\sigma(\mathbf{u}) n_s \cdot \tau_s = 0, \quad \text{on } \Gamma, \tag{18}$$

for any constant α, β such that $\alpha\beta \neq \omega^2 \rho_f$.

Based on the new form of the interface conditions, we propose the following iterative algorithms. Algorithm 1 can be regarded as a Jacobi type algorithm, and Algorithm 2 as a Gauss-Seidel type algorithm.

Algorithm 1
Step 1 $\forall\ p^0 \in H^1(\Omega_f)$, $\mathbf{u}^0 \in \mathbf{H}^1(\Omega_s)$.
Step 2 For $k \geq 0$, define $(p^{k+1}, \mathbf{u}^{k+1})$ such that

$$\begin{aligned}
-\frac{\omega^2}{c^2} p^{k+1} - \Delta p^{k+1} &= g_f, && \text{in } \Omega_f,\\
\frac{i\omega}{c} p^{k+1} + \frac{\partial p^{k+1}}{\partial n_f} &= 0, && \text{on } \Gamma_f,\\
\frac{\partial p^{k+1}}{\partial n_f} + \alpha p^{k+1} &= -\omega^2 \rho_f \mathbf{u}^k \cdot n_s - \alpha \sigma(\mathbf{u}^k) n_s \cdot n_s, && \text{on } \Gamma;\\
-\omega^2 \rho_s \mathbf{u}^{k+1} - \operatorname{div}(\sigma(\mathbf{u}^{k+1})) &= \mathbf{g}_s, && \text{in } \Omega_s,\\
i\omega \mathcal{A}_s \mathbf{u}^{k+1} + \sigma(\mathbf{u}^{k+1}) n_s &= 0, && \text{on } \Gamma_s,\\
\sigma(\mathbf{u}^{k+1}) n_s \cdot n_s + \beta \mathbf{u}^{k+1} \cdot n_s &= -p^k - \frac{\beta}{\omega^2 \rho_f} \frac{\partial p^k}{\partial n_f}, && \text{on } \Gamma,\\
\sigma(\mathbf{u}^{k+1}) n_s \cdot \tau_s &= 0, && \text{on } \Gamma.
\end{aligned}$$

Algorithm 2
Step 1 $\forall\ \mathbf{u}^0 \in \mathbf{H}^1(\Omega_s)$.
Step 2 For $k \geq 0$, define $(p^k, \mathbf{u}^{k+1})$ such that

$$\begin{aligned}
-\frac{\omega^2}{c^2} p^k - \Delta p^k &= g_f, && \text{in } \Omega_f,\\
\frac{i\omega}{c} p^k + \frac{\partial p^k}{\partial n_f} &= 0, && \text{on } \Gamma_f,\\
\frac{\partial p^k}{\partial n_f} + \alpha p^k &= -\omega^2 \rho_f \mathbf{u}^k \cdot n_s - \alpha \sigma(\mathbf{u}^k) n_s \cdot n_s, && \text{on } \Gamma;\\
-\omega^2 \rho_s \mathbf{u}^{k+1} - \operatorname{div}(\sigma(\mathbf{u}^{k+1})) &= \mathbf{g}_s, && \text{in } \Omega_s,\\
i\omega \mathcal{A}_s \mathbf{u}^{k+1} + \sigma(\mathbf{u}^{k+1}) n_s &= 0, && \text{on } \Gamma_s,\\
\sigma(\mathbf{u}^{k+1}) n_s \cdot n_s + \beta \mathbf{u}^{k+1} \cdot n_s &= -p^k - \frac{\beta}{\omega^2 \rho_f} \frac{\partial p^k}{\partial n_f}, && \text{on } \Gamma,\\
\sigma(\mathbf{u}^{k+1}) n_s \cdot \tau_s &= 0, && \text{on } \Gamma.
\end{aligned}$$

The main result of this section is the following convergence theorem.

Theorem 2 *Suppose $\alpha = i\omega\hat{\alpha}$, $\beta = i\omega\hat{\beta}$ where $\hat{\alpha} > 0$ and $\hat{\beta} > 0$. Then the sequence $\{(p^k, \mathbf{u}^k)\}$ generated by Algorithm 1 and Algorithm 2 satisfies*

(1). $p^k \longrightarrow p$ strongly in $H^1(\Omega_f)$, *(2). $\mathbf{u}^k \longrightarrow \mathbf{u}$ strongly in $\mathbf{H}^1(\Omega_s)$.*

Proof: Let $r^n = p - p^n$ and $\mathbf{e}^n = \mathbf{u} - \mathbf{u}^n$. The proof follows the same idea used in ([BF97, Des91, Fen97]). Here the "pseudo–energy" E_n for Algorithm 1 is defined as

$$E_n = \left\| \frac{\partial r^n}{\partial n_f} + \alpha r^n \right\|^2_{L^2(\Gamma)} + \left(\frac{\omega^2 \rho_f}{\beta} \right)^2 \|\sigma(\mathbf{e}^n) n_s \cdot n_s + \beta \mathbf{e}^n \cdot n_s\|^2_{L^2(\Gamma)},$$

and for Algorithm 2 is given by

$$E_n = \left\| \frac{\partial r^{n-1}}{\partial n_f} + \alpha r^{n-1} \right\|^2_{L^2(\Gamma)} + \left(\frac{\omega^2 \rho_f}{\beta} \right)^2 \|\sigma(\mathbf{e}^n) n_s \cdot n_s + \beta \mathbf{e}^n \cdot n_s\|^2_{L^2(\Gamma)}.$$

It is not hard to show that the "pseudo–energy" E_n is decreasing, and it decreases fast enough to guarantee the desired convergence of the iterates $\{(p^n, \mathbf{u}^n)\}$ when the relaxation parameters are chosen appropriately. See [BF97] for the details of a similar argument.

The above algorithms have a drawback with respect to implementation. For instance, solving for $(p^k, \mathbf{u}^{k+1})$ in Step 2 of Algorithm 2 requires the normal derivatives $\sigma(\mathbf{u}^k) n_s \cdot n_s$ and $\frac{\partial p^k}{\partial n_f}$ on the interface. Consequently, one is forced to use non–standard or hybrid finite element methods in order to implement Algorithm 1 and 2. This drawback can be easily avoided through a simple modification. For example, the modification to Algorithm 2 is given as follows.

Algorithm 3
Step 1 $\forall h_f^0 \in L^2(\Gamma)$.
Step 2 For $k \geq 0$, define $(p^k, \mathbf{u}^{k+1})$ such that

$$\begin{aligned}
-\tfrac{\omega^2}{c^2} p^k - \Delta p^k &= g_f, && \text{in } \Omega_f, \\
\tfrac{i\omega}{c} p^k + \tfrac{\partial p^k}{\partial n_f} &= 0, && \text{on } \Gamma_f, \\
\tfrac{\partial p^k}{\partial n_f} + \alpha p^k &= h_f^k, && \text{on } \Gamma, \\
h_s^k &= (-1 + \tfrac{\alpha\beta}{\omega^2 \rho_f}) p^k - \tfrac{\beta}{\omega^2 \rho_f} h_f^k; && \text{on } \Gamma; \\
-\omega^2 \rho_s \mathbf{u}^{k+1} - \operatorname{div}(\sigma(\mathbf{u}^{k+1})) &= \mathbf{g}_s, && \text{in } \Omega_s, \\
i\omega \mathcal{A}_s \mathbf{u}^{k+1} + \sigma(\mathbf{u}^{k+1}) n_s &= 0, && \text{on } \Gamma_s, \\
\sigma(\mathbf{u}^{k+1}) n_s \cdot n_s + \beta \mathbf{u}^{k+1} \cdot n_s &= h_s^k, && \text{on } \Gamma, \\
\sigma(\mathbf{u}^{k+1}) n_s \cdot \tau_s &= 0, && \text{on } \Gamma, \\
h_f^{k+1} &= (-\omega^2 \rho_f + \alpha\beta) \mathbf{u}^{k+1} \cdot n_s - \alpha h_s^k, && \text{on } \Gamma.
\end{aligned}$$

The equivalence of Algorithm 2 and 3 can be seen formally from the following identity

$$\begin{aligned}
\sigma(\mathbf{u}^{k+1}) n_s \cdot n_s + \beta \mathbf{u}^{k+1} \cdot n_s = h_s^k &= \left(-1 + \frac{\alpha\beta}{\omega^2 \rho_f} \right) p^k - \frac{\beta}{\omega^2 \rho_f} {h_f}^k \\
= \left(-1 + \frac{\alpha\beta}{\omega^2 \rho_f} \right) p^k - \frac{\beta}{\omega^2 \rho_f} \left(\frac{\partial p^k}{\partial n_f} + \alpha p^k \right) &= -p^k - \frac{\beta}{\omega^2 \rho_f} \frac{\partial p^k}{\partial n_f}.
\end{aligned}$$

The modification to Algorithm 1 can be constructed similarly. Following the proof of Theorem 2, it can be shown that the statement of Theorem 2 also holds for modified algorithms.

Numerical Experiments

In this section we present the numerical results of two test problems in order to validate the theoretical analysis established in the previous section and to demonstrate the effectiveness of the proposed domain decomposition algorithms. To test the domain decomposition algorithm, we perform two sets of numerical experiments. In the first set, the true solution is known; in the second, the true solution is not known explicitly. In all experiments, $\Omega_f = [0,1] \times [0,1]$, $\Omega_s = [1,2] \times [0,1]$ and the mesh size is approximately 0.1. We use Algorithm 3 to generate all domain decomposition solutions, and the piecewise linear finite element method is employed as the discretization method in each experiment.

For the first set of experiments, we chose the following source functions

$$\mathbf{g}_s \equiv 0.$$
$$g_f = \sin^2(\pi x)\sin^2(\pi y)(4\pi^2 - \omega^2) - 2\pi^2[\sin^2(\pi x)\cos^2(\pi y) + \cos^2(\pi x)\sin^2(\pi y)].$$

Given the above sources, it is easy to show that the true solution is $\mathbf{u}(x,y) = 0$, $p(x,y) = \sin^2(\pi x)\sin^2(\pi y)$

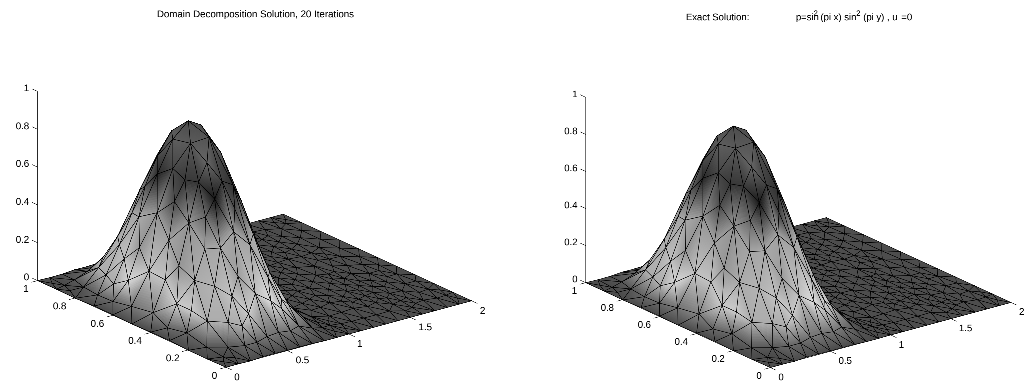


Figure 1 Exact solution vs. domain decomposition solution

Figure 1 shows the true solution and the domain decomposition solution. Table 1 shows the L^∞–norm and the L^2–norm of the error. In table 1, $(p, \mathbf{u})$ denotes the true solution of (10)–(15). The numerical results indicate that after about 5 iterations the domain decomposition solution already has the accuracy of the global finite element solution (see [FLW97a]).

n	$\|\|p^n - p\|\|_{L^2}$	$\|\|p^n - p\|\|_{L^\infty}$	$\|\|\mathbf{u}^n - \mathbf{u}\|\|_{L^2}$	$\|\|\mathbf{u}^n - \mathbf{u}\|\|_{L^\infty}$
5	.00774713	.00150090	.00737340	.00184845
10	.00750596	.00151935	.00731775	.00172242
20	.00750852	.00152213	.00731816	.00172427

Table 1 Errors of domain decomposition solution

In the second experiment, we choose the source functions

$$g_f(x,y) = x^2 e^y, \qquad \mathbf{g}_s(x,y) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

In this test, because the true solution of the problem is not known explicitly and the whole domain finite element solution is not easily obtained, we tested the accuracy of our domain decomposition solutions by calculating the relative error of successive iterates. Figure 22 shows the domain decomposition solution after 30 iterations. Note that the graph shows the real part of the first coordinate of the solution $\mathbf{u}$. Table 2 shows the L^2–norm and the L^∞–norm of the relative errors of the domain decomposition solutions after 5, 10, 20 and 30 iterations, respectively.

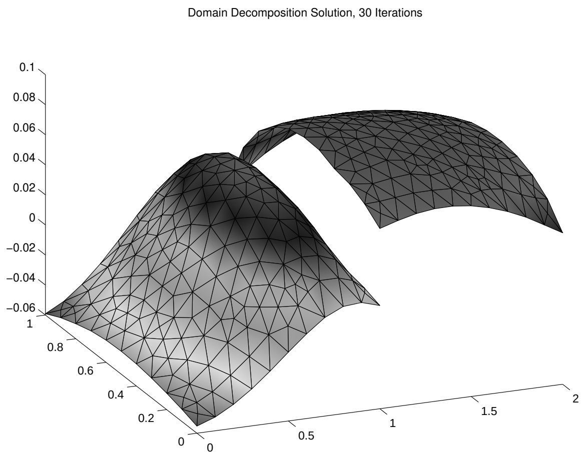


Figure 2 Domain decomposition solution after 30 interactions

n	$\frac{\|p^n - p^{n+1}\|_{L^2}}{\|p^{n+1}\|_{L^2}}$	$\frac{\|p^n - p^{n+1}\|_{L^\infty}}{\|p^{n+1}\|_{L^\infty}}$	$\frac{\|\mathbf{u}^n - \mathbf{u}^{n+1}\|_{L^2}}{\|\mathbf{u}^{n+1}\|_{L^2}}$	$\frac{\|\mathbf{u}^n - \mathbf{u}^{n+1}\|_{L^\infty}}{\|\mathbf{u}^{n+1}\|_{L^\infty}}$
5	.00762755	.01312341	.00133775	.00313325
10	6.825e-05	2.292e-05	1.697e-05	7.146e-05
20	6.103e-08	2.175e-07	1.736e-08	7.198e-08
30	8.856e-11	3.640e-10	2.442e-11	9.257e-11

Table 2 Relative errors of domain decomposition solution

REFERENCES

[BF97] Bennethum L. S. and Feng X. (1997) A domain decomposition method for solving a helmholtz–like problem in elasticity based on the wilson nonconforming element. *R.A.I.R.O., Anal. Numér.* 31: 1–25.

[Bou87] Boujot J. (1987) Mathematical formulation of fluid–structure interaction problems. *R.A.I.R.O., Anal. Numér.* 21: 239–260.

[Des91] Després B. (1991) Domain decomposition method and helmholtz problem. In *Mathematical and Numerical Aspects of Wave Propagation Phenomena*, pages 44–52. SIAM, Philadelphia.

[DOAG91] Demkowicz L., Oden J. T., Ainsworth M., and Geng P. (1991) Solution of elastic scattering problems in linear acoustic using h-p boundary element methods. *J. Comp. Appl. Math.* 36: 29–63.

[EM79] Engquist B. and Majda A. (1979) Radiation boundary conditions for acoustic and elastic wave calculations. *Comm. Pure Appl. Math.* 32: 313–357.

[Fen97] Feng X. (1997) Analysis of a domain decomposition method for the nearly elastic wave equations based on mixed finite element methods. *IMA J. Numer. Anal.* 14: 1–22.

[FLW97a] Feng X., Lee P., and Wei Y. (1997) Finite element methods and domain decomposition algorithms for a fluid–solid interaction problem. Technical Report 1496, Institute of Mathematics and Applications, University of Minnesota.

[FLW97b] Feng X., Lee P., and Wei Y. (1997) Formulation and mathematical analysis of a fluid–solid interaction problem. Technical Report 1495, Institute of Mathematics and Applications, University of Minnesota.

[Lio90] Lions P. L. (1990) On the schwartz alternating method iii. In *Proceedings of Third International Symposium on Domain Decomposition Method for Partial Differential Equations*, pages 202–223. SIAM, Philadelphia.

[MIB95] Makridakis C., Ihlenburg F., and Babuska I. (1995) Analysis and finite element methods for a fluid–structure interaction problem in one dimension. Technical Report BN-1183, IPST, University of Maryland at College Park.

[QPV92] Quarteroni A., Pasquarelli F., and Valli A. (1992) Heterogeneous domain decomposition: principles, algorithms, applications. In *Proceedings of Fifth International Symposium on Domain Decomposition Method for Partial Differential Equations*, pages 129–150. SIAM, Philadelphia.

[SBG96] Smith B. E., Bjorstad P. E., and Gropp W. D. (1996) *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations.* Cambridge University Press, New York.

23

On the convergence of the generalized asynchronous multisplitting block two-stage relaxation methods for the large sparse systems of mildly nonlinear equations

D.J. Evans[1], Zhong-Zhi Bai[2]

Introduction

Consider the mildly nonlinear system

$$Ax = G(x), \tag{1}$$

where $A \in \mathbb{R}^{n\times n}$ is a large sparse nonsingular matrix and $G : \mathbb{R}^n \to \mathbb{R}^n$ is a nonlinear mapping of certain smooth properties.

To solve this system efficiently, Bai [1] established a class of sequential two-stage iterative methods by taking into account concrete properties of the involved matrix

1 *Department of Computing, The Nottingham Trent University Burton Street, Nottingham NG1 4BU, England Email: D.J. Evans@lboro.ac.uk*
2 Supported by the National Natural Science Foundation project 19601036.
State Key Laboratory of Scientific/Engineering Computing Institute of Computational Mathematics and Scientific/Engineering Computing The Academy of Mathematics and Systems Sciences Chinese Academy of Sciences, P.O.Box 2719, Beijing 100080, P.R.China Email: bzz@lsec.cc.ac.cn
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

and mapping. Then, based on the matrix multisplitting technique, Bai [2] presented efficient parallel generalizations of the above sequential two-stage iterative methods. To exploit the parallel efficiency of the high-speed multiprocessor systems as far as possible, Bai and Huang [7] further proposed the asynchronous multisplitting two-stage iterative methods. These asynchronous methods have the potential of converging much faster than their synchronous counterparts in [2], in particular, when there is load imbalance. When the matrix $A \in \mathbb{R}^{n\times n}$ is a point H-matrix and the mapping $G : \mathbb{R}^n \to \mathbb{R}^n$ is a point P-bounded mapping, the convergence of the afore-mentioned two-stage iterative methods were discussed in detail in [1], [2] and [7], respectively, under suitable conditions imposed upon the multisplittings of the matrix $A \in \mathbb{R}^{n\times n}$. Moreover, several different models of asynchronous multisplitting block two-stage relaxation methods for solving the mildly nonlinear system (1) were proposed in [11], and their global convergence properties were studied in depth for the case when the matrix $A \in \mathbb{R}^{n\times n}$ is a block H-matrix of different types and the mapping $G : \mathbb{R}^n \to \mathbb{R}^n$ is a block P-bounded mapping. All these results favoured a variety of practical sequential and parallel methods for solving the large sparse block system of mildly nonlinear equations (1), and they also have reliable theoretical guarantees for the convergence of these methods. We remark that the parallel multisplitting two-stage iterative methods for the large sparse systems of linear equations were discussed in [8], [10] and [13].

In this paper, based on the above existing results we will discuss a class of generalized asynchronous multisplitting two-stage block relaxation methods for solving the block system of mildly nonlinear equations (1). For the convenience of our statements, we assume that the matrix $A \in \mathbb{R}^{n\times n}$ is partitioned into $N \times N$ blocks $A_{\ell j} \in \mathbb{R}^{n_\ell \times n_j}$, with $\sum_{j=1}^{N} n_j = n$, i.e.,

$$A \in \mathbb{L}_n = \{A \in \mathbb{R}^{n\times n} \mid A = (A_{\ell j}), A_{\ell j} \in \mathbb{R}^{n_\ell \times n_j}, 1 \le \ell, j \le N\}$$

and A_{jj} being nonsingular for $j = 1, 2, \ldots, N$, and the vectors x and $G(x)$ are partitioned into subvectors $x_j \in \mathbb{R}^{n_j}$ and $G_j(x) \in \mathbb{R}^{n_j}$, $j = 1, 2, \ldots, N$, in a way conformally with the partition of A, i.e.,

$$x \in V_n = \{x \in \mathbb{R}^n \mid x = (x_1^T, \ldots, x_N^T)^T, x_j \in \mathbb{R}^{n_j}, 1 \le j \le N\}.$$

We will study convergence properties of these generalized asynchronous multisplitting block two-stage relaxation methods for the nonsingular matrices including block H-matrices of different types (see e.g., [3], [4], [5], [6] and [12]), and nonlinear mappings including block P-bounded mapping (see e.g., [11]). Therefore, the two-stage relaxation methods for solving the large sparse systems of mildly nonlinear equations (1) and their corresponding global convergence are dealt with in a unified manner.

The rest of this paper is organized as follows: We describe the new generalized asynchronous multisplitting block two-stage relaxation methods in section 23; Some preliminary results used in this paper is presented in section 23; In section 23, we analyze the global convergence of these generalized asynchronous multisplitting block two-stage relaxation methods.

Establishments of the new methods

Assume that the multiprocessor system consists of α processors and consider the splittings $A = B_i - C_i$, $i = 1, 2, \ldots, \alpha$, and a set of block diagonal nonnegative matrices E_i, $i = 1, 2, \ldots, \alpha$, such that $\sum_{i=1}^{\alpha} E_i = I$ (the identity matrix). Let $\tau_\ell^{(i)}(p)$ be a nonnegative integer that represents the index of the ℓ-th block element of a currently available global iterate which the i-th processor uses to compute its p-th local approximation, and $J_i(p)$ be a nonempty subset of the integer set $\mathbb{N} = \{1, 2, \ldots, N\}$ that satisfies $\ell \in J_i(p)$ if and only if the i-th processor starts its computation of the ℓ-th block element of a new iterate at the p-th step. As is customary in the descriptions and analyses of asynchronous methods, we assume that the superscripts $\tau_\ell^{(i)}(p)$ and the subsets $J_i(p)$, $p \in N_0 = \{0, 1, 2, \ldots\}$, satisfy the following conditions:

(a) $\tau_\ell^{(i)}(p) \leq p$ for all $i \in \{1, 2, \ldots, \alpha\}$, $\ell \in \mathbb{N}$ and $p \in N_0$;
(b) $\lim_{p \to \infty} \tau_\ell^{(i)}(p) = \infty$ for all $i \in \{1, 2, \ldots, \alpha\}$ and $\ell \in \mathbb{N}$; and
(c) *The set* $\{p \in N_0 \mid \ell \in J_i(p)\}$ is infinite for all $i \in \{1, 2, \ldots, \alpha\}$ and $\ell \in \mathbb{N}$.

Moreover, to describe the new asynchronous multisplitting block two-stage relaxation method we let γ_ℓ and $\omega_\ell (\neq 0)$, $\ell \in \mathbb{N}$, be two groups of relaxation parameters and for $i = 1, 2, \ldots, \alpha$, $D_i = \mathrm{Diag}(B_i)$ be the block diagonal matrices of $B_i = (B_{\ell j}^{(i)}) \in \mathbb{L}_n$, $L_i = (L_{\ell j}^{(i)}) \in \mathbb{L}_n$ be strictly block lower triangular matrices and $U_i = (U_{\ell j}^{(i)}) \in \mathbb{L}_n$ block zero-diagonal matrices, satisfying $B_i = D_i - L_i - U_i$. Denote by $E_i = Diag(E_{11}^{(i)}, E_{22}^{(i)}, \ldots, E_{NN}^{(i)})$. Then the new Generalized Asynchronous Multisplitting Block Two-stage (GAMBT) Accelerated OverRelaxation (AOR) method, or in short, the GAMBT-AOR method, can be described as follows.

THE GAMBT-AOR METHOD: Given an initial vector $x^0 \in \mathbb{R}^n$. Supposing we have approximations $x^0, x^1, \ldots, x^p$ to the solution $x^* \in \mathbb{R}^n$ of the block system of mildly nonlinear system (1). Then the next approximation $x^{p+1} = ((x_1^{p+1})^T, (x_2^{p+1})^T, \ldots, (x_N^{p+1})^T)^T$ is obtained element by element from

$$x_\ell^{p+1} = \sum_{i \in \mathbb{N}_\ell(p)} E_{\ell\ell}^{(i)} x_\ell^{p+1,i} + \sum_{i \notin \mathbb{N}_\ell(p)} E_{\ell\ell}^{(i)} x_\ell^p, \qquad \ell = 1, 2, \ldots, N, \tag{2}$$

where for each $\ell \in J_i(p)$, $x_\ell^{p+1,i} = x_\ell^{p,i,s_i(p)}$ is computed from the recursive formula

$$\begin{aligned} x_\ell^{p,i,k+1} \;=\; & B_{\ell\ell}^{(i)^{-1}} \Bigg\{ \gamma_\ell \sum_{j<\ell; j \in J_i(p)} L_{\ell j}^{(i)} x_j^{p,i,k} + (\omega_\ell - \gamma_\ell) \sum_{j<\ell; j \in \mathbb{N}} L_{\ell j}^{(i)} x_j^{\tau_j^{(i)}(p)} \\ & + \gamma_\ell \sum_{j<\ell; j \in \mathbb{N} \setminus J_i(p)} L_{\ell j}^{(i)} x_j^{\tau_j^{(i)}(p)} + \omega_\ell \sum_{j \neq \ell} U_{\ell j}^{(i)} x_j^{\tau_j^{(i)}(p)} \\ & + \omega_\ell \left(\sum_{j=1}^{N} C_{\ell j}^{(i)} x_j^{\tau_j^{(i)}(p)} + G_\ell(\ldots, x_j^{\tau_j^{(i)}(p)}, \ldots) \right) \Bigg\} \\ & + (1 - \omega_\ell) x_\ell^{\tau_\ell^{(i)}(p)}, \quad k = 0, 1, \ldots, s_i(p) - 1, \end{aligned} \tag{3}$$

with the starting point $x_\ell^{p,i,0} = x_\ell^{\tau_\ell^{(i)}(p)}$; and $\mathbb{N}_\ell(p) = \{i \mid \ell \in J_i(p), i = 1,2,\ldots,\alpha\}$ for all $\ell \in \mathbb{N}$ and $p \in N_0$. Here, γ_ℓ and ω_ℓ, $\ell \in \mathbb{N}$, are relaxation and acceleration parameters, respectively.

Some important special cases of the GAMBT-AOR method are:

(a) the asynchronous multisplitting block two-stage Gauss-Seidel method, which corresponds to the case of taking the relaxation parameter pair $(\gamma_\ell, \omega_\ell)$ to be $(1,1)$;
(b) the asynchronous multisplitting block two-stage SOR method, which corresponds to the case of taking the relaxation parameter pair $(\gamma_\ell, \omega_\ell)$ to be (ω, ω);
(c) the asynchronous multisplitting block two-stage AOR method, which corresponds to the case of taking the relaxation parameter pair $(\gamma_\ell, \omega_\ell)$ to be (γ, ω); and
(d) the generalized asynchronous multisplitting block two-stage SOR method, which corresponds to the case of taking the relaxation parameter pair $(\gamma_\ell, \omega_\ell)$ to be $(\omega_\ell, \omega_\ell)$.

Define $x_\ell^{p,i,k} = x_\ell^{\tau_\ell^{(i)}(p)}$ for all $\ell \in \mathbb{N}\backslash J_i(p)$, $i \in \{1,2,\ldots,\alpha\}$ and $k \in \{0,1,\ldots,s_i(p)-1\}$. If we introduce the projection operators $\mathcal{P}_\ell : V_n \to \mathbb{R}^{n_\ell}(\ell = 1,2,\ldots,N)$ by $\mathcal{P}_\ell(x) = x_\ell$ for any $x \in V_n$, and the matrices

$$\left\{\begin{array}{lcl} M_i(\gamma_\ell,\omega_\ell) & = & \frac{1}{\omega_\ell}(D_i - \gamma_\ell L_i), \\ N_i(\gamma_\ell,\omega_\ell) & = & \frac{1}{\omega_\ell}\left((1-\omega_\ell)D_i + (\omega_\ell - \gamma_\ell)L_i + \omega_\ell U_i\right), \end{array}\right. \qquad i = 1,2,\ldots,\alpha, \quad \ell \in \mathbb{N},$$

then after direct calculations, the GAMBT-AOR method can be briefly expressed in the matrix-vector form:

$$x_\ell^{p+1} = \sum_{i \in \mathbb{N}_\ell(p)} E_{\ell\ell}^{(i)} \mathcal{P}_\ell(y^{p+1,i,\ell}) + \sum_{i \notin \mathbb{N}_\ell(p)} E_{\ell\ell}^{(i)} x_\ell^p, \quad \ell = 1,2,\ldots,N, \tag{4}$$

where

$$\begin{aligned} y^{p+1,i,\ell} \;=\; & \left(M_i(\gamma_\ell,\omega_\ell)^{-1} N_i(\gamma_\ell,\omega_\ell)\right)^{s_i(p)} x^{\tau^{(i)}(p)} \\ & + \sum_{k=0}^{s_i(p)-1} \left(M_i(\gamma_\ell,\omega_\ell)^{-1} N_i(\gamma_\ell,\omega_\ell)\right)^k M_i(\gamma_\ell,\omega_\ell)^{-1}(C_i x^{\tau^{(i)}(p)} + G(x^{\tau^{(i)}(p)})) \end{aligned} \tag{5}$$

and

$$x^{\tau^{(i)}(p)} = (x_1^{\tau_1^{(i)}(p)}, x_2^{\tau_2^{(i)}(p)}, \ldots, x_N^{\tau_N^{(i)}(p)})^T. \tag{6}$$

Preliminaries

The orderings and the point absolute values in $\mathbb{R}^n$ and $\mathbb{R}^{n\times n}$ are defined according to elements, respectively. A nonsingular matrix $A \in \mathbb{R}^{n\times n}$ is called an M-matrix if it has non-positive off-diagonal entries and it is monotone (i.e., $A^{-1} \geq 0$). Let $D_A = \mathrm{diag}(A)$ be the point diagonal matrix of A and $B_A = D_A - A$. Then A is an M-matrix if and only if D_A is positive diagonal, B_A is nonnegative and $\rho(D_A^{-1}B_A) < 1$.

Define

$$\mathbb{L}_{n,I} = \{A = (A_{\ell j}) \in \mathbb{L}_n \mid A_{\ell\ell} \in \mathbb{R}^{n_\ell \times n_\ell} \text{ nonsingular}, \ell = 1, \ldots, N\},$$

which is a subset of $\mathbb{L}_n$. For a matrix $A \in \mathbb{L}_n$, let $D(A) = \mathrm{Diag}(A_{11}, \ldots, A_{NN})$, i.e., its block-diagonal part. Thus, $A \in \mathbb{L}_{n,I}$ if and only if $A \in \mathbb{L}_n$ and $D(A)$ is nonsingular.

For any matrix $A \in \mathbb{L}_{n,I}$ we define its type-I and type-II comparison matrices $\langle A \rangle = (\langle A \rangle_{\ell j}) \in \mathbb{R}^{N\times N}$ and $\langle\langle A \rangle\rangle = (\langle\langle A \rangle\rangle_{\ell j}) \in \mathbb{R}^{N\times N}$ as $\langle A \rangle_{\ell\ell} = \|A_{\ell\ell}^{-1}\|^{-1}$, $\langle A \rangle_{\ell j} = -\|A_{\ell j}\|$, $\ell \neq j$, and $\langle\langle A \rangle\rangle_{\ell\ell} = 1$, $\langle\langle A \rangle\rangle_{\ell j} = -\|A_{\ell\ell}^{-1} A_{\ell j}\|$, $\ell \neq j$, $\ell, j = 1, 2, \ldots, N$, respectively. For $A \in \mathbb{L}_n$, we define its block absolute value by $[A] = (\|A_{\ell j}\|) \in \mathbb{R}^{N\times N}$. The definition for a vector $v \in V_n$ is analogous. Here $\|\cdot\|$ is any consistent matrix norm satisfying $\|I\| = 1$. Note that these concepts directly reduce to the corresponding point ones in the literature when they are understood in the pointwise sense. We refer to [3] for the detailed properties about these block absolute values.

$A \in \mathbb{L}_{n,I}$ is said to be a Type-I (Type-II) block H-matrix if $\langle A \rangle$ ($\langle\langle A \rangle\rangle$) is an M-matrix in $\mathbb{R}^{N\times N}$ ($A \in H_B^I$ ($A \in H_B^{II}$)). It follows that $H_B^I \subset H_B^{II}$ with the inclusion being strict. For $A \in \mathbb{L}_{n,I}$, a splitting $A = M - N$ is called an H_B^I-compatible (H_B^{II}-compatible) splitting if

$$\langle A \rangle = \langle M \rangle - [N] \quad (\langle\langle A \rangle\rangle = \langle\langle M \rangle\rangle - [D(M)^{-1}N]).$$

A mapping $G : \mathbb{R}^n \to \mathbb{R}^n$ is called block P-bounded if there exists a nonnegative matrix $P \in \mathbb{R}^{N\times N}$ such that

$$[G(x) - G(y)] \leq P[x - y]$$

holds for all $x, y \in \mathbb{R}^n$. We refer to [3] for the concrete properties about the H_B^I and H_B^{II} matrix classes.

The following two lemmas are very useful for proving the global convergence of the GAMBT-AOR method.

Lemma 1 [11] *Let $A \in \mathbb{R}^{n\times n}$ be nonsingular. Then the block system of weakly nonlinear equations (1) has a unique solution provided either of the following two conditions holds:*
(a) $A \in H_B^I$, G is block P-bounded, and $\rho(\langle A \rangle^{-1} P) < 1$.
(b) $A \in H_B^{II}$, G is block P-bounded, and $\rho(\langle\langle A \rangle\rangle^{-1}[D(A)^{-1}]P) < 1$.

Lemma 2 [9] *Let $\{H_p^{(i)}\}_{p\in N_0} (i = 1, 2, \ldots, \alpha)$ be sequences of nonnegative matrices in $\mathbb{R}^{N\times N}$, $\widetilde{E}_i = diag(\widetilde{e}_1^{(i)}, \widetilde{e}_2^{(i)}, \ldots, \widetilde{e}_N^{(i)}) (i = 1, 2, \ldots, \alpha)$ be nonnegative diagonal matrices in $\mathbb{R}^{N\times N}$ satisfying $\sum_{i=1}^{\alpha} \widetilde{E}_i \leq I$, and $\{\epsilon^p\}_{p\in N_0}$ be sequence in $\mathbb{R}^N$ defined by*

$$\epsilon_\ell^{p+1} = \sum_{i\in\mathbb{N}_\ell(p)} \widetilde{e}_\ell^{(i)} \epsilon_\ell^{\tau_\ell^{(i)}(p)} + \sum_{i\notin\mathbb{N}_\ell(p)} \widetilde{e}_\ell^{(i)} \epsilon_\ell^p, \qquad \ell = 1, 2, \ldots, N, \quad p = 0, 1, 2, \ldots$$

with $\{\mathbb{N}_\ell(p)\}_{p\in N_0} (\ell = 1, 2, \ldots, N)$ and $\{\tau_\ell^{(i)}(p)\}_{p\in N_0} (\ell = 1, 2, \ldots, N, i = 1, 2, \ldots, \alpha)$ being described as in section 23. Then $\lim_{p\to\infty} \epsilon^p = 0$ holds for any $\epsilon^0 \in \mathbb{R}^N$, provided there exist a constant $\theta \in [0, 1)$ and a positive vector $v \in \mathbb{R}^N$ such that $H_p^{(i)} v \leq \theta v$ $(i = 1, 2, \ldots, \alpha, p \in N_0)$.

The global convergence theory

In this section, we will prove the global convergence of the GAMBT-AOR method when the matrix $A \in \mathbb{L}_n$ is a block H-matrix of different types and when the mapping $G : \mathbb{R}^n \to \mathbb{R}^n$ is a block P-bounded mapping.

Theorem 1 *Let $A \in H_B^I$ $(H_B^{II}) \subset \mathbb{L}_{n,I}$. Let the splittings $A = B_i - C_i$, $i = 1, 2, \ldots, \alpha$, be H_B^I-compatible (H_B^{II}-compatible) such that $D(B_i) = D(A)$, $i = 1, 2, \ldots, \alpha$, the splittings $B_i = D_i - L_i - U_i$, $i = 1, 2, \ldots, \alpha$, satisfy $\langle B_i \rangle = \langle D_i \rangle - [L_i] - [U_i] (i = 1, 2, \ldots, \alpha)$ for Type-I case (and $\langle\langle B_i \rangle\rangle = I - [D_i^{-1} L_i] - [D_i^{-1} U_i] (i = 1, 2, \ldots, \alpha)$ for Type-II case), and the weighting matrices E_i, $i = 1, 2, \ldots, \alpha$, satisfy $\sum_{i=1}^{\alpha} [E_i] \leq I$. Assume further that $G : \mathbb{R}^n \to \mathbb{R}^n$ is a block P-bounded mapping such that*

$$\rho(\langle A \rangle^{-1} P) < 1 \quad (\rho(\langle\langle A \rangle\rangle^{-1} [D(A)^{-1}] P) < 1).$$

Then, the GAMBT-AOR method converges to the unique solution of the system of mildly nonlinear equations (1), for any initial vector $x^0 \in \mathbb{R}^n$ and any sequence of numbers of inner iterations $s_i(p) \geq 1$, $i = 1, 2, \ldots, \alpha$, $p \in N_0$, provided the relaxation parameters γ_ℓ and ω_ℓ, $\ell \in \mathbb{N}$, satisfy $0 \leq \gamma_\ell \leq \omega_\ell$ $(\ell \in \mathbb{N})$ and

$$0 < \omega_\ell < \frac{2}{1 + \rho(D_{\langle A \rangle}^{-1}(B_{\langle A \rangle} + P))} \quad \left(0 < \omega_\ell < \frac{2}{1 + \rho(B_{\langle\langle A \rangle\rangle} + [D(A)^{-1}] P)}\right), \quad \ell \in \mathbb{N}.$$

Proof: By Lemma 1 we know that there exists a unique vector $x^* \in \mathbb{R}^n$ such that $Ax^* = G(x^*)$. Thus, if we let $\varepsilon^p = x^p - x^*$ be the error at the p-th iteration of the GAMBT-AOR method, then according to (4)-(6), $\{\varepsilon^p\}_{p \in N_0}$ satisfies

$$\varepsilon_\ell^{p+1} = \sum_{i \in \mathbb{N}_\ell(p)} E_{\ell\ell}^{(i)} \mathcal{P}_\ell (y^{p+1,i,\ell} - x^*) + \sum_{i \notin \mathbb{N}_\ell(p)} E_{\ell\ell}^{(i)} \varepsilon_\ell^p \tag{7}$$

for all $\ell \in \mathbb{N}$, where

$$\begin{aligned} y^{p+1,i,\ell} - x^* &= \left(M_i(\gamma_\ell, \omega_\ell)^{-1} N_i(\gamma_\ell, \omega_\ell)\right)^{s_i(p)} \varepsilon^{\tau^{(i)}(p)} \\ &\quad + \sum_{k=0}^{s_i(p)-1} \left(M_i(\gamma_\ell, \omega_\ell)^{-1} N_i(\gamma_\ell, \omega_\ell)\right)^k M_i(\gamma_\ell, \omega_\ell)^{-1} \\ &\quad \times \left(C_i \varepsilon^{\tau^{(i)}(p)} + G(x^{\tau^{(i)}(p)}) - G(x^*)\right). \end{aligned} \tag{8}$$

Let us denote by $\widetilde{P} = [D(A)^{-1}] P$, and for $i = 1, 2, \ldots, \alpha$,

$$\widetilde{B}_i = D(A)^{-1} B_i, \quad \widetilde{C}_i = D(A)^{-1} C_i, \quad \widetilde{L}_i = D(A)^{-1} L_i, \quad \widetilde{U}_i = D(A)^{-1} U_i$$

and

$$\left\{ \begin{array}{lcl} \widetilde{M}_i(\gamma_\ell, \omega_\ell) &=& \frac{1}{\omega_\ell}(I - \gamma_\ell \widetilde{L}_i), \\ \widetilde{N}_i(\gamma_\ell, \omega_\ell) &=& \frac{1}{\omega_\ell}((1 - \omega_\ell) I + (\omega_\ell - \gamma_\ell) \widetilde{L}_i + \omega_\ell \widetilde{U}_i), \end{array} \right. \quad \ell \in \mathbb{N}.$$

Then we have

$$\left\{ \begin{array}{lcl} \langle \widetilde{M}_i(\gamma_\ell, \omega_\ell) \rangle & = & \langle\langle \widetilde{M}_i(\gamma_\ell, \omega_\ell) \rangle\rangle = \frac{1}{\omega_\ell}(I - \gamma_\ell [\widetilde{L}_i]) \equiv \mathcal{M}_i(\gamma_\ell, \omega_\ell), \\ [\widetilde{N}_i(\gamma_\ell, \omega_\ell)] & \leq & \frac{1}{\omega_\ell}(|1 - \omega_\ell| I + (\omega_\ell - \gamma_\ell)[\widetilde{L}_i] + \omega_\ell [\widetilde{U}_i]) \equiv \mathcal{N}_i(\gamma_\ell, \omega_\ell) \end{array} \right. \tag{9}$$

for $i = 1, 2, \ldots, \alpha$ and $\ell \in \mathbb{N}$. By using these last inequalities, taking block absolute values on both sides of (7), applying the block P-bounded property of G, writing $\mathcal{C}_i = [\widetilde{C}_i]$ and $\mathcal{P} = \widetilde{P}$, and noticing that $\mathcal{M}_i(\gamma_\ell, \omega_\ell)$ $(i = 1, 2, \ldots, \alpha, \ell \in \mathbb{N})$ are M-matrices in $\mathbb{R}^{N \times N}$, we know from (8)-(9) that the error $\{\varepsilon^p\}_{p \in N_0}$ of GAMBT-AOR method satisfies

$$[\varepsilon_\ell^{p+1}] \leq \sum_{i \in \mathbb{N}_\ell(p)} [E_{\ell\ell}^{(i)}] e_\ell^T T_p^{(i,\ell)} [\varepsilon^{\tau^{(i)}(p)}] + \sum_{i \notin \mathbb{N}_\ell(p)} [E_{\ell\ell}^{(i)}][\varepsilon_\ell^p], \quad \ell \in \mathbb{N}, \quad p \in N_0 \tag{10}$$

for both Type-I and Type-II cases, where $e_\ell \in \mathbb{R}^N$ is the ℓ-th unit basis vector, and

$$\begin{aligned} T_p^{(i,\ell)} \; = \; & \left(\mathcal{M}_i(\gamma_\ell, \omega_\ell)^{-1} \mathcal{N}_i(\gamma_\ell, \omega_\ell)\right)^{s_i(p)} \\ & + \sum_{k=0}^{s_i(p)-1} \left(\mathcal{M}_i(\gamma_\ell, \omega_\ell)^{-1} \mathcal{N}_i(\gamma_\ell, \omega_\ell)\right)^k \mathcal{M}_i(\gamma_\ell, \omega_\ell)^{-1} (\mathcal{C}_i + \mathcal{P}). \end{aligned} \tag{11}$$

For Type-I case, similarly to [11] we can demonstrate that there exist a constant $\theta \in [0, 1)$ and a positive vector $v \in \mathbb{R}^N$ such that

$$T_p^{(i,\ell)} v \leq \theta v, \qquad i = 1, 2, \ldots, \alpha, \quad \ell \in \mathbb{N}, \quad p \in N_0.$$

Now, defining the sequence $\{\epsilon^p\}_{p \in N_0}$ according to $\epsilon^0 = [\varepsilon^0]$ and

$$\epsilon_\ell^{p+1} = \sum_{i \in \mathbb{N}_\ell(p)} \widetilde{e}_\ell^{(i)} e_\ell^T T_p^{(i,\ell)} \epsilon^{\tau^{(i)}(p)} + \sum_{i \notin \mathbb{N}_\ell(p)} \widetilde{e}_\ell^{(i)} \epsilon_\ell^p, \qquad \ell = 1, 2, \ldots, N, \quad p \in N_0,$$

where

$$\widetilde{E}_i = diag(\widetilde{e}_1^{(i)}, \widetilde{e}_2^{(i)}, \ldots, \widetilde{e}_N^{(i)}) = [E_i], \quad i = 1, 2, \ldots, \alpha,$$

we can immediately deduce that $\{\epsilon^p\}_{p \in N_0}$ is a majorizing sequence of $\{[\varepsilon^p]\}_{p \in N_0}$. That is to say, $[\varepsilon^p] \leq \epsilon^p$ holds for all $p \in N_0$. By making use of Lemma 2 we know that $\lim_{p \to \infty} \epsilon^p = 0$. Therefore, $\lim_{p \to \infty} [\varepsilon^p] = 0$ and then, $\lim_{p \to \infty} \varepsilon^p = 0$.

Quite analogous to the proof of Type-I case we can also get $\lim_{p \to \infty} \varepsilon^p = 0$ for Type-II case.

Theorem 1 immediately leads to the following convergence theories for the asynchronous multisplitting block two-stage Gauss-Seidel, SOR and AOR methods, as well as the generalized asynchronous multisplitting block two-stage SOR method.

Theorem 2 *Let $A \in H_B^I$ $(H_B^{II}) \subset \mathbb{L}_{n,I}$. Let the splittings $A = B_i - C_i$, $i = 1, 2, \ldots, \alpha$, be H_B^I-compatible (H_B^{II}-compatible) such that $D(B_i) = D(A)$, $i = 1, 2, \ldots, \alpha$, the splittings $B_i = D_i - L_i - U_i$, $i = 1, 2, \ldots, \alpha$, satisfy $\langle B_i \rangle = \langle D_i \rangle - [L_i] - [U_i] (i = 1, 2, \ldots, \alpha)$ for Type-I case (and $\langle\langle B_i \rangle\rangle = I - [D_i^{-1} L_i] - [D_i^{-1} U_i] (i = 1, 2, \ldots, \alpha)$ for Type-II case), and the weighting matrices E_i, $i = 1, 2, \ldots, \alpha$, satisfy $\sum_{i=1}^{\alpha} [E_i] \leq$*

I. Assume further that $G : \mathbb{R}^n \to \mathbb{R}^n$ is a block P-bounded mapping such that $\rho(\langle A\rangle^{-1}P) < 1$ ($\rho(\langle\langle A\rangle\rangle^{-1}[D(A)^{-1}]P) < 1$). Then for any initial vector $x^0 \in \mathbb{R}^n$ and any sequence of numbers of inner iterations $s_i(p) \geq 1$, $i = 1, 2, \ldots, \alpha$, $p \in N_0$:

(a) the asynchronous multisplitting block two-stage Gauss-Seidel method converges to the unique solution of the system of mildly nonlinear equations (1)*;*

(b) the asynchronous multisplitting block two-stage SOR method converges to the unique solution of the system of mildly nonlinear equations (1)*, provided the relaxation parameter ω satisfies*

$$0 < \omega < \frac{2}{1 + \rho(D_{\langle A\rangle}^{-1}(B_{\langle A\rangle} + P))} \qquad \left(0 < \omega < \frac{2}{1 + \rho(B_{\langle\langle A\rangle\rangle} + [D(A)^{-1}]P)}\right);$$

(c) the asynchronous multisplitting block two-stage AOR method converges to the unique solution of the system of mildly nonlinear equations (1)*, provided the relaxation parameters γ and ω satisfy $0 \leq \gamma \leq \omega$, and*

$$0 < \omega < \frac{2}{1 + \rho(D_{\langle A\rangle}^{-1}(B_{\langle A\rangle} + P))} \qquad \left(0 < \omega < \frac{2}{1 + \rho(B_{\langle\langle A\rangle\rangle} + [D(A)^{-1}]P)}\right);$$

(d) the generalized asynchronous multisplitting block two-stage SOR method converges to the unique solution of the system of mildly nonlinear equations (1)*, provided the relaxation parameters ω_ℓ ($\ell \in \mathbb{N}$) satisfy*

$$0 < \omega_\ell < \frac{2}{1 + \rho(D_{\langle A\rangle}^{-1}(B_{\langle A\rangle} + P))} \qquad \left(0 < \omega_\ell < \frac{2}{1 + \rho(B_{\langle\langle A\rangle\rangle} + [D(A)^{-1}]P)}\right), \qquad \ell \in \mathbb{N}.$$

REFERENCES

Z.-Z. Bai. A class of two-stage iterative methods for systems of weakly nonlinear equations. *Numerical Algorithms*, 14:295–319, 1997.

Z.-Z. Bai. Parallel multisplitting two-stage iterative methods for large sparse systems of weakly nonlinear equations. *Numerical Algorithms*, 15:347–372, 1997.

Z.-Z. Bai. Parallel matrix multisplitting block relaxation methods. *Mathematica Numerica Sinica*, 17:238–252, 1995. In Chinese.

Z.-Z. Bai. A class of blockwise matrix multi-splitting multi-parameter block relaxation methods. *Proceeding of the Third China-Japan Joint Seminar on Numerical Mathematics*, Z. C. Shi and M. Mori eds., Science Press of China, Beijing/New York, 1998.

Z.-Z. Bai. Asynchronous parallel matrix multisplitting block relaxation iteration methods. *Numerical Mathematics, A Journal of Chinese Universities (Chinese Series)*, 19:28–39, 1997. In Chinese.

Z.-Z. Bai. A class of asynchronous parallel multisplitting blockwise relaxation methods. *Parallel Computing*, accepted for publication, 1999.

Z.-Z. Bai and Y.-G. Huang. Asynchronous multisplitting two-stage iterations for systems of weakly nonlinear equations. *Journal of Computational and Applied Mathematics* , 93:13-33, 1998.

Z.-Z. Bai, J.-C. Sun and D.-R. Wang. A unified framework for the construction of various matrix multisplitting iterative methods for large sparse system of linear equations. *Computers and Mathematics with Applications*, 32(12):51-76, 1996.

Z.-Z. Bai, D.-R. Wang and D. J. Evans. A class of asynchronous matrix multi-splitting multi-parameter relaxation iterations. *Journal of Computational Mathematics*, 16(3):221-238, 1998.

Z.-Z. Bai, D.-R. Wang and D. J. Evans. On the convergence of asynchronous nested matrix multisplitting methods for linear systems. *Journal of Computational Mathematics*, accepted for publication, 1997.

Z.-Z. Bai, V. Migallón, J. Penadés and D. B. Szyld. Block and asynchronous two-stage methods for mildly nonlinear systems. *Numerische Mathematik*, accepted for publication, 1998.

D. J. Evans and Z.-Z. Bai. Blockwise matrix multi-splitting multi-parameter block relaxation methods. *International Journal of Computer Mathematics*, 64:103-118, 1997.

D. J. Evans and Z.-Z. Bai. On the convergence of two-stage multisplitting methods for linear systems. *Neural, Parallel and Scientific Computations*, 6:279-295, 1998.

24

Domain Decomposition Capabilities for the Mortar Finite Volume Element Methods

Richard E. Ewing [1], Raytcho D. Lazarov [2], Tao Lin [3], &

Yanping Lin [4]

Introduction

Since the introduction of the *mortar method* as a coupling technique between the spectral and finite element methods (see, e.g. [BM97, BMP94, BMSL89]), it has become the most important technique in domain decomposition methods for non-matching grids. The active research by the scientific computation community in this field is motivated by its flexibility and great potential for large scale parallel computation (see, e.g. [BM94]). A good description of the mortar element method can be found in [Bel97, BDM90, BMP94, Cas]. The nonconforming finite element mortar method has been studied in [BMP94], where optimal order convergence in H^1-norm was demonstrated. Three-dimensional mortar finite element analysis has been given in [BM97]. Non-mortar mixed finite element approximations for second order elliptic

1 Institute for Scientific Computation, Texas A&M University, College Station, TX 77843-3404, USA This author is partially supported by the EPA Grant #825207-01-1. email: ewing@isc.tamu.edu

2 Address and support same as above. email: lazarov@math.tamu.edu

3 Department of Mathematics, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061, USA This author is partially supported by NSF #DMS-9704621. email: tlin@math.vt.edu

4 Department of Mathematics, University of Alberta, Edmonton, Alberta T6G 2G1, Canada This author is partially supported by NSERC #OGP0121508.

Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

problems have been discussed in [AY97].

The above mentioned mortar elements are defined on non-matching grids with non-overlapping subdomains. Recently, the overlapping mortar linear finite element method was studied in [CDS99], where several additive Schwarz preconditioners have been proposed and analyzed and extensive numerical examples to support the theoretical results have been reported.

To the authors' best knowledge, there has not been a study for the mortar finite volume element method. In the past 10 years the finite (control) volume method has drawn serious attension both form mathematicians, engineers, and physicists as an attractive solution technique for various applied problems (see, e.g. [BV96]). Following the notations and the approach of Ben Belgacem [Bel97], we apply the mortar technique to derive two control volume schemes based on: (1) finite volume element approximation of the solution in the subdomains and finite element approximation for Lagrange multipliers on the interfaces; (2) finite volume approximations for both the solution in the subdomains and the Lagrange multipliers on the interfaces. It has been shown on variuous test problems that the latter schemes converge much faster (5-6 times) than the former schemes. In this respect we have found evidence from our numerical experiments and we believe that in the finite volume element methods if r-th order piecewise polynomials are used on the subdomains, then $(r-1)-$th order polynomials should be used on the interface for the Lagrange multipliers. For both types of schemes, we have obtained in [ELLL98] optimal order H^1-norm error estimates under the regularity assumption that $u \in H^{1+\tau_k}(\Omega_k)$ for $0 < \tau_k \leq 1$ where $\overline{\Omega} = \cup \overline{\Omega}_k$.

Mortar Finite Element Approximation

We shall use the notations from [Bel97]. We break up the initial domain Ω into K non-overlapping subdomains $\{\Omega_k\}_{1\leq k \leq K}$, which are assumed to be polygonally shaped and arranged in such a way that the intersection of two subdomains $\overline{\Omega}_l \cap \overline{\Omega}_k$ as well as the intersection $\partial\Omega \cap \partial\Omega_k$ is either empty or reduced to a vertex or to a common edge. If two subdomains Ω_k and Ω_l are adjacent, Γ_{kl} is the common interface, and $\mathbf{n}_{kl}$ is the unit normal from Ω_k to Ω_l. Let $\overline{k}$ denote the set of all indices so that kl is meaningful. For any k, let $H^1_*(\Omega_k)$ denote the space $H^1(\Omega_k)$ if the measure of $\partial\Omega_k \cap \partial\Omega$ is zero; otherwise it coincides with the subspace of $H^1(\Omega_k)$ involving all functions whose trace is zero over the set $\partial\Omega_k \cap \partial\Omega$:

$$H^1_*(\Omega_k) = \left\{ v_k \in H^1(\Omega_k) : \quad v_k|_{\partial\Omega_k \cap \partial\Omega} = 0, \text{ if } \operatorname{meas}(\partial\Omega_k \cap \partial\Omega) \neq 0 \right\}.$$

Set the space

$$X = \left\{ v \in L^2(\Omega) : \quad v_k = v|_{\Omega_k} \in H^1_*(\Omega_k) \right\} = \prod_{k=1}^{K} H^1_*(\Omega_k)$$

equipped with the norm: $||u||_X = \left(\sum_{k=1}^{K} ||v_k||^2_{H^1(\Omega_k)} \right)^{1/2}$. Let

$$H_0(\operatorname{div}, \Omega) = \{ q \in H(\operatorname{div}, \Omega) : \quad \mathbf{q} \cdot \mathbf{n}|_{\partial\Omega} = 0 \},$$

where $H(\mathrm{div},\Omega)$ is the space of all vector-functions in $(L^2(\Omega))^2$ whose weak divergence is in $L^2(\Omega)$. The trace of these function on the boundary $\partial\Omega$ is understood in the appropriate weak sense. The characterization of $H_0^1(\Omega)$ can be made:

$$H_0^1(\Omega) = \left\{ v \in X : \quad \sum_{k=1}^{K} (\mathbf{q}\cdot\mathbf{n}, v)_{\partial\Omega_k} = 0, \quad \mathbf{q} \in H_0(\mathrm{div},\Omega) \right\}.$$

Now we define the space M of those $\psi = (\psi_1, \cdots, \psi_K)$ with components $\psi_k \in H_*^{-1/2}(\partial\Omega_k)$ such that there weak traces on the boundaries represent a weak trace of a function in $H_0(\mathrm{div},\Omega)$, i.e.

$$M = \{\psi : \text{ there exists } \mathbf{q} \in H_0(\mathrm{div},\Omega) \text{ s.t. for } k = 1, \cdots, K,\ \psi_k = \mathbf{q}\cdot\mathbf{n}_k\}.$$

The space M is provided with the norm

$$||\psi||_M = \inf\left\{ ||\mathbf{q}||_{H(\mathrm{div},\Omega)} : \quad \mathbf{q} \in H_0(\mathrm{div},\Omega), \quad \mathbf{q}\cdot\mathbf{n}_k = \psi_k, \forall k \right\},$$

where $H_*^{-1/2}(\partial\Omega_k)$ is the dual space of $H_*^{1/2}(\partial\Omega_k)$ with $< \cdot,\cdot >_{*,\partial\Omega_k}$ pairing, $H_*^{1/2}(\partial\Omega_k) = H^{1/2}(\partial\Omega_k)$ if $\partial\Omega_k \cap \partial\Omega = \emptyset$ and $H_*^{1/2}(\partial\Omega_k) = H_{00}^{1/2}(\partial\Omega_k \setminus \partial\Omega)$ if $\partial\Omega_k \cap \partial\Omega \neq \emptyset$. Basically speaking the constraints on the distributions $\psi \in M$ imply that the jumps across the interfaces Γ_{kl} vanish.

We now define the bilinear form: $B : X \times M \to R$ by

$$B(v,\phi) = \sum_{k=1}^{K} < v_k, \phi_k >_{*,\partial\Omega_k},$$

so that it follows from Hahn-Banach Theorem that

$$H_0^1(\Omega) = \{v \in X, \quad B(v,\phi) = 0, \quad \phi \in M\}.$$

Similarly, the bilinear form $A : X \times X \to \mathcal{R}$ is defined by

$$A(u,v) = \sum_{k=1}^{K} \int_{\Omega_k} \nabla u_k \cdot \nabla v_k \, dx.$$

We consider the following model problem: find $u \in H_0^1(\Omega)$ such that

$$A(u,v) = (f,v), \quad v \in H_0^1(\Omega). \tag{1}$$

Its primal hybrid formulation is therefore defined by: find $(u,\psi) \in X \times M$ such that

$$\begin{array}{lcll} A(u,v) + B(v,\psi) & = & (f,v), & v \in X, \\ B(u,\phi) & = & 0, & \phi \in M. \end{array} \tag{2}$$

We have the following equivalent result: Problem (2) has a unique solution $(u,\psi) \in X \times M$, and the first component $u \in H_0^1(\Omega)$ is also the solution of problem (1). Moreover, we have

$$\psi_k = A\nabla u_k \cdot \mathbf{n}_k, \quad k = 1, \cdots, K \quad \text{and} \quad ||u||_{H^1(\Omega)} + ||\psi||_M \leq C||f||_{L^2(\Omega)}.$$

Finite Volume Element Approximation

Let the triangulation $\mathcal{T}_{h_k}$ of each subdomain Ω_k, $1 \le k \le K$, be such that

$$\overline{\Omega}_k = \cup_{T \in \mathcal{T}_{h_k}} \overline{T}, \quad h_k = \max_{T \in \mathcal{T}_{h_k}} h_T, \quad \text{and} \quad h_T = \sup_{x,y,\in T} d(x,y).$$

For piecewise linear finite element subspaces of $H_*^1(\Omega_k)$ on $\mathcal{T}_{h_k}$, we set

$$X_{\delta,k} = \left\{ v_{\delta,k} \in C(\Omega_k) : \quad v_{\delta,k}|_T \in P_1(T), \quad T \in \mathcal{T}_{h_k}, \quad v_{\delta,k}|_{\partial\Omega \cap \partial\Omega_k} = 0 \right\},$$

and the global finite element spaces

$$X_\delta = \prod_{k=1}^{K} X_{\delta,k}, \quad \text{where} \quad \delta = (h_1, h_2, \cdots, h_K).$$

Notice that the trace of the triangulation $\mathcal{T}_{h_k}$ over Γ_{kl}, $1 \le k \le K$, $l \in \overline{k}$, with vertices $v_{1,kl}$ and $v_{2,kl}$ results in a regular triangulation denoted by t_{kl}, where $\overline{k}$ is the class of the indices $l \in \underline{k}$ with $l > k$ and $\underline{k}$ is denotes the set of all indices l so that kl exist. The trace space $W_{\delta,kl}$ of the functions in $X_{\delta,k}$ is given by (see Figure 2):

$$W_{\delta,kl} = \left\{ \phi_{\delta,kl} \in C(\Gamma_{kl}) : \quad t \in t_{kl}, \quad \phi_{\delta,kl} \in P_1(t) \right\},$$

the approximation of the local Lagrange multiplier is defined as

$$M_{\delta,kl} = \left\{ \phi_{\delta,kl} \in W_{\delta,kl} : \quad t \in t_{kl}, \quad \text{or} \quad \phi_{\delta,kl} \in P_0(t) \quad \text{if} \quad v_{1,kl} \quad or \quad v_{2,kl} \in t \right\},$$

and the global finite element space on the interface is

$$M_\delta = \prod_{k=1}^{K} \prod_{l \in \overline{k}} M_{\delta,kl}.$$

We now define a bilinear form on $X_\delta \times M_\delta$ by

$$B(v_\delta, \phi_\delta) = \sum_{k=1}^{K} < v_{\delta,k}, \phi_{\delta,k} >_{*,\partial\Omega_k} = \sum_{k=1}^{K} \sum_{l \in \overline{k}} \int_{\Gamma_{kl}} \phi_{\delta,kl}(v_{\delta,k} - v_{\delta,l}) ds.$$

Thus, the mortar finite element approximation of the solution of (2) is defined by (see, e.g. [Bel97, BDM90, BMP94]):

$$\begin{array}{rcll} A(u_\delta, v_\delta) + B(v_\delta, \psi_\delta) &=& (f, v_\delta), & v_\delta \in X_\delta, \\ B(u_\delta, \phi_\delta) &=& 0, & \phi_\delta \in M_\delta. \end{array} \tag{3}$$

If the space V_δ of nonconforming approximations of functions in $H_0^1(\Omega)$ is introduced by:

$$V_\delta = \left\{ v_\delta \in X_\delta : \quad B(v_\delta, \phi_\delta) = 0, \quad \phi_\delta \in M_\delta \right\},$$

then the problem (3) is equivalent to the problem of finding $u_\delta \in V_\delta$ such that

$$A(u_\delta, v_\delta) = (f, v_\delta) \quad v_\delta \in V_\delta. \tag{4}$$

Now we shall introduce the mortar finite volume element approximation of the model problem (3). For a given triangulation $\mathcal{T}_{h_k}$, we construct a dual mesh $\mathcal{T}^*_{h_k}$ based upon $\mathcal{T}_{h_k}$ whose elements are called control volumes.

There are various ways of introducing regular control volume grids $\mathcal{T}^*_\delta$. In the most popular control volume partitions, the medicenter of the finite element T is connected with the midpoints of the edges of T. These types of volumes can be introduced for any finite element partition $\mathcal{T}_{h_k}$ and leads to relatively simple calculations. If the vertex is on the interface Γ_{kl}, then "half" control volume (shaded regions in Figure 1) is used.

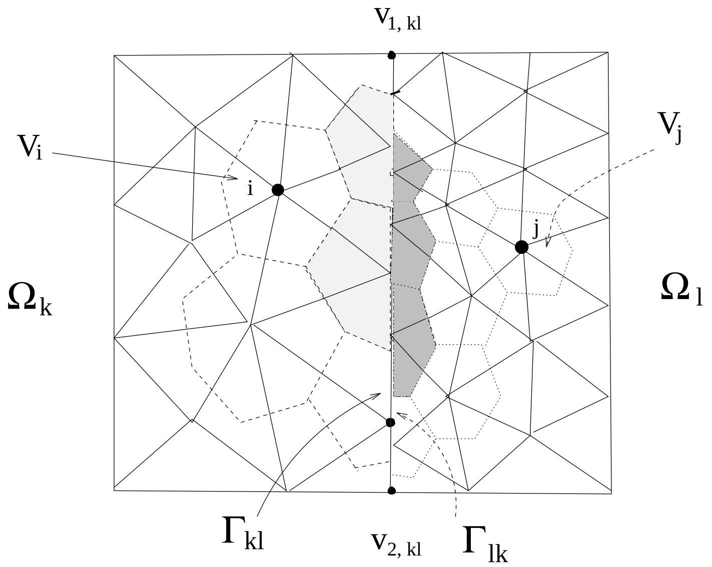


Figure 1 Interfaces Γ_{kl} and Γ_{lk} with $v_{1,kl}$ and $v_{2,kl}$ as two end points, triangulation $\mathcal{T}_{h_k}$ and $\mathcal{T}_{h_l}$, and the volumes in Ω_k and Ω_l. The triangulation t_{kl} and t_{lk} are different on the interface due to non-matching grids.

For the finite element space X_δ we can define its dual volume element space $X^*_\delta = \prod_{k=1}^K X^*_{\delta,k}$, where

$$X^*_{\delta,k} = \{v_k \in L^2(\Omega_k) : \ v_k|_V \ \text{ is constant over } V \in \mathcal{T}^*_{h_k} \ \text{ and } v_k|_{\partial\Omega\setminus\partial\Omega_k} = 0\}.$$

Obviously, $X^*_{\delta,k}$ =span$\{\chi_{i,k}(V) : \ V \in \mathcal{T}^*_{h_k}\}$, where $\chi_{i,k}$ is the characteristic function of the volume $V_{i,k}$. Let $I_{h_k} : C(\Omega_k) \to X_{\delta,k}$ be the interpolation operator and $I^*_{h_k} : C(\Omega_k) \to X^*_{\delta,k}$ be the piecewise constant interpolation operator, that is

$$I^*_{h_k} u = \sum_{x_{i,k} \in N_{h_k}} u_{i,k}\chi_{i,k}(x), \text{ where } u_{i,k} = u(x_{i,k}).$$

Then we set $I_\delta = \prod_{k=1}^K I_{h_k}$ and $I^*_\delta = \prod_{k=1}^K I^*_{h_k}$. With the above preparation, we can combine the finite volume approximation (see, e.g. [Cai91, ELL98, LC94, Mis98]) with the mortar approach to define our *mortar finite volume element method*: find $(u_\delta, \psi_\delta) \in X_\delta \times M_\delta$ such that

$$\begin{array}{lcll} A(u_\delta, I_\delta^* v_\delta) + B(v_\delta, \psi_\delta) & = & (f, I_\delta^* v_\delta), & v_\delta \in X_\delta, \\ B(u_\delta, \phi_\delta) & = & 0, & \phi_\delta \in M_\delta, \end{array} \tag{5}$$

where

$$A(u_\delta, I_\delta^* v_\delta) = -\sum_{k=1}^{K} \sum_{j \in N_{h_k}} v_{j,k} \int_{\partial V_{j,k}} A(x) \nabla u_{\delta,k} \cdot \mathbf{n}_k ds,$$

$$(f, I_\delta^* v_\delta) = \sum_{k=1}^{K} \sum_{j \in N_{h_k}} v_{j,k} \int_{V_{j,k}} f(x) dx.$$

This problem is equivalent to the following problem: find $u_\delta \in V_\delta$ such that

$$A(u_\delta, I_\delta^* v_\delta) = (f, I_\delta^* v_\delta), \quad v_\delta \in V_\delta. \tag{6}$$

Remark: We keep the same piecewise linear element spaces on the interfaces and formulate our mortar finite volume approximations only on the subdomains. This alone in fact is enough to preserve the basic feature of finite volume element method, that is, both (5) and (6) are locally conservative. The weak compatibility condition of the spaces X_δ and M_δ are satisfied automatically:

$$\{\phi_\delta : \quad B(v_\delta, \phi_\delta) = 0, \quad \forall v_\delta \in X_\delta\} = \{0\}, \tag{7}$$

which guarantees that there is no spurious modes generated for the normal derivatives of the solution using this discretization. In other words, our mortar finite volume element formulation has the nice properties of mortar finite element method.

In [ELLL98] we have introduced another formulation of the mortar finite volume element method with piecewise constant volume element approximation on the interfaces. The stability, convergence and error estimates for this type of scheme can be obtained in the framework presented above. Similarly, methods for geometrically nonconforming subdomains or overlapping domains as those shown on Figure 24 can be introduced as well (see [ELLL98]).

Error Estimates

The following error estimate has been proved in [ELLL98]:

Theorem 1 *Assume that $\mathcal{T}_\delta$ is regular, then the unique solution pair $(u_\delta, \psi_\delta) \in X_\delta \times M_\delta$ exists for the finite volume element mortar formulation and satisfies the error estimates:*

$$||u - u_\delta||_X \leq C \sum_{k=1}^{K} h_k ||u||_H^2(\Omega_k) + C \sum_{k=1}^{K} h_k ||f||_{L^2(\Omega)}.$$

A similar estimate is valid for the M-norm of the error in the Lagrange multipliers as well.

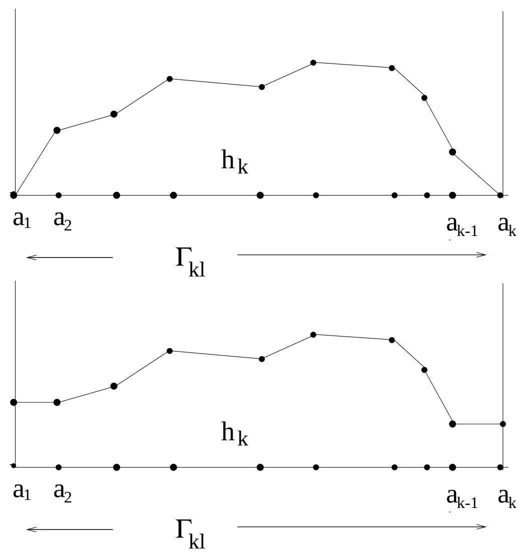


Figure 2 Left: a function from $W_{\delta,kl}$; right: a function from $M_{\delta,kl}$.

REFERENCES

[AY97] Arbogast T. and Yotov I. (1997) A non-mortar mixed finite element method for elliptic problems on non-matching multi-block grids. *Computer Methods in Applied Mechanics and Engineering* 149: 255–265.

[BDM90] Bernardi C., Debit N., and Maday Y. (1990) Coupling finite element and spectral methods: First results. *Math. Comput.* 54: 21–39.

[Bel97] Belgacem F. (1997) The mortar finite element method with Lagrange multiplier. preprint.

[BM94] Belgacem F. and Maday Y. (1994) Nonconforming spectral element methodology tuned to parallel implementation. *Computer Methods in Applied Mechanics and Engineering* 116: 59–67.

[BM97] Bernardi C. and Maday Y. (1997) The mortar element method for three dimensional finite elements. *M2NA* 31: 269–301.

[BMP94] Bernardi C., Maday Y., and Patera A. (1994) *A new nonconcoforming approach to domain decomposition: the mortar element method.* 299.

[BMSL89] Bernardi C., Maday Y., and Sacchi-Landriani G. (1989) Nonconforming matching conditions for coupling spectral finite element methods. *Applied Numer. Math.* 54: 64–84.

[BV96] Benkhaldoun F. and Vilsmeier R. (eds) (1996) *Proc. Int. Conference on Finite Volumes for Complex Applications.*

[Cai91] Cai Z. (1991) On the finite volume element method. *Numer. Math.* 58: 713–735.

[Cas] Casarin M.Schwarz preconditioners for spectral and mortar finite element methods with applications to incompressible fluids. Technical report, Department of Computer Sciences.

[CDS99] Cai X.-C., Dryja M., and Sarkis M. (1999) Overlapping nonmatching grid mortar element methods for elliptic problems. *SIAM J. Numer. Anal.* 35:

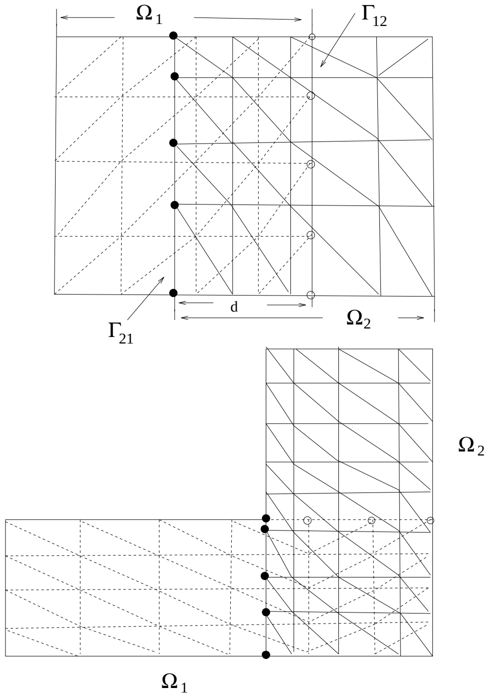


Figure 3 Overlapping subdomains.

581–606.

[ELL98] Ewing R., Lazarov R., and Lin Y. (1998) Finite volume element approximations of nonlocal reactive flows in porous media. Technical report, Texas A&M University, Institute for Scientific Computation.

[ELLL98] Ewing R., Lazarov R., Lin T., and Lin Y. (1998) The mortar finite volume element methods and domain decomposition. Technical report, Texas A&M University, Institute for Scientific Computation.

[LC94] Li R. and Chen Z. (1994) *The generalized difference method for differential equations*. Jilin University Publishing House.

[Mis98] Mishev I. (1998) Finite volume methods on Voronoi meshes. *Numer. Methods for Partial Differential Equations* 14: 193–212.

25

FETI-H: a scalable domain decomposition method for high frequency exterior Helmholtz problems

Charbel Farhat, Antonini Macedo and Radek Tezaur [1]

Introduction

The finite element discretization of the exterior Helmholtz problem leads to a system of equations that can be written as

$$\begin{aligned} \widetilde{\mathbf{K}}\mathbf{u} &= \mathbf{f}, \quad \text{where} \\ \widetilde{\mathbf{K}} &= \mathbf{K} - k^2\mathbf{M} + ik\mathbf{M}_S \end{aligned} \tag{1}$$

Matrices $\mathbf{K}$ and $\mathbf{M}$ are the so-called stiffness and mass matrix of the problem, and $\mathbf{f}$ its right-hand side vector. Matrix $\mathbf{M}_S$ is induced by the Sommerfeld radiation condition and is non zero only at the degrees of freedom lying on the outer boundary of the computational domain. In the absence of the Sommerfeld condition — that is, for the interior Helmholtz problem — $\widetilde{\mathbf{K}} = \mathbf{K} - k^2\mathbf{M}$ is usually an indefinite matrix. In this sense, $\widetilde{\mathbf{K}} = \mathbf{K} - k^2\mathbf{M} + ik\mathbf{M}_S$ is also often called an indefinite matrix. The large scale systems of equations resulting from realistic acoustic scattering applications have led to a great interest in the development of Krylov-subspace, multigrid and domain decomposition (DD) based iterative methods [1, 2, 3, 4, 19, 20, 24] for solving problem

1 Department of Aerospace Engineering Sciences, and Center for Aerospace Structures, University of Colorado at Boulder, Boulder, CO 80309-0429, U. S. A.; e-mail: charbel@colorado.edu, antonini@colorado.edu, radek.tezaur@colorado.edu

Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

(1).

Here, we present a Lagrange multiplier based two-level DD method for solving iteratively large-scale systems of equations arising from the finite element discretization of high-frequency exterior Helmholtz problems. The proposed method, which is introduced in Section 25 and presented in its simplest form in Section 25 of this paper, is essentially an extension of the regularized version [11] of the FETI (Finite Element Tearing and Interconnecting) method [13, 14, 15, 21] to indefinite problems. Its two key ingredients are the regularization of each subdomain matrix by a complex interface mass matrix, and the preconditioning of the interface problem by an auxiliary coarse problem constructed in Section 25 to enforce at each iteration the orthogonality of the residual to a set of carefully chosen planar waves. In Section 25, we show numerically that the proposed method is scalable with respect to the subdomain size, and the wavenumber.

Domain decomposition with Lagrange multipliers

The DD method presented in this paper is based on the two-level FETI method [13, 9]. Our focus on FETI is motivated by our experience with dual domain decomposition algorithms, and justified by the optimal convergence properties of the FETI method for second-order elasticity and fourth-order plate and shell problems [9, 13, 22, 21]. More specifically, our objective is the extension of the FETI method to exterior Helmholtz problems.

For the sake of clarity, we consider first the case where Ω is partitioned into two non-overlapping subdomains Ω^1 and Ω^2, and the formulation of the problem does not include the Sommerfeld condition (i.e., the interior Helmholtz problem). In Section 25, we generalize the proposed method to the case of arbitrary mesh decompositions and the exterior Helmholtz problem. (In practice, it is the mesh associated with Ω that is decomposed into subdomains, and therefore in this paper we consider only subdomains with matching interfaces).

Let $\mathbf{K}^s$, $\mathbf{M}^s$, $\hat{\mathbf{K}}^s = \mathbf{K}^s - k^2\mathbf{M}^s$, and $\mathbf{f}^s$ denote respectively the stiffness matrix, mass matrix, problem matrix, and right-hand side vector associated with subdomain Ω^s, $s = 1, 2$, and let $\mathbf{u}^s$ denote the restriction to Ω^s of the solution of problem (1). We partition each vector $\mathbf{u}^s$ into two components

$$\mathbf{u}^s = \begin{bmatrix} \mathbf{u}_i^s \\ \mathbf{u}_b^s \end{bmatrix} \tag{2}$$

where the subscripts i and b designate the internal and interface boundary unknowns of a given subdomain, respectively.

Given an interface matrix $\mathbf{S}_{bb}$ — that is, a matrix defined on the interface between subdomains Ω^1 and Ω^2 — we construct the following modified Lagrangian [11].

$$\begin{aligned} \mathcal{L}(\mathbf{v}^1, \mathbf{v}^2, \lambda) \quad &= \tfrac{1}{2}\mathbf{v}^{1^T}\hat{\mathbf{K}}^1\mathbf{v}^1 - \mathbf{f}^{1^T}\mathbf{v}^1 + \tfrac{1}{2}\mathbf{v}^{2^T}\hat{\mathbf{K}}^2\mathbf{v}^2 - \mathbf{f}^{2^T}\mathbf{v}^2 \\ &+\lambda^T(\mathbf{B}^1\mathbf{v}^1 + \mathbf{B}^2\mathbf{v}^2) + \tfrac{1}{2}(\mathbf{v}_b^{1^T}\mathbf{S}_{bb}\mathbf{v}_b^1 - \mathbf{v}_b^{2^T}\mathbf{S}_{bb}\mathbf{v}_b^2) \\ &= L(\mathbf{v}^1, \mathbf{v}^2, \lambda) + \tfrac{1}{2}(\mathbf{v}_b^{1^T}\mathbf{S}_{bb}\mathbf{v}_b^1 - \mathbf{v}_b^{2^T}\mathbf{S}_{bb}\mathbf{v}_b^2) \end{aligned}$$

where each of $\mathbf{B}^1$ and $\mathbf{B}^2$ is a signed Boolean matrix that extracts from a subdomain vector its interface boundary component, λ is a vector of discrete Lagrange multipliers defined on the interface between Ω^1 and Ω^2, and the superscript T designates the transpose of a quantity.

Solving problem (1) is equivalent to finding the stationary points $\mathbf{u}^1$ and $\mathbf{u}^2$ of the modified Lagrangian $\mathcal{L}$. Indeed, $L(\mathbf{v}^1, \mathbf{v}^2, \lambda)$ is the classical Lagrangian function of a two-subdomain problem, and the quantity $\frac{1}{2}(\mathbf{v}_b^{1^T}\mathbf{S}_{bb}\mathbf{v}_b^1 - \mathbf{v}_b^{2^T}\mathbf{S}_{bb}\mathbf{v}_b^2)$ depends only on the traces of $\mathbf{v}^1$ and $\mathbf{v}^2$ on the interface between Ω^1 and Ω^2. Since the vector of Lagrange multipliers λ enforces the continuity equation $\mathbf{B}^1\mathbf{v}^1 + \mathbf{B}^2\mathbf{v}^2 = 0$ on the interface between Ω^1 and Ω^2, it follows that the stationary points $\mathbf{u}_1$ and $\mathbf{u}_2$ of $\mathcal{L}$ are independent of the choice of the interface matrix $\mathbf{S}_{bb}$.

Let $\mathbf{S}_I^s$ denote the subdomain matrix defined as zero inside Ω^s and as $\mathbf{S}_{bb}$ on the interface boundary between Ω^1 and Ω^2. Using the same partitioning as in (2), $\mathbf{S}_I^s$ can be written as

$$\mathbf{S}_I^s = \begin{bmatrix} 0 & 0 \\ 0 & \mathbf{S}_{bb} \end{bmatrix} \qquad s = 1,\ 2$$

The Euler equations associated with the modified Lagrangian $\mathcal{L}$ are then given by

$$\begin{aligned} (\hat{\mathbf{K}}^1 + \mathbf{S}_I^1)\mathbf{u}^1 &= (\mathbf{K}^1 - k^2\mathbf{M}^1 + \mathbf{S}_I^1)\mathbf{u}^1 = \mathbf{f}^1 - \mathbf{B}^{1^T}\lambda \\ (\hat{\mathbf{K}}^2 - \mathbf{S}_I^2)\mathbf{u}^2 &= (\mathbf{K}^2 - k^2\mathbf{M}^2 - \mathbf{S}_I^2)\mathbf{u}^2 = \mathbf{f}^2 - \mathbf{B}^{2^T}\lambda \\ \mathbf{B}^1\mathbf{u}^1 + \mathbf{B}^2\mathbf{u}^2 &= 0 \end{aligned} \qquad (3)$$

The role of the interface matrix $\mathbf{S}_{bb}$ is now clear. For some given discretization of both subdomains Ω^1 and Ω^2, the prescribed wavenumber k may correspond to a resonant frequency of Ω^1 and/or Ω^2. In other words, k^2 may coalesce with an eigenvalue of either or both pencils $(\mathbf{K}^1,\ \mathbf{M}^1)$ and $(\mathbf{K}^2,\ \mathbf{M}^2)$. In such an event, either or both local problems described in (3) become ill-posed when $\mathbf{S}_{bb} = 0$. Hence, the purpose of a carefully constructed $\mathbf{S}_{bb}$ is to prevent the singularity of the subdomain matrix problems. We note that this issue has already been addressed in the literature, albeit with a different perspective (for example, see [1, 4]).

However, we would like to emphasize that as far as the design of a DD based iterative solver is concerned, it is not the potential singularity of a subdomain matrix problem that is problematic as much as the characterization of this singularity. In fact, for ill-posed Helmholtz subdomain problems, the solution of (3) can be written using the generalized inverse of the local problems. For elasticity problems ($k = 0$), the subdomain matrices can be singular, and as shown in [7, 13, 14, 21, 15], the ill-posed nature of the subdomain problems can be exploited to construct an auxiliary coarse problem. This "coarse grid" can then be used to propagate the error globally, accelerate convergence, and ensure scalability with respect to the subdomain size H.

However, for Helmholtz problems, the null space of the local matrices cannot be easily determined. Indeed, given a subdomain discretization, it is practically impossible to determine whether k^2 is a true eigenvalue of the pencil $(\mathbf{K}^s,\ \mathbf{M}^s)$, or whether it is numerically "close" to an eigenvalue of that pencil. It is not easy either to determine the multiplicity of that eigenvalue. And most importantly, whether k^2 coalesces or not with an eigenvalue of the pencil $(\mathbf{K}^s,\ \mathbf{M}^s)$ depends on the size of the mesh h, which complicates the issues further. For all these reasons, for Helmholtz problems, it is preferable to regularize the subdomain matrices $\mathbf{K}^s$ with an interface matrix $\mathbf{S}_{bb}$ as proposed once in the regularized FETI method [11], rather than attempt to compute a general form of the subdomain solutions using null spaces of subdomain matrices. It remains to address the issue of how to construct a regularizing interface matrix $\mathbf{S}_{bb}$.

The regularized FETI method for complex problems

Previous work on stabilized finite element methods for the discretization of the Helmholtz equation [16, 17] suggests that a good choice for $\mathbf{S}_{bb}$ is the complex interface mass matrix

$$[\mathbf{S}_{bb}]_{lm} \;=\; ik[\mathbf{M}_{bb}]_{lm} \;=\; ik\int_{\Omega^1\cap\Omega^2} \phi_l\phi_m d\xi$$

where $i=\sqrt{-1}$, and ϕ_l and ϕ_m are the finite element shape functions associated with node l and node m on the interface between subdomains Ω^1 and Ω^2. Indeed, $\mathbf{M}_{bb}$ is positive definite, and it can be shown (see Theorem 1 in [12]) that

$$\begin{aligned}&\mathbf{K}^s - k^2\mathbf{M}^s \pm ik\mathbf{M}_I^s\\ &\text{where}\\ &\mathbf{M}_I^s \;=\; ik\begin{bmatrix} 0 & 0\\ 0 & \mathbf{M}_{bb}\end{bmatrix}\end{aligned} \tag{4}$$

is non singular for any value of k and independently of the value of the mesh size h.

We now extend our consideration to multiple subdomains and the exterior problem. For $s=1,\ldots,N_s$, we define $\widetilde{\mathbf{K}}^s = \hat{\mathbf{K}}^s$ if the subdomain Ω_s does not touch the external artificial boundary and $\widetilde{\mathbf{K}}^s = \hat{\mathbf{K}}^s + ik\mathbf{M}_S^s$ otherwise, where $\hat{\mathbf{K}}^s$ is defined as in the previous section. The modified Lagrangian formulation presented here can be related to alternative transmission conditions for the subdomain interfaces (see [12]) both within a FETI framework [3] and other approaches [1, 4]. For the case of N_s subdomains, this formulation becomes

$$\begin{aligned}\mathcal{L}(\mathbf{v}^s,\lambda) \;=\;& \sum_{s=1}^{s=N_s}(\frac{1}{2}\mathbf{v}^{s^T}\widetilde{\mathbf{K}}^s\mathbf{v}^s - \mathbf{f}^{s^T}\mathbf{v}^s) + \lambda^T\sum_{s=1}^{s=N_s}\mathbf{B}^s\mathbf{v}^s\\ &+ \sum_{s=1}^{s=N_s}\sum_{\Omega^s\cap\Omega^q\neq\{\emptyset\}}\frac{1}{2}(\mathbf{v}_b^{s^T}\mathbf{S}_{bb}^{s,q}\mathbf{v}_b^s - \mathbf{v}_b^{q^T}\mathbf{S}_{bb}^{s,q}\mathbf{v}_b^q)\end{aligned}$$

where $\mathbf{S}_{bb}^{s,q}$ is an interface matrix with nonzero values only on $\Omega^s\cap\Omega^q$, and constructed as the mass matrix associated with the degrees of freedom lying on $\Omega^s\cap\Omega^q$

$$\begin{aligned}\mathbf{S}_{bb}^{s,q} &= \epsilon^{s,q}ik\mathbf{M}_{bb}^{s,q}\\ [\mathbf{M}_{bb}^{s,q}]_{lm} &= \int_{\Omega^s\cap\Omega^q}\phi_l\phi_m d\xi\\ \epsilon^{q,s} = -\epsilon^{s,q} &= \pm 1\end{aligned}$$

Using a notation similar to that of Eq. (4), the Euler equations associated with the above modified Lagrangian can be written as

$$(\widetilde{\mathbf{K}}^s + ik\mathbf{M}_I^s)\mathbf{u}^s \;=\; \mathbf{f}^s - \mathbf{B}^{s^T}\lambda \tag{5}$$

$$\sum_{s=1}^{s=N_s} \mathbf{B}^s \mathbf{u}^s = 0 \tag{6}$$

where

$$\mathbf{M}_I^s = \begin{bmatrix} 0 & 0 \\ 0 & \sum\limits_{\Omega^s \cap \Omega^q \neq \{\emptyset\}} \epsilon^{s,q} \mathbf{M}_{bb}^{s,q} \end{bmatrix} \tag{7}$$

From Eqs. (5,7) and Theorem 1 in [12], it follows that, if $\mathbf{M}_I^s$ has a constant sign — that is, if $\forall q \;/\; \Omega^s \cap \Omega^q \neq \emptyset \;\; \epsilon^{s,q} = 1$, or $\forall q \;/\; \Omega^s \cap \Omega^q \neq \emptyset \;\; \epsilon^{s,q} = -1$ — the subdomain problem matrix $\widetilde{\mathbf{K}}^s + ik\mathbf{M}_I^s$ is non singular for any value of k and any value of the mesh size h. Also, in [12] an algorithm is proposed to determine the signs of the interfaces for arbitrary partitions that, in conjunction with Theorem 1 in [12], ensures that the resulting subdomain problem matrices $\widetilde{\mathbf{K}}^s + ik\mathbf{M}_I^s$ are always non singular.

From (6), we find that the interface problem associated with the regularized subdomain equations (5) is given by

$$\mathbf{F}_I \lambda = \mathbf{d} \tag{8}$$

where

$$\mathbf{F}_I = \sum_{s=1}^{s=N_s} \mathbf{B}^s (\widetilde{\mathbf{K}}^s + ik\mathbf{M}_I^s)^{-1} \mathbf{B}^{s^T}$$

$$\mathbf{d} = \sum_{s=1}^{s=N_s} \mathbf{B}^s (\widetilde{\mathbf{K}}^s + ik\mathbf{M}_I^s)^{-1} \mathbf{f}^s$$

Note that $\mathbf{F}_I$ is symmetric even if not Hermitian. For this reason, we choose the generalized conjugate residuals method (GCR) (cf. [23]) to solve the interface problem (8).

The FETI-H method

The methodology we follow here for preconditioning the regularized FETI method for exterior Helmholtz problems is based on the ideas proposed in [8] and [10]. Essentially, we propose to precondition at each iteration the interface residual generated by the GCR algorithm, by solving an auxiliary second-level problem obtained by projecting the interface problem (8) onto a suitable coarse space.

Let $\mathbf{r}^k$ denote the k-th residual associated with the solution by the GCR algorithm of the interface problem (8)

$$\mathbf{r}^k = \mathbf{d} - \mathbf{F}_I \lambda^k$$

The convergence of the regularized FETI method can be accelerated by modifying the GCR algorithm so that, at every iteration k, the interface residual $\mathbf{r}^k$ is orthogonal to a subspace represented by an interface matrix $\mathbf{Q}$

$$\mathbf{Q}^T \mathbf{r}^k = 0 \tag{9}$$

Indeed, condition (9) is a weighted-residual weak form of $\mathbf{r}^k = 0$, and therefore its effect at each iteration k is to reduce the error until $\mathbf{r}^k \to 0$. For example, if n_I

denotes the size of the interface problem, constructing an interface matrix $\mathbf{Q}$ with n_I linearly independent columns guarantees that after modification to enforce at every iteration $\mathbf{Q}^T\mathbf{r}^k = 0$, the regularized FETI method converges in one iteration. However, the subspace represented by the interface matrix $\mathbf{Q}$ must be chosen "coarse" enough to keep the overhead associated with enforcing $\mathbf{Q}^T\mathbf{r}^k = 0$ affordable. Eq. (9) is called in reference [10] an "optional admissible solution" constraint.

A straightforward approach to enforce this condition is to introduce the additional Lagrange multiplier $\mu = \mathbf{Q}\gamma$, where γ is a vector of additional unknowns, and modify the GCR algorithm to compute

$$\tilde{\lambda}^k \;=\; \lambda^k + \mu^k \;=\; \lambda^k + \mathbf{Q}\gamma^k \tag{10}$$

Substituting Eq. (10) into Eq. (9) gives

$$\mathbf{Q}^T\mathbf{F}_I\mathbf{Q}\gamma^k \;=\; \mathbf{Q}^T(\mathbf{d} - \mathbf{F}_I\lambda^k) \tag{11}$$

which shows that at each iteration k, γ^k can be obtained from the solution of an auxiliary "second-level coarse FETI" problem, which represents the projection of the regularized FETI interface problem (8) onto the "coarse" subspace represented by $\mathbf{Q}$.

From Eqs. (10,11), it follows that $\tilde{\lambda}^k$ can be computed as

$$\tilde{\lambda}^k \;=\; \mathbf{P}\lambda^k + \lambda^0$$

where $\mathbf{P}$ is the projector given by

$$\mathbf{P} \;=\; \mathbf{I} - \mathbf{Q}(\mathbf{Q}^T\mathbf{F}_I\mathbf{Q})^{-1}\mathbf{Q}^T\mathbf{F}_I$$

and λ^0 is given by

$$\lambda^0 \;=\; \mathbf{Q}(\mathbf{Q}^T\mathbf{F}_I\mathbf{Q})^{-1}\mathbf{Q}^T\mathbf{d} \tag{12}$$

which transforms the original regularized FETI interface problem into

$$\mathbf{P}^T\mathbf{F}_I\mathbf{P}\lambda \;=\; \mathbf{F}_I\mathbf{P}\lambda \;=\; \mathbf{P}^T\mathbf{d} \tag{13}$$

as $\mathbf{P}^T\mathbf{F}_I\mathbf{P} \;=\; \mathbf{F}_I\mathbf{P}$. We solve this problem by a projected generalized conjugate residuals method with the initial approximation given by (12). We note that this can also be interpreted as the GCR method for the interface problem (8) with a right preconditioner $\mathbf{P}$.

The coarse problem represented by $\mathbf{Q}$ is chosen in the form

$$\mathbf{Q} \;=\; \left[\; \mathbf{B}^1\mathbf{Q}^1 \quad \cdots \quad \mathbf{B}^s\mathbf{Q}^s \quad \cdots \quad \mathbf{B}^{N_s}\mathbf{Q}^{N_s} \;\right]$$

where $\mathbf{Q}^s$ is a matrix of local coarse vectors in subdomain Ω^s. In each subdomain, motivated by the fact that the solution can be approximated by a superposition of planar waves, we define column j of the matrix $\mathbf{Q}^s$ as

$$\mathbf{Q}^s_j(-) = e^{ik\boldsymbol{\Theta}_\mathbf{j}^\mathbf{T}\mathbf{x}(-)},$$

where $-$ indicates a degree of freedom corresponding to a given interface point, $\mathbf{x}(-)$ its nodal coordinates and $\boldsymbol{\Theta}_\mathbf{j}$ is the unitary "coarse direction" vector. In 2D, we choose the coarse directions as

$$\boldsymbol{\Theta}_\mathbf{j} = [\cos\theta_\mathbf{j}, \sin\theta_\mathbf{j}], \quad \text{where} \quad \theta_\mathbf{j} = (\mathbf{j}-\mathbf{1}) \times \frac{\mathbf{2}\pi}{\mathbf{N}_\theta}, \quad \mathbf{j} = \mathbf{1}, \ldots, \mathbf{N}_\theta;$$

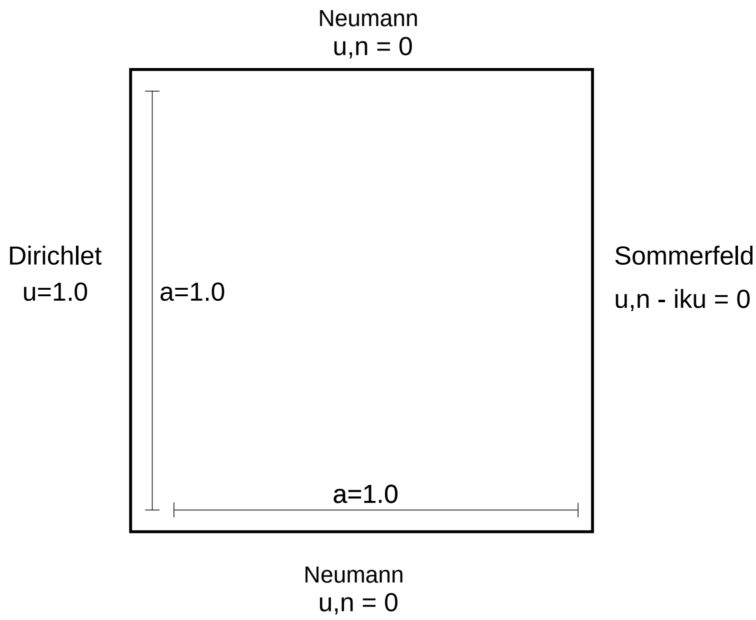


Figure 1 The guided wave problem

We choose N_θ among even integers in order to include opposite directions.

The 3D coarse directions are generated using the following algorithm. At each mesh node of coordinates $[x_n, y_n, z_n]$, we consider a cube centered at this node and uniformly discretize its surface by points of coordinates $[x_j, y_j, z_j], j = 1, \ldots, N_\theta$. The coarse direction vectors $\mathbf{\Theta_j}$ are then obtained by normalizing the vectors $[x_j - x_n, y_j - y_n, z_j - z_n]$. In this case, the choice of N_θ is given by the number of points on the surface of the cube.

We note that $\mathbf{Q}$ has $N_s \times N_\theta$ columns. Depending on the parameters of the problem, some of them may be linearly dependent, in which case they are filtered out during the factorization of the second-level problem matrix $\mathbf{Q}^T \mathbf{F}_I \mathbf{Q}$.

We refer to the regularized FETI method for complex problems equipped with the preconditioner presented herein as the FETI-H method.

Numerical results

To demonstrate the numerical scalability of the FETI-H method, we consider a guided wave problem in two and three dimensions. The 2D version of this problem is depicted in Fig. 25. The 3D version corresponds to a cube where one face is subjected to the Dirichlet boundary condition, the opposite face to the Sommerfeld condition, and the other faces to the Neumann boundary condition. We consider various configurations of this problem and demonstrate numerically the scalability of the FETI-H method with respect to the subdomain size H and the wavenumber k. Results concerning the scalability with respect to the mesh size h can be found in [12].

ka	N_θ	N_S	Size of the coarse problem	Number of iterations
20	4	25	88	33
20	4	49	180	31
20	4	81	304	28
20	8	25	176	19
20	8	49	354	20
20	8	81	600	18
20	16	25	262	18
20	16	49	471	19
20	16	81	713	18
40	4	25	88	70
40	4	49	180	56
40	4	81	304	61
40	8	25	176	25
40	8	49	354	25
40	8	81	600	22
40	16	25	262	19
40	16	49	471	22
40	16	81	713	22
60	4	25	88	148
60	4	49	180	141
60	4	81	304	110
60	8	25	176	50
60	8	49	354	42
60	8	81	600	22
60	16	25	262	18
60	16	49	471	18
60	16	81	713	18

Table 1 The guided wave problem in 2D – $h = 1/315$, total size of the problem = 99225 dofs

In both cases, we perform the same computations but for three different values of the wavenumber corresponding to $ka = 20, 40$, and 60 in 2D, and $ka = 5, 10$, and 15 in 3D. We also consider several $1/H \times 1/H$ mesh partitions with 25, 49 and 81 subdomains. We report the obtained performance results in Table 1 for the 2D case, and Table 2 for the 3D one. Note that in these tables, the size of the coarse problem is computed as the number of linearly independent columns found during the factorization of the second-level problem matrix $\mathbf{Q}^T\mathbf{F}_I\mathbf{Q}$. Hence, this size is less or equal to $N_\theta \times N_s$.

The results summarized in both Table 1 and Table 2 confirm the scalability of the FETI-H method with respect to both the wavenumber and the number of subdomains.

ka	N_θ	N_S	Size of the coarse problem	Number of iterations
5	0	25	0	42
5	0	64	0	99
5	0	125	0	73
5	2	27	178	26
5	2	64	450	26
5	2	125	908	27
5	3	27	428	26
5	3	64	579	27
5	3	125	1211	29
5	4	27	461	28
5	4	64	602	28
5	4	125	1269	34
10	0	27	0	67
10	0	64	0	151
10	0	125	0	116
10	2	27	178	31
10	2	64	450	30
10	2	125	908	33
10	3	27	597	22
10	3	64	1276	31
10	3	125	2537	41
10	4	27	823	29
10	4	64	1441	36
10	4	125	2835	41
15	0	27	0	112
15	0	64	0	331
15	0	125	0	251
15	2	27	178	47
15	2	64	450	38
15	2	125	908	42
15	3	27	598	24
15	3	64	1471	27
15	3	125	2993	26
15	4	27	1067	23
15	4	64	1626	44
15	4	125	3118	47

Table 2 The guided wave problem in 3D – $h = 1/60$, total size of the problem = 226981 dofs

Acknowledgements

The first and third authors acknowledge the partial support by the Office of Naval Research under Grant N-00014-95-1-0663. The second author acknowledges the financial support by a Brazilian CAPES Fellowship, Process No. 1233/95-2.

REFERENCES

J. Benamou and B. Desprès. A domain decomposition method for the Helmholtz equation and related optimal control problems. Rapport de recherche INRIA No. 2791, Février 1996.

X.-C. Cai, M. A. Casarin Jr., F. W. Elliott Jr., and O. B. Widlund. Two-level algebraic multigrid for the Helmholtz problem. In *Contemporary Mathematics 218 (1998)*, pages 390–398.

A. de La Bourdonnaye, C. Farhat, A. Macedo, F. Magoulès, and F. X. Roux. A non-overlapping domain decomposition method for the exterior Helmholtz problem. In *Contemporary Mathematics 218 (1998)*, pages 42–66.

B. Desprès. Décomposition de domaine et problème de Helmholtz. *Compte Rendu de l'Académie des Sciences*, No 311 (Série I):313–316, 1990.

Q. V. Dihn, R. Glowinski, and J. Periaux. Solving elliptic problems by domain decomposition methods with applications. In A. Schoenstadt, editor, *Elliptic Problem Solvers II.* Academic Press, 1984.

M. R. Dorr. A domain decomposition preconditioner with reduced rank interdomain coupling. *Appl. Numer. Math.*, 8:333–352, 1991.

C. Farhat. A Lagrange multiplier based divide and conquer finite element algorithm. *J. Comput. Sys. Engrg.*, 2:149–156, 1991.

C. Farhat, P. S. Chen, and J. Mandel. A scalable Lagrange multiplier based domain decomposition method for implicit time-dependent problems. *Internat. J. Numer. Meths. Engrg.*, 38:3831–3854, 1995.

C. Farhat, P. S. Chen, J. Mandel, and F. X. Roux. The two-level FETI method for plate problems - Part II: extension to shells, parallel implementation and performance results. *Comput. Meths. Appl. Mech. Engrg.*, 155:153–180, 1998.

C. Farhat, P. S. Chen, F. Risler, and F. X. Roux. A unified framework for accelerating the convergence of iterative substructuring methods with Lagrange multipliers. *Internat. J. Numer. Meths. Engrg.*, 42:257–288, 1998.

C. Farhat, P. S. Chen, and F. X. Roux. The dual Schur complement method with well-posed local Neumann problems: regularization with a perturbed Lagrangian formulation. *SIAM J. Sc. Stat. Comput.*, 14:752–759, 1993.

C. Farhat, A. Macedo, and M. Lesoinne. A two-level domain decomposition method for the iterative solution of high frequency exterior Helmholtz problems. Technical Report CU-CAS-98-06, Center for Aerospace Structures, University of Colorado, 1998. Numerische Mathematik, in press.

C. Farhat and J. Mandel. The two-level FETI method for static and dynamic plate problems - Part I: an optimal iterative solver for biharmonic systems. *Comput. Meths. Appl. Mech. Engrg.*, 155:129–152, 1998.

C. Farhat, J. Mandel, and F. X. Roux. Optimal convergence properties of the FETI domain decomposition method. *Comput. Meths. Appl. Mech. Engrg.*, 115:367–388, 1994.

C. Farhat and F. X. Roux. An unconventional domain decomposition method for an efficient parallel solution of large-scale finite element systems. *SIAM J. Sc. Stat. Comput.*, 13:379–396, 1992.

L. Franca, C. Farhat, M. Lesoinne, and A. Russo. Unusual stabilized finite element methods and residual-free bubbles. *Internat. J. Numer. Meths. in*

Fluids, 27:159–168, 1998.

L. Franca, C. Farhat, A. Macedo, and M. Lesoinne. Residual-free bubbles for the Helmholtz equation. *Internat. J. Numer. Meths. Engrg.*, 40:4003–4009, 1997.

R. Glowinski and M. F. Wheeler. Domain decomposition and mixed finite element methods for elliptic problems. In R. Glowinski, G. Golub, G. Meurant, and J. Périaux, editors, *Proc. First SIAM Conference on Domain Decomposition Methods for Partial Differential Equations*, pages 144–172. SIAM, 1988.

B. Lee, T. Manteuffel, S. McCormick, and J. Ruge. Multilevel first-order system least squares for Helmholtz equation. In *Procs. of 2nd International Conf. on Approx. and Num. Meths. for the Solution of the Maxwell Equations, Washington, D.C, 1993, John Wiley and Sons.*

M. Malhotra, R. Freund, and P. M. Pinsky. Iterative solution of multiple radiation and scattering problems in structural acoustics using a block quasi-minimal residual algorithm. *Comp. Meths. Appl. Mech. Engrg.*, 146:173–196, 1997.

Jan Mandel and Radek Tezaur. Convergence of a substructuring method with Lagrange multipliers. *Numerische Mathematik*, 73:473–487, 1996.

Jan Mandel, Radek Tezaur, and Charbel Farhat. A scalable substructuring method by Lagrange multipliers for plate bending problems. SIAM J. Numer. Anal., to appear.

Y. Saad. *Iterative methods for sparse linear systems.* PWS Publishing Company, Boston, 1995.

P. Vaněk, J. Mandel, and Marian Brezina. Two-level algebraic multigrid for the Helmholtz problem. In *Contemporary Mathematics 218 (1998)*, pages 349–356.

26

Domain Decomposition with Local Fourier Bases applied to Frontal Polymerisation problems

M. Garbey , D. Tromeur-Dervout [1] [2]

Introduction

This paper describes a domain decomposition technique and high order accurate method that allows solving a nonlinear system of reaction diffusion equations coupled with Navier Stokes equations in a channel with local Fourier approximations. The present method is a generalization of the Israeli et al (1993) method which applies domain decomposition with local Fourier bases to the Helmoltz's problem. In the present work several new difficulties occur. First the problem is an unsteady and non linear problem which makes the periodic extension delicate to construct in terms of stability and accuracy. Secondly we use a stream function biharmonic formulation of the incompressible Navier Stokes equation in two space dimensions: the application of domain decomposition with local Fourier bases to a fourth order operator is more difficult to achieve than for a second order operator. This present work is a generalization of our previous work on combustion by the authors [GTD97] and the use of Fourier bases to solve non periodic problems [GTD98a][Gar97].

Our previous 2D numerical simulation of quasi-planar frontal polymerization (FP) used an adaptative domain decomposition method based on a piecewise C^1 Chebyshev polynomial approximation in the direction of propagation of the front and a Fourier approximation in a direction parallel to the front for periodic as well

1 This work was supported by the Région Rhône Alpes
2 CDCSP - University Lyon 1,69622 Villeurbanne France
{garbey,dtromeur}@cdcsp.univ-lyon1.fr, http://cdcsp.univ-lyon1.fr
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

as nonperiodic boundary conditions. This adaptative method is very efficient and accurate but it requires a priori knowledge of the structure of the solution. We refer to [GTD97, GTD98b] for the parallel efficiency and the numerical accuracy of this method.

In this paper we use the same problem as a test case, but the methodology that we develop is based on a totally different philosophy. We do not use adaptivity and compute the solution with (local) pseudo-spectral Fourier approximations in such way that we can use a fast direct solver per subdomain and a large total number of discretization points efficiently on a parallel computer. In particular, by introducing a non-overlapping domain decomposition, we avoid the problem of global transposition of matrices induced by a two dimensional fast Fourier transform on a parallel computer. In addition, we use no a priori knowledge on the structure of the solution since there is no space adaptativity but possibly filtering to remove spurious oscillations. We expect the question of adaptativity to be resolved as a separate issue, using appropriate numerical generation of a 2D mapping applied to a regular grid structure [J.A98] [LMS98].

The outline of this article is as follows. Section 26 describes the governing equations of the model problem. Section 26 (Section 26) describes the algorithms for the stream function solution without domain decomposition (with domain decomposition). Section 26 deals with the numerical accuracy and stability of the method. Section 26 presents a numerical result obtained with the new methodology for a non-planar flame front. Our conclusions are given in Section 26.

Governing Equations

We consider a model that couples a reaction-diffusion system for the temperature T and the reactant concentration C, describing a one-step exothermic chemical process, with the incompressible Navier-Stokes equation using the Boussinesq approximation; it is worth noticing that we use the biharmonic formulation of Navier-Stokes with Ψ as the stream function. We refer to [GTD98b] for the precise statement of the problem and parameters, and we simply recall the basic equations

$$\partial T/\partial t + (\partial\Psi/\partial z)(\partial T/\partial x) - (\partial\Psi/\partial x)(\partial T/\partial z) = \Delta T + C \exp\frac{ZT}{1+\delta(1-T)},$$

$$\partial C/\partial t + (\partial\Psi/\partial z)(\partial C/\partial x) - (\partial\Psi/\partial x)(\partial C/\partial z) = \epsilon\Delta C - C \exp\frac{ZT}{1+\delta(1-T)},$$

$$\partial\Delta\Psi/\partial t = (\partial\Psi/\partial x)(\partial\Delta\Psi/\partial z) - (\partial\Psi/\partial z)(\partial\Delta\Psi/\partial x) + \Delta\Delta\Psi - RP\vec{g}.\vec{\nabla}T, \quad (1)$$

$$T\to 0,\ C\to 1,\ \Psi\to 0,\ as\ z\to -\infty,\ \ T\to 1,\ C\to 0,\ \Psi\to 0,\ as\ z\to +\infty$$

$$\Psi(0,z)=0,\ \partial\Psi/\partial x(0,z)=0,\ \Psi(L,z)=0, \partial\Psi/\partial x(L,z)=0, \forall z \quad (2)$$

$$\partial T/\partial x(0,z)=0,\ \partial T/\partial x(L,z)=0,\ \partial C/\partial x(0,z)=0,\ \partial C/\partial x(L,z)=0,\ \forall z$$

Many different possible nonlinear regimes of the solution exist. In addition, the systems have a well-known one dimensional traveling wave solution $(T_0(z), C_0(z), \Psi\equiv 0)$. The existence and stability of these solutions depend on the specific value of the bifurcation or control parameters, such as the Zeldovich number Z or the Rayleigh number R.

In addition Z is large, thus making the problem difficult to compute. We are mainly interested in using this problem as a nontrivial test case to demonstrate and validate the feasibility of our approach with local Fourier bases.

A First Algorithm without Domain Decomposition to Solve the NS Equation

We consider the following semi-implicit Euler scheme for the time discretisation of equation (1) as in [GTD98b]:

$$\left\{\begin{array}{l} \Delta\Psi^{n+1} - dt\Delta\Delta\Psi^{n+1} = dt\ ((\partial\Psi^n/\partial x)(\partial\Delta\Psi^n/\partial z) - (\partial\Psi^n/\partial z)(\partial\Delta\Psi^n/\partial x) \\ \qquad\qquad +\Delta\Psi^n - dt\ f(x,z,t_n), \quad (x,z) \in (0,L)\times(-H,H), \\ \Psi^{n+1}(0,z) = 0, \quad (\partial\Psi^{n+1}/\partial x)(0,z) = 0, \quad z \in (-H,H), \\ \Psi^{n+1}(L,z) = 0, \quad (\partial\Psi^{n+1}/\partial x)(L,z) = 0, \quad z \in (-H,H). \end{array}\right. \tag{3}$$

We look for a z-periodic solution of period H with H large. As a matter of fact Ψ vanishes exponentially when $|z|$ goes to infinity and if H is large enough for the boundary conditions to have no influence on the dynamic of the combustion wave. For simplicity of notation, we will assume in the following that $H = \pi$. f and Ψ at time t_n are approximated by the following discrete Fourier expansion:

$$f(x,z,t_n) = \Sigma_{k=\frac{-N_z}{2},\frac{N_z}{2}-1} \hat{f}_k^n(x)\ e^{iky},\ \Psi^n(x,z,t_n) = \Sigma_{k=\frac{-N_z}{2},\frac{N_z}{2}-1} \hat{\Psi}_k^n(x)\ e^{iky},$$

with $y = \frac{(z+H)\ \pi}{H}$. Let $F^n(x,z)$ be the right hand side of equation (3). From the approximation,

$$(\partial\Psi^n/\partial x)(\partial\Delta\Psi^n/\partial y) - (\partial\Psi^n/\partial y)(\partial\Delta\Psi^n\partial x)$$
$$\approx i\Sigma_{k=\frac{-N_z}{2},\frac{N_z}{2}-1}[e^{iky}\Sigma_{k_1+k_2=k}(\hat{\Psi^n}'_{k_1}\ k_2(\hat{\Psi^n}''_{k_2} - k_2^2\hat{\Psi^n}_{k_2}) - \hat{\Psi^n}_{k_1}\ k_1(\hat{\Psi^n}'''_{k_2} - k_2^2\hat{\Psi^n}'_{k_2}))] \quad ,$$

we obtain the discrete Fourier expansion of F^n. We then have to solve the uncoupled fourth order N_z ODE's at each time step. For $k = 0$, we have:

$$L_0[\hat{\Psi}_0^{n+1}] = (\hat{\Psi}_0^{n+1})'' - dt\ (\hat{\Psi}_0^{n+1})'''' = \hat{F}_0^n,\ x \in (0,L), \tag{4}$$
$$\hat{\Psi}_0^{n+1}(0) = (\hat{\Psi}_0^{n+1})'(0) = \hat{\Psi}_0^{n+1}(L) = (\hat{\Psi}_0^{n+1})'(L) = 0,$$

and for $k \neq 0$:

$$\begin{array}{r} L_k[\hat{\Psi}_k^{n+1}] = (\hat{\Psi}_k^{n+1})'' - k^2\hat{\Psi}_k^{n+1} - dt\ ((\hat{\Psi}_k^{n+1})'''' \\ -2k^2(\hat{\Psi}_k^{n+1})'' + k^4\hat{\Psi}_k^{n+1}) = \hat{F}_k^n,\ x \in (0,L), \\ \hat{\Psi}_k^{n+1}(0) = (\hat{\Psi}_k^{n+1})'(0) = \hat{\Psi}_k^{n+1}(L) = (\hat{\Psi}_k^{n+1})'(L) = 0. \end{array} \tag{5}$$

In order to compute $\hat{\Psi}_k^{n+1}(x)$ with a Fourier transform in the x direction as well as in the z direction, we use the technique as in [GTD98b] (see also [IVA93] [IVA94a] [IVA94b]) based on the superposition principle and the construction of a smooth periodic extension of the right hand sides. We split the unknowns as follows:

$$\hat{\Psi_k}^{n+1}(x) = \hat{\Psi_{kF}}^{n+1}(x) + \hat{\Psi_{kC}}^{n+1}(x), x \in [0,L] \tag{6}$$

This splitting is efficient because one can use a fast Fourier transform to compute $\hat{\Psi}_{k\,F}^{n+1}$ and analytical formulae for $\hat{\Psi}_{k\,C}^{n+1}$. In what follows, we will omit subscript k and $n+1$ to simplify the notations. Let $d > 0, d \in \mathbb{R}$, and let $[0, L+d]$ be an extension of the domain $(0, L)$. Let $\bar{F}_k(x)$ be a smooth periodic extension of $\hat{F}_k(x)$ on the interval $[0, L+d]$. We look for the $L+d$ periodic solution of the nonhomogeneous problem

$$L_k[\hat{\Psi}_F] = \bar{F},\ x \in (0, L+d),\ \hat{\Psi}_F\ L+d\ periodic,\ \hat{\Psi}_F\ \in\ C^{n+4}(\mathbb{R}) \tag{7}$$

Then we retrieve the homogeneous Neuman and Dirichlet boundary conditions satisfied by $\hat{\Psi}_k^{n+1}$ by computing the corrector term Ψ_C. These corrector terms satisfy the ODE problems

$$\begin{cases} L_k[\hat{\Psi}_C] = 0,\ x \in (0, L), \\ \hat{\Psi}_C(0) = -\hat{\Psi}_F(0),\ (\hat{\Psi}_C)'(0) = -(\hat{\Psi}_F)'(0),\ \hat{\Psi}_C(L) = -\hat{\Psi}_F(L),\ (\hat{\Psi}_C)'(L) = -(\hat{\Psi}_F)'(L) \end{cases}$$

Since the operators L_k are fourth order linear operators with constant coefficients, one can compute the basis functions for the four dimensional vector space of the solutions once and for all. The solution is written as $\hat{\Psi}_{kC} = \alpha_k v_k(x) + \beta_k w_k(x) + \gamma_k r_k(x) + \delta_k s_k(x)$, and the bases functions are explicitely given with formulae, for $k = 0$:

$$v_0(x) = exp(-\frac{1}{\sqrt{dt}}(x-0)),\ w_0(x) = exp(-(\frac{1}{\sqrt{dt}}(L-x)),\ r_0(x) = x,\ s_0(x) = (L-x), \tag{8}$$

and for $k \neq 0$:

$$\begin{aligned} v_k(x) &= exp(-\sqrt{k^2 + \frac{1}{dt}}\,(x-0)) \qquad & w_k(x) &= exp(-\sqrt{k^2 + \frac{1}{dt}}\,(L-x)) \\ r_k(x) &= exp(-|k|\,(x-0)) \qquad & s_k(x) &= exp(-|k|\,(L-x)) \end{aligned}$$

The coefficients $(\alpha_k, \beta_k, \gamma_k, \delta_k)$ are the solution of four-by-four linear systems that are solved at each time step. The boundary conditions on the left and on the right are numerically decoupled when the wave number k is large enough. The derivatives of $\hat{\Psi}_{kC}$ are readily computed from the previous formula. The derivatives of $\hat{\Psi}_F^k$ follow from its discrete Fourier expansion. We assemble the right hand side of equation (3) at each time step using the splitting (6) for the derivatives as well.

Our approach differs from that of Israeli et al in the way we compute a sufficiently regular periodic extension of the right hand sides. Let us recall that the smoothness of this extension is the essential limitation on the spectral accuracy of the method. More precisely, if the right hand side has regularity C^q, the numerical scheme is of order $q+4$ at most.

When the right hand-side is a given analytical function that can be defined for the interval $[0, L+d]$, one could use a so-called bell function B (as in Israeli et al) that is equal to one in [O,L] and zero in the vicinity of $L + d/2$. B times the right-hand side is then a smooth periodic function of period $L+d$. [IVA94b] [IVA94a] show evidence of the accuracy of this method for the Laplace equation or the Helmoltz problem. However, in our computation the right hand side is given *numerically* at each time step *only within* the physical domain of computation. We therefore use a numerical procedure to derive a smooth extension of this function. We proceed as follows.

We consider the exact or computed values of the derivatives of $\bar{F}_k$ up to order q at $x = 0$ and $x = L$. The classic Hermite interpolation allows us to define a polynomial function P on $[L, L+d]$ of degree $2q+1$ that interpolates function $\bar{F}_k$ with the following conditions:

$$\text{for } j = 0, ..., q,\ P^{(j)}(L) = \hat{F}_k^{(j)}(L),\ P^{(j)}(L+d) = \hat{F}_k^{(j)}(0).$$

The extended right hand side is then

$$\bar{F}_k = \begin{cases} \hat{F}_k,\ \forall x \in [0, L] \\ P(x),\ \forall x \in]L, L+d[. \end{cases} \tag{9}$$

In practice we require a C^2 continuity condition at the end points $x = 0$ and $x = L+d$ and the derivatives are computed by using sixth order one sided finite differences.

Domain Decomposition Algorithm

The next step is to introduce a domain decomposition in the x direction for each k-mode equation (1). We split $[0, L]$ into nd nonoverlapping subdomains of equal sizes, which we denote generically as $(0, l)$.

For each subdomain we apply the same splitting of the unknowns as described above, and in addition we impose C^3 continuity on the solution at the artificial interfaces. We compute the extension of the right hand sides for each subdomain and its corresponding periodic solution with a local Fourier discrete approximation. This part of the algorithm is exactly what we had for the single domain case, but it is applied for each subdomain in parallel.

Note that for mode 0, the operator with a periodic boundary condition is singular. The solution is then defined up to a shift. However the corrector term substracts this shift and the superposition principle gives the unique solution of (4).

We compute the corrector term for each subdomain in order to retrieve the C^3 continuity of the solution at the artificial interfaces. Let us denote by $(v_k^j, w_k^j, r_k^j, s_k^j)$ the set of basis functions for the corrector in each subdomain j and by $(\alpha_k^j, \beta_k^j, \gamma_k^j, \delta_k^j)$ the corresponding coefficients; in the local coordinate system of the subdomain, the basis functions are identical to the monodomain basis function. With two subdomains for example, the matrix of the interface problem for the unknown coefficient vector $\vec{T} = (\alpha_k^1, \beta_k^1, \gamma_k^1, \delta_k^1, \alpha_k^2, \beta_k^2, \gamma_k^2, \delta_k^2)^T$ writes

$$\begin{pmatrix}
v\,{}_k^1(0) & w\,{}_k^1(0) & r\,{}_k^1(0) & s\,{}_k^1(0) & 0 & 0 & 0 & 0 \\
v'\,{}_k^1(0) & w'\,{}_k^1(0) & r'\,{}_k^1(0) & s'\,{}_k^1(0) & 0 & 0 & 0 & 0 \\
v\,{}_k^1(l) & w\,{}_k^1(l) & r\,{}_k^1(l) & s\,{}_k^1(l) & -v\,{}_k^2(0) & -w\,{}_k^2(0) & -r\,{}_k^2(0) & -s\,{}_k^2(0) \\
v'\,{}_k^1(l) & w'\,{}_k^1(l) & r'\,{}_k^1(l) & s'\,{}_k^1(l) & -v'\,{}_k^2(0) & -w'\,{}_k^2(0) & -r'\,{}_k^2(0) & -s'\,{}_k^2(0) \\
v''\,{}_k^1(l) & w''\,{}_k^1(l) & r''\,{}_k^1(l) & s''\,{}_k^1(l) & -v''\,{}_k^2(0) & -w''\,{}_k^2(0) & -r''\,{}_k^2(0) & -s''\,{}_k^2(0) \\
v'''\,{}_k^1(l) & w'''\,{}_k^1(l) & r'''\,{}_k^1(l) & s'''\,{}_k^1(l) & -v'''\,{}_k^2(0) & -w'''\,{}_k^2(0) & -r'''\,{}_k^2(0) & -s'''\,{}_k^2(0) \\
0 & 0 & 0 & 0 & v\,{}_k^2(l) & w\,{}_k^2(l) & r\,{}_k^2(l) & s\,{}_k^2(l) \\
0 & 0 & 0 & 0 & v'\,{}_k^2(l) & w'\,{}_k^2(l) & r'\,{}_k^2(l) & s'\,{}_k^2(l)
\end{pmatrix}$$

Note that for large wave number k, the local interface problems are fully decoupled because of the exponential decay of the basis function. In such a case, the domain decomposition algorithm only requires local communication between node processing adjacent domains.

Numerical Accuracy of the Domain Decomposition Method

We now consider the accuracy of the domain decomposition with respect to the number of Fourier modes and the number of subdomains. It is important to notice that problems (4, 5) are singular perturbation problems when the time step dt is small and/or the wave number k is large. The sensitiviy of the method with respect to dt and k will also be tested. In order to test the numerical accuracy of our domain decomposition method, we consider each fourth order wave equation analogous to (5) with k replaced by $k\frac{\pi}{H}$ separately and choose the right hand side $\hat{F}_k^n$ such that the fourth order polynomial $x^2 * (x - 4\pi)^2/(16 * \pi^4)$ is the exact steady solution. The size of the domain of computation is given by $H = 25$ and $L = 2\pi$. In the following, we use the time marching scheme starting from the trivial initial condition until convergence to the steady solution is reached. We measure the difference between the converged numerical solution and the exact steady solution in maximum norm.

Table 1 gives the error in a maximum norm for the computed solution corresponding to the wave number $k = 0$ and $k = 1$ with a space step of order 0.05 and several numbers of subdomains. The total number of discretisation points in a physical domain is $2N_x$. The number of discretization points used for the extension is $2N_r$ for each subdomain. The number of Fourier modes per subdomain depends on the number of domains n_d and it is $N_x/n_d + N_r$. The time step is $dt = 0.1$

We see in all cases that $N_r = 16$ gives better results than $N_r = 8$. More precisely the accuracy depends strongly on the size of the extension only for large number of subdomains. This means that if the number of Fourier modes per subdomain is large enough the accuracy is not so sensitive to the size of the extension. The results of this table for mode $k = 1$ are always better than the corresponding results for the zero mode no matter what the shift is.

In particular, the error decreases with the number of subdomains for mode one. We observe the inverse for the zero mode except when $N_x = 256$ and $N_r = 16$, which is when the number of Fourier modes per subdomain is large.

Let us now consider four subdomains and study the sensitivity of the method to the time step, wave number k and the size of the extension per subdomain. Table 2 gives the error in maximum norm depending on the percentage of the extension i.e $100 \times d/(L + d)$ and the number of Fourier modes N_y for different couples (k, dt) of time steps and wave numbers. N_y is the number of Fourier modes per subdomain *including* the extension.

First we observe that the accuracy of the method deteriorates when the time step goes to zero: this is no surprise since the problem becomes increasingly singular as dt goes to zero. Eventually the time dependant schemes diverge for the zero mode equation if the space step is larger than the boundary layer thickness $\sqrt{dt}$ or if the extension is not large enough. The comparison between results for $k = 1$ and $k = 10$ or 30 with the same time step shows that accuracy deteriorates when k grows. The thickness of the boundary layer of the wave equation is $\sqrt{\frac{1}{k^2} + dt}$ and this phenomenon can be interpreted as above. However the amplitude of the solution Ψ_k should decrease with the wave number k in two dimensional problems and this phenomenon should not significantly affect the accuracy of the overall solution.

In all cases it is also significantly better to take a large extension of the subdomain

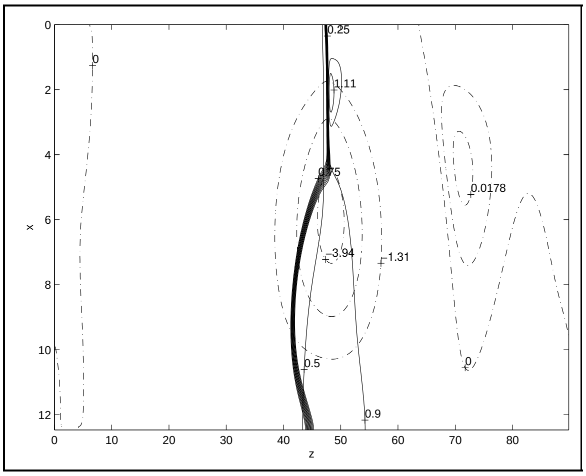


Figure 1 Superposition of the stream function contour lines to the hot spot formation; localization of the front is given by the thick line

rather than a small one. We speculate that small extensions lead to stability problems of the time marching scheme.

The method is at least of order 5 for mode 0 and for $d/(L+d) = 50\%$ or 25%. The order of the method for the higher modes 0 is less dependant on the value $d/(L+d)$, but differs with the mode value: (order 3.2 for k=30, order 4.3 for k=10). Nevertheless the decrease of the order of convergence is compensated for by the fact that the module of the Fourier coefficients decreases steeply with increasing mode value.

Results on FP process in Liquid (i.e R=0.5)

In order to solve our model problem in Frontal polymerization given in Section 26, we have combined the domain decomposition algorithm given above for the Navier Stokes equation with the analogous algorithm for a second order system of reaction diffusions describing the exothermic one-step chemical process [GTD98a]. We refer to that paper for preliminary results on the efficiency of our parallel algorithm. A detailed description of the implementation and parallel efficiency of the domain decomposition will be reported elsewhere. We have applied this methodology to compute non-planar flame front.

Figure 1 shows the effect of hydrodynamics on the structure of the flame front when the channel is horizontal. Thin solid lines represent the temperature isovalues while dash lines represent the stream function isovalues. The thick solid line represents the location of the front, centered on the level set $C = 0.5$. The computation was done with a total of $N_x = 112$ by $M_z = 256$ modes on a physical domain of size $[0, 4\pi] \times [0, 90]$. The Zeldovich number is equal to 6., the Rayleigh number is equal to 1.5, the Prandlt number is $P = 1$, the mass diffusion is given by $\epsilon = 0.02$, and the time step is set to $dt = 0.01$. This solution is a travelling wave moving toward the left with a hot spot close to the top wall of the horizontal channel. Gravity is vertical. The location of the hot spot and the front curvature of the concentration profile are closely related to the circular motion of the flow.

Conclusions

We have developed and implemented a domain decomposition methodology based on local Fourier approximations and the superposition principle to solve incompressible Navier Stokes equation in two space dimensions.

This approach has been generalized to solve a frontal polymerisation model. It allows us to tackle complex two D nonlinear regimes without any a priori hypothesis on the structure of the flame front.

REFERENCES

[Gar97] Garbey M. (1997) On some applications of the superposition principle with (local) fourier basis. *Preprint CDCSP-97-02* to appear in SIAM J. of Scientific Computing.

[GTD97] Garbey M. and Tromeur-Dervout D. (1997) Massively parallel computation of stiff propagating fronts. *Combustion Theory and Modelling* 1(3): 271–294.

[GTD98a] Garbey M. and Tromeur-Dervout D. (1998) Application of local fourier basis to domain decomposition in combustion problems. In Ecer A. and al (eds) *Proc. Parallel CFD98*. North-Holland, Taiwan. to appear.

[GTD98b] Garbey M. and Tromeur-Dervout D. (1998) A new parallel solver for non periodic incompressible navier-stokes equations with fourier basis: Application to frontal polymerisation. *J. Comp. Phys.* 145(1): 316–331.

[IVA93] Israeli M., Vozovoi L., and Averbuch A. (1993) Spectral multidomain technique with local fourier basis. *J. of Scientific Computing* 8(2): 135–149.

[IVA94a] Israeli M., Vozovoi L., and Averbuch A. (1994) Domain decomposition methods with local fourier basis for parabolic problems. *Comtemporary Mathematics* 157.

[IVA94b] Israeli M., Vozovoi L., and Averbuch A. (1994) Spectral multidomain technique with local fourier basis. *J. of Scientific Computing* 9(3): 311–325.

[J.A98] J.A.Mackenzie (1998) The efficient generation of simple two-dimensionnal adaptive grids. *SIAM J. Scientific Compting* 19(4): 1340–1365.

[LMS98] L.S. Mulholland W. H. and Sloan D. (1998) Pseudospectral solution of near-singular problems using numerical coordinate transformations based on adaptivity. *SIAM J. Scientific Compting* 19(4): 1261–1289.

Mode 0	number of subdomains				Mode 1	number of subdomains			
N_x=256	2	4	8	16	N_x=256	2	4	8	16
N_r=16	4.6e-5	1.9e-5	1.3e-5	9.3e-6	N_r=16	4.2e-7	2.6e-6	8.8e-7	4.2e-7
N_r=8	4.6e-5	4.6e-4	7.7e-4	1.4e-3	N_r=8	1.2e-5	5.5e-6	2.6e-6	1.3e-6
N_x=128	2	4	8	16	N_x=128	2	4	8	16
N_r=16	7.7e-5	8.8e-5	1.1e-4	1.3e-4	N_r=16	1.8e-5	1.0e-5	4.5e-6	2.4e-6
N_r=8	1.6e-4	6.3e-4	1.2e-3	2.0e-3	N_r=8	7.9e-5	3.9e-5	1.8e-5	9.9e-6

Table 1 accuracy with respect to the number of subdomains

	$k=0$, $dt=0.1$				$k=1$, $dt=0.1$			
N_x	50%	25%	12.5%	6.25%	50%	25%	12.5%	6.25%
16	7.55e-3	4.33e-1	div	5.39e-1	1.65e-4	8.59e-4	5.80e-3	9.70e-3
32	8.42e-5	7.62e-3	4.05e-1	div	6.73e-6	3.47e-5	3.02e-4	2.46e-3
64	2.52e-6	1.56e-5	7.04e-3	3.94e-1	2.44e-7	1.02e-6	1.07e-5	1.20e-4
128	6.00e-7	1.24e-6	4.77e-6	6.74e-3	7.75e-9	2.77e-8	3.41e-7	4.46e-6
	$k=0$, $dt=0.01$				$k=1$, $dt=0.01$			
N_x	50%	25%	12.5%	6.25%	50%	25%	12.5%	6.25%
16	3.75e-2	1.22e-0	div	div	3.30e-4	1.73e-3	1.13e-2	1.98e-2
32	7.90e-4	7.49e-2	8.37e-1	div	1.48e-5	7.57e-5	6.51e-4	5.31e-3
64	2.45e-5	1.54e-4	7.17e-2	8.46e-1	5.53e-7	2.27e-6	2.38e-5	2.66e-4
128	6.05e-7	4.77e-6	5.38e-5	6.90e-2	1.73e-8	6.14e-8	7.58e-7	9.91e-6
	$k=0$, $dt=0.001$				$k=1$, $dt=0.001$			
N_x	50%	25%	12.5%	6.25%	50%	25%	12.5%	6.25%
16	div	div	div	div	div	7.27e-3	div	div
32	3.34e-3	6.61e-1	div	div	7.10e-5	4.11e-4	3.57e-3	2.87e-2
64	1.67e-4	1.42e-3	div	div	3.67e-6	1.39e-5	1.48e-4	1.65e-3
128	5.26e-6	4.48e-5	5.30e-4	div	1.14e-7	3.86e-7	4.85e-6	6.38e-5
	$k=10$, $dt=0.1$				$k=30$, $dt=0.1$			
N_x	50%	25%	12.5%	6.25%	50%	25%	12.5%	6.25%
16	2.92e-4	1.72e-3	8.37e-3	2.09e-2	2.89e-4	1.66e-3	7.06e-3	2.20e-2
32	3.74e-5	2.29e-4	1.46e-3	8.74e-3	3.99e-5	2.54e-4	1.55e-3	7.52e-3
64	3.79e-6	2.05e-5	1.38e-4	1.04e-3	5.02e-6	3.23e-5	2.18e-4	1.45e-3
128	2.56e-7	1.05e-6	7.02e-6	6.06e-5	5.57e-7	3.26e-6	2.19e-5	1.62e-4
	$k=10$, $dt=0.01$				$k=30$, $dt=0.01$			
N_x	50%	25%	12.5%	6.25%	50%	25%	12.5%	6.25%
16	2.96e-4	1.74e-3	8.45e-3	2.10e-2	2.90e-4	1.66e-3	7.07e-3	2.20e-2
32	3.79e-5	2.32e-4	1.48e-3	8.82e-3	3.99e-5	2.55e-4	1.55e-3	7.53e-3
64	3.84e-6	2.07e-5	1.40e-4	1.05e-3	5.02e-6	3.23e-5	2.18e-4	1.45e-3
128	2.59e-7	1.07e-6	7.12e-6	6.15e-5	5.57e-7	3.26e-6	2.20e-5	1.63e-4
	$k=10$, $dt=0.001$				$k=30$, $dt=0.001$			
N_x	50%	25%	12.5%	6.25%	50%	25%	12.5%	6.25%
16	3.32e-4	1.94e-3	9.22e-3	2.23e-2	2.94e-4	1.69e-3	7.15e-3	2.22e-2
32	4.23e-5	2.58e-4	1.64e-3	9.60e-3	4.05e-5	2.58e-4	1.57e-3	7.61e-3
64	4.29e-6	2.33e-5	1.57e-4	1.18e-3	5.09e-6	3.28e-5	2.21e-4	1.47e-3
128	2.93e-7	1.22e-6	8.12e-6	6.97e-5	5.65e-7	3.31e-6	2.23e-5	1.65e-4

Table 2 Sensitivity of the methods to parameters k and dt

27

Lower Dimensional Interpolation in Overlapping Composite Mesh Difference Methods

Serge Goossens[1] & Xiao-Chuan Cai[2]

setcountertheorem0

Introduction

We propose a modified Composite Mesh Difference Method (CMDM) in which a lower dimensional interpolation can be used along the interface of the nonmatching grids. The advantage of this approach is that fewer interpolation points are needed while the same order of global accuracy is preserved. This is important especially for distributed memory implementations since smaller amounts of data need to be communicated among the overlapping subdomains. A CMDM on two subdomains has been described by Starius [Sta77], while Cai et al. [CMS98] have studied the case of many subdomains. We focus on the 2D Poisson's equation. Our results show that it is possible to obtain global second order accuracy on nonmatching grids with a local coupling using only

1 Dept. of Computer Science, K. U. Leuven, Celestijnenlaan 200A, B3001 Leuven, Belgium (Serge.Goossens@cs.kuleuven.ac.be, www.cs.kuleuven.ac.be/~serge/).
Part of this work was carried out during the visit of the first author to the University of Colorado at Boulder. The financial support for this visit by the FWO-Vlaanderen is gratefully acknowledged. This research is financed by a specialisation scholarship of the Flemish Institute for the Promotion of Scientific and Technological Research in Industry (IWT).
2 Dept. of Computer Science, University of Colorado at Boulder, Boulder, CO 80309-0430, USA (cai@cs.colorado.edu, www.cs.colorado.edu/~cai/).
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

4 points at the interface in the discretisation equations. In contrast, the overlapping nonmatching mortar method [CDS99] requires a global mortar projection involving all the mesh points on the interface.

Composite Mesh Difference Method

We briefly describe our CMDM for solving the second order elliptic partial differential equation $\mathcal{L}u = f$ in Ω with a Dirichlet boundary condition $u = g$ on $\partial\Omega$. Given a domain Ω consisting of p nonoverlapping subdomains Ω_i such that $\bar{\Omega} = \cup_{i=1}^{p} \bar{\Omega}_i$, we independently construct a grid of size h_i on each extended subdomain Ω_i' of Ω_i. Due to the extension of the subdomains these grids overlap. We denote by $\Gamma_i = \partial\Omega_i' \cap \partial\Omega$ the intersection of the boundaries $\partial\Omega_i'$ and $\partial\Omega$.

Assumption 1 *The truncation error* $\alpha_i(x) = (\mathcal{L}_{h_i} u_{h_i} - \mathcal{L}u)(x)$ *is of order* p_i*:*

$$||\alpha_i||_\infty \leq C_{\alpha_i} h_i^{p_i} ||u||_{p_i+2,\infty,\Omega_i'}, \tag{1}$$

where C_{α_i} *is a constant independent of the mesh size* h_i *and* $||u||_{k,\infty,\Omega_i'}$ *denotes the Sobolev norm for the space* $W_\infty^k(\Omega_i')$.

Assumption 2 *The interpolation operator* $\mathcal{I}_i$ *only uses values from* $\cup_{j\neq i}\bar{\Omega}_j$ *and no values from* Ω_i'. *The interpolation error* $\beta_i(x) = (u - \mathcal{I}_i u)(x)$ *is of order* q_i*:*

$$||\beta_i||_\infty \leq C_{\beta_i} h_i^{q_i} ||u||_{q_i,\infty}. \tag{2}$$

Let $\mathcal{I}_i$ be the interpolation matrix and $\sigma_i = ||\mathcal{I}_i||_\infty$ the norm of $\mathcal{I}_i$. Note that for piecewise linear or bilinear interpolation $\sigma_i = 1$ and for quadratic or cubic interpolation, $\sigma_i = 5/4$ in 1D and $\sigma_i = 25/16$ in 2D. We denote by $\sigma = \max_i \sigma_i$ the largest of all the interpolation constants.

The *global discretisation* $u_h = (u_{h_1}, u_{h_2}, \cdots, u_{h_p})$ on the composite grid is obtained by coupling the local discretisations through the requirement that the solution matches the interpolation of the discrete solutions from adjacent grids. The system of equations consists of p subproblems, each having the following form:

$$\begin{cases} \mathcal{L}_{h_i} u_{h_i} = f_{h_i} & \text{in } \Omega_i', \\ u_{h_i} = g_{h_i} & \text{on } \Gamma_i, \\ u_{h_i} = z_{h_i} = \mathcal{I}_i u_h & \text{on } \partial\Omega_i' \setminus \Gamma_i. \end{cases} \tag{3}$$

Assumption 3 *The local finite difference discretisations (3) are stable in the maximum norm and satisfy a strong discrete maximum principle, i.e. a constant* K_i *independent of* h_i *and a constant* $0 \leq \rho_i < 1$ *exist so that*

$$||u_{h_i}||_{\infty,\bar{\Omega}_i} \leq K_i ||f_{h_i}||_{\infty,\Omega_i'} + \max\{||g_{h_i}||_{\infty,\Gamma_i}, \rho_i ||z_{h_i}||_{\infty,\partial\Omega_i'\setminus\Gamma_i}\}. \tag{4}$$

The *contraction factor* $0 \leq \rho_i < 1$ measures the error reduction. This can be seen by considering the homogeneous problem, i.e. $f = 0$ and $g = 0$.

Assumption 4 *The product of the interpolation constant and the contraction factor is less than 1*

$$\tau = \max_i (\rho_i \sigma) < 1. \tag{5}$$

Under the above assumptions Cai et al. [CMS98] proved the maximum norm stability of the global discretisation and showed that the following error bound holds.

Theorem 2 *The error in the discrete solution satisfies*

$$\sum_{i=1}^{p} \|e_{h_i}\|_\infty \leq \left(1 + \frac{\sigma}{1-\tau}\right) \left(\sum_{i=1}^{p} K_i \|\alpha_i\|_\infty + \sum_{i=1}^{p} \|\beta_i\|_\infty\right). \tag{6}$$

Schwarz Alternating Method

Since the CMDM described above is a contraction mapping, the resulting system of equations can be solved by repeatedly solving the p subproblems (3) in parallel, where z_{h_i} is computed from the previous iteration. The convergence rate of this iteration is bounded by the contraction factor τ of the mapping.

Theorem 3 *The iterates* $\{u_h^{(n)}\}$ *converge to the exact discrete solution* u_h *and*

$$d(u_h^{(n)}, u_h) \leq \tau^n d(u_h^{(0)}, u_h). \tag{7}$$

Here $d(w_h, v_h) = \max_i \{\|w_{h_i} - v_{h_i}\|_{\infty, \bar{\Omega}_i'}\}$.

This is a parallel variant of the Schwarz alternating method. Rather than using the additive Schwarz method as a solver, it is better to use it as a preconditioner in a Krylov subspace method. Convergence proofs of the Schwarz algorithm based on a maximum principle can be found in [CMS98, Mil65].

We remark that when solving (3) using iterative methods, the total arithmetic cost is determined by the subdomain mesh sizes, while the communication cost, for implementations on distributed memory machines, depends on the interface interpolation operators. The rest of this paper is devoted to the interpolation issue.

Interface Interpolation Schemes

In this section, we discuss several interface interpolation schemes. We restrict ourselves to the two subdomains case, i.e. $\Omega = \Omega_1' \cup \Omega_2'$ where $\Omega_1' = [0, l_1] \times [0, 1]$ and $\Omega_2' = [l_2, 2] \times [0, 1]$. We assume $l_1 > 1$ and $l_2 < 1$. The usual five-point finite difference method is used in the two subdomains.

The standard stencil with bilinear interpolation

Since both the standard five-point stencil and bilinear interpolation are second order, the error bound (6) shows that the resulting CMDM is also second order. However this scheme does not satisfy the *consistent interpolation* condition defined by Goossens et al. [GCR98]. We describe the inconsistency present in this approach. Let $(0, 0)$ be the local coordinates of a mesh point in Ω_1' that is next to the interface. To define the finite difference stencil, the value at $(h, 0)$ has to be obtained from Ω_2 through interpolation. More precisely, the stencil S at $(0, 0)$ has the form

$$S = -4u(0,0) + u(0,-h) + u(0,h) + u(-h,0) + v, \tag{8}$$

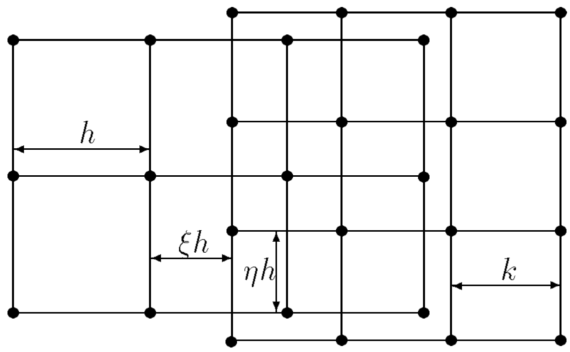


Figure 1 The scaled local coordinates (ξ,η) used in the interpolation.

where v is computed using a bilinear interpolation for $u(h,0)$, i.e. $v = (1-\xi)(1-\eta)u(h-\xi k, -\eta k)+(1-\xi)\eta u(h-\xi k,(1-\eta)k)+\xi(1-\eta)u(h+(1-\xi)k,-\eta k)+\xi\eta u(h+(1-\xi)k,(1-\eta)k)$. Figure 1 shows the scaled local coordinates (ξ,η) used in the interpolation on the overlapping meshes. Expanding (8) we find the inconsistent discretisation for the nodes where the bilinear interpolation is used:

$$\frac{S}{h^2} - (u_{xx}+u_{yy}) = \frac{\gamma_k^2}{2}\left(\xi(1-\xi)u_{xx} + \eta(1-\eta)u_{yy}\right) + \mathcal{O}(h), \tag{9}$$

where $\gamma_k = k/h$ is the ratio of the mesh sizes. Note that the scheme is consistent only if ξ and η are either 0 or 1, which implies that the two meshes match each other on the interface. We show numerical results illustrating the effect of this inconsistent discretisation.

Theorem 4 *The standard stencil with bilinear interpolation results in a second order scheme on every extended subdomain Ω_i':*

$$|e_p| \le C_{\Phi_a} max\left\{C_I, max_{p_j\in\mathcal{J}}\left(\xi(1-\xi)u_{xx}+\eta(1-\eta)u_{yy}\right)\frac{\gamma_k^2}{2\tilde{E}_2}\right\}h^2 + C_D h^2 + \mathcal{O}(h^3), \tag{10}$$

$\forall p \in \Omega_i'$, where $C_I = (M_{xxxx}+M_{yyyy})/(48E_1)$ is a constant depending on the derivatives u_{xxxx} and u_{yyyy}. The constant C_{Φ_a} denotes the maximum of the nonnegative function Φ over the grid points where a Dirichlet boundary condition is used for the subdomain Ω_i' and $C_D \ge 0$ is a constant associated with the accuracy of the Dirichlet boundary conditions on the internal boundaries $\partial\Omega_i' \setminus \Gamma_i$. The set $\mathcal{J}$ contains all the points where the inconsistent discretisation (9) is used. The constants E_1 and $\tilde{E}_2$ are lower bounds for $\mathcal{L}_{h_i}\Phi$.

Second Order Scheme on a Modified Stencil

In order to obtain a consistent discretisation, we construct a second order accurate difference formula on the modified stencil depicted in Fig. 2. The idea is to slightly modify the meshes along their boundaries so that the interpolation is needed only

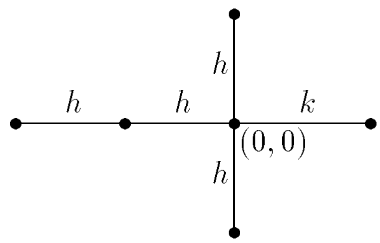


Figure 2 Modified stencil for the second order scheme.

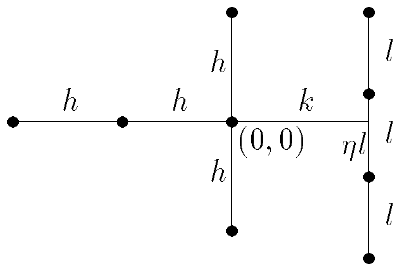


Figure 3 Stencil for the second order scheme with 1D cubic interpolation.

in one of the directions in contrast to the standard bilinear method, which needs information in both x and y directions. The mesh width $k \geq h$ is selected so that a grid line in the other mesh is matched and no interpolation along the x-axis is required. The point $u(-2h,0)$ is needed to obtain a second order discretisation in the stencil

$$S = c_{0,0}u(0,0)+c_{0,-1}u(0,-h)+c_{0,1}u(0,h)+c_{-1,0}u(-h,0)+c_{1,0}u(k,0)+c_{-2,0}u(-2h,0) \tag{11}$$

with $c_{0,0} = -1 - 3/\gamma_k$, $c_{0,-1} = c_{0,1} = 1$, $c_{-1,0} = 2(2-\gamma_k)/(\gamma_k+1)$, $c_{1,0} = 6/(\gamma_k(\gamma_k+1)(\gamma_k+2))$ and $c_{-2,0} = (\gamma_k - 1)/(\gamma_k+2)$. The truncation error is given as

$$\frac{S}{h^2} - (u_{xx} + u_{yy}) = \frac{h^2}{12}\left((3\gamma_k - 2)u_{xxxx} + u_{yyyy}\right) + \mathcal{O}(h^3). \tag{12}$$

Note that for $1 \leq \gamma_k \leq 2$ this stencil satisfies a discrete maximum principle.

Modified Stencil with Linear Interpolation

We now consider the effect of replacing $u(k,0)$ by an interpolation formula along the y-axis. First of all we show that a consistent approximation exists when linear interpolation is used, i.e. $v = (1-\eta)u(k,-\eta l) + \eta u(k,(1-\eta)l)$ is used instead of $u(k,0)$. We use the standard stencil and seek the coefficients in

$$S = c_{0,0}u(0,0) + c_{0,-1}u(0,-h) + c_{0,1}u(0,h) + c_{-1,0}u(-h,0) + c_v v \tag{13}$$

so that a consistent approximation to $(u_{xx} + u_{yy})$ results. Setting $c_{0,-1} = c_{0,1} = 1 - \eta(1-\eta)\gamma_l^2/(\gamma_k(\gamma_k+1))$, $c_{-1,0} = 2/(\gamma_k+1)$, $c_v = 2/(\gamma_k(\gamma_k+1))$ and $c_{0,0} = 2\eta(1-\eta)\gamma_l^2 - (\gamma_k+1)^2/(\gamma_k(\gamma_k+1))$, where $\gamma_l = l/h$, results in

$$\frac{S}{h^2} - (u_{xx} + u_{yy}) = \mathcal{O}(h). \tag{14}$$

If $\gamma_l \leq 2\sqrt{\gamma_k(\gamma_k+1)}$ this formula satisfies a discrete maximum principle. For η equal to 0 or 1 a first order discretisation on a modified five-point stencil is obtained. Note that the difference formula has to be modified to account for the low order interpolation.

Modified Stencil with Cubic Interpolation

The interpolation along the x-axis can be avoided by using a modified stencil. We show that cubic interpolation along the y-axis results in a second order scheme. This is equivalent to constructing a second order accurate difference formula on the modified stencil depicted in Fig. 3.

Theorem 5 *It is not possible to obtain a second order accurate discretisation of* $(u_{xx} + u_{yy})$ *at the center point* $(0,0)$ *if only 3 or fewer points are used along the line* $x = k$ *and none of these points has a zero* y*-coordinate.*

Theorem 6 *The only second order accurate discretisation of* $(u_{xx}+u_{yy})$ *at the center point* $(0,0)$ *using only 4 points along the line* $x = k$ *with none of these points having a zero* y*-coordinate, is the second order scheme (11) on the modified stencil with cubic interpolation along the line* $x = k$ *for the point* $(k,0)$*.*

We seek the coefficients in the stencil

$$\begin{aligned} S \;=\; & c_{0,0}u(0,0) + c_{0,-1}u(0,-h) + c_{0,1}u(0,h) + c_{-1,0}u(-h,0) + c_{-2,0}u(-2h,0) + \\ & c_{1,-1}u(k,-(1+\eta)l) + c_{1,0}u(k,-\eta l) + c_{1,1}u(k,(1-\eta)l) + c_{1,2}u(k,(2-\eta)l). \end{aligned} \tag{15}$$

Setting $c_{0,0} = -1 - 3/\gamma_k$, $c_{0,-1} = c_{0,1} = 1$, $c_{-1,0} = 2(2-\gamma_k)/(\gamma_k+1)$, $c_{-2,0} = (\gamma_k - 1)/(\gamma_k + 2)$, $c_{1,-1} = -\eta(1-\eta)(2-\eta)/N$, $c_{1,0} = 3(1-\eta)(2-\eta)(\eta+1)/N$, $c_{1,1} = 3(2-\eta)\eta(\eta+1)/N$ and $c_{1,2} = -\eta(1-\eta)(\eta+1)/N$, where $N = \gamma_k(\gamma_k+1)(\gamma_k+2)$ results in a second order accurate discretisation

$$\frac{S}{h^2} - (u_{xx} + u_{yy}) = \mathcal{O}(h^2). \tag{16}$$

The truncation error for this scheme is

$$\alpha = \left[\frac{(3\gamma_k - 2)}{12}u_{xxxx} + \left(\frac{1}{12} - \frac{(2-\eta)(1-\eta)\eta(\eta+1)\gamma_l^4}{4\gamma_k(\gamma_k+1)(\gamma_k+2)}\right)u_{yyyy}\right]h^2 + \mathcal{O}(h^3). \tag{17}$$

It is clear that the coefficients $c_{1,-1}$, $c_{1,0}$, $c_{1,1}$ and $c_{1,2}$ are equal to the product of the coefficient $c_{1,0}$ in (11) and the cubic Lagrange interpolation polynomials.

Numerical results

Our testcase is taken from [CMS98] and concerns the solution of $-\nabla^2 u = f$ on $\Omega = \Omega_1 \cup \Omega_2$, where $\Omega_1 = [0,1] \times [0,1]$ and $\Omega_2 = [1,2] \times [0,1]$. The r.h.s. f and the Dirichlet boundary conditions g are chosen so that the exact solution is $u(x,y) = (\sin(\pi x) + \sin(\pi x/2))\sin(\pi y)$. The overlapping subdomains are $\Omega_1' = [0,1.4] \times [0,1]$ with $h_1 = 0.2 \times 2^{-l}$ and $\Omega_2' = [0.75,2] \times [0,1]$ with $h_2 = 0.25 \times 2^{-l}$.

In Table 1 we list the L_∞-norm of the error for the standard stencil with bilinear interpolation. For a second order scheme, the ratio between two successive error norms should be 4 when the mesh size is halved. These results show ratios alternating between 4.11 and 3.89 and between 4.18 and 3.83! This is due to the presence of the inconsistency as shown by (9) which results in a dependency of the error on

Table 1 Results for the standard stencil with bilinear interpolation.

l	ξ	$\|e_{\Omega'_1}\|_\infty$	ratio	$\|e_{\Omega'_2}\|_\infty$	ratio
0	0.6	5.53e-2		6.34e-2	
1	0.2	1.40e-2	3.94	1.31e-2	4.82
2	0.4	3.62e-2	3.88	3.47e-3	3.80
3	0.8	8.79e-4	4.11	8.26e-4	4.19
4	0.6	2.26e-4	3.89	2.16e-4	3.82
5	0.2	5.50e-5	4.11	5.17e-5	4.18
6	0.4	1.41e-5	3.89	1.35e-5	3.83

Table 2 Results for standard stencil with bicubic interpolation (columns 2–5) and for modified stencil with 1D cubic interpolation (columns 6–9).

l	$\|e_{\Omega'_1}\|_\infty$	ratio	$\|e_{\Omega'_2}\|_\infty$	ratio	$\|e_{\Omega'_1}\|_\infty$	ratio	$\|e_{\Omega'_2}\|_\infty$	ratio
0	5.20e-2		4.14e-2		4.98e-2		8.41e-2	
1	1.34e-2	3.885	1.19e-2	3.4810	1.34e-2	3.7168	1.51e-2	5.5828
2	3.33e-3	4.0164	3.01e-3	3.9605	3.31e-3	4.0523	3.67e-3	4.0997
3	8.34e-4	3.9986	7.50e-4	4.0072	8.34e-4	3.9687	8.22e-4	4.4717
4	2.08e-4	4.0010	1.88e-4	4.0012	2.08e-4	4.0017	1.95e-4	4.2099
5	5.21e-5	4.0003	4.69e-5	4.0004	5.21e-5	3.9999	4.78e-5	4.0868
6	1.30e-5	4.0000	1.17e-5	3.9998	1.30e-5	4.0004	1.18e-5	4.0393

$\xi(1-\xi)$, i.e. the relative position of the interface in the other mesh. For this testcase the dominant term in the error bound (10) is $e \approx (\xi(1-\xi)c_1 + c_2)\, h^2$, where c_1 and c_2 are constants independent of ξ and h. With this expression, we can estimate the ratio γ_e between two successive error norms. When the mesh is refined by halving the mesh size, i.e. $h_{i+1} = h_i/2$, we have

$$\gamma_e = \frac{\|e_{\Omega'_{h_i}}\|_\infty}{\|e_{\Omega'_{h_{i+1}}}\|_\infty} = \frac{c_1\,(\xi_i(1-\xi_i)+\gamma_c)\,h_i^2}{c_1\,(\xi_{i+1}(1-\xi_{i+1})+\gamma_c)\,h_{i+1}^2} = \frac{\xi_i(1-\xi_i)+\gamma_c}{\xi_{i+1}(1-\xi_{i+1})+\gamma_c}\,4 \tag{18}$$

where $\gamma_c = c_2/c_1$. For Ω'_1 we find $\gamma_c = 2.7491$ which results in ratios γ_e of 4.11 and 3.89 while for Ω'_2 we have $\gamma_c = 1.6178$ resulting in ratios γ_e of 4.18 and 3.83. The accuracy of the scheme depends on the relative position of the interface in the other mesh. Apart from this phenomenon the scheme is second order, since fitting a power of the mesh size $\|e_{\Omega'_1}\|_\infty \approx \kappa h^\lambda$ yields $\lambda = 1.9929$. The second order accuracy can also be seen when the mesh is refined twice, i.e. the mesh size is divided by 4, in this case the factor $\xi(1-\xi)$ does not change and we get ratios of 15.9973 (for $\|e_{\Omega'_1}\|_\infty$) and of 15.9966 (for $\|e_{\Omega'_2}\|_\infty$) between two successive error norms, which is very close to the theoretical value of 16.

In Table 2 we show the results for the standard stencil with bicubic interpolation, which is a very expensive interpolation since it requires 16 points, and for the modified stencil with 1D cubic interpolation, which only requires 4 points. It is clear that both these schemes are second order and that our method is as accurate as the classical approach with expensive bicubic interpolation.

In order to see the effect of the overlap, we fix the mesh sizes to be $h_1^{-1} = 320$ and $h_2^{-1} = 256$ and vary the overlap according to $\delta_1 = 2 \times 2^\chi h_1$ and $\delta_2 = 2^\chi h_2$ for

Table 3 Effect of overlap on the convergence rate of the Schwarz method and on the accuracy for the standard stencil with bilinear interpolation (columns 2–4) and for the modified stencil with 1D cubic interpolation (columns 5–7).

χ	n	$\|e_{\Omega_1}\|_\infty$	$\|e_{\Omega_2}\|_\infty$	n	$\|e_{\Omega_1}\|_\infty$	$\|e_{\Omega_2}\|_\infty$
0	846	4.00e-4	4.00e-4	369	1.31e-5	6.49e-6
1	436	1.89e-4	1.90e-4	300	1.31e-5	6.49e-6
2	224	7.99e-5	7.93e-5	161	1.31e-5	6.48e-6
3	116	3.12e-5	3.19e-5	96	1.31e-5	6.48e-6
4	60	9.97e-6	1.04e-5	53	1.30e-5	6.51e-6
5	32	1.24e-5	7.47e-6	30	1.30e-5	6.73e-6
6	17	1.41e-5	6.82e-6	16	1.30e-5	7.52e-6
7	10	1.33e-5	6.53e-6	9	1.31e-5	9.19e-6

the values of χ listed in Table 3. We list the number of additive Schwarz iterations required to satisfy the convergence criterion of $\|r_n\|_2 \leq 10^{-10}\|r_0\|_2$ and the L_∞-norm of the error in the nonoverlapping subdomains Ω_1 and Ω_2. From the results it is clear that the global accuracy of the first method increases as the overlap increases, thus necessitating substantial overlap, while our method reaches the attainable accuracy even with minimal overlap. As expected the number of additive Schwarz iterations decreases, as the overlap increases.

Concluding remarks

We have studied several interface interpolation schemes for overlapping nonmatching grids finite difference methods. A scheme based on the combination of 1D cubic interpolation and a six-point stencil is proposed and produces a consistent and globally second order method. We have also shown numerically that minimum overlap is required to achieve the accuracy, and that larger overlap reduces the number of Schwarz iterations but does not change the accuracy. The method is cheaper than the interpolation method required by the theory of [CH90] and the mortar based method proposed in [CDS99].

REFERENCES

[CDS99] Cai X.-C., Dryja M. and Sarkis M. (1999) Overlapping nonmatching grid mortar element methods for elliptic problems. *SIAM J. Numer. Anal.* 36: 581–606.

[CH90] Chesshire G. and Henshaw W. D. (1990) Composite overlapping meshes for the solution of partial differential equations. *J. Comp. Phys.* 90: 1–64.

[CMS98] Cai X.-C., Mathew T. P. and Sarkis M. (1998) Maximum norm analysis of overlapping nonmatching grid discretizations of elliptic equations. Technical Report CU-CS-866-98, Department of Computer Science, University of Colorado at Boulder.

[GCR98] Goossens S., Cai X.-C. and Roose D. (1998) Overlapping nonmatching grids method: some preliminary studies. In Mandel J., Farhat C. and Cai X.-C. (eds) *Domain Decomposition Methods 10*, Contemporary Mathematics, pages

254–261. AMS.
[Mil65] Miller K. (1965) Numerical analogs to the Schwarz alternating procedure. *Numer. Math.* 7: 91–103.
[Sta77] Starius G. (1977) Composite mesh difference methods for elliptic boundary value problems. *Numer. Math.* 28: 243–258.

28

Domain Decomposition for Indefinite Weakly Singular Integral Equations

N. HEUER [1]

INTRODUCTION

We propose and analyze a preconditioner based on domain decomposition for the p-version of the boundary element method in three dimensions. We consider indefinite weakly singular integral equations on surfaces and use quadrilateral elements for the boundary discretization. The GMRES method is used as iterative solver for the linear systems. For a non-overlapping method, we prove that the numbers of GMRES-iterations (which are required to solve the systems up to a given accuracy) grow only polylogarithmically with the polynomial degree.

Solving first-kind weakly singular integral equations by the Galerkin method does not require continuous ansatz functions. Therefore, without performing any Schur complement technique, simple non-overlapping methods are sufficient to define efficient preconditioners. In [Heua] this is shown in the symmetric, positive definite case and here we extend those results to the more general situation of indefinite operators. For this extension we follow the ideas of Cai, Widlund [CW92] (who deal with finite element systems) and Stephan, Tran [ST98] (who deal with boundary integral equations on curves). In this paper, we consider problems in the three-dimensional space. That means we have to deal with integral operators on surfaces.

As model problem we consider the Dirichlet problem for the scalar Helmholtz

1 Institut für Wissenschaftliche Datenverarbeitung, Universität Bremen, Postfach 33 04 40, 28334 Bremen, Germany email: heuer@iwd.uni-bremen.de
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

equation exterior to a screen. The problem reads as follows. *For given* $g \in H^{1/2}(\Gamma)$ *find* $u \in H^1_{\mathrm{loc}}(\Omega)$ *with* $\Omega := I\!R^3 \setminus \bar{\Gamma}$ *satisfying*

$$(\Delta + k^2)u = 0 \quad in \quad \Omega, \qquad u = g \quad on \quad \Gamma, \tag{1}$$

$$\left.\begin{array}{ll} \frac{\partial u}{\partial r} - iku = o(r^{-1}) & for \quad k \neq 0 \quad or \\ u = O(r^{-1}) & for \quad k = 0 \end{array}\right\} \quad as \quad r := |x| \to \infty. \tag{2}$$

Here, Γ is a plane surface with polygonal boundary such that it can be decomposed into quadrilaterals. We note that our method also works for polyhedral surfaces and triangular meshes. We consider wave numbers k with $\Im(k) \geq 0$ and $|k|$ small. The latter condition is to ensure the existence of a Gårding inequality with moderate constants for the boundary integral operator V_k, i.e. the costant γ_1 in (4) is not much smaller than γ_2, cf. [Ste87]. If this were not the case our method would require a very small coarse mesh size H, cf. Lemma 3 below.

The Dirichlet screen problem (1), (2) appears in the scattering theory of acoustic fields u by obstacles. From [Ste87] we know that for $\Im(k) \geq 0$ this problem is uniquely solvable. Further, $u \in H^1_{\mathrm{loc}}(\Omega)$ is the solution of the Dirichlet screen problem if and only if the jump $[\partial u/\partial n]$ of the normal derivative of u across Γ is the solution to the weakly singular integral equation

$$V_k \left[\frac{\partial u}{\partial n}\right](x) := \frac{1}{4\pi} \int_\Gamma \left[\frac{\partial u}{\partial n}\right](y) \frac{e^{ik|x-y|}}{|x-y|}\, dS_y = g(x) \quad (x \in \Gamma). \tag{3}$$

Moreover, by [Cos88, Ste87], V_k is continuous,

$$V_k : \tilde{H}^{-1/2+s}(\Gamma) \to H^{1/2+s}(\Gamma) \quad (-1/2 \leq s \leq 1/2)$$

and strongly elliptic, i.e., it satisfies a Gårding inequality: *There exist constants* $\gamma_1, \gamma_2 > 0$ *such that for all* $v \in \tilde{H}^{-1/2}(\Gamma)$

$$\Re\langle V_k v, v\rangle \geq \gamma_1 \|v\|^2_{\tilde{H}^{-1/2}(\Gamma)} - \gamma_2 \|v\|^2_{\tilde{H}^{-1}(\Gamma)}. \tag{4}$$

Here, $\langle\cdot,\cdot\rangle$ denotes the $L^2(\Gamma)$-inner product. The space $\tilde{H}^{-s}(\Gamma)$ for $s > 0$ is the dual space of $H^s(\Gamma)$ which is the usual Sobolev space for integer s and is obtained by interpolation for non-integer s, see, e.g., [LM72]. The operator $V := V_0$, which is the main part of V_k, is symmetric, positive definite and induces an equivalent norm in $\tilde{H}^{-1/2}(\Gamma)$. The remainder $V_k - V$ is of lower order than V_k and is, as well as its adjoint $V_k' - V$, bounded:

$$|\langle (V_k - V)v_1, v_2\rangle| \leq c \min\{\|v_1\|_{\tilde{H}^{-1}(\Gamma)} \|v_2\|_{\tilde{H}^{-1/2}(\Gamma)},\ \|v_1\|_{\tilde{H}^{-1/2}(\Gamma)} \|v_2\|_{\tilde{H}^{-1}(\Gamma)}\} \tag{5}$$

for any $v_1, v_2 \in \tilde{H}^{-1/2}(\Gamma)$, see [Ste87].

We define boundary element spaces $S^0_p(\Gamma_h)$ of piecewise polynomials of degree p on quasi-uniform quadrilateral meshes $\bar{\Gamma}_h = \sum_{j=1}^J \bar{\Gamma}_j$ of size h:

$$S^0_p(\Gamma_h) := \{f \in L^2(\Gamma);\ f|_{\Gamma_j} \in P_p(\Gamma_j), j = 1, \dots, J\} \subset \tilde{H}^{-1/2}(\Gamma)$$

Here, $P_p(\Gamma_j)$ denotes the set of polynomials on Γ_j which are at most of degree p in each of the variables. The Galerkin scheme for the numerical solution of the integral equation (3) then is: *Find* $u_N \in S_p^0(\Gamma_h)$ *such that*

$$\langle V_k u_N, w\rangle = \langle g, w\rangle \quad \textit{for any} \quad w \in S_p^0(\Gamma_h). \tag{6}$$

Here, the subscript N is used to denote the dimension of $S_p^0(\Gamma_h)$. By the strong ellipticity of V_k this method converges quasi-optimally.

Since V_k is a first kind integral operator of order minus one the spectral condition number of the stiffness matrix A in (6) behaves (when using standard basis functions) like $O(h^{-1}p^2)$, cf., e.g., [HW77, Heu96]. Especially for the p-version we are considering, a preconditioner is necessary for the efficient solution of (6). The stiffness matrix A is in general non-Hermitian and indefinite and we use the GMRES method [SS86] as iterative solver. For real linear systems the rate of convergence of the GMRES method depends on the two numbers

$$\Lambda_0 = \inf_{v\in \mathbb{R}^N} \frac{a(v, Av)}{a(v,v)}, \quad \text{and} \quad \Lambda_1 = \sup_{v\in\mathbb{R}^N} \frac{\|Av\|_a}{\|v\|_a}.$$

The quantity Λ_0 is the minimum eigenvalue of the symmetric part of A and Λ_1 is the matrix norm of A which is induced by the norm $\|\cdot\|_a$. In the complex case we have to consider, instead of Λ_0 and Λ_1 as defined above, the expressions

$$\Lambda_0 = \inf_{v\in \mathbb{C}^N} \Re\frac{a(v, Av)}{a(v,v)}, \quad \text{and} \quad \Lambda_1 = \sup_{v\in\mathbb{C}^N} \frac{\|Av\|_a}{\|v\|_a}.$$

For $a(\cdot,\cdot)$ we use the inner product which is given by V, i.e., $a(\cdot,\cdot) = \langle V\cdot,\cdot\rangle \simeq \|\cdot\|^2_{\tilde H^{-1/2}(\Gamma)}$. From [EES83] we cite the following result: *If* $\Lambda_0 > 0$ *then the GMRES method converges and for the residual corresponding to the* j*th iterate* $r_j = g - Au_j$ *there holds*

$$\|r_j\|_a \le \Big(1 - \frac{\Lambda_0^2}{\Lambda_1^2}\Big)^{j/2} \|r_0\|_a.$$

Here, g is the right hand side vector of the linear system (6). In order to reduce the number of iterations of the GMRES method we need to find a preconditioner for the matrix A which bounds the ratio Λ_0/Λ_1 from below.

Throughout the paper c denotes a positive generic constant.

NON-OVERLAPPING ADDITIVE SCHWARZ METHOD

In order to define our preconditioner we need to decompose the boundary element space. This is done by introducing a coarse mesh $\bar\Gamma_H = \cup_{j=1}^n \bar G_j$ ($H \ge h$) which must be compatible with Γ_h. We note that the case $\Gamma_H = \Gamma_h$ is included. Now we decompose

$$S_p^0(\Gamma_h) = X_0 + X_1 + \cdots + X_n \tag{7}$$

where X_0 is the space of piecewise constant functions on the coarse mesh Γ_H and

$$X_j := \{v|_{G_j};\ v \in S_p^0(\Gamma_h),\ \langle v, 1\rangle_{L^2(G_j)} = 0\}, \quad j = 1,\dots,n.$$

We define the projection operators $\mathcal{P}_j : S^0_p(\Gamma_h) \to X_j$, $j = 0,\dots,n$, and $P_j : S^0_p(\Gamma_h) \to X_j$, $j = 1,\dots,n$, such that, for given $v \in S^0_p(\Gamma_h)$,

$$\langle V_k \mathcal{P}_j v, \phi\rangle = \langle V_k v, \phi\rangle \quad \forall \phi \in X_j \quad \text{and} \quad \langle V P_j v, \phi\rangle = \langle V_k v, \phi\rangle \quad \forall \phi \in X_j.$$

Two types of the additive Schwarz operator are now defined by $\mathcal{P} = \mathcal{P}_0 + \mathcal{P}_1 + \cdots + \mathcal{P}_n$ and $P = \mathcal{P}_0 + P_1 + \cdots + P_n$. The additive Schwarz method consists in solving, instead of the system (6), one of the equations

$$\mathcal{P} u_N = f_1 \quad \text{or} \quad P u_N = f_2.$$

The right hand sides $f_1 = (\mathcal{P}_0 + \mathcal{P}_1 + \cdots + \mathcal{P}_n) u_N$ and $f_2 = (\mathcal{P}_0 + P_1 + \cdots + P_n) u_N$ can be computed without knowing the solution u_N of the original Galerkin system (6) since only $\langle V_k u_N, \phi\rangle$ for $\phi \in S^0_p(\Gamma_h)$ needs to be known. But $V_k u_N = g$ when testing against functions of the ansatz space $S^0_p(\Gamma_h)$. The operators $\mathcal{P}_j$ $(j > 0)$ are computed by locally inverting the indefinite operator V_k whereas for P_j we only need to solve positive definite problems for V.

The efficiency of the additive Schwarz operators $\mathcal{P}$ and P is proved by the following theorem.

Theorem 1 *There exist positive constants c_1, c_2, c_3, h_0 such that, if $0 < H \le h_0$, there holds for $p > 0$ and for any $u \in S^0_p(\Gamma_h)$*

$$\Re\langle V\mathcal{P}u, u\rangle \ge c_1 \Big(c_2 \big(1 + \log(\frac{H}{h}(p+1))\big)^{-2} - (1 + \log 1/H)^{1/2} H^{1/2} \Big) \langle V u, u\rangle$$

and

$$\|\mathcal{P}u\|_{\tilde{H}^{-1/2}(\Gamma)} \le c_3 \|u\|_{\tilde{H}^{-1/2}(\Gamma)}.$$

Analogous estimates hold for the operator P.

Technical details for the proof of this theorem are given in the next section.

TECHNICAL DETAILS

We collect all the technical details which are needed to investigate the decomposition (7) within $\tilde{H}^{-1/2}(\Gamma)$ and we show how these results apply to indefinite operators. For the generalization of efficient decompositions within $\tilde{H}^{-1/2}(\Gamma)$ (which is the symmetric, positive definite case) to efficient preconditioners for indefinite operators on surfaces we extend the theory for integral operators on curves of Stephan and Tran [ST98], see also [CW92]. The sketch of a proof of Theorem 1 is given at the end of this section.

The first lemma, which we cite from [Heua], proves that the non-overlapping decomposition (7) is almost optimal within the space $\tilde{H}^{-1/2}(\Gamma)$. This is equivalent to the boundedness of the spectral condition number of the additive Schwarz operator (defined by this decomposition) when considering symmetric, positive definite weakly singular integral operators.

Lemma 1 [Heua] *Let $X_0, \ldots, X_n$ be the subspaces of the non-overlapping decomposition (7). There exist constants $c_1, c_2 > 0$ such that for any $u \in S_p^0(\Gamma_h)$ with arbitrary representation $u = \sum_{j=0}^n v_j$ (with $v_j \in X_j$) there exist $u_j \in X_j$ with $u = \sum_{j=0}^n u_j$ and*

$$c_1\big(1 + \log(\frac{H}{h}(p+1))\big)^{-2} \sum_{j=0}^n \|u_j\|^2_{\tilde{H}^{-1/2}(\Gamma)} \le \|u\|^2_{\tilde{H}^{-1/2}(\Gamma)} \le c_2 \sum_{j=0}^n \|v_j\|^2_{\tilde{H}^{-1/2}(\Gamma)}.$$

The constants $c_1, c_2 > 0$ do not depend on the mesh size h or the polynomial degree p.

The following lemma is used to estimate the coarse grid component of the additive Schwarz operator.

Lemma 2 *There exists $h_0 > 0$ such that, for all $H \in (0, h_0)$, the projection operator $\mathcal{P}_0 : S_p^0(\Gamma_h) \to X_0$ is well defined. Moreover, there exists a constant $c > 0$ such that, for $H \in (0, h_0)$, there holds for any $u \in S_p^0(\Gamma_h)$*

$$\|\mathcal{P}_0 u\|_{\tilde{H}^{-1/2}(\Gamma)} \le c\|u\|_{\tilde{H}^{-1/2}(\Gamma)}$$

and

$$\|u - \mathcal{P}_0 u\|_{\tilde{H}^{-1}(\Gamma)} \le c(1 + \log 1/H)^{1/2} H^{1/2} \|u\|_{\tilde{H}^{-1/2}(\Gamma)}.$$

Proof. The projection operator $\mathcal{P}_0$ is the standard h-version Galerkin projection operator (restricted onto $S_p^0(\Gamma_h)$) with respect to V_k. Since V_k is strongly elliptic the boundedness of $\mathcal{P}_0$ and the convergence

$$\|u - \mathcal{P}_0 u\|_{\tilde{H}^{-1}(\Gamma)} \le c(\epsilon) H^{1/2-\epsilon} \|u\|_{\tilde{H}^{-1/2}(\Gamma)} \quad \text{for any} \quad u \in S_p^0(\Gamma_h)$$

($\epsilon > 0$) follows by standard arguments, see, e.g., [Wen90, Ste87]. Here, $c(\epsilon)$ is a constant which is not bounded when $\epsilon \to 0$, in general. This is due to the reduced regularity of the solution to the equation $V_k u = f$ near the boundary of a screen, even for smooth right hand side f. This regularity is used in the Aubin-Nitsche trick which give error estimates in lower norms than the energy norm $\tilde{H}^{-1/2}(\Gamma)$, cf. [CS88, Wen90]. When one uses precise norm estimates of the singularities appearing in the images of V_k^{-1}, then the Aubin-Nitsche trick yields $c(\epsilon) = O(\epsilon^{-1/2})$ (see [Heub]) and taking $\epsilon := -1/\log H$ this proves the second assertion of the lemma.

The next lemma will be used to bound the lower order norm in the Gårding inequality (4) for functions with small support. This gives the positive definiteness of the real part of V_k on local subspaces of $S_p^0(\Gamma_h)$.

Lemma 3 *Let u be a function from any one of the local subspaces $X_1, X_2, \ldots, X_n$ in (7), that means* $\operatorname{diam}(\operatorname{supp}(u)) = O(H)$. *Then there exist $c, C, h_0 > 0$ such that, for any $H \in (0, h_0)$, there holds*

$$\|u\|_{\tilde{H}^{-1}(\Gamma)} \le C H^{1/2} \|u\|_{\tilde{H}^{-1/2}(\Gamma)} \tag{8}$$

and

$$\Re\langle V_k u, u\rangle \ge c\|u\|^2_{\tilde{H}^{-1/2}(\Gamma)}.$$

Proof. The subspaces X_j are constructed such that $\int_\Gamma u\,dS = 0$ for $u \in X_j$ $(j > 0)$. For functions with integral mean zero the $\tilde{H}^{-s}$-norms $(0 \le s \le 1)$ are scalable with respect to the diameter of the supports, see Lemma 3 in [Heua], and this proves assertion (8). The second assertion follows from (8) by Gårding's inequality (4) choosing H small enough.

The following result is used to prove Theorem 1. It is based on the previous lemmas and extends the almost-optimality result of Lemma 1 to the indefinite situation of the operators $\mathcal{P}$ and P.

Lemma 4 *There exist $c_1, c_2, h_0 > 0$ such that, for $H \in (0, h_0)$ and $p > 0$, there holds for any $u \in S_p^0(\Gamma_h)$*

$$c_1\big(1 + \log(\frac{H}{h}(p+1))\big)^{-2}\|u\|^2_{\tilde{H}^{-1/2}(\Gamma)} \le \sum_{j=0}^{N} \|\mathcal{P}_j u\|^2_{\tilde{H}^{-1/2}(\Gamma)} \le c_2\|u\|^2_{\tilde{H}^{-1/2}(\Gamma)}.$$

The same estimates hold if one replaces $\mathcal{P}_j$ by P_j for $j > 0$.

Proof. This lemma is the counter part to Lemma 2.5 in [ST98] and for a proof we refer to that reference. The main ingredients are provided by the previous Lemmas 1, 2, 3 and by Gårding's inequality (4).

Proof of Theorem 1. Using the technical details provided in this section Theorem 1 follows analogously as in [Heub] where hypersingular operators are considered. In fact, one only needs to substitute $\tilde{H}^{1/2}(\Gamma)$ by $\tilde{H}^{-1/2}(\Gamma)$ and $L^2(\Gamma)$ by $\tilde{H}^{-1}(\Gamma)$ in the proof of the main theorems in [Heub]. All the needed details are then given by Lemmas 1, 2, 4 and by (5) and (8) above. See also the proof of Theorem 2.1 in [ST98] where the details for integral operators on curves are given.

NUMERICAL RESULTS

We consider the Dirichlet screen problem (1), (2) for the homogeneous Helmholtz equation with $\Gamma = (0,1)^2 \times \{0\} \subset \mathbb{R}^3$. As right hand side g in (1) we simply take the function 1. This choice does not influence the behavior of the resulting stiffness matrix. For the definition of the ansatz space $S_p^0(\Gamma_h)$ we use uniform meshes Γ_h consisting of squares and take piecewise polynomials of degree p that need not be continuous across element interfaces. As basis functions we use tensor products of Legendre polynomials which are scaled to have unit L^2-norm.

For the non-overlapping additive Schwarz method we simply take the coarse mesh $\Gamma_H = \Gamma_h$, i.e., $H/h = 1$. Figure 1 shows the parameters Λ_0 and Λ_1 for different wave numbers and full local solvers ("I", additive Schwarz operator $\mathcal{P}$) and positive definite local solvers ("P", additive Schwarz operator P). In these cases Theorem 1 proves an asymptotic bound like $(1 + \log p)^{-2}$ for the minimum eigenvalue Λ_0 of the real Hermitian part of the preconditioned stiffness matrix. The norm Λ_1 of the preconditioned stiffness matrix is proven to be bounded. Both theoretical estimates seem to be confirmed by the numerical results. The requirement of Theorem 1 that the mesh size h must be small enough does depend on the actual polynomial degree. In particular, to ensure convergence of the GMRES method, we must have $\Lambda_0 > 0$, i.e., h must be small enough such that

$$c_2(1+\log p)^{-2} - (1+\log 1/h)^{1/2}h^{1/2} > 0.$$

Here, c_2 is an unknown constant and, nevertheless, there is no problem to satisfy this condition since the dependence on the polynomial degree is only logarithmic. Our numerical experiments do not reflect the theoretically needed dependence of h on p in this situation. The condition on the mesh size is of asymptotic nature and may become a restriction for higher polynomial degrees and for larger wave numbers.

Table 1 presents the numbers of iterations of the GMRES method for different values of k which are needed to reduce the initial residual by a factor of 10^{-6}. The results demonstrate that the dependence of the iteration numbers of the GMRES method on the wave number is rather weak. Indeed, the numbers increase only slightly with k (and p) and are almost constant in h.

REFERENCES

[Cos88] Costabel M. (1988) Boundary integral operators on Lipschitz domains: Elementary results. *SIAM J. Math. Anal.* 19: 613–626.

[CS88] Costabel M. and Stephan E. P. (1988) Duality estimates for the numerical approximation of boundary integral equations. *Numer. Math.* 54: 339–353.

[CW92] Cai X.-C. and Widlund O. B. (1992) Domain decomposition algorithms for indefinite elliptic problems. *SIAM J. Sci. Stat. Comput.* 13: 243–258.

[EES83] Eisenstat S. C., Elman H. C., and Schultz M. H. (1983) Variational iterative methods for non-symmetric systems of linear equations. *SIAM J. Numer. Anal.* 20: 345–357.

[Heua] Heuer N.Additive Schwarz method for pseudo-differential operators of order minus one in $\mathbb{R}^3$. Submitted for publication.

[Heub] Heuer N.Additive Schwarz methods for indefinite hypersingular integral equations in $\mathbb{R}^3$ – the p-version. *Appl. Anal.* To appear.

[Heu96] Heuer N. (1996) Efficient algorithms for the p-version of the boundary element method. *J. Integral Equations Appl.* 8(3): 337–361.

[HW77] Hsiao G. C. and Wendland W. L. (1977) A finite element method for some integral equations of the first kind. *J. Math. Anal. Appl.* 58: 449–481.

[LM72] Lions J. L. and Magenes E. (1972) *Non-Homogeneous Boundary Value Problems and Applications I.* Springer–Verlag, New York.

[SS86] Saad Y. and Schultz M. H. (1986) GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.* 7(3): 856–869.

[ST98] Stephan E. P. and Tran T. (1998) Domain decomposition algorithms for indefinite hypersingular integral equations – the h- and p-versions. *SIAM J. Sci. Comput.* 19(4): 1139–1153.

[Ste87] Stephan E. P. (1987) Boundary integral equations for screen problems in $\mathbb{R}^3$. *Integral Equations Operator Theory* 10: 257–263.

[Wen90] Wendland W. L. (1990) Boundary element methods for elliptic problems. In Schatz A. H., Thomée V., and Wendland W. L. (eds) *Mathematical Theory of Finite and Boundary Element Methods*, volume 150 of *DMV Seminar*, pages 219–276. Birkhäuser, Basel.

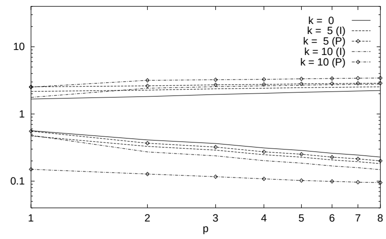


Figure 1 Values of Λ_0 and Λ_1 (p-version, $h^{-1} = 2$) with full local solvers (I) and positive definite local solvers (P).

Table 1 Numbers of GMRES iterations (different wave numbers, full local solvers and positive definite local solvers (I, P)).

		k:	0	5		10	
N	h^{-1}	p	(P)	(I)	(P)	(I)	(P)
4	2	0	2	2	2	2	2
36	2	2	7	7	7	7	7
100	2	4	9	9	9	9	11
196	2	6	9	10	9	10	11
324	2	8	10	10	10	10	11
36	2	2	7	7	7	7	7
144	4	2	13	14	14	15	15
324	6	2	13	15	15	17	17
576	8	2	13	14	14	17	17
900	10	2	13	14	14	16	16

29

Numerical Simulation of Wave Propagation Phenomena in Vocal Tract and Domain Decomposition Method

T. KAKO[1], & T. KANO[2]

INTRODUCTION

We develop a finite element approximation method for the Helmholtz equation in some unbounded region Ω_0:

$$-\Delta u - \omega^2 u = 0 \text{ in } \Omega_0 \tag{1}$$

and apply the method to the wave propagation phenomena in a vocal tract. For various time frequencies ω, we solve the Helmholtz equation in an unbounded acoustic region Ω_0 a part of which forms a vocal tract. Then we investigate the resonance phenomena of the sound wave propagation which is important to characterize vowels and consonants. We use a two dimensional model as well as one dimensional model called Webster's horn equation.

In this research, we confine our study to the case in which the outer region consists of a semi-infinite cylinder, and we assume that the original three dimensional acoustic region is planar which enables us to reduce the problem into a two dimensinal one after the Fourier mode decomposition with respect to the perpendicular direction to the corresponding plane of the planar region.

1 Department of Computer Science & Information Mathematics, The University of Electro-Communications, Chofu, Tokyo 182-8585, JAPAN. email: kako@im.uec.ac.jp
2 Address same as above.
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

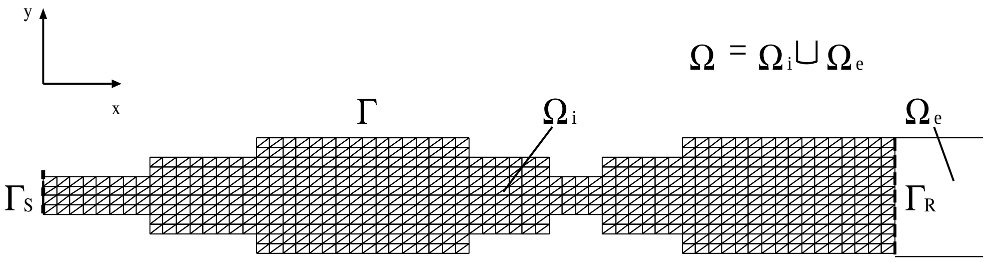


Figure 1 Acoustic region in 2D case

Assuming the radiation condition at infinity, we derive a radiation boundary condition on the artificial boundary which forms a part of the boundary of the bounded sub-region of the original unbounded acoustic region. In the radiation condition, we use the Dirichlet to Neumann map for the Helmholtz problem in the unbounded outer cylindrical region.

We introduce a one dimesional Webster's model which corresponds to the 2-D plane region using the width of the vocal tract. There is a good coincidence between the results of 1-D and 2-D models as far as the magnitude of the modulus of a wave number is not large, i.e. less than some constant. The results justify the use of the 1-D Webster's model as the approximation of the 2-D model in case that the incident wave from the vocal cord contains only low time frequency modes.

MATHEMATICAL MODELS

We consider the case where the outer unbounded region consists of a semi-infinite cylinder $\Omega_{0,e}$ with a bounded 2-D cross section $S_{0,e}$:

$$\Omega_{0,e} = \{\mathbf{x} = (x, y, z) \mid x_0 < x < +\infty,\ (y, z) \in S_{0,e}\} \tag{2}$$

and we assume that the original three dimensional acoustic region is planer:

$$\Omega_0 = (\Omega_i \cup \Omega_e) \times (0, z_0) \ \text{ with } \ \Omega_{0,e} = \Omega_e \times (0, z_0) \ \text{ and } \ S_{0,e} = (0, y_0) \times (0, z_0), \tag{3}$$

which enables us to reduce the problem into a two dimensinal one after the Fourier mode decomposition with respect to the perpendicular z-direction to the corresponding xy plane of the planar region.

Now, consider the following wave equation:

$$\begin{aligned} (\frac{\partial^2}{\partial t^2} - \Delta)u(t, \mathbf{x}) &= f(t, \mathbf{x}) \ \text{ in } R_t \times \Omega_{\mathbf{x}}, \ \ \Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}, & (4) \\ (\alpha\frac{\partial}{\partial n} + \beta)u(t, \mathbf{x}) &= g(t, \mathbf{x}) \ \text{ on } R_t \times \partial\Omega_{\mathbf{x}}, & (5) \end{aligned}$$

with some initial condition for u and $\partial u/\partial t$, and consider a stationary time harmonic solution: $u(t, \mathbf{x}) = e^{i\omega t}u(\mathbf{x})$ for inhomogeneous data: $f(t, \mathbf{x}) = e^{i\omega t}f(\mathbf{x})$ and $g(t, \mathbf{x}) =$

$e^{i\omega t}g(\mathbf{x})$. Then u satisfies the Helmholtz equation:

$$(-\Delta-\omega^2)u(\mathbf{x}) = f(\mathbf{x}) \ \text{ in } \Omega, \tag{6}$$

$$(\alpha\frac{\partial}{\partial n}+\beta)u(\mathbf{x}) = g(\mathbf{x}) \ \text{ on } \partial\Omega, \tag{7}$$

with some radiation condition at infinity ($|\mathbf{x}|\to+\infty$).

DOMAIN DECOMPOSITION INTO INNER AND OUTER REGIONS

We consider the case of outgoing wave and derive a radiation condition on the artificial boundary $S_e=\{x_0\}\times(0,y_0)$ which forms a part of the boundary of the bounded sub-region Ω_i of the original unbounded acoustic region $\Omega=\Omega_i\cup\Omega_e$ (Figure 1).

The radiation condition is then given by the Dirichlet to Neumann map Λ_y for the Helmholtz problem in the unbounded outer cylindrical region Ω_e (see [Mas87] in the case of obstacle scattering):

$$\frac{\partial}{\partial x}u(x_0,y)=\Lambda_y u(x_0,y), \tag{8}$$

where

$$\Lambda_y u=\Lambda u\equiv\sum_{n=0}^{\infty}\gamma_n C_n(u)\cos(\frac{n\pi}{y_0}y)$$

with

$$C_n(u)=\begin{cases}\frac{1}{y_0}\displaystyle\int_0^{y_0}u(x_0,y)dy & (n=0),\\ \frac{2}{y_0}\displaystyle\int_0^{y_0}u(x_0,y)\cos(\frac{n\pi}{y_0}y)dy & (n\geq 1),\end{cases}$$

$$\gamma_n=\begin{cases} i\zeta_n, & \zeta_n=\{\omega^2-(\frac{n\pi}{y_0})^2\}^{1/2}, \quad 0<\frac{n\pi}{y_0}<\omega,\\ -\eta_n, & \eta_n=\{(\frac{n\pi}{y_0})^2-\omega^2\}^{1/2}, \quad \omega\leq\frac{n\pi}{y_0}.\end{cases}$$

Then, we have the following Helmholtz equation in the inner region Ω_i:

$$(-\omega^2-\Delta)u=0 \ \text{ in } \Omega_i, \tag{9}$$

$$\frac{\partial u}{\partial n}=0 \ \text{ on } \Gamma, \quad \frac{\partial u}{\partial n}=g_S \ \text{ on } \Gamma_S, \quad \frac{\partial u}{\partial n}=\Lambda u \ \text{ on } \Gamma_R\equiv S_e.$$

WEBSTER'S HORN EQUATION

Now, we introduce a one dimensional approximation of the original problem which is called Webster's horn equation:

$$-\frac{\partial v}{\partial t} = \frac{A(x)}{\rho}\frac{\partial u}{\partial x}, \tag{10}$$

$$-\frac{\partial u}{\partial t} = \frac{\rho c^2}{A(x)}\frac{\partial v}{\partial x}, \tag{11}$$

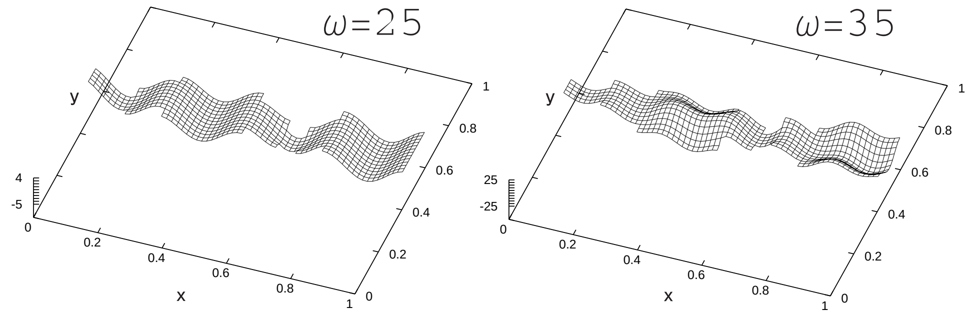


Figure 2 Acoustic field for some wave numbers

which can be reduced to the form of the second order equation:

$$\frac{\partial^2 u}{\partial t^2} - \frac{1}{A(x)} c^2 \frac{\partial}{\partial x}(A(x)\frac{\partial u}{\partial x}) = 0, \tag{12}$$

where $A(x)$ corresponds to the width of the original 2-D or 3-D region of the vocal tract. The corresponding time harmonic equation becomes to be

$$-A(x)\frac{d}{dx}\{\frac{1}{A(x)}\frac{d}{dx}u(x)\} - \omega^2 u(x) = 0. \tag{13}$$

NUMERICAL RESULTS AND ERROR ANALYSIS

As for the discretization, we adopt the finite element method for respective 1-D and 2-D models using a finite element subspace of piecewise linear continuous functions, and compare the numerical results for various wave numbers ω. There is a good coincidence between the results of 1-D and 2-D cases for the frequency responce as far as the magnitude of the modulus of a wave number (frequency) is not large (see Figure 2,3). The results justify the use of the 1-D Webster's equation model as the approximation of the 2-D model in case that the incident wave from the vocal cord contains only low frequency modes

Next, we give the convergence results for the finite element calculation. First, we write the two domensional problem in the following weak form:

Find $u \in \mathcal{V} \equiv H^1(\Omega)$:

$$a(u,v) + b(u,v) = (f,v)(= a_0(g,v)) \qquad \forall v \in \mathcal{V} \tag{14}$$

where

$$a(u,v) = a_0(u,v) + b_2^{\infty}(u,v), \;\; b(u,v) = b_1(u,v) + b_2^0(u,v)$$

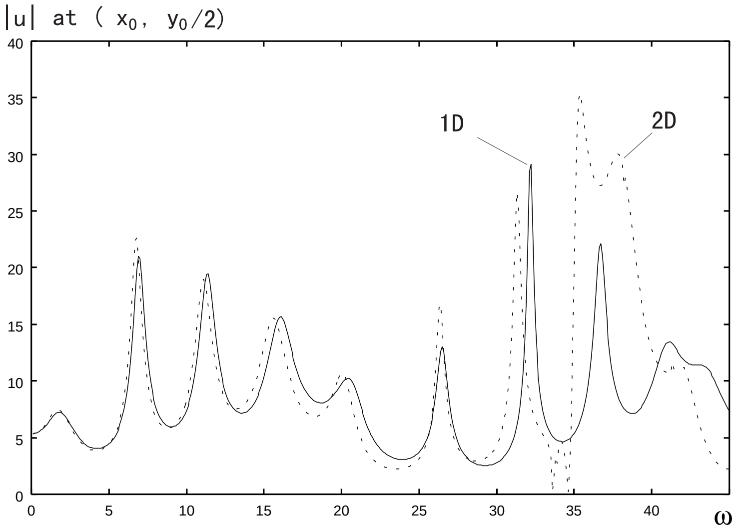


Figure 3 Comparison between the results of 1-D and 2-D cases for the frequency responce

with

$$\begin{aligned}
a_0(u,v) &= \int_\Omega \nabla u \cdot \overline{\nabla v} + u\overline{v}dxdy, \\
b_1(u,v) &= -(\omega^2+1)\int_\Omega u\overline{v}dxdy, \\
b_2(u,v) &= (\Lambda u(x_R,\cdot), v(x_R,\cdot)) = b_2^0(u,v) + b_2^\infty(u,v), \\
b_2^0(u,v) &= \sum_{0\le\frac{n\pi}{L}<\omega} \gamma_n C_n(u)C_n(v), \\
b_2^\infty(u,v) &= \sum_{\omega\le\frac{n\pi}{L}} \gamma_n C_n(u)C_n(v).
\end{aligned}$$

Based on this formulation we introduce a finite element method:
Find $u_h \in \mathcal{V}_h \subset \mathcal{V}$:

$$a(u_h,v_h) + b(u_h,v_h) = (f,v_h)(= a_0(g,v_h)) \qquad \forall v_h \in \mathcal{V}_h. \tag{15}$$

Concerning the above finite element methods, we have the followings:

1. $b_2^\infty(u,v)$ is a nonnegative sesquilinear form, and hence $a(u,v)$ is an inner product in $\mathcal{V}$;
2. $b_2^0(u,v)$ and hence b is a compact form with respect to $a(u,v)$ in $\mathcal{V}$;
3. Applying the results of Mikhlin [Mik64], we get the convergence of the approximation (see [Kak81],[LK98]);

4. After the finite element approximation, we can approximate $b_2^\infty(u, v)$ by the one with a finite summation up to the N-th Fourier mode, and the approximation converges to the original finite element solution in the finite dimensional space $\mathcal{V}_h$ as N tends to infinity.

CONCLUDING REMARKS

We summerise our results as follows:

1. Mathematical models for the speech production problem are formulated. In the outer infinite region, the exact solution is given and the radiation boundary condition is proposed by using the Dirichlet to Neumann mapping on the artificial boundary between inner and outer regions;
2. A comparison between 1-D and 2-D calculations is given. When the time frequency of a sound source is low enough, the coincidence of the two calculations is good and 1-D Webster's horn equation model can be used to simulate the resonance phenomena of the vocal tract. A discrepancy becomes large when the frequency is larger than the value above which the single mode approximation is violated due to transversal higher modes (see Figure 2);
3. Numerical analysis of the 2-D problem is given. Using the property of the Dirichlet to Neumann mapping, we can prove the convergence of the finite element approximation, and then the approximation of the Dirichlet to Neumann mapping by its finite Fourier mode summation can be justified in the finite dimensional approximation subspace.

REFERENCES

[Kak81] Kako T. (1981) Approximation of the scattering state by means of the radiation boundary condition. *Math. Meth. in the Appl. Sci.* 3: 506–515.

[LK98] Liu X.-J. and Kako T. (1998) Higher order radiation boundary condition and finite element method for scattering problem. *Adv. Math. Sci. Appl.* 8: 801–819.

[Mas87] Masmoudi M. (1987) Numerical solution for exterior problems, numer. math. *Numer. Math.* 51: 87–101.

[Mik64] Mikhlin S. (1964) *Variational Methods in Mathematical Physics*. Pergamon, Oxford.

30

Optimal Shape of Pine for Sound Absorption in Water

H. KAWARADA [1] & H. SUITO [2]

Introduction

This paper deals with the phenomena of sound propagation near pine timber in water. A plane incident wave with frequency ω and incident angle (α, β) comes from the right hand side through medium 1(water) toward an absorbing plate made of medium 2 (pine); In fact, pine has been widely used as an absorbing material in the past.

In this case, some part of the incident wave is transmitted into the pine and the rest is reflected into the water at the surface plane.

In the first part of this paper, the problem of boundary matching is studied; in other words, the sound field of the total system is obtained using treatment of domain decomposition type. The computed domain is a rectangular parallelepiped, which is divided into two parts faced by several types of shapes.

The sound fields composed of incident and reflected waves in the water and the transmitted wave in the pine are computed separately. The total solution is obtained by coupling the two solutions computed in each domain to satisfy transmission conditions defined on an interfacial boundary. Here, let us construct the cost function of a Dirichlet-Neumann type on the boundary, which is directly minimized by means of Fuzzy Optimization Method (FOM)[KOPS98, KS97].

In order to evaluate the shape of the interfacial boundary in pursuit of a better absorption of the sound field into the pine, we proceed to the second part; i.e., shape optimization. We denote the flat interfacial boundary by Γ_0 and expect that

1 Faculty of engineering, Chiba University, Inage-ku, Chiba, 263-8522, JAPAN. email: kawarada@applmath.tg.chiba-u.ac.jp
2 Address same as above. email: suito@applmath.tg.chiba-u.ac.jp
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

an appropriate Γ deformed from Γ_0 brings about the increase of the amplitude of the transmitted wave into pine. The minimization problem, the cost of which is represented by the amplitude defined on the fixed flat surface in the pine, is proposed with Γ regarded as a control variable. Here, the deformation of Γ is represented by a superposition of the product of two scaling functions for a wavelet. The admissible set due to this choice seems to be considerably restricted, but the merit of this use is an appearance of wedge-shaped patterns involved in them, which have been used as the shape of absorption materials for a long time. Finally, the minimization problem constructed is also directly solved by means of FOM.

The reason why we have not used elegant techniques developed in the framework of domain decomposition is due to the requirement to test the validity and the utility of direct minimization by use of FOM. At the present stage we have confirmed the numerical solvability of the problems, but cannot discuss the suitability of this strategy without comparison with various othrt methods approached from a range of viewpoints.

The algorithm to solve the minimization problems numerically is divided into two parts;

1. 2D/3D Helmholtz solver for deformed region;
   Here we solve the Helmholtz equation in the deformed region by the use of finite difference approximation. The mesh generation method of an elliptic type is effectively adopted and also the GPBi-CG method[Zha97] is used to solve a discretized Helmholtz equation defined in the computed domain, where the preconditioning technique causes a surprising effect to save an iteration number.
2. Fuzzy Optimization Method;
   This is used to solve the minimization problem. This method is briefly reviewed in this paper.

The contents of this paper are as follows; 1. Introduction, 2. Applications, 3. Deformation of the interfacial boundary, 4. Shape optimization, 5. Numerical Solution of Helmholtz Eq., 6. Hybrid Algorithm by FOM and GAs, 7. Numerical Results, 8. Conclusions.

Applications

Optimal shape design problems for;

1. Anti-noise wall for high-speed trains (Shinkansen) running through densely populated areas,
2. Sonar domes at the bows or bottom of a ship,
3. Input surface of a solar battery,
4. Noise absorption for engine rooms,
5. Etc,

could be treated in almost the same framework as that introduced in this paper.

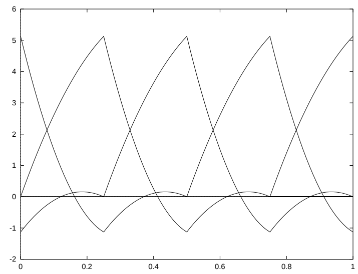


Figure 1 Scaling functions

Deformation of the Interfacial Boundary

Scaling functions for wavelet

Wedge-shaped patterns often arise in wavelets. A scaling function for wavelets is introduced as follows;
Let

$$\eta_0(x) = \begin{cases} 1 & x \in [0,1], \\ 0 & else, \end{cases} \tag{1}$$

and

$$\begin{aligned} f_0(x) &= (-0.585x^2 + 1.867x)\eta_0(x), & (2)\\ f_1(x) &= (1.170x^2 - 2.734x + 1.282)\eta_0(x), & (3)\\ f_2(x) &= (0.585x^2 + 0.867x - 0.282)\eta_0(x). & (4) \end{aligned}$$

Then

$$\phi(x) = f_0(x) + f_1(x-1) + f_2(x-2) \tag{5}$$

satisfies

1. supp $\phi = [0,3]$
2. $\int_R \phi(x)dx = 1$
3. An auto-correlation function $\phi^{[2]}(x) = \int_R \phi(x+y)\phi(y)dy$ satisfies $\phi^{[2]}(m) = \delta_{m,0}$ $(m \in Z)$.

Define

$$\phi_{L,m}(x) = \sqrt{N_L} \cdot \phi(N_L x - m) \quad (m \in Z) \tag{6}$$

where $N_L = 2^L$.

Then $\{\phi_{L,m}\}$ constitutes an orthonormal set, i.e.,

$$\int_R \phi_{L,m}\phi_{L,m'}dx = \delta_{m,m'}. \tag{7}$$

Figure 1 shows an example of $y = \phi_{L,m}(x)$, where $L = 2$ and $m = 1, 2, \cdots, 6$.

Admissible Set for the Deformed Surface

Represent the deformed surface by means of a superposition of $\phi_{L,m}(y)\phi_{L,m'}(z)$, i.e.,

$$\Gamma = \sum_{m,m'} \gamma_{mm'} \cdot \phi_{L,m}(y)\phi_{L,m'}(z). \tag{8}$$

The admissible set for the deformation of the interfacial boundary is defined by

$$\mathcal{A}_1 = \{\gamma_{mm'} \in R| \, |\gamma_{mm'}| \leq K \; (m, m' = 1, 2, 3, \cdots, M_1)\}. \tag{9}$$

Shape optimization

Configuration

- $\Gamma_{top}, \Gamma_{side}$ and Γ_{bottom} are rigid boundaries, i.e., the density of these walls is infinite. Hence, sound waves are entirely reflected at these boundaries.
- Γ_{in} is a vibrating plate which makes sound wave.
- Ω_1 is occupied by water.
- Ω_2 is made of pine, the role of which is to absorb the sound wave coming through Ω_1.
- Γ is the boundary between Ω_1 and Ω_2. We will try to optimize its shape to absorb the sound wave into Ω_2 as completely as possible.
- Ω_3 is a so-called Fictitious Domain, i.e., an artificial domain to approximate the boundary condition at infinity. In this domain, the Helmholtz eq. with a complex wave number is assumed, which is derived from Navier-Stokes eq. including a viscosity term. A sound wave transmitted from Ω_2 is almost completely damped in this domain and is not reflected into Ω_2.
- Γ_{absorb}, on which the amplitude of the absorbed sound wave in Ω_2 is computed.
- Γ_{out}, on which the sound wave does not exist because of the damping effect in the domain Ω_3.
- $\Omega = \Omega_1 \cup \Gamma \cup \Omega_2 \cup \Omega_3 = (0, l_x) \times (0, l_y) \times (0, l_z)$
- $u^{(i)}(x, y, z)$ $(i = 1, 2, 3)$: Complex sound pressure.
- k_i $(i = 1, 2, 3)$: Wave number.
- λ_i $(i = 1, 2, 3)$: Wave length.
- c_i $(i = 1, 2, 3)$: Sound speed.
- ω : Angular velocity of the incident wave.

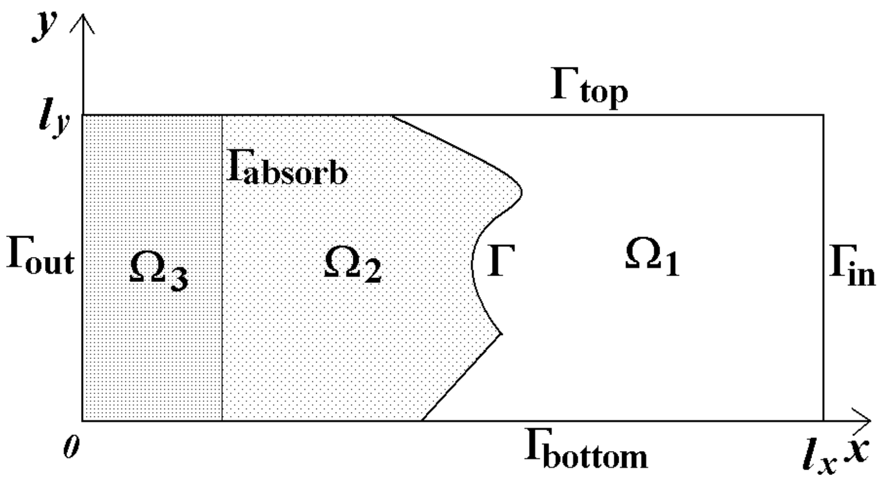


Figure 2 Projection of computed domain into $x - y$ plane

- ρ_i $(i = 1, 2, 3)$: Density of medium.
- $\mathbf{n}$: Outward normal vector on the boundaries.
- Γ : Interfacial boundary between Ω_1 and Ω_2.
- α, β : An incident angle of plane wave,

where $i = 1$ means water, $i = 2$ means pine and $i = 3$ means the fictitious domain.

Physical data

Table 30 shows the physical data of media used in our problem.

Minimization with the Constraint

Define the state equation;

$$\begin{cases} (\triangle + k_i^2)\, u^{(i)}(\Gamma, a) = 0 & \text{in } \Omega_i, \quad (i = 1, 2, 3), \\ u^{(1)}(\Gamma, a) = u^{(2)}(\Gamma, a) = a & \text{on } \Gamma, \\ \dfrac{\partial u^{(i)}(\Gamma, a)}{\partial n} = 0 & \text{on } \Gamma_{top} \cup \Gamma_{bottom}, \\ & \quad \cup \Gamma_{side} (i = 1, 2, 3), \\ u^{(1)}(\Gamma, a) = e^{ik_1 \cos\beta l_x} e^{ik_1 \sin\beta \cos\alpha y} e^{ik_1 \sin\beta \cos\alpha z} & \text{on } \Gamma_{in}, \\ u^{(2)}(\Gamma, a) = 0 & \text{on } \Gamma_{out}, \end{cases} \tag{10}$$

and the cost function;

Table 1 Physical data of media

	Water	Pine
$\rho\ (kg/m^3)$	998.9	450.0
$k\ (rad/m)$	53.51	25.5
$\lambda\ (m)$	0.12	0.25
$c\ (m/sec)$	1467.8	3599.4
$\omega\ (rad/sec)$	7.85×10^4	7.85×10^4

$$J_c(\Gamma, a) = -\int_{\Gamma_{absorb}} \left| u^{(2)}(\Gamma, a) \right|^2 d\Gamma$$
$$+\frac{1}{\varepsilon}\int_{\Gamma} \left| \frac{1}{\rho_1}\frac{\partial u^{(1)}(\Gamma, a)}{\partial n} - \frac{1}{\rho_2}\frac{\partial u^{(2)}(\Gamma, a)}{\partial n} \right|^2 d\Gamma, \quad (\varepsilon > 0). \tag{11}$$

The Dirichlet datum a is defined on Γ by

$$a = \sum_{mm'} a_{mm'} \cos(\frac{\pi m}{l_y} y) \cos(\frac{\pi m'}{l_z} z). \tag{12}$$

A The admissible set for a is represented by

$$\mathcal{A}_2 = \{a_{mm'} \in \mathbf{C} \ \ (m, m' = 0, 1, 2, \cdots, M_2) | |a_{mm'}| \leq L\}. \tag{13}$$

The minimization problem is;

$$[Pr_c]: \quad \text{Minimize} J_c(\Gamma, a) \ \text{ for } \ (\Gamma, a) \in \mathcal{A} = \mathcal{A}_1 \times \mathcal{A}_2.$$

Numerical Solution of Helmholtz Eq. in the Deformed Region

In order to compute the sound field in a domain with a complicated boundary, the following coordinate transformation is used;

1. Generate a mesh system in the deformed domain, which is the transformation from a physical domain to a computational one;

$$\begin{aligned} x &= x(\xi, \eta, \zeta), \\ y &= y(\xi, \eta, \zeta), \\ z &= z(\xi, \eta, \zeta). \end{aligned} \tag{14}$$

2. Transform differential operators by use of (14).
3. Transform Helmholtz eq. by use of (14).

Discretized Helmholtz Eq. in the Computational Domain

Helmholtz equation in the physical domain;

$$(\partial_{xx} + \partial_{yy} + \partial_{zz})u + k^2 u = 0, \tag{15}$$

is written in the computational domain as follows;

$$\begin{aligned}
&\Big(\frac{a_{11}}{J}\partial_\xi(\frac{a_{11}}{J}\partial_\xi + \frac{a_{12}}{J}\partial_\eta + \frac{a_{13}}{J}\partial_\zeta) + \frac{a_{12}}{J}\partial_\eta(\frac{a_{11}}{J}\partial_\xi + \frac{a_{12}}{J}\partial_\eta + \frac{a_{13}}{J}\partial_\zeta)\\
&+\frac{a_{13}}{J}\partial_\zeta(\frac{a_{11}}{J}\partial_\xi + \frac{a_{12}}{J}\partial_\eta + \frac{a_{13}}{J}\partial_\zeta) + \frac{a_{21}}{J}\partial_\xi(\frac{a_{21}}{J}\partial_\xi + \frac{a_{22}}{J}\partial_\eta + \frac{a_{23}}{J}\partial_\zeta)\\
&+\frac{a_{22}}{J}\partial_\eta(\frac{a_{21}}{J}\partial_\xi + \frac{a_{22}}{J}\partial_\eta + \frac{a_{23}}{J}\partial_\zeta) + \frac{a_{23}}{J}\partial_\zeta(\frac{a_{21}}{J}\partial_\xi + \frac{a_{22}}{J}\partial_\eta + \frac{a_{23}}{J}\partial_\zeta)\\
&+\frac{a_{31}}{J}\partial_\xi(\frac{a_{31}}{J}\partial_\xi + \frac{a_{32}}{J}\partial_\eta + \frac{a_{33}}{J}\partial_\zeta) + \frac{a_{32}}{J}\partial_\eta(\frac{a_{31}}{J}\partial_\xi + \frac{a_{32}}{J}\partial_\eta + \frac{a_{33}}{J}\partial_\zeta)\\
&+\frac{a_{33}}{J}\partial_\zeta(\frac{a_{31}}{J}\partial_\xi + \frac{a_{32}}{J}\partial_\eta + \frac{a_{33}}{J}\partial_\zeta)\Big)\, u + k^2 u = 0
\end{aligned} \tag{16}$$

where

$$a_{11} = y_\eta z_\zeta - y_\zeta z_\eta,\ a_{12} = y_\zeta z_\xi - y_\xi z_\zeta,\ a_{13} = y_\xi z_\eta - y_\eta z_\xi, \tag{17}$$

$$a_{21} = z_\eta x_\zeta - x_\eta z_\zeta,\ a_{22} = x_\xi z_\zeta - x_\zeta z_\xi,\ a_{23} = x_\eta z_\xi - x_\xi z_\eta, \tag{18}$$

$$a_{31} = x_\eta y_\zeta - y_\eta x_\zeta,\ a_{32} = y_\xi x_\zeta - x_\xi y_\zeta,\ a_{33} = x_\xi y_\eta - y_\xi x_\eta, \tag{19}$$

$$J = \begin{vmatrix} x_\xi & x_\eta & x_\zeta \\ y_\xi & y_\eta & y_\zeta \\ z_\xi & z_\eta & z_\zeta \end{vmatrix}. \tag{20}$$

The boundary conditions defined on the boundaries of the physical domain are each transformed in a similar way. (16) is discretized by second order central differences.

Mesh Generation of an Elliptic Type

In order to generate a smooth mesh structure along the deformed boundary, an elliptic type of mesh generation method is adopted. In this method, the following three Laplace equations are solved in the computational domain;

$$\xi_{xx} + \xi_{yy} + \xi_{zz} = 0, \tag{21}$$

$$\eta_{xx} + \eta_{yy} + \eta_{zz} = 0, \tag{22}$$

$$\zeta_{xx} + \zeta_{yy} + \zeta_{zz} = 0, \tag{23}$$

where $\xi = \xi(x, y, z), \eta = \eta(x, y, z)$ and $\zeta = \zeta(x, y, z)$.

The numbers of mesh points along ξ, η and ζ directions are 161, 21 and 21, respectively.

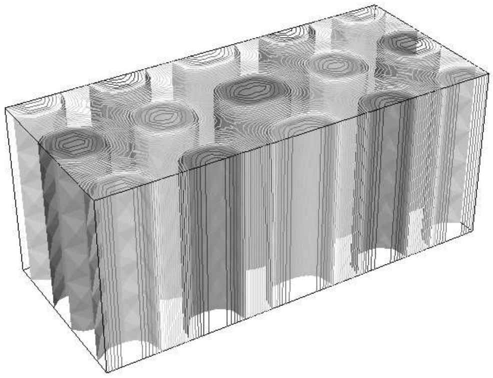


Figure 3 Exact solution

Algorithm to Solve the Discretized Helmholtz Eq.

The discretized Helmholtz eq. for (16) constitutes a large scale of linear system of equations. In order to solve this system of equations, GPBi-CG method[Zha97] is adopted.

Furthermore, in order to validate our Helmholtz solver, we compare numerical solutions with an exact solution;

$$\begin{aligned} u &= \exp(ik\cos\beta x)\exp(ik\sin\beta y) \\ &+ \exp(ik\cos\beta x)\exp(-ik\sin\beta y) \\ &= 2\exp(ik\cos\beta x)\cos(k\sin\beta y). \end{aligned} \tag{24}$$

Figure 3 shows the iso-surfaces of a real part of the exact solution.

Figures 4 and 5 show iso-surfaces of a real part of the numerical solution and some grid lines in the cases where the interfacial boundary Γ is flat and deformed, respectively.

In both cases, a good agreement between the numerical solutions and the exact solution is obtained. And it guarantees that our Helmholtz solver is reasonably reliable.

The CPU time needed to solve once Helmholtz eq. is approximately 20 seconds by DEC Alpha 533MHz.

Hybrid Algorithm by FOM and GAs

In this section, Fuzzy optimization Method (FOM) is briefly summarized.

1. Fuzzy Optimization Method[KS97]
   Modification of the steepest descent method by means of stochastic Fuzzy ruling.

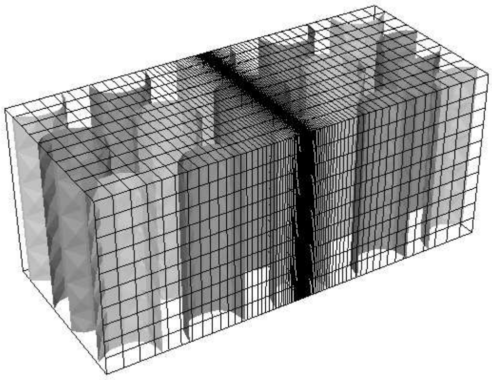


Figure 4 Numerical solution by orthogonal mesh

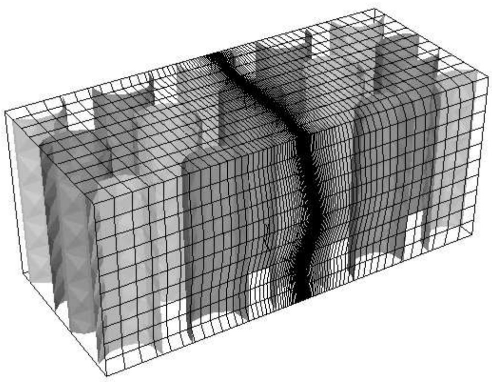


Figure 5 Numerical solution by boundary fitting mesh

2. Single-start FOM[KS97]
   Combination of FOM and Mountain Crossing Algorithm (MCA).
3. Multi-start FOM[KOPS98]
   An increase of the number of initial points. Let operators F, M and R be defined as follows;

   - F : Algorithm due to single-start Fuzzy optimization method,
   - M : Mountain crossing algorithm,
   - R : Rearrangement algorithm by GAs.

   Then the algorithm of Multi-start FOM is stated in the following way;

   (a) Give an initial population W^0 (the set of patrols).
   (b) Compute $U^n := FW^n$ (the set of local minimizers obtained).
   (c) Compute $V^n := MU^n$ (the set of quasi-local maximizers obtained).
   (d) Compute $W^n := RV^n$ (the set of rearranged patrols).
   (e) Increase the generation number $n := n + 1$ and repeat the steps from (b) to (d) until the generation number n is beyond the preset one.

4. Hierarchical Multi-start FOM (To be published)

   (a) Decompose the searching domain into an appropriate number of sub-domains.
   (b) Search local minimizers by Multi-start FOM in each sub-domain.
   (c) Rearrange the sub-domains by GAs or Multi-start FOM.
   (d) Increase the generation number $n := n + 1$ and repeat steps (b) and (c) until the generation number n is beyond the preset one.

Numerical Results

An incident angle is chosen $(\alpha = 0, \beta = 30(> \beta_c))$ in all cases. β_c is a critical angle at which the complete reflection occurs in the case of a flat interfacial boundary;

$$\sin \beta_c = \frac{k_2}{k_1}. \tag{25}$$

First, the boundary matching problem is considered. In this case, the interfacial boundary is flat and the cost function is defined as follows;

$$J_b(a) = \int_\Gamma \left| \frac{1}{\rho_1} \frac{\partial u^{(1)}(a)}{\partial n} - \frac{1}{\rho_2} \frac{\partial u^{(2)}(a)}{\partial n} \right|^2 d\Gamma. \tag{26}$$

Figure 6 shows the absolute value of complex sound pressure after the optimization process terminates.

Since incident angle β is larger than the critical angle β_c, numerical results show that a sound wave is almost completely reflected into the water region.

We then optimize the shape of the interfacial boundary under the constraint of the matching of transmission condition, i.e., Shape Optimization Problem. Figure 7

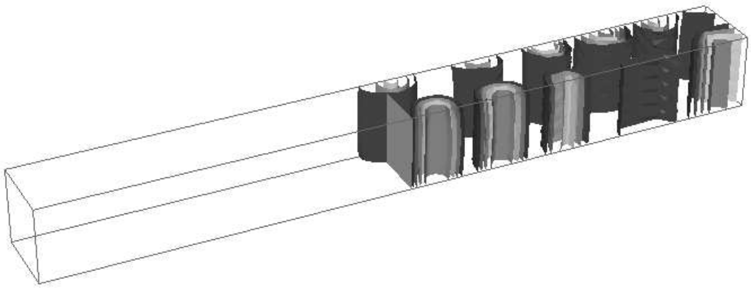


Figure 6 Boundary matching problem (absolute value)

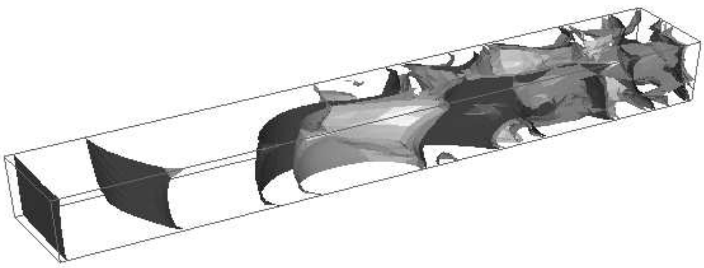


Figure 7 Shape optimization (absolute value)

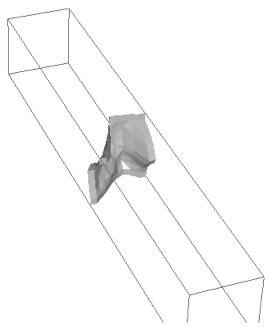


Figure 8 Optimal shape of the interfacial boundary Γ

shows the absolute value of complex sound pressure after the optimization process terminates.

As a result of shape optimization of the interfacial boundary, numerical results show that a large portion of the sound wave is transmitted into the pine region.

Figure 8 shows the optimal shape of interfacial boundary Γ after the 15th iteration.

Conclusions

Carrying out the computations in this paper, one of the difficulties we had to overcome was to construct a 2D/3D Helmholtz solver in the deformed region. The technique of preconditioning in iteration contributed much to solving large scale discretized equations. The description of variation in the usual application of shape design techniques was not adopted here, i.e., it means there was no use of the variation distributed on the lattice points. The deformation of interfacial boundary is represented by a superposition of the scaling functions in wavelets, that make it not only easy to compute the deformed boundary in each iterative step of the optimization, but also encapsulates the nature of the present problem. The use of FOM as the minimization technique demonstrated its validity and utility by producing satisfactory numerical results. Thanks to the combined use of the above-mentioned techniques, it seems that reasonable results can be obtained from a numerical point of view.

REFERENCES

[KOPS98] Kawarada H., Ohtomo T., Periaux J., and Suito H. (1998) Multi-start fuzzy optimization method. In *GAKUTO International Series, Mathematical Sciences and Applications, Vol. 11*, pages 338–345. Gakko Tosho, Tokyo, Japan.

[KS97] Kawarada H. and Suito H. (1997) Fuzzy optimization method. In *Computational Science for the 21th Century*, pages 642–651. John Wiley & Sons.

[Zha97] Zhang S.-L. (1997) GPBi-CG : Generalized product-type methods based on bi-cg for solving nonsymm *SIAM J. on Scientific Computing* 18-2: 537–551.

31

How Scalable is Domain Decomposition in Practice?

D. E. Keyes[1]

Introduction

The convergence rates and, therefore, the overall parallel efficiencies of additive Schwarz methods are often dependent on subdomain granularity. Except when effective coarse-grid operators and intergrid transfer operators are known, so that optimal multilevel preconditioners can be constructed, the number of iterations to convergence tends to increase with granularity for elliptically-controlled problems, for either fixed or memory-scaled problem sizes. The communication overhead per iteration also grows with granularity when global operations, such as inner products, are required as part of the acceleration scheme. The growth is sublinear on reasonable networks, but not *a priori* dismissable as processor counts enter the thousands.

In practical large-scale applications, however, the convergence rate degradation of single-level additive Schwarz is sometimes not as serious as the scalar, linear elliptic theory would suggest. Its effects are mitigated by several factors, including pseudo-transient nonlinear continuation and dominant intercomponent coupling that can be captured fully in a point-block ILU preconditioner. Furthermore, computations with complex models may be so dominated by fully concurrent work at each gridpoint that global communication constitutes a small tax, albeit one that grows with the number

1 Departments of Mathematics & Statistics and Computer Science, Old Dominion University, Norfolk, VA 23529-0162 and ICASE, NASA Langley Res. Ctr., Hampton, VA 23681-2199 and ISCR, Lawrence Livermore Nat. Lab., Livermore, CA 94551; `keyes@icase.edu`. Supported in part by the National Science Foundation under grant ECS-9527169, by NASA under contracts NAS1-19480 and NAS1-97046, and by Argonne National Laboratory under contract 982232402.

Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

of processors while the point-concurrent work stays fixed. We illustrate these favorable claims for the practicality of domain decomposition with encouraging scalabilities for a legacy unstructured-grid Euler flow application code, parallelized with the pseudo-transient Newton-Krylov-Schwarz (ΨNKS) algorithm using the freely available PETSc library on contemporary high-end platforms. We note some impacts on performance of the horizontal (distributed) and vertical (hierarchical) aspects of the memory system and consider architecturally motivated algorithmic variations for their amelioration.

Scalability

Domain decomposition has been "reinvented" regularly for several reasons: it is natural for task-based divide-and-conquer algorithms on problems with multi-physics and/or multi-discretizations, it provides a framework for combination and reuse of legacy software, and it organizes "out of core"-style access to massive amounts of memory. However, the main motivation for the focus on domain decomposition methods in the past two decades is scalable performance through data parallelism. Few other algorithms for PDEs possess the same combination of *arbitrary concurrency degree* with *modest convergence degradation*.

Definitions of scalability

Several types of scalability are formally defined in the parallel computing literature, and numerous other operational definitions of scalability exist in different scientific subcommunities, making semantic digression a necessity before any consideration of the scalability of domain decomposition methods. Given a performance metric that is a function of parameters $\{\pi_1, \pi_2, \ldots\}$, we study the variation of the metric as the π_i are varied, independently or on some constrained manifold. If the metric exhibits acceptable behavior over some region of parameter space, performance is said to "scale" over that region. Scalability of interesting numerical algorithms is rarely uniform over the entire parameter space, so claims of scalability should ordinarily be accompanied by a statement of applicable domain.

Consider parameters N, the discrete problem dimension, and P, the number of processors employed, and performance metric η, the parallel efficiency. The variation of $\eta(N,P)$ as P varies with fixed N is the "fixed-problem-size" scaling. The variation of $\eta(N,P)$ as N and P vary with N/P fixed is the "fixed-storage-per-processor" scaling. If computational work is not linear in N, then a fixed-work-per-processor scaling may be useful to consider, in addition. The expression of scalability may be inverted, e.g., the isoefficiency function, $N(P)$, of an algorithm may be sought such that $\eta(N(P),P) = \text{constant}$, as P varies.

Fixed-storage-per-processor is arguably the most interesting limit from an architectural point of view, for three reasons. First, the proportion of memory to processors is typically fixed (at purchase). Second, work and communication are designed to scale as different powers of N/P in domain decomposition methods. If N/P varies, the ratio of communication to work varies, becoming more parasitic as N/P becomes smaller. Third, the performance of a single processor-memory unit may vary considerably with local problem size, due to cache effects. There are often thresholds of workingset size, across which performance jumps (see, e.g., [CSG98]). Careful attention to data locality smooths out these thresholds, providing a range of problem size over which performance is nearly level, but if such thresholds are not controlled for, they may obscure parallel performance evaluation. Keeping N/P constant avoids the possibility of threshold effects. Despite the aesthetic superiority of fixed-storage-per-processor scaling, fixed-size scaling is often performed in practice because, for instance, grid generation is more responsive to discretization demands than parallelization opportunities. Since fixed-size scalability is more difficult to achieve (because of the second point above), we measure it, rather than fixed-storage-per-processor scalability, in all but one of the examples in Section 4.

Absolute efficiency on P processors is defined as the speedup divided by P,

$$\eta(N,P) = \frac{1}{P} \cdot \frac{T(N,1)}{T(N,P)}, \tag{1}$$

where $T(N,P)$ is the execution time. The execution time on one processor of a fixed-size problem large enough to be worthy of the combined memories of $P \gg 1$ is often inflated by memory system thrashing, leading to superunitary efficiencies. Over smaller ranges of variation of processor number, it is useful to define the relative efficiency, in going from Q to P processors $(P > Q)$,

$$\eta(N,P|Q) = \frac{Q \cdot T(N,Q)}{P \cdot T(N,P)}. \tag{2}$$

In iterative methods, time T can be factored into a total number of iterations, I, and an average cost per iteration, C. Domain decomposition methods are typically implemented in an SPMD style that assigns one subdomain per processor, which has the effect of changing the algorithm, mathematically, when P changes, by inducing finer diagonal blocks in the underlying global preconditioner. Thus, domain decomposition has an algorithmic efficiency:

$$\eta_{alg}(N,P|Q) = \frac{I(N,Q)}{I(N,P)}. \tag{3}$$

The remaining factor in the overall efficiency is the implementation efficiency:

$$\eta_{impl}(N,P|Q) \equiv \eta/\eta_{alg} = \frac{Q \cdot C(N,Q)}{P \cdot C(N,P)}. \tag{4}$$

Whereas T and I (and hence η and η_{alg}) are measured, C (and hence η_{impl}) is usually inferred from their ratio.

Problems with scalability metrics

Viewing the parallel performance of domain decomposition (or any other) methods for PDEs through such a simple metric as efficiency is fraught with pitfalls. As is well known [Bai91], efficiency can be improved while absolute performance deteriorates at all granularities by employing methods that perform wasteful local arithmetic, and thus drive up the fraction of the overall work that is concurrent. Hence, efficiency should be measured only after tuning subdomain solver arithmetic complexity to a practical minimum. Fixed-storage-per-processor metrics require varying the discrete problem size, N, along with P. However, increasing the resolution of the problem often degrades linear conditioning and increases nonlinear stiffness, e.g., by giving better definition to near-singularities, increasing the difficulty of the discrete problem. From the point of view of solving the continuous problem, this is highly desirable, and constitutes a main motivation for doing parallel computing in the first place. However, it means that a simple metric like fixed-storage-per-processor scalability is meaningful only for rare problems with self-similarity in the fine scales. This usually obtains only after all multiscale physics is resolved, in a physically irrelevant asymptotic regime. Finally, as mentioned above, fixed-problem-size efficiencies can be greater than 100% for hierarchical memory computers when, as P grows, workingsets suddenly drop into cache. They should be measured only over a range of N/P where single-processor performance is fairly flat.

Theoretical scalability of DD

In this section, we build an extremely simple model of the scalability of domain decomposition methods, while illustrating a methodology that is extensible to more comprehensive models. Reflecting the factorization of efficiency into algorithmic and implementation terms, we separate convergence and implementation issues. We restrict attention to a scalar, linear, elliptically dominated PDE.

Convergence of Schwarz methods

For a general exposition of Schwarz methods for linear problems, see [SBG96]. Assume a d-dimensional isotropic problem. Consider a unit aspect ratio domain with quasi-uniform mesh parameter h and quasi-uniform subdomain diameter H. Then $N = h^{-d}$ and, under the one-subdomain-per-processor assumption, $P = H^{-d}$. Consider four preconditioners: point Jacobi, subdomain Jacobi, 1-level additive Schwarz (subdomain Jacobi with overlapped subdomains), and 2-level additive Schwarz (overlapped subdomains with a global coarse problem with approximately one degree of freedom per subdomain). The first two can be thought of as degenerate Schwarz methods (with zero overlap, and possibly point-sized subdomains). Consider acceleration of the Schwarz method by a Krylov method such as conjugate gradients or one of its many generalizations to nonsymmetric problems, e.g., GMRES. Krylov-Schwarz iterative methods typically converge in a number of iterations that scales as the square-root of the condition number of the Schwarz-preconditioned system. Table 1 lists the expected number of iterations to achieve a given reduction ratio in the residual norm. (Here we gloss over unresolved issues in 2-norm and operator-norm convergence definitions, but see [CZ98].) The first line of this table pertains to diagonally-scaled CG, a very common default parallel implicit method, but one which is *not* very algorithmically scalable, since increasing the problem resolution N degrades the iteration count. The results in this table were first derived for symmetric definite operators with exact solves on each subdomain, but they have been extended to operators with nonsymmetric and indefinite components and inexact solves on each subdomain. The intuition behind this table is the following: Errors propagate from the interior to the boundary in steps that are proportional to the largest implicit aggregate in the preconditioner,

Table 1 Iteration count scaling of Schwarz-preconditioned Krylov methods

Preconditioning	Iteration Count in 2D	Iteration Count in 3D
Point Jacobi	$\mathcal{O}(N^{1/2})$	$\mathcal{O}(N^{1/3})$
Subdomain Jacobi	$\mathcal{O}((NP)^{1/4})$	$\mathcal{O}((NP)^{1/6})$
1-level Additive Schwarz	$\mathcal{O}(P^{1/2})$	$\mathcal{O}(P^{1/3})$
2-level Additive Schwarz	$\mathcal{O}(1)$	$\mathcal{O}(1)$

pointwise or subdomainwise. The use of overlap avoids the introduction of high-energy field discontinuities at subdomain boundaries. The 2-level method projects out low-wavenumber errors at the price of solving a global problem.

Only the 2-level method scales perfectly in convergence rate (constant, independent of N and P), like a traditional multilevel iterative method. However, the 2-level method shares with multilevel methods a nonscalable cost-per-iteration from the necessity of solving a coarse-grid system of size $\mathcal{O}(P)$. Unlike recursive multilevel methods, a 2-level Schwarz method may have a rather fine coarse grid, e.g., $H = \mathcal{O}(h^{1/2})$, which makes it less scalable overall. Parallelizing the coarse grid solve is ultimately necessary. Neither extreme of a fully distributed or a fully redundant coarse solve is optimal, but somewhere in between.

Per-iteration parallel complexity of Schwarz methods

Krylov-Schwarz methods fit perfectly into the bulk synchronous model of parallel complexity, consisting of balanced concurrent computations on each subdomain, with costs that may be taken as proportional to the volume, N/P. Assume a distributed-memory architecture with a toroidal mesh interconnect and a global reduction time of $CP^{1/d}$. Assume no coarse-grid solve. For simplicity, we neglect the cost of neighbor-only communication relative to arithmetic and global reductions.

The first line of Table 2 shows the estimated execution time per iteration in the left column and the overall execution time (factoring in the number of iterations for 1-level additive Schwarz from Table 1) in the right column. All of the work terms (matrix-vector multiplies, subdomain preconditioner sweeps or incomplete factorizations, DAXPYs, and local summations of inner product computations) are contained in A, and, since it is given in units of time, A also reflects per-processor floating-point performance, including local memory system effects. C includes the cost of synchronization, including algorithmic synchronization frequency relative to the work term and the actual cost of each synchronization. If the network architecture changes, the functional form in P of the synchronization term changes. A fuller model would contain a term of the form $B(N/P)^{2/3}$.

The second line of Table 2 shows the optimal number of processors to employ on a problem of size N, based on the parallel complexity in the first line. The work term falls in P and the communication term rises; setting the P-derivative of their sum to zero yields the P that minimizes overall execution time. The marginal value of the ${P_{opt}}^{\mathrm{th}}$ processor may be small, but $\mathcal{O}(N^{d/d+1})$ processors may be employed without

Table 2 Execution time scaling of Schwarz-preconditioned Krylov methods

	Time per iter.	Time overall
$T(N,P)$	$A\left(\frac{N}{P}\right)+CP^{1/d}$	$\left[A\left(\frac{N}{P}\right)+CP^{1/d}\right]\cdot P^{1/d}$
P_{opt}	$(dA/C)^{\frac{d}{d+1}}\cdot N^{\frac{d}{d+1}}$	$((d-1)A/C)^{\frac{d}{d+1}}\cdot N^{\frac{d}{d+1}}$
$T(N,P_{opt})\vert_{d=2}$	$\propto A^{1/3}C^{2/3}\cdot N^{1/3}$	$\propto A^{2/3}C^{1/3}\cdot N^{2/3}$
$T(N,P_{opt})\vert_{d=3}$	$\propto A^{1/4}C^{3/4}\cdot N^{1/4}$	$\propto A^{1/2}C^{1/2}\cdot N^{1/2}$

encountering the phenomenon of "speeddown."

Isoefficiency scaling, under these algorithmic and architectural assumptions, is $P\propto N^{\frac{d}{d+1}}$, or $N\propto P^{\frac{d+1}{d}}$. Observe that this requires more memory per processor as processors increase. If, on the other hand, as is architecturally customary, $N\propto P$ and we solve the largest possible problem as we scale, then in 3D, as C dominates,

$$\eta_{impl}(P|1)=\frac{AN+C}{AN+CP^{1+(1/d)}}\sim P^{-1/d}, \tag{5}$$

$$\eta_{overall}(P|1)=\frac{AN+C}{[AN+CP^{1+(1/d)}]\cdot P^{1/d}}\sim P^{-2/d}. \tag{6}$$

Execution time increases in inverse proportion to efficiency in the fixed-storage-per-processor scaling.

Practical scalability of DD

In practical applications of domain decomposition to large-scale nonlinear problems, the preceding linear scaling estimates for elliptically dominated systems are baselines, at best. Nonlinear problems are approached through a sequence of linear problems, each with its own Jacobian matrix. Nonlinear problems typically employ parameter continuation, such as pseudo-transience, which adds diagonal dominance to these Jacobians, making the linear convergence estimates above *pessimistic*. Nonlinear problems often become "stiffer" as they are better resolved, dragging out the continuation process, requiring more linear systems to be solved, and making the overall estimates above *optimistic* for the overall scaling, inasmuch as they were predicated on a single linear system. Because of these complications that lie outside of any convenient theory, the value of domain decomposition in practical problems is ultimately problem-specific.

Background on a legacy CFD code

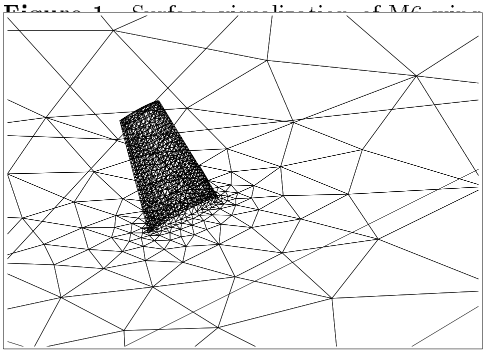
FUN3D is a tetrahedral vertex-centered unstructured grid code developed by W. K. Anderson (LaRC) for compressible and incompressible Euler and Navier-Stokes equations and described elsewhere in these proceedings [KKS99]. It employs pseudo-timestepping for nonlinear continuation towards steady state. One of the solvers in the legacy vector-oriented code is Newton-Krylov with global point-block-ILU preconditioning, which is competitive with FAS multigrid in 2D sequential contexts.

When rewritten in the ΨNKS framework (see, e.g., [KK98b] for the continuation theory, [GKMT98] for software engineering aspects, [KKS98] for architectural aspects, and [CGK+98] for an aerodynamic application of the 2-level method) using PETSc [BGMS98], its flow of control can be represented by the following four-deep nest of loops, the innermost of which is a concurrent `doall`:

```
do l = 1, n_time
    SELECT TIME-STEP
    do k = 1, n_Newton
        compute nonlinear residual and Jacobian
        do j = 1, n_Krylov
            doall i = 1, n_Precon
                solve subdomain problems concurrently
            enddoall
            perform Jacobian-vector product
            ENFORCE KRYLOV BASIS CONDITIONS
            update optimal coefficients
            CHECK LINEAR CONVERGENCE
        enddo
        perform DAXPY update with robustness conditions
        CHECK NONLINEAR CONVERGENCE
    enddo
enddo
```

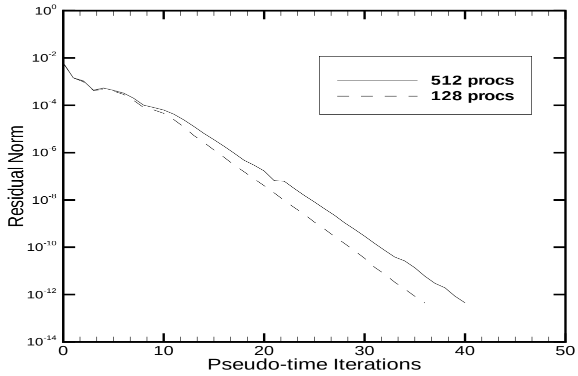
The code is applied to the M6 wing, a coarse surface triangulation of which appears in Fig. 1. Sample convergence histories for a fixed-size incompressible case

Figure 1. Surface simplication of M6 wing



of approximately 2.8 million vertices (approximately 11 million degrees of freedom) on 128 and on 512 processors of a T3E-900 are shown in Fig. 2. To translate the history of the 2-norm of the nonlinear steady-state residual back to the domain decomposition theory, it is necessary to note that the linear Newton correction GMRES iterations are truncated to a fixed (timestep-independent) Krylov dimension. Thus, a weaker preconditioner leads to a poorer Newton correction for a given amount of work, which ultimately increases the number of pseudo-timesteps to convergence. Contrary to the $P^{1/3}$ scalar elliptic theoretical scaling a four-fold increase in P leads to only

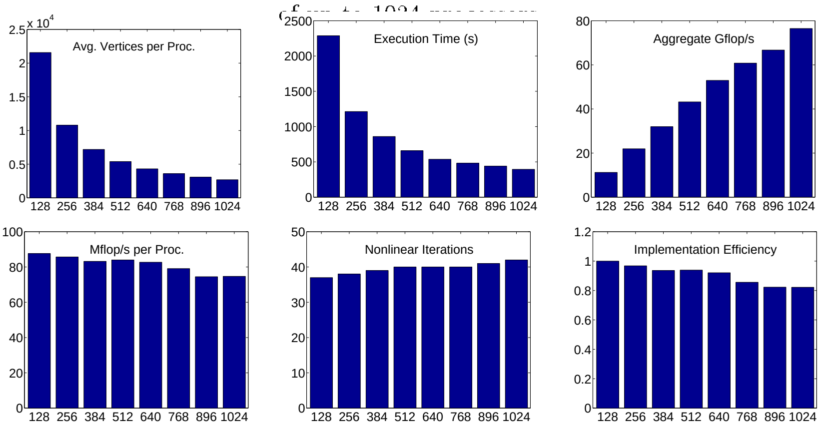
Figure 2 M6 wing convergence history on T3E



a 10% increase in overall iterations. (The results in Fig. 2 represent a significant improvement over earlier results on the same problem in [KKS98]. More aggressive CFL advancement and less linear work per Newton correction make the difference.)

A fuller set of performance profiles for this problem on 128 through 1024 processors of a T3E-1200 appears in Fig. 3.

Figure 3 Scalability profile for fixed-size M6 wing computations on T3E-1200 of up to 1024 processors



Though our limited catalog of M6 grids does not permit a perfect fixed-storage-per-processor scaling, Table 3 comes close. In the final column may be seen a good demonstration of scalable implementation efficiency, and in the first two columns to the right of the double bar may be seen a good illustration of the non-self-similarity of nonlinear problems.

Table 3 Parallel scalability of M6 Euler flow with fixed storage per processor

Vertices, N	P	N/P	Steps	Time	Time/Step
357,900	80	4,474	78	559.93s	7.18s
53,961	12	4,497	36	265.72s	7.38s
9,428	2	4,714	19	131.07s	6.89s

Algorithmic issues in improving scalability

Though the results of Section 4 (and in the companion paper [KKS99]) are encouraging, there is reason to hope for significant improvements in computational performance on problems of contemporary scale (10^6 vertices on 10^2 processors) and there is reason to strive earnestly for further improvements at the next scale, typical of the ASCI vision, namely 10^8 vertices on 10^4 processors. Improvements will have to come largely from more complex algorithms since systems software is advancing very slowly at the high-performance end, and most architectural "advances" drive the percentage of peak available to PDE applications lower, while raising the "theoretical" peak performance. We briefly describe three such improvements algorithmic under investigation; for additional detail, see [Key99].

Improvement in algorithmic efficiency

A region of strong nonlinearity embedded in a region of nearly linear behavior is characteristic of many important PDE problems (e.g., noise, flame, and crack propagation). Such problems may be said to be "nonlinearly stiff"; progress of Newton's method is held hostage to a low-dimensional feature and the expensive-to-obtain Newton correction is small over most of the domain. For an illustration involving shocked transonic flow, see [CGK+98].

For such problems, we propose cyclicly permuting Newton-Krylov-Schwarz methodology into Schwarz-Newton-Krylov. Also known as "nonlinear Schwarz," its two-domain description is as follows:

> Given: $u \in \mathcal{R}^n$, coming from a discretization on Ω, and $f : \mathcal{R}^n \to \mathcal{R}^n$, with $f(u) = 0$ a governing system (including implicitly posed BCs) on Ω. Given: an overlapping partition of Ω into Ω_1 and Ω_2, with induced partionings on u into u_1 and u_2 and f into f_1 and f_2. Given: initial iterates $u_1^{(0)}$ and $u_2^{(0)}$. Iterate $k = 0, 1, \ldots$, until convergence:
>
> Solve $f_1(u_1, u_2^{(k)}) = 0$ for $u_1^{(k+1)}$;
>
> Solve $f_2(u_1^{(k+1)}, u_2) = 0$ for $u_2^{(k+1)}$.

This is the multiplicative version; the additive companion has $u_1^{(k)}$ instead of $u_1^{(k+1)}$ in the second equation. Each subdomain solve (partitioned based on isolation of strongly and weakly nonlinear regions) can be further partitioned for parallel solution by NKS. The overlap region can coincide with an entire inner subdomain. There is a nonoverlapping analog known as Schur-Newton-Krylov, in which the iteration is reduced to a lower-dimensional interface. Automatic detection of the small regions responsible for the nonlinear stiffness may be possible through an indicator such as tensoricity [Key99].

Improvement in parallel implementation efficiency

Profiling some large-scale runs on 128 and 512 processors, we observe that the percentage of time devoted to global inner products increases from 6% to 9%. The growing percentage of execution time consumed by the global reduction step of inner products is the dominant effect on overall efficiency. Independent measurements with barriers indicate that only about 1% of the 9% penalty is attributable to "time on the wire"; the rest is slight load imbalance in subdomain surface work, amplified by a granularity of 512. Extrapolation to 10^4 processors without amelioration of this problem is unacceptable.

We look, in part, to multiobjective load balancing [KK98a], and, in part, to multithreading for synchronization relief. The typical critical path in a nonlinear implicit method is `..., solve, bound_step, update, solve, bound_step, update, solve, ...` There are other useful operations to perform, off the critical path, such as refreshing the Jacobian, testing convergence, adapting continuation and algorithmic parameters, performing interprocessor communication and disk I/O, visualization, filtering, compression, feature extraction, etc. Some of these "off the path" tasks share considerable data locality with critical path tasks, and can be performed during otherwise wasted cycles when a single-threaded code is data-starved. Thus, the scalability of "naked" sparse linear problems with limited arithmetic work between synchronizations, may be unrealistically pessimistic, relative to the richer "marketbasket" of work and synchronization in full simulations.

Other ways of reducing inefficiency attributable to synchronization include: reducing the penalty of each synchronization step by load balancing surface work phase simultaneously with dominant volume work phase; reducing the frequency of synchronizations by employing more speculative control using fewer norms, and/or less stable numerics in projection steps using fewer inner products; and reducing the globality of most synchronization steps by means of the nonlinear Schwarz algorithm of the previous subsection.

Improvement in uniprocessor implementation efficiency

As alluded to above, the high-end superscalar pipelined RISC processors of contemporary parallel machines are typically utilized at a small fraction (e.g., 10%) of their theoretical peak performance by sparse PDE algorithms. Blocking of all of the data defined at each gridpoint improves considerably over other storage schemes, in part by reducing the memory traffic of integers describing the sparsity and in part by permitting unrolling of inner loops over components. This effect is illustrated in [GKMT98]. Additional illustrations of major improvements (ranging from a factor of 2.5 on the Intel Pentium II to a factor of 7.5 on the IBM Power2 SuperChip) in memory locality are presented in [KKS99]. Beyond the intuitive data structure transformations described therein to enhance memory locality, we look to semiautomated data structure reorderings, by applying genetic optimization techniques to the problem of memory layout. Such optimization techniques require an objective function, which should be faster to evaluate than executing a large-scale program and measuring its execution time. The concept of "memtropy" is proposed in [Key99] as a surrogate objective, and a domain decomposition-like blocking for a sequential relaxation algorithm on a structure grid is shown to be an improvement on memtropy, relative to natural ordering. As mentioned in Section 2, improving local memory locality renders parallel computations less sensitive to workingset threshold effects, and thus makes fixed-problem-size scaling more meaningful.

Summary

Schwarz-style domain decomposition is not a truly scalable algorithm in the theoretical sense, in that its parallel efficiency degrades due to both algorithmic and implementation effects as $P \to \infty$ for elliptic problems. However, in practical large-scale nonlinear applications, Schwarz methods are often much better behaved than the theory for model problems predicts. Besides concurrency features, the memory locality of domain decomposition methods makes them highly suitable for advanced architectures, and many avenues remain open for conquering practical difficulties arising from complex applications and architecture, with correspondingly complex algorithmic adaptations.

It is difficult to define scalability metrics that are both practically useful and "fair" for nonlinear PDE-based problems, whether the candidate algorithm is domain decomposition, or anything else. Therefore, a multicriteria evaluation of any parallel PDE code is recommended, including fixed-size and fixed-storage-per-processor scaling, sustained per-processor computational rate as a percentage of peak, and optimality of tuning of the single subdomain code.

Acknowledgements

The author thanks W. Kyle Anderson of the NASA Langley Research Center for providing FUN3D, Dinesh Kaushik of Old Dominion University for his work in porting FUN3D to the parallel environment, and Satish Balay, Bill Gropp, Lois McInnes, and Barry Smith of Argonne National Laboratory who developed the PETSc software employed in obtaining the results of this paper. Computer time was supplied by DOE (through Argonne and NERSC) and by SGI-Cray.

REFERENCES

[Bai91] Bailey D. (August 1991) How to fool the masses when reporting results on parallel computers. *Supercomputing Review* pages 54–55.

[BGMS98] Balay S., Gropp W. D., McInnes L. C., and Smith B. F. (1998) The Portable, Extensible Toolkit for Scientific Computing, version 2.0.22. `http://www.mcs.anl.gov/petsc`.

[CGK+98] Cai X.-C., Gropp W. D., Keyes D. E., Melvin R. G., and Young D. P. (1998) Parallel Newton-Krylov-Schwarz algorithms for the transonic full potential equation. *SIAM J. Sci. Comput.* 19: 246–265.

[CSG98] Culler D. E., Singh J. P., and Gupta A. (1998) *Parallel Computer Architecture: A Hardware/Software Approach.* Morgan-Kaufmann.

[CZ98] Cai X.-C. and Zou J. (1998) Some observations on the l^2 convergence of the additive Schwarz preconditioned GMRES method. Tech Report CU-CS-865-98, Univ. of Colorado at Boulder.

[GKMT98] Gropp W. D., Keyes D. E., McInnes L. C., and Tidriri M. D. (1998) Globalized Newton-Krylov-Schwarz algorithms and software for parallel implicit CFD. ICASE TR 98-24.

[Key99] Keyes D. E. (1999) Trends in algorithms for nonuniform applications on hierarchical distributed architectures. To appear in the Proceedings of a Workshop on Computational Aerosciences in the 21st Century (Salas, ed.), Kluwer.

[KK98a] Karypis G. and Kumar V. (1998) Multilevel algorithms for multi-constraint graph partitioning. Univ. of Minn. CS Dept. TR 98-019.

[KK98b] Kelley C. T. and Keyes D. E. (1998) Convergence analysis of pseudo-transient continuation. *SIAM J. Num. Anal.* 35: 508–523.

[KKS98] Kaushik D. K., Keyes D. E., and Smith B. F. (1998) On the interaction of architecture and algorithm in the domain-based parallelization of an unstructured grid incompressible flow code. In *Proceedings of the Tenth Int. Conf. on Domain Decomposition Methods*, pages 311–319. AMS.

[KKS99] Kaushik D. K., Keyes D. E., and Smith B. F. (1999) NKS methods for compressible and incompressible flows on unstructured grids. In these Proceedings.

[SBG96] Smith B. F., Bjørstad P. E., and Gropp W. D. (1996) *Domain Decomposition: Parallel Multilevel Algorithms for Elliptic Partial Differential Equations.* Cambridge Univ. Press.

32

A non-overlapping DDM of Robin-Robin type for parabolic problems

G. LUBE, L. MÜLLER, F.-C. OTTO[1]

Introduction

We consider a time-dependent advection-reaction-diffusion model. An A-stable implicit semidiscretization in time leads to a sequence of elliptic problems. Here we use the discontinuous Galerkin method DG(r). It allows time-step control based on a-posteriori estimates.
Domain decomposition methods (later referred to as DDM) provide a promising tool for solving the arising elliptic problems. Overlapping methods require a minimal overlap which depends on the recent time step [Kus90], [Ran94]. This results in a cumbersome implementation if the time step varies. Therefore we prefer non-overlapping DDM. There is some recent progress in such methods for elliptic problems using interface conditions of Robin type [Gas96], [Nat95], [Aug98], [Alo98].
In Sec. 2 we introduce the semidiscrete problem. In Sec. 3 we apply, for DG(0), the DDM to the fully discretized, and SUPG-stabilized problem. The main result is an a-posteriori error estimate of the discrete DDM using the interface error, which can be used as a stopping criterion. Additionally we get a bound for a parameter in the interface condition which depends on the (variable) time steps. Then we discuss briefly the extension to DG(1). Numerical results are given in Sec. 4.
For a domain G we denote by $W^{k,p}(G)$ the Sobolev space with norm $\|\cdot\|_{k,p,G}$ and seminorm $|\cdot|_{k,p,G}$. $(\cdot,\cdot)_G$ and $\|\cdot\|_G$ are the inner product and the norm in $L^2(G)$. In

1 Universität Göttingen, Fakultät für Mathematik, D–37083 Göttingen, Germany
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

case of $G = \Omega$ we usually omit the index. For a sufficiently smooth curve $S \subset \overline{\Omega}$, we denote by $\langle \cdot, \cdot \rangle_S$ the inner product in $L^2(S)$ or, whenever needed, the duality product between $H^{-1/2}(S)$ and $H^{1/2}(S)$. C is a generic constant not depending on relevant parameters.

Stable semidiscretization in time

In a bounded domain $\Omega \subset \mathbf{R}^d, d \leq 3$, we consider

$$\begin{aligned} \frac{\partial u}{\partial t} + L_\epsilon u := \frac{\partial u}{\partial t} - \epsilon \Delta u + \vec{b} \cdot \nabla u + cu &= f \qquad \text{in } (0,T] \times \Omega, && (1) \\ u = 0 \quad \text{on } (0,T] \times \partial\Omega; \qquad u &= u_0(x) \quad \text{in } \{0\} \times \Omega. && (2) \end{aligned}$$

Assume that $\nabla \cdot \vec{b} \equiv 0$ and $c \geq 0$.
The discontinuous Galerkin method allows a systematic construction of A-stable, implicit, and high-order schemes for problem (1)-(2). Galerkin orthogonality is a basic ingredient of a-posteriori estimates used for time-step control. We set $\partial_t^q v := (\frac{\partial}{\partial t})^q v$, $\mathbf{V} := W_0^{1,2}(\Omega)$ with norm $\| \cdot \|_{1,\Omega}$. $\mathbf{V}^*$ is the dual space.
The weak formulation of (1)-(2) reads :
Find $u \in W(0,T) \equiv \{z \in L^2(0,T;\mathbf{V}) | \ \partial_t z \in L^2(0,T;\mathbf{V}^*)\}$ *s.t.* $\forall v \in L^2(0,T;V)$

$$\int_0^T (\partial_t u, v) \ dt + \int_0^T B_G(u,v) \ dt = \int_0^T (f,v) \ dt, \quad u(0) = u_0, \tag{3}$$

$$B_G(u,v) \equiv (\epsilon \nabla u, \nabla v) + \frac{1}{2} \Big((\vec{b} \cdot \nabla u, v) - (\vec{b} \cdot \nabla v, u) \Big) + (cu, v). \tag{4}$$

Let $0 = t_0 < t_1 < \ldots < t_{M+1} = T$ be a sequence of discrete time levels with time step $\tau(t) \equiv \tau_m := t_{m+1} - t_m, t \in I_m \equiv [t_m, t_{m+1})$. $\Pi_r(I_m)$ denotes the space of polynomials of degree $r \in \mathbf{N_0}$ with coefficients in $\mathbf{V}$. Furthermore we set $v_\pm^m := \lim_{s \to +0} v(t_m \pm s)$, $[v^m] := v_+^m - v_-^m$.
The DG(r)-method requires the solution of elliptic problems resp. systems on each time slab $I_m \times \Omega$, $m = 0, \ldots., M$

$$\begin{aligned} \textit{Find } u_\tau^m \in \Pi_r(I_m) \quad \textit{s.t.} \quad & A_G^m(u_\tau^m, v) = F_G^m(v) \quad \forall v \in \Pi_r(I_m), && (5) \\ A_G^m(v,w) \quad \equiv \quad & \frac{1}{\tau_m} \{ (\partial_t v, w)_{I_m} + B_G^m(v,w) + (v_+^m, w_+^m) \}, \\ F_G^m(w) \quad \equiv \quad & \frac{1}{\tau_m} \{ (f,w)_{I_m} + (v_-^m, w_+^m) \}, \quad u_-^0 = u_0 \end{aligned}$$

where $(v,w)_{I_m} \equiv \int_{I_m} (v,w) \ dt$ and $B_G^m(v,w) \equiv \int_{I_m} B_G(v,w) \ dt$.
We apply the DG(r)-method with $r = 0$ resp. $r = 1$ for long time resp. time accurate calculations. Error estimates can be derived for problem (5), say for time-independent coefficients $\vec{b}$ and c, which extend the results in [Tho97], Ch.12.

Proposition 1 *Assume that $\partial_t^{r+1} u$ is smooth. Then we obtain for the semidiscrete DG(r)−method the a-priori error estimate*

$$\|u_\tau^m(t_m) - u(t_m)\|^2 \quad \leq \quad C \sum_{i=0}^{m-1} \tau_i^{2(r+1)} \int_{I_i} \|\partial_t^{r+1} u\|_1^2 \ dt$$

and for $r=0$ an a-posteriori estimate with $C \sim \left(1+\|b\|_\infty\sqrt{t_m}\right)\sqrt{\log\frac{t_m}{\tau_m}+1}$

$$\|u_\tau^m(t_m)-u(t_m)\| \le C \max_{i\le m-1} \tau_i \left(\sup_{t\in I_i}\|f(t)\| + \left\|\frac{u^i-u^{i-1}}{\tau_i}\right\|\right).$$

The a-posteriori estimate is useful for an adaptive time-step control. As a first step for $r=0$, one can take the jumps u^i-u^{i-1} as an error indicator.

Robin-Robin DDM for the elliptic problems

DD iteration for the semidiscrete problem

Now we apply a non-overlapping DDM to the semidiscrete problems (5) arising from the DG(0)-method, i.e. with constant (in time) u^m on I_m. Setting $\vec{b}_m = \frac{1}{\tau_m}\int_{I_m}\vec{b}\,dt$, $c_m = \frac{1}{\tau_m}\int_{I_m} c\,dt$, $\tilde{c}_m = c_m + \frac{1}{\tau_m}$, $f_\tau^m = \frac{1}{\tau_m}\left(\int_{I_m} f\,dt + u^{m-1}\right)$, the variational problem (5) is related to the elliptic problem

$$L_{\epsilon,\tau}^m u^m \equiv -\epsilon\Delta u^m + \vec{b}_m\cdot\nabla u^m + \tilde{c}_m u^m = f_\tau^m \quad \text{in } \Omega; \qquad u^m = 0 \quad \text{on } \partial\Omega. \tag{6}$$

Let $\overline{\Omega} = \cup_k \overline{\Omega}_k$ be a non-overlapping partition with $\Gamma_{kj} \equiv \partial\Omega_k \cap \partial\Omega_j, j \neq k$, $\Gamma_k \equiv \partial\Omega_k \setminus \partial\Omega$. Now we seek for $n \in \mathbf{N}$ a sequence of approximations $u_{n;k}^m$ to $u^m|_{\Omega_k}$ on Ω_k using the additive iteration-by-subdomain method of Robin-Robin type

$$\begin{aligned} L_{\epsilon,\tau}^m u_{n;k}^m &= f_\tau^m \ \text{ in } \Omega_k; \qquad u_{n;k}^m = 0 \quad \text{on } \partial\Omega_k\cap\partial\Omega, & (7)\\ \epsilon\frac{\partial u_{n;k}^m}{\partial n_k} + \rho_k^m u_{n;k}^m &= \epsilon\frac{\partial u_{n-1;j}^m}{\partial n_k} + \rho_k^m u_{n-1;j}^m \quad \text{on } \Gamma_{kj},\ j\neq k. & (8)\end{aligned}$$

The main problem is the design of the ρ_k^m. In [Aug98] we obtained convergence to the solution of (6) for $\rho_k^m \equiv -\frac{1}{2}\vec{b}_m\cdot\vec{n}_k + z_k^m,\ z_k^m > 0$.
Denoting by $A_G^{m;k}(\cdot,\cdot)$ and $F_G^{m;k}(\cdot)$ the restrictions of $A_G^m(\cdot,\cdot)$ resp. $F_G^m(\cdot)$ to Ω_k, the weak formulation of (7), (8) can be written as :

$$\begin{aligned} \textit{Find } u_{n;k}^m &\in \mathbf{V}(\Omega_k) := \mathbf{V}|_{\Omega_k} \ s.t.\ \forall v\in\mathbf{V}(\Omega_k),\ \forall\phi\in\mathbf{V}|_{\Gamma_{kj}} \\ A_G^{m;k}(u_{n,k}^m, v) &+ \left\langle z_k^m u_{n,k}^m, v\right\rangle_{\Gamma_k} = F_G^{m;k}(v) + \sum_{j(\neq k)}\left\langle\lambda_{n-1;jk}^m, v\right\rangle_{\Gamma_{kj}}, & (9)\\ \left\langle\lambda_{n;kj}^m,\phi\right\rangle_{\Gamma_{kj}} &= \left\langle\left(z_k^m+z_j^m\right)u_{n;k}^m - \lambda_{n-1;jk}^m,\phi\right\rangle_{\Gamma_{kj}} & (10)\end{aligned}$$

Stabilized Galerkin method and domain decomposition

Now we consider a fully discrete version of the DDM using a SUPG-stabilized FEM. Let $\mathcal{T}_h$ be an admissible triangulation of Ω with simplicial elements K. Furthermore assume that the macroelement partition $\{\Omega_k\}_k$ is aligned with $\mathcal{T}_h$. Let $\mathbf{V}_h \subset \mathbf{V}$ be the subspace of piecewise polynomials of degree $l\in\mathbf{N}$. The discrete space $\Pi_{\tau,h}(I_m)$ in each slab $I_m\times\Omega$ is the set of polynomials of degree r w.r.t. t with coefficients in $\mathbf{V}_h$. In the singularly perturbed case we consider the stabilized Galerkin FEM:

$$\textit{Find } U_{\tau,h}^m \in \Pi_{\tau,h}(I_m) \quad s.t. \quad A_{SG}^m(U_{\tau,h}^m, v) = F_{SG}^m(v) \quad \forall v\in\Pi_{\tau,h}(I_m), \tag{11}$$

$$A^m_{SG}(v,w) \equiv A^m_G(v,w) + \sum_K \delta^m_K \left(L^m_{\epsilon,\tau} v, \vec{b}_m \cdot \nabla w \right)_K ,$$
$$F^m_{SG}(w) \equiv F^m_G(w) + \sum_K \delta^m_K \left(f^m_\tau, \vec{b}_m \cdot \nabla w \right)_K .$$

The usual Galerkin FEM corresponds to $\delta^m_K = 0$. (11) is approximately consistent to the continuous problem (6). The parameter set $\{\delta^m_K\}$ is determined in such a way that (11) yields a stable and accurate method. If only steady state calculations (with time-independent coefficients b and c) are considered, one can replace the semidiscrete residual $L^m_{\epsilon,\tau} v - f^m_\tau$ by $L_\epsilon v - f$, i.e. the approximation $\frac{1}{\tau_m}(u^m - u^{m-1})$ of $\partial_t u$ is not used in the stabilizing terms.
We denote by $A^{m;k}_{SG}(\cdot,\cdot)$ and $F^{m;k}_{SG}(\cdot)$ the restrictions of $A^m_{SG}(\cdot,\cdot)$ resp. $F^m_{SG}(\cdot)$ to Ω_k. Furthermore, $\Pi^k_{\tau,h}(I_m)$ is the restriction of $\Pi_{\tau,h}(I_m)$ to $I_m \times \Omega_k$. The discrete DDM (for $n \in \mathbf{N}$) consists in the parallel solution of problems on $I_m \times \Omega_k$:

$$\textit{Find } U^m_{n;k} \in \Pi^k_{\tau,h}(I_m) \quad s.t. \quad \forall v \in \Pi^k_{\tau,h}(I_m),\ \forall \phi \in \Pi_{\tau,h}(I_m)|_{\Gamma_{kj}}$$
$$A^{m;k}_{SG}(U^m_{n;k}, v) + \left\langle z^m_k U^m_{n;k}, v \right\rangle_{\Gamma_k} = F^{m;k}_{SG}(v) + \sum_{j(\neq k)} \left\langle \Lambda^m_{n-1;jk}, v \right\rangle_{\Gamma_{kj}}, \tag{12}$$
$$\left\langle \Lambda^m_{n;kj}, \phi \right\rangle_{\Gamma_{kj}} = \left\langle \left(z^m_k + z^m_j \right) U^m_{n;k} - \Lambda^m_{n-1;jk}, \phi \right\rangle_{\Gamma_{kj}} . \tag{13}$$

The subproblems are well-posed: Standard arguments for stabilized FEM for elliptic problems yield for all $v \in \Pi^k_{\tau,h}(I_m)$ the coercivity estimate

$$2A^{m;k}_{SG}(v,v) \geq |||v|||^2_{SG;\Omega_k} \equiv \epsilon |v|^2_{1,2,\Omega_k} + \|\sqrt{\tilde{c}_m} v\|^2_{0,2,\Omega_k} + \sum_{K \in \Omega_k} \delta^m_K \|\vec{b}_m \cdot \nabla v\|^2_{0,2,K}.$$

Existence and uniqueness of $U^m_{n;k}$ follow from Lax-Milgram Lemma.

A-priori and a-posteriori estimates for the Robin-Robin method

Here we consider the simplified case $\overline{\Omega} = \overline{\Omega_1} \cup \overline{\Omega_2}$ with $\Omega_1 \cap \Omega_2 = \emptyset$, $\partial\Omega_i \cap \partial\Omega \neq \emptyset$ and $H_i \equiv \mathrm{diam}(\Omega_i)$. The interface is $\Gamma \equiv \partial\Omega_1 \cap \partial\Omega_2$. Furthermore, set $\mathbf{V}(\Omega_i) \equiv \mathbf{V}|_{\Omega_i}$, $\mathbf{V}_{i,h} \equiv \mathbf{V}_h|_{\Omega_i}$, $W \equiv H^{1/2}_{00}(\Gamma)$, $W_h \equiv \mathbf{V}_h|_\Gamma$. We can prove [Lub98]

Theorem 1 *Let $0 < Z \leq z^m_1 = z^m_2 \in L^\infty(\Gamma)$, $\Lambda^m_0 \in L^2(\Gamma)$. Then the solutions $(U^m_{n;k})$ of (12)-(13) converge to the solution of (11) as*

$$|||U^m_{n;k} - U^m|_{\Omega_k}|||_{SG,\Omega_k} \to 0, \quad n \to \infty, \quad k = 1,2.$$

In order to have a stopping criterion for the DD iteration in each time-step, we derive an a-posteriori error estimate which controls the subdomain error $U^m_{n;k} - U^m_{\tau,h}|_{\Omega_k}$ w.r.t. $|||\cdot|||_{SG;\Omega_k}$ in terms of the interface error. Later on we apply

Lemma 1 *[Lub98] The trace mapping $Tr_i : V(\Omega_i) \to L^2(\Gamma)$, the extension operator $Tr_i^{-1} : W \to \mathbf{V}(\Omega_i)$ and the injection of $L^2(\Gamma)$ into W are continuous:*

$$\|Tr_i v\|_{0,2,\Gamma} \leq C_{Tr_i} |v|_{1,2,\Omega_i} \quad \forall v \in \mathbf{V}(\Omega_i), \quad C_{Tr_i} \approx \sqrt{H_i}, \tag{14}$$
$$|Tr_i^{-1}\phi|_{1,2,\Omega_i} \leq C_{Tr_i^{-1}} \|\phi\|_W \quad \forall \phi \in W, \quad C_{Tr_i^{-1}} \approx 1, \tag{15}$$
$$\|\phi\|_{0,2,\Gamma} \leq C_I \|\phi\|_W \quad \forall \phi \in W, \quad C_I \approx \sqrt{H_i}. \tag{16}$$

Furthermore we need Friedrichs inequality

$$\|v\|_{0,2,\Omega_i} \leq C_{F,i} |v|_{1,2,\Omega_i} \quad \forall v \in \mathbf{V}(\Omega_i) \quad C_{F,i} \approx H_i. \tag{17}$$

For convenience, we skip index m of the time slab, e.g. $A^k_{SG}(\cdot,\cdot) := A^{m;k}_{SG}(\cdot,\cdot)$ resp. $F^k_{SG}(\cdot) := F^{m;k}_{SG}(\cdot)$. Let U_i be the restriction of the global discrete solution of (11) to Ω_i. Then the *a-posteriori estimate* reads

Theorem 2 *Let $U \in \mathbf{V}_h$ with $U|_{\Omega_i} =: U_i \in \mathbf{V}_{i,h}$ resp. (U_1^n, U_2^n) be the solutions of the global discrete problem (11) resp. of the two domain variant of the discrete DDM (12)-(13). Then there exists a constant $C > 0$ such that for all $n \in \mathbf{N}$ holds*

$$|||U_1^{n+1} - U_1|||_{SG,\Omega_1} + |||U_2^n - U_2|||_{SG,\Omega_2} \le C \max_{i=1,2} K_{i,i} \; \|U_1^{n+1} - U_2^n\|_W, \tag{18}$$

$$K_{i,1} := \epsilon^{-1/2} \|z_i\|_{0,\infty,\Gamma} \; C_I \; C_{Tr_i}, \tag{19}$$

$$K_{i,2} := \frac{1}{2\sqrt{\epsilon}} \sup_{t\in I_m} \|\vec{b}\cdot\vec{n}_i\|_{0,\infty,\Gamma} C_I \; C_{Tr_i} \tag{20}$$

$$+ C_{Tr_i^{-1}} \left(\sqrt{\epsilon} + \left(\frac{1}{\sqrt{\tau_m}} + \sup_{t\in I_m} \sqrt{\|c\|_{0,\infty,\Omega_i}} \right) C_{F,i} + \sqrt{\tau_m} \sup_{t\in I_m} \|\vec{b}\|_{0,\infty,\Omega_i} \right).$$

The estimate remains valid if Ω_1 and Ω_2 are interchanged.

Sketch of the proof: (i) Problem (11) is equivalent to find $(U_1, U_2) \in \mathbf{V}_{1,h} \times \mathbf{V}_{2,h}$ and $\Lambda_1, \Lambda_2 \in W_h^*$ s.t. $U_1 = U_2$ on Γ and

$$\begin{aligned} A^i_{SG}(U_i, V_i) + \langle z_i U_i, V_i\rangle_\Gamma &= F^i_{SG}(V_i) + \langle \Lambda_j \,, V_i\rangle_\Gamma \qquad \forall V_i \in \mathbf{V}_{i,h}, &(21)\\ \langle \Lambda_i, \phi\rangle_\Gamma &= \langle (z_i + z_j)\, U_i - \Lambda_j, \phi\rangle_\Gamma \qquad \forall \phi \in W_h. &(22)\end{aligned}$$

(ii) Set $E_i^n := U_i^n - U_i$ and $\eta_i^n := \Lambda_i^n - \Lambda_i$. Taking the difference of (21),(22) and (12),(13), we obtain for $i \neq j$ the error problem

$$\begin{aligned} A^i_{SG}(E_i^{n+1}, V_i) \quad &+ \quad \left\langle z_i E_i^{n+1}, V_i\right\rangle_\Gamma = \left\langle \eta_j^n, V_i\right\rangle_\Gamma \quad \forall V_i \in \mathbf{V}_{i,h}, &(23)\\ \left\langle \eta_i^{n+1}, \phi\right\rangle_\Gamma \quad &= \quad \left\langle (z_i+z_j)E_i^{n+1} - \eta_j^n, \phi\right\rangle_\Gamma \quad \forall \phi \in W. &(24)\end{aligned}$$

(iii) Let $z_i \in L^\infty(\Gamma)$ and $K_{i,j}$ as in (19),(20). Then basically using Lemma (1) we obtain for all $w \in W_h$ that

$$\begin{aligned} \left|\langle z_i w, E_i^{n+1}\rangle_\Gamma\right| \quad &\le \quad K_{i,1}\; |||E_i^{n+1}|||_{SG,\Omega_i} \|w\|_W, &(25)\\ \left|\langle \eta_j^{n-1} - z_i E_i^n, w\rangle_\Gamma\right| \quad &\le \quad K_{i,2}\; |||E_i^n|||_{SG,\Omega_i} \|w\|_W, \qquad i \neq j. &(26)\end{aligned}$$

(iv) Starting from (23), (24), we find after some manipulations

$$\begin{aligned} A^1_{SG}(E_1^{n+1}, V_1) \quad + \quad A^2_{SG}(E_2^n, V_2) &= \left\langle \eta_2^n - z_1 E_1^{n+1}, V_1\right\rangle_\Gamma + \left\langle \eta_1^{n-1} - z_2 E_2^n, V_2\right\rangle_\Gamma \\ &= \left\langle z_1(E_2^n - E_1^{n+1}), V_1\right\rangle_\Gamma + \left\langle \eta_1^{n-1} - z_2 E_2^n, V_2 - V_1\right\rangle_\Gamma . &(27)\end{aligned}$$

Set now $V_1 := E_1^{n+1}$, $V_2 := E_2^n$ in (27). We derive lower and upper bounds using the coercivity of A^i_{SG} and step (iii)

$$\begin{aligned} \frac{1}{4}\left(|||E_1^{n+1}|||_{1,SG} + |||E_2^n|||_{2,SG}\right)^2 \quad &\le \quad A^1_{SG}(E_1^{n+1}, E_1^{n+1}) + A^2_{SG}(E_2^n, E_2^n) \\ &\le \max\left(K_{1,1}; K_{2,2}\right)\left(|||E_1^{n+1}|||_{1,SG} + |||E_2^n|||_{2,SG}\right) \|E_2^n - E_1^{n+1}\|_W . &(28)\end{aligned}$$

Noting that $E_2^n - E_1^{n+1} = U_1^{n+1} - U_2^n$ on Γ this implies the assertion. $\square$

The a-posteriori estimate of Theorem 2 gives some information on a suitable choice of the parameter function z_i. A reasonable upper bound is then

$$\|z_1\|_{0,\infty,\Gamma} \leq \frac{\sqrt{\epsilon}}{C_I C_{Tr_1}} K_{2,2}, \qquad \|z_2\|_{0,\infty,\Gamma} \leq \frac{\sqrt{\epsilon}}{C_I C_{Tr_2}} K_{1,2}. \tag{29}$$

According to Lemma 1, we can rewrite (29) for $i \neq j$ as follows:

$$\begin{aligned} \|z_i\|_{0,\infty,\Gamma} \leq \frac{1}{2} \sup_{t \in I_m} \|\vec{b} \cdot \vec{n}_i\|_{0,\infty,\Gamma} + \frac{\epsilon}{\sqrt{H_1 H_2}} \\ + \sqrt{\frac{\epsilon}{H_1 H_2}} \left(\left(\frac{1}{\sqrt{\tau_m}} + \sup_{t \in I_m} \sqrt{\|c\|_{0,\infty,\Omega_j}} \right) H_j + \sqrt{\tau_m} \sup_{t \in I_m} \|\vec{b}\|_{0,\infty,\Omega_2} \right). \end{aligned} \tag{30}$$

Finally we select z_i such that the first r.h.s. term in (30) is replaced with $\frac{1}{2}|\vec{b} \cdot \vec{n}_i|$. Observe that time step τ_m is a critical parameter in (30).

Extension to the DG(1)-case

Let us briefly discuss the application to the *time-accurate* DG(1)-variant: We use for the semidiscrete problem (5) ansatz and test functions

$$u^m(t) := \frac{t_{m+1} - t}{\tau_m} u_0^m + \frac{t - t_m}{\tau_m} u_1^m, \quad v^m(t) := \frac{t_{m+1} + t_m - 2t}{\tau_m} v_0^m + v_1^m$$

with $u_s^m, v_s^m \in \mathbf{V_h}$, $s = 0, 1$. Then we arrive at the elliptic system

$$\frac{\tau_m}{6} B_G(u_0^m - u_1^m, v_0^m) + (u_0^m, v_0^m) = \frac{\tau_m}{6}(f^m - f^{m+1}, v) + (u_1^{m-1}, v), \tag{31}$$

$$\frac{\tau_m}{2} B_G(u_0^m + u_1^m, v_1^m) + (u_1^m, v_1^m) = \frac{\tau_m}{2}(f^m + f^{m+1}, v) + (u_1^{m-1}, v). \tag{32}$$

We propose an efficient iterative decoupling of this system starting from the second equation for $u_1^{m;l}$ with an initial guess for $u_0^{m;0}$ and then solving the first equation for $u_0^{m;l}$ using the last approximation $u_1^{m;l}$. As a result, in each time slab we have to solve a sequence of two different stationary problems. This is again done using the full discretization by a stabilized FEM and the non-overlapping DD method.

Numerical results

We present some results for problem (1)–(2) in $\Omega = (0,1)^2$ showing a reasonable performance of the (discrete) DD method for the time-accurate case. The computations are performed on a triangular mesh with $h = \frac{1}{128}$ with a 2×2 partition for the DD case and using two iterations of decoupling of (31), (32).

Example 1 Consider the heat equation with $\epsilon = 1$, $\vec{b} = \vec{0}$, $c = 0$ and the exact solution $u = \frac{x^2+y^2}{2} \sin(10^3 t)$ which is highly oscillatory in time. In Fig.1 we present the discrete L^2-error norms for the sequential case with different time-steps which shows the better accuracy of the DG(1) method and for the DDM with either fixed

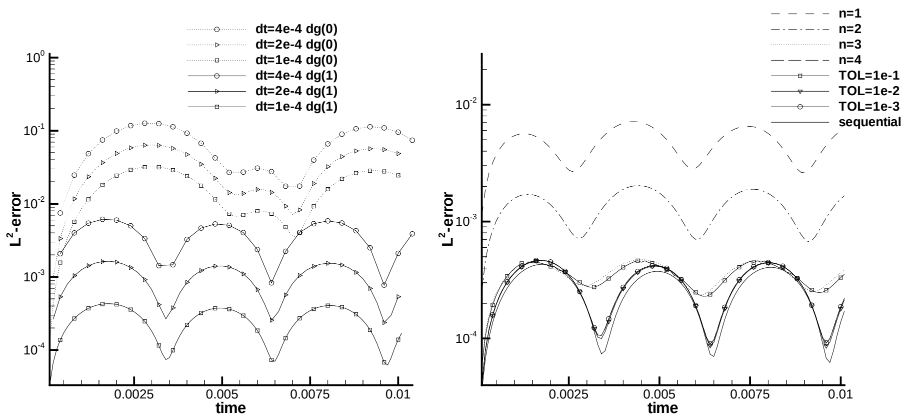


Figure 1 Example 1 (a) sequential (b) DD method

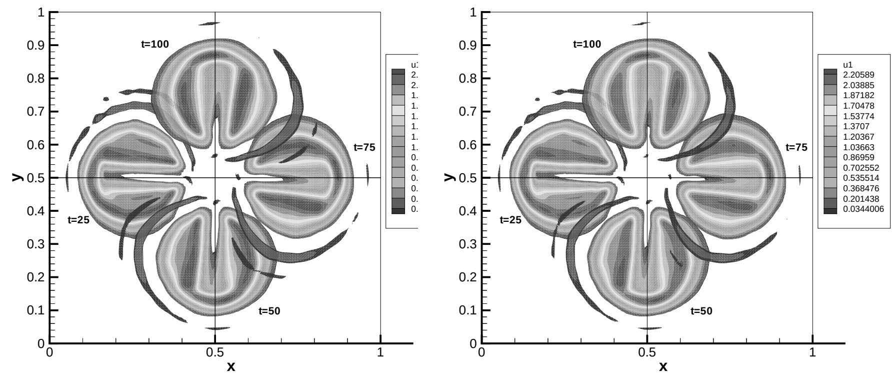


Figure 2 Example 2 (a) sequential (b) DD method

number of DD-steps and the stopping criterion (interface error $\leq$ TOL) arising from the a-posteriori analysis (cf. Thm. 2).

Example 2 Consider an advection-dominated problem with $\epsilon = 10^{-8}$, $c = 0$, $\vec{b} = (-x_2 + 0.5, x_1 - 0.5)^T$. The initial condition is a circular cylinder with a slot being advected with the rotating flow field. In the limit case $\epsilon = 0$, we obtain a periodic solution with period $T = 2\pi$. This is a hard test problem for both the discretization and the interface condition of the DDM. The solution after one rotation without and with domain decomposition is presented in Figure 2. Typical wiggles of the discrete solution around the discontinuity of the initial profile are observed. No reflections can be seen for the DDM thus showing the high quality of the interface condition.

Concluding remarks

The semidiscretization of parabolic problems by the discontinuous Galerkin method results in a sequence of elliptic problems. We propose the solution of these problems using a stabilized FEM and a non-overlapping DDM of Robin/ Robin type. An a-posteriori result allows to control the convergence of the discrete solutions in the subdomains via convergence of the interface data. Furthermore we obtain some information how to design the interface condition depending e.g. on a variable time step. This approach can be easily extended to certain systems of advection-reaction-diffusion equations (for the linear case cf. [Alo98]).
Our main interest is currently the application to coupled models of the incompressible Navier-Stokes equations with scalar advection-diffusion-reaction problems (e.g. from the k/ϵ turbulence model). The research code *Parallel NS* has been implemented on different platforms under a message passing system, fore more detail for the stationary case cf. [Ott98] and [Mue99] for the time-dependent case. Furthermore we refer to [Ott98] for some foundation of the DDM for the linearized Navier-Stokes problem.

REFERENCES

[Alo98] Alonso, A., Trotta, L., Valli, A. (1998) Coercive domain decomposition algorithms for advection-diffusion equations and systems. *J. Comput. Appl. Math.* 96: 57–76.

[Aug98] Auge, A., Kapurkin, A., Lube, G., Otto, F.-C. (1998) A note on domain decomposition of singularly perturbed elliptic problems. In P.E. Bjorstadt et al. (ed) *Ninth Intern. Conf. on Domain Decomposition Methods*, pages 163–170. DDM.org.

[Gas96] Gastaldi, F., Gastaldi, L., Quarteroni, A. (1996) Adaptive domain decomposition methods for advection dominated equations. *East–West J. Numer. Anal.* 3: 165–206.

[Kus90] Kusnetzov, Y.A. (1990) Domain decomposition methods for unsteady convection–diffusion problems. In Glowinski R. and Lions J. (eds) *Proc. 9th Int. Conf. on Comput. Meth. Appl. Sci. Engin.*, pages 211–227. SIAM, Philadelphia.

[Lub98] Lube, G., Müller, L., Otto, F.C. (1998) A non-overlapping DDM for the advection-diffusion problem using the streamline-diffusion FEM. *Preprint* .

[Mue99] Mueller, H. (1999) *(in preparation)*. PhD thesis, TU Dresden, ILR.

[Nat95] Nataf, F., Rogier, F. (1995) Factorization of the convection–diffusion operator and the Schwarz algorithm. *Math. Model. Meth. Appl. Sc.* 5: 67–93.

[Ott98] Otto, F.C., G. Lube (1998) A non–overlapping domain decomposition method for the Oseen equations. *Math. Model. Meth. Appl. Sc.* 8(6): 1091–1117.

[Ran94] Rannacher, R., Zhou, G. (1994) Analysis of a domain-splitting method for nonstationary convection-diffusion problems. *East–West J. Num. Anal.* 2: 151–172.

[Tho97] Thomee V. (1997) *Galerkin Finite Element Methods for Parabolic Problems*. Springer.

33

Communication Latency Hiding in a Parallel Conjugate Gradient Method

Kevin McManus[1], Steve Johnson[2], Mark Cross[3]

Introduction

The diagonally preconditioned Conjugate Gradient Method (CGM) is commonly used in a range of applications for the iterative solution of $\mathbf{Ax} = \mathbf{b}$. The method lends itself to parallelisation by Domain Decomposition (DD) using a Single Program Multi-Data (SPMD) mapping onto Distributed Memory (DM) parallel platforms. While it is possible to achieve good parallel performance it is often the case that performance is limited by the communication performance of the parallel system. Performance of the parallel CGM method can be improved by reducing the impact of communication start-up latency and finite inter-processor communication bandwidth without modification of the algorithm. This invites an examination of the factors affecting parallel performance.

Unstructured mesh applications using Finite Element or Finite Volume techniques to solve differential equations with iterative solution procedures are commonly parallelised using overlapping DD methods based on mesh partitioning. Decomposition of the mesh into P sub-domains distributed over P processors allows concurrent loop execution with global consistency maintained through the exchange of the data between the sub-domains as required to satisfy the dependencies of the integration

1 Centre for Numerical Modelling and Process Analysis, University of Greenwich, London, SE18 6PF, UK email: k.mcmanus@gre.ac.uk
2 Address same as above.
3 Address same as above.
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

method. Extension of the sub-domains to include all required data leads to each sub-domain being surrounded by a layer that overlaps adjacent sub-domains. Message passing is used to communicate data from the processor on which it is assigned into the overlap of the processor(s) on which it is accessed [RL90]. Renumbering of each sub-domain from global to local numbering permits an implementation without global data structures and hence a scalable mapping onto DM parallel hardware [MCJ95].

For many applications, execution time is dominated by the execution of loops in the solution of $\mathbf{Ax} = \mathbf{b}$. Amdahl's law [Amd67] gives the maximum theoretical speed-up for a parallel code running on an ideal machine. With a large problem on each processor Amdahl's law predicts near linear speed-up for parallel solvers when scaling with a constant problem size per processor. But linear speed-up is seldom achieved in practice. The parallel overhead resulting from the time needed to communicate the necessary data between processors restricts the achieved performance. Two strategies that can help to hide the communication overheads are examined in a parallel conjugate gradient method.

Parallel Conjugate Gradient Method

CGM is an established non-stationary iterative method for symmetric positive definite systems providing rapid convergence and $O(m)$ computational cost per step, where m is the number of non-zero components of $\mathbf{A}$. Preconditioning may be used to improve the condition number of the matrix $\mathbf{A}$ [GL89, BBC$^+$94]. For a positive definite matrix preconditioner $\mathbf{M}$

$$\mathbf{Ax} = \mathbf{b} \quad \equiv \quad \mathbf{M^{-1}Ax} = \mathbf{M^{-1}b} \tag{1}$$

If the eigenvalues of $\mathbf{M^{-1}A}$ are clustered better than the eigenvalues of $\mathbf{A}$ then the preconditioned problem may converge in fewer iterations than the original problem. The Jacobi preconditioner is the diagonal of the $\mathbf{A}$ matrix that has the effect of scaling the quadratic form along the coordinate axes [She94]. While not the most effective preconditioner in reducing iterations its simplicity provides computational efficiency. A modification [LL94] which transforms of $\mathbf{Ax} = \mathbf{b}$ into $\mathbf{\check{A}\check{x}} = \mathbf{\check{b}}$ where the components of $\mathbf{\check{A}}$, $\mathbf{\check{x}}$ and $\mathbf{\check{b}}$ are

$$\check{a_{ij}} = \frac{aij}{\sqrt{a_{ii}a_{jj}}}, \quad \check{x_i} = x_i\sqrt{a_{ii}}, \quad \check{b_i} = \frac{b_i}{\sqrt{a_{ii}}} \tag{2}$$

results in the diagonal of $\mathbf{\check{A}}$ being the identity matrix $\mathbf{I}$ and so the Jacobi preconditioner $\mathbf{M} = \mathbf{I}$. The CGM iteration can then be expressed as

$$\mathbf{u^{(k)}} = \mathbf{Ap^{(k-1)}} \tag{3}$$

$$\alpha^{(k)} = \frac{\rho^{(k-1)}}{\mathbf{p^{(k-1)T}u^{(k)}}} \tag{4}$$

$$\mathbf{x^{(k)}} = \mathbf{x^{(k-1)}} + \alpha^{(\mathbf{k})}\mathbf{p^{(k-1)}} \tag{5}$$

$$\mathbf{r^{(k)}} = \mathbf{r^{(k-1)}} - \alpha^{(\mathbf{k})}\mathbf{u^{(k)}} \tag{6}$$

$$\rho^{(k)} = \mathbf{r^{(k)T}r^{(k)}} \tag{7}$$

$$\beta^{(k)} = \frac{\rho^{(k)}}{\rho^{(k-1)}} \tag{8}$$

$$\mathbf{p^{(k)}} = \mathbf{r^{(k)}} + \beta^{(\mathbf{k})}\mathbf{p^{(k-1)}} \tag{9}$$

An overlapping DD implementation of this method requires three communications within the iterative loop. One overlap update is required for the matrix multiply and each of the two inner products require a global summation. The integration stencil results in the distributed **A** matrix on each processor being non-square, having coefficients that address **x** in the overlap. Values of **x** must therefore be copied into the overlap before they are addressed by the matrix-vector multiplication. Such an overlap update communication only requires communication between adjacent subdomains and so may be expected to scale well with increasing P. A distributed inner product loop results in each processor calculating a local inner product which must be summed to produce a global inner product. A binary tree communication pattern is used to minimise the number of messages required to calculate a global summation.

Substituting for $\mathbf{r^{(k)}}$ in Equation (7) gives

$$\rho^{(k)} = (\mathbf{r^{(k-1)}} - \alpha^{(\mathbf{k})}\mathbf{u^{(k)}})^{\mathbf{T}}(\mathbf{r^{(k-1)}} - \alpha^{(\mathbf{k})}\mathbf{u^{(k)}}) \tag{10}$$

which can be expanded to produce

$$\rho^{(k)} = \mathbf{r^{(k-1)T}r^{(k-1)}} + \alpha^{(\mathbf{k})\mathbf{2}}\mathbf{u^{(k)T}u^{(k)}} + \mathbf{2}\alpha^{(\mathbf{k})}\mathbf{r^{(k-1)T}u^{(k)}} \tag{11}$$

substituting $\mathbf{r^{(k-1)T}r^{(k-1)}}$ from Equation (7) now gives

$$\rho^{(k)} = \rho^{(k-1)} + \alpha^{(k)}(\alpha^{(k)}\mathbf{u^{(k)T}u^{(k)}} + \mathbf{2r^{(k-1)T}u^{(k)}}) \tag{12}$$

Calculation of $\rho^{(k)}$ now requires two inner products instead of one but no longer requires $\mathbf{r^{(k)}}$ and so may be moved forward in the loop to the same point at which $\alpha^{(k)}$ is calculated.

$$\mathbf{u^{(k)}} = \mathbf{Ap^{(k-1)}} \tag{13}$$

$$\alpha^{(k)} = \frac{\rho^{(k-1)}}{\mathbf{p^{(k-1)T}u^{(k)}}} \tag{14}$$

$$\rho^{(k)} = \rho^{(k-1)} + \alpha^{(k)}(\alpha^{(k)}\mathbf{u^{(k)T}u^{(k)}} + \mathbf{2r^{(k-1)T}u^{(k)}}) \tag{15}$$

$$\mathbf{x^{(k)}} = \mathbf{x^{(k-1)}} + \alpha^{(\mathbf{k})}\mathbf{p^{(k-1)}} \tag{16}$$

$$\mathbf{r^{(k)}} = \mathbf{r^{(k-1)}} - \alpha^{(\mathbf{k})}\mathbf{u^{(k)}} \tag{17}$$

$$\beta^{(k)} = \frac{\rho^{(k)}}{\rho^{(k-1)}} \tag{18}$$

$$\mathbf{p^{(k)}} = \mathbf{r^{(k)}} + \beta^{(\mathbf{k})}\mathbf{p^{(k-1)}} \tag{19}$$

The three inner products in Equations (14) and (15) can now be calculated using only a single commutative operation to perform the three global summations [BJG94]. If it takes less time to calculate an inner product than the time required to calculate a global summation then this algebraic modification will improve parallel performance.

Performance Estimation

A simple cost model can provide a predictor for parallel performance. The time t_{iter} required for each iteration is the sum of the calculation time t_{calc} and the communication time t_{comm}. An estimate for the calculation time can be calculated from the number of floating point operations in each iteration and the floating point rate of the processor. Unfortunately such an estimate is unlikely to be an accurate reflection of practice as the time for each individual floating point operation is often highly dependent upon the success or otherwise of cache memory systems and superscalar pipeline processor architectures.

Communication time can be characterised in terms of the communication start-up latency which is the time required to send a single word message and the communication bandwidth or rate usually quoted in megabytes per second. The communication time consists of the time required for an overlap update t_{update} and the time required for a global commutative t_{global}. The communication time for the original CGM

$$t_{comm} = t_{update} + 2t_{global} \tag{20}$$

is reduced by the modification to

$$t_{comm} = t_{update} + t_{global} \tag{21}$$

The global commutative time can be approximated as

$$t_{global} \approx 2nt_{latency} \tag{22}$$

where n is the number of levels required in the binary tree communication

$$n = 1 + \lceil \log_2(P-1) \rceil \tag{23}$$

The overlap update time can be approximated as

$$t_{update} \approx 2S(t_{data} + t_{latency}) \tag{24}$$

where S is the maximum degree of sub-domain interconnection, t_{data} is the average time required to communicate an overlap data message. This can be approximated from the surface area of the sub-domain and the communication bandwidth B.

$$t_{data} \approx \frac{(\pi(\frac{6N}{P})^2)^{\frac{1}{3}}}{SB} \tag{25}$$

The achieved bandwidth (excluding latency) is a function of message length which can be crudely approximated as

$$B \approx B_{MAX}\left(1 - \left(\frac{0.8}{\log(\frac{\pi(\frac{6N}{P})^2)^{\frac{1}{3}}}{S})}\right)\right) \tag{26}$$

where B_{MAX} is the maximum bandwidth. The maximum bandwidth for parallel machines varies over the range of $10^5 - 10^8$ words per second. Communication start-up latency $t_{latency}$ varies over the range of 5×10^{-6} to 10^{-3} seconds. Again these can only be estimated as practical considerations such as communication patterns and message contention will have a noticeable effect.

As mentioned in Section 33 if the modified CGM is to return improved performance then the time to calculate an inner product t_{dot} must be less than the time required to perform a global summation.

$$t_{dot} < t_{global} \tag{27}$$

As each inner product requires N/P floating point product and sum calculations the inequality may be expressed as

$$\frac{N}{P}(t_{fp-product} + t_{fp-add}) < 2(1 + \log_2(P - 1))t_{latency} \tag{28}$$

In practice it is not actually helpful to decompose the dot product as t_{dot} will be dependent on the cache and superscalar pipeline performance of the processor which are functions of the vector length N/P and vector storage. Run-time evaluation of t_{dot} for the given problem size together with a run-time measurement of $t_{latency}$ provides an accurate determination of which solver to use for best performance. For a current generation platform t_{dot} may be around 10^{-8} seconds and $t_{latency}$ around 10^{-5} seconds. With a small number of processors, 8 for example, then the modification will be beneficial for N less than 61,000 ($\frac{N}{P} \approx 7,600$). For large numbers of processors, 256 for example, this figure rises to around 4.6 million ($\frac{N}{P} \approx 18,000$). The graph in Figure 1 shows that the expected loss of performance at low P is significantly less than the potential gain in performance which becomes most apparent when efficiency falls to around 50%. This figure is largely a function of the calculation to communication ratio of the platform. The development of communication technology (memory bandwidth) continues to be out-paced by the improvements in processor throughput [KKS97] and so this figure is expected to increase.

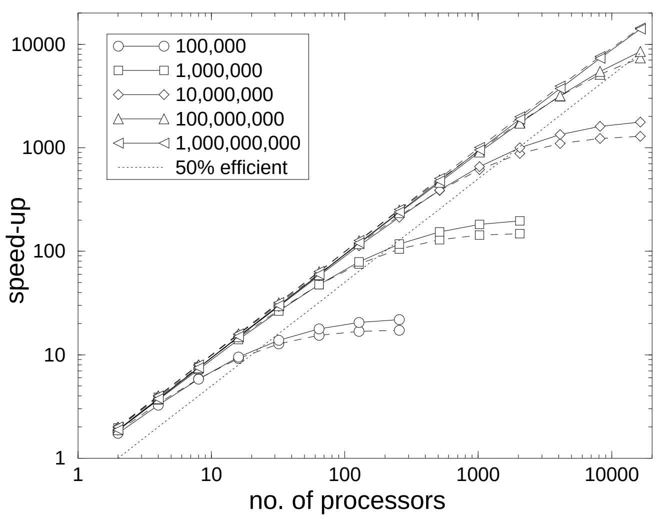


Figure 1 Predicted Speed-up for the Original and the Modified CGM.

The predicted speed-up for the CGM iteration in Figure 1 is an extrapolation the performance of current generation hardware to examine the potential scalability of parallel applications. Without cache effects and constant N t_{dot} scales linearly with P but t_{global} scales as $P\log_2(P)$ and so it was hoped that the curves in Figure 1 would diverge accordingly. However, some of the performance gain is offset with very large P by the overlap update becoming dominant as the bandwidth drops for very short message lengths.

Asynchronous Communication

Asynchronous or non-blocking communication calls to allow calculation to overlap communication. Communication subroutines can initialise a communication and return from the subroutine call before completion of the communication. The communication can then be tested for completion (synchronised) at some future point in the code execution. Calculations that are independent of the communicated data can be performed immediately subsequent to an asynchronous communication call and synchronisation effected prior to the point at which the communicated data is accessed [KKS97]. Unstructured mesh parallel applications with local numbering schemes lend themselves to such communication hiding as it is a simple matter to identify the parts of each sub-domain that are required by other processors and renumber the distributed unstructured mesh so that the dependent data is numbered before the independent data with the overlaps being numbered last. Some of the loops can be split so that part of the loop is executed concurrently with the overlap update communication.

$$\mathbf{u^{(k)}} = \mathbf{Ap^{(k-1)}} \qquad \texttt{DEPT..TOTAL} \tag{29}$$

$$\textit{synchronise}$$

$$\mathbf{u^{(k)}} = \mathbf{Ap^{(k-1)}} \qquad \texttt{1..DEPT} \tag{30}$$

$$\alpha^{(k)} = \frac{\rho^{(k-1)}}{\mathbf{p^{(k-1)T}u^{(k)}}} \tag{31}$$

$$\rho^{(k)} = \rho^{(k-1)} + \alpha^{(k)}(\alpha^{(k)}\mathbf{u^{(k)T}u^{(k)}} + \mathbf{2r^{(k-1)T}u^{(k)}}) \tag{32}$$

$$\mathbf{x^{(k)}} = \mathbf{x^{(k-1)}} + \alpha^{(\mathbf{k})}\mathbf{p^{(k-1)}} \tag{33}$$

$$\mathbf{r^{(k)}} = \mathbf{r^{(k-1)}} - \alpha^{(\mathbf{k})}\mathbf{u^{(k)}} \tag{34}$$

$$\beta^{(k)} = \frac{\rho^{(k)}}{\rho^{(k-1)}} \tag{35}$$

$$\mathbf{p^{(k)}} = \mathbf{r^{(k)}} + \beta^{(\mathbf{k})}\mathbf{p^{(k-1)}} \qquad \texttt{1..DEPT} \tag{36}$$

$$\textit{asynchronous exchange}$$

$$\mathbf{p^{(k)}} = \mathbf{r^{(k)}} + \beta^{(\mathbf{k})}\mathbf{p^{(k-1)}} \qquad \texttt{DEPT..TOTAL} \tag{37}$$

The dependent data `1..DEPT` forms a layer on the inside surface of a sub-domain corresponding to the overlaps of adjacent sub-domains. Equation (36) calculates $\mathbf{p^{(k)}}$ over this inner surface layer and these values are sent to the adjacent sub-domain(s). At this point a receive is also posted to obtain the values of $\mathbf{p^{(k)}}$ required in the overlap. Calculation of $\mathbf{p^{(k)}}$ in Equation (37) and $\mathbf{u^{(k)}}$ in Equation (29) overlaps the communication time. The communication is synchronised before calculating the values of $\mathbf{u^{(k)}}$ that require the communicated data.

The success or otherwise of this strategy depends upon the implementation of the parallel system. If message passing is implemented using dedicated hardware then the potential for improvement is significant. If however the communication is implemented entirely in software then there is little to be gained as the processor is required to perform both calculation and communication. Each asynchronous overlap update also requires an additional synchronisation message which can lead to performance degradation in comparison with synchronous communication.

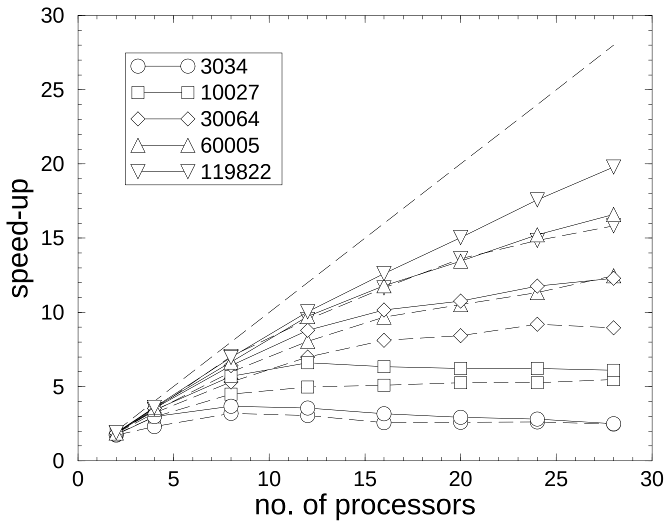


Figure 2 Speed-up using synchronous (dotted line) and asynchronous (solid line) communications in a CGM solver.

The curves in Figure 2 demonstrate a clear performance improvement using a Transtech Paramid in which each Intel i860 processor based compute node is equipped with an Inmos T800 transputer that is used to implement message passing. This system allows the i860 to continue processing while the T800 handles the communication. For performance improvement the asynchronous scheme requires that there is sufficient calculation between the message initialisation and the message synchronisation to compensate for the cost of the synchronisation. Performance gains on other platforms have been less successful as a consequence of both hardware and software limitations.

Discussion

The conjugate gradient method presented has a number of advantages that have led to it being regularly used in several codes. Parallelisation of the method does not require algorithmic modification and therefore results in parallel code for which convergence is largely independent of P. The modifications described have the potential to improve parallel performance in some cases. Modification of the algorithm in order to further reduce communication and hence improve parallel performance is possible [DER93] but this can lead to convergence and stability problems. Further improvement in the scalability of the parallel CGM will therefore require communication costs to be lowered in relation to compute costs. But the communication performance of parallel machines continues to lag behind the developments in processor performance. In particular, inter-processor communication start-up latency is often very high in proportion to the calculation rate. However, the impact of latency and finite bandwidth can be effectively reduced if the parallel system provides truly asynchronous (buffered) message passing.

REFERENCES

[Amd67] Amdahl G. M. (1967) Validity of the single-processor approach to achieving large scale computing capabilities. In *Proc AFIPS*, pages 483–485.

[BBC+94] Barrett R., Berry M., Chan T., Demmel J., Donato J., Dongarra J., Eijkhout V., Pozo R., Romine C., and van der Vorst H. (1994) *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods.* Netlib.

[BJG94] Bjørstad P. E., Jr. W. M. C., and Grosse E. (1994) Parallel domain decomposition applied to coupled transport equations. In Keyes D. E. and Xu J. (eds) *Domain Decomposition Methods in Scientific and Engineering Computing*, pages 369–380. American Mathematical Society. Proceedings of the 7th International Conference on Domain Decomposition, October, 1993.

[DER93] D'Azvedo E., Eijkhout V., and Romine C. (January 1993) Reducing communication costs in the conjugate gradient algorithm on distributed memory multiprocessors. Technical Report S-93-185, Oak Ridge National Laboratory and the University of Tenessee. Lapack Working Note 56.

[GL89] Golub G. H. and Loan C. F. V. (1989) *Matrix Computations: Second Edition.* John Hopkins University Press.

[KKS97] Keyes D. E., Kaushik D. K., and Smith B. F. (1997) Prospects for CFD on petaflops systems. Technical report, ICASE NASA Langley.

[LL94] Lai C.-H. and Liddell H. M. (1994) Preconditioned conjugate gradient methods on the DAP. In *The Mathematics of Finite Elements and Applications VI*, pages 145–156. Academic Press.

[MCJ95] McManus K., Cross M., and Johnson S. (1995) Integrated flow and stress using an unstructured mesh on distributed memory parallel systems. In *Proceedings PCFD'94*, pages 287–294.

[RL90] Robinson G. and Lonsdale R. (April 1990) Fluid dynamics in parallel using an unstructured mesh. Internal report, UKAEA.

[She94] Shewchuck J. R. (March 1994) An introduction to the conjugate gradient method without the agonizing pain. Technical Report CMU-CS-94-125, Carnegie Mellon University.

34

Implementational Aspects of Prewavelet Sparse Grid Methods

A. Niedermeier[1], S. Zimmer[2]

Introduction

For sparse grid discretizations of elliptic PDEs [Zen90, Bun98, BD98], prewavelet bases [CW92] have been introduced to construct fast iterative schemes [GO94, GO95]. In this paper, we want to recall that the implementation of the prewavelet preconditioner is simple, since it fits in the usual tensor product scheme underlying the sparse grid approach. The only extra work we have to take care of, are one-dimensional basis transformations and a diagonal scaling; all other calculations, especially the application of the stiffness matrix, are done in the hierarchical basis. Another way to use prewavelets is to implement a complete solver with explicit use of the prewavelets. At first glance, this seems to be more involved than its hierarchical basis counterpart, due to the larger support of the basis functions. But for the Helmholtz equation on the unit cube, it turns out to be a much simpler code than in the usual hierarchical basis, since it decomposes completely into one-dimensional subproblems.

In the following two sections, we describe the discretization with the piecewise d-linear tensor product basis (hierarchical or prewavelet) that, in the case of more than one spatial dimension, can be used for sparse grid discretizations. Due to the tensor product structure of our discretization, we may first restrict the discussion of the corresponding algorithms to the one-dimensional case that we will later apply sequentially in each coordinate direction. We will pay special attention to the exchange of information between different levels, since this is the crucial part for

1 Zentrum Mathematik, Technische Universität München, D-80290 München
2 Institut für Informatik, Technische Universität München, D-80290 München
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

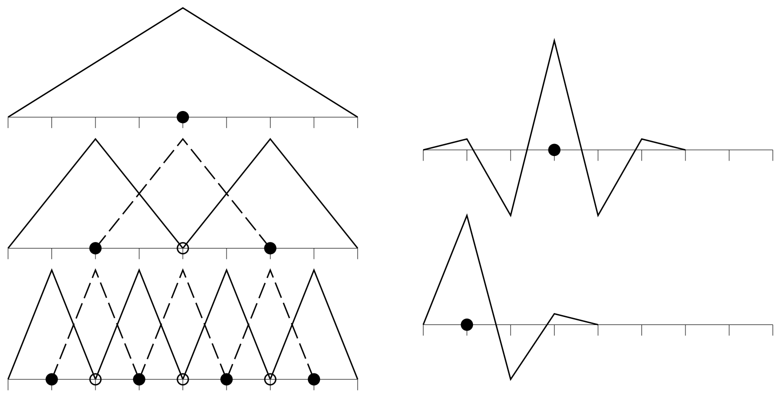


Figure 1 Left: the hierarchical basis B_3^H (solid lines) and the semidefinite system B_3^E (solid and dashed lines). Right: prewavelets for $l = 3$.

the generalization from one to an arbitrary number of dimensions, see [BD98] as an example.

Finally, we will demonstrate the application of the stiffness matrix for the Helmholtz equation in the prewavelet basis and give some numerically computed condition numbers.

The one-dimensional case

For the one-dimensional case, let Ω_l be the regular grid on $[0,1]$ with mesh size $h_l = 2^{-l}$ and inner grid points $\{x_{l,i}, 1 \le i \le 2^l - 1\}$, V_l be the space of functions $u : \Omega \to \mathbf{R}$, piecewise linear with respect to Ω_l, and with $u(0) = u(1) = 0$.

The nodal basis B_l^N and the hierarchical basis B_l^H for V_l consist of the dilated and translated standard hat function $\phi(x) := \max(1 - |x|, 0)$, corresponding to the grid points: $\phi_{l,i}(x) := \phi((x - i \cdot h_l)/h_l)$. Namely, we have $B_l^N := \{\phi_{l,i}, 1 \le i \le 2^l - 1\}$ and $B_l^H := \{\phi_{k,i}, 1 \le k \le l \text{ and } 1 \le i \le 2^k - 1, i \text{ odd}\}$. The hierarchical basis leads to a splitting

$$V_l = \bigoplus_{k=0}^{l} W_k \tag{1}$$

with the hierarchical increments $W_k := \text{span}\{\phi_{k,i}, 1 \le i \le 2^k - 1, i \text{ odd}\}$.

The non-orthogonality of (1) with respect to the L_2-Norm leads to difficulties in the construction of efficient solvers for elliptic PDEs. It can be overcome by the introduction of prewavelets [CW92, GO94, GO95], which provide partial L_2-orthogonality for the price of basis functions with a larger support.

The prewavelet basis B_l^P of V_l is constructed by collecting basis functions of the orthogonal complements with respect to the L_2-Norm of V_{k-1} in V_k, $2 \le k \le l$. The natural choice here are those basis functions $\psi_{l,i}$ with minimum support:

$$\psi_{l,i} := \frac{1}{10}\phi_{l,i-2} - \frac{3}{5}\phi_{l,i-1} + \phi_{l,i} - \frac{3}{5}\phi_{l,i+1} + \frac{1}{10}\phi_{l,i+2}$$

for $3 \leq i \leq 2^l - 3$, i odd, and two special cases on each level $l > 1$ for the two grid points next to the boundary:

$$\psi_{l,1} := \frac{9}{10}\phi_{l,1} - \frac{3}{5}\phi_{l,2} + \frac{1}{10}\phi_{l,3},$$

$\psi_{l,2^l-1}$ symmetrical, and finally $\psi_{1,1} := \phi_{1,1}$. Then, we define $B_l^P := \{\psi_{k,i}, 1 \leq k \leq l \text{ and } 1 \leq i \leq 2^k - 1, i \text{ odd}\}$. The corresponding splitting

$$V_l = \bigoplus_{k=0}^{l} \tilde{W}_k$$

with the hierarchical increments $\tilde{W}_k := \text{span}\{\psi_{k,i}, 1 \leq i \leq 2^k - 1, i \text{ odd}\}$, is orthogonal with respect to the L_2-Norm, but each $\psi_{k,i}$ (sufficiently far away from the boundary) overlaps with four other $\psi_{k,j}$ of the same level l (two on each side) without orthogonality.

We assume that the coefficients u_l^H with respect to the hierarchical basis and coefficients u_l^P with respect to the prewavelets are arranged in a binary tree.

The use of a generating system [Gri94a, Gri94b]

$$B_l^E := \bigcup_{k=0}^{l} B_k^N$$

that contains the nodal bases of all levels will be useful — here not for the construction of multilevel methods, but as an intermediate storage for the transformation between different bases. The storage for the additional coefficients in the extended vector u_l^E is provided by lists of length $l - k$ in the nodes of depth k in the binary tree.

Now, we turn to the transformation of the prewavelet coefficients u_l^P into the hierarchical basis. Besides the use of this operation in the context of a prewavelet preconditioner (where it is basically the only module in the solution process that knows about prewavelets) it can be used as a final step after the direct computation of the solution in prewavelet coefficients, and it can serve as a template for the general structure of all the modules we will need for our solver. On each level, the transformation from the prewavelet basis into the nodal basis is simply the application of a 5-point-stencil with the coefficients $(\frac{1}{10}, -\frac{3}{5}, 1, -\frac{3}{5}, \frac{1}{10})$ — with obvious modifications for the points next to the boundary. Since this nodal basis is part of the generating system, we get by level-wise application a transformation from the prewavelet basis into the B_l^E. To conclude the transformation, we could append a second sweep that runs from the finest towards the coarsest grid and eliminates the coefficients of the basis functions that do not belong to the hierarchical basis. However, it is more economic to avoid the overhead of two tree traversions and combine both sweeps, which is simply a reordering of the operations.

For the extension to d-dimensional problems, we will keep this algorithm on a one-dimensional binary tree as black box. However, it is important to note that the

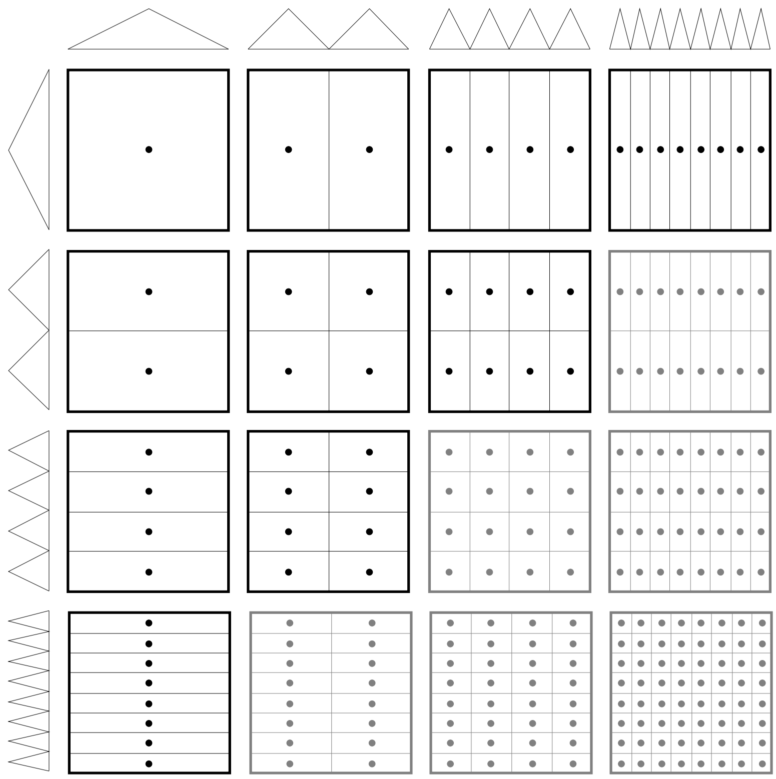


Figure 2 The scheme of grids corresponding to the hierarchical increments $W_{i,j}$ in $\mathbf{K^F}$ (all grids) and $\mathbf{K^S}$ (black grids only) for $d = 2$, $l = 3$.

information transport is only bottom-up: the output coefficients for one specific level k do only depend on the input coefficients of this and the finer grids $k \leq k' \leq l$.

The reverse transformation can be implemented in a similar manner, but it requires the solution of a pentadiagonal linear system of equations on each level. Here, the transport of information is bottom-up, too. The adjoint operators are constructed by the execution of the adjoints of their elementary operations in reverse order. Here, the flow direction changes: they are of top-down type.

The d-dimensional case

The general d-dimensional case is a straightforward tensor product approach. On the unit cube $[0,1]^d$ we define for the multi-index $\mathbf{l} := (l_1, \ldots, l_d)$ the grid $\Omega_\mathbf{l}$ with mesh sizes $\mathbf{h_l} := (2^{-l_1}, ..., 2^{-l_d})$, grid points $\mathbf{x_{l,i}} := (x_{l_1,i_1}, \ldots, x_{l_d,i_d})$, piecewise d-linear ansatz functions

$$\phi_{\mathbf{l},\mathbf{i}}(\mathbf{x}) := \prod_{k=1}^{l} \phi_{l_k,i_k}(x_k),$$

$$\psi_{\mathbf{l},\mathbf{i}}(\mathbf{x}) := \prod_{k=1}^{l} \psi_{l_k,i_k}(x_k),$$

and hierarchical increments $W_{\mathbf{k}} := \operatorname{span}\{\phi_{\mathbf{k},\mathbf{i}}, 1 \leq i_j \leq 2^{k_j} - 1, i_j \text{ odd for all } 1 \leq j \leq d\}$.

With these $W_{\mathbf{k}}$ we can build our finite element space

$$V_{\mathbf{l}} := \bigoplus_{\mathbf{k} \in \mathbf{K}} W_{\mathbf{k}}$$

for some index set $\mathbf{K}$. The usual full tensor product space is given by $\mathbf{K^F} := \{\mathbf{l}, l_j \leq l, 1 \leq j \leq d\}$ for some fixed l. For reasons of efficiency, thinking about the cost/benefit ratio of the different grids, however, we also allow subspaces hereof, where the only condition on $\mathbf{K}$ is to preserve hierarchy:

$$\mathbf{k} \in \mathbf{K} \Rightarrow \forall \mathbf{k}' \leq \mathbf{k} : \mathbf{k}' \in \mathbf{K}$$

with the component-wise relation $\mathbf{k}' \leq \mathbf{k} :\Leftrightarrow \forall 1 \leq j \leq d : k'_j \leq k_j$. A typical choice for $\mathbf{K}$, which results in a sparse grid scheme, is

$$\mathbf{K^S} := \{\mathbf{l}, \sum_{j=1}^{d} l_j \leq l + d - 1\}, \tag{2}$$

see, for example, [Zen90, Bun98].

From the implementational viewpoint, the major difference between $\mathbf{K^F}$ and schemes like $\mathbf{K^S}$ is the absence of a finest grid $\Omega_{(l,\ldots,l)}$ in the latter case. This makes the usual way of implementation of hierarchical basis algorithms impossible, which uses a transformation onto the finest grid and there the nodal basis. In our case, the operations like application of the stiffness matrix have to be implemented explicitly on the hierarchical basis.

A suitable data structure for the coefficient vectors provides a nested structure of a binary trees, one for each coordinate direction $k = 1, \ldots, l$ with the $k-1$-dimensional structures as node values — except for $k = 1$, where the actual coefficients are stored.

The general principle to extend our one-dimensional algorithms — one may for example think of the transformation from the previous section — consists of one outer loop over the dimensions in which we set up a set of one-dimensional binary trees, namely for those grid points whose coordinates differ only in the k-th component, one for each value of this component. On each of these trees, we perform the one-dimensional algorithm.

With respect to the computational domain, this leads to a decomposition into a possibly large number of subdomains that are treated as one-dimensional objects. In contrast to most domain decomposition methods, the orientation of the subdomains is not fixed, but changes with the loop over the dimensions — there is no explicit mechanism for communication between the subdomains.

For the full tensor product space, one can easily convince oneself about the soundness of this algorithm, if we consider a test vector with all but one coefficient zero, and take into account the linearity of the operators.

For sparse grid spaces, which are no tensor product spaces themselves, but subspaces thereof, things are more difficult, since the flow of information in the full grid scheme in some cases uses finer grids as intermediate storage. They are no longer guaranteed to exist — the assumption on the index set is an assumption about the existence of coarser grids. As an example, consider the scheme from figure 2, and a one-dimensional algorithm that is applied first along horizontal lines and then along vertical lines in the scheme. If both top-down and bottom-up transports take place, information interchange from the lower left grid $\Omega_{(1,4)}$ to the upper right grid $\Omega_{(4,1)}$ would require the finest grid $\Omega_{(4,4)}$.

The direction of the information transport (bottom-up, e.g.) is the important fact — the d-dimensional algorithms work if, and only if, after a top-down transport towards finer grids, no bottom-up transport in any other coordinate direction is necessary.

Therefore, the transformations between u_l^P and u_l^H are no special problem — all transport is done bottom-up. The adjoint transformations can also be handled with this technique, since here all one-dimensional operators work top-down.

Note that we do not need permanent storage for the additional coefficients of the d-dimensional semidefinite system $B_{\mathbf{l}}^E$, since we use them only as intermediate storage within the one-dimensional black boxes. This is important, since the number of extra coefficients for fixed mesh size h grows roughly like 2^d.

Application: The Helmholtz equation

For the solution of the Helmholtz equation $\lambda u - \Delta u = f$ on Ω with Dirichlet boundary conditions, we have to evaluate the integrals

$$\lambda \cdot \int_\Omega u(\mathbf{x}) \cdot v(\mathbf{x})\, d\mathbf{x} + \int_\Omega \nabla u(\mathbf{x}) \cdot \nabla v(\mathbf{x})\, d\mathbf{x}$$

for $u, v \in V_K$ by application of the stiffness matrix. Since for tensor product functions

$$u(\mathbf{x}) := \prod_{j=1}^{d} u_j(x_j), \quad v(\mathbf{x}) := \prod_{j=1}^{d} v_j(x_j),$$

the above integrals can be split up into the one-dimensional integrals

$$\prod_{j=1}^{d} \int_0^1 u_j(x) \cdot v_j(x)\, dx, \tag{3}$$

$$\sum_{i=1}^{d} \int_0^1 u_i'(x) \cdot v_i'(x)\, dx \cdot \prod_{j \neq i} \int_0^1 u_j(x) \cdot v_j(x)\, dx, \tag{4}$$

the one-dimensional building blocks for the application of the stiffness matrix, are the application of the one-dimensional stiffness matrices A and B corresponding to

$$a(u,v) := \int_0^1 u'(x) \cdot v'(x)\, dx \quad \text{and} \quad b(u,v) := \int_0^1 u(x) \cdot v(x)\, dx.$$

Table 1 Condition numbers for the scaled prewavelet stiffness matrix on sparse grids (2), $\lambda = 0$ (top) and different values of λ for fixed $d = 3$ and $l = 7$ (bottom).

l	$d=1$	$d=2$	$d=3$	$d=4$	$d=5$	$d=6$	$d=7$	$d=8$
2	5.2	4.5	4.1	3.8	3.6	3.4	3.2	3.1
3	7.8	8.4	8.0	7.8	7.5	7.3	7.1	6.9
4	10.1	12.5	14.3	13.9	13.4	13.0	12.7	12.3
5	12.0	16.2	21.0	24.9	24.3	23.7	23.1	22.6
6	13.1	19.8	27.2	35.7	43.3	-	-	-
7	14.1	23.1	33.7	46.2	-	-	-	-
8	14.8	25.5	40.0	-	-	-	-	-

λ	0	10^2	10^4	10^6	10^8
κ	33.7	28.8	11.9	8.6	8.6

Due to our tensor product bases — and again the linearity of the operators — they are sufficient for the d-dimensional problem.

The one-dimensional algorithms are of quite different character for the hierarchical basis and for the prewavelets, respectively. The hierarchical basis is a-orthogonal, A^H is therefore diagonal, there is no exchange of information between the levels. The mass matrix B^H, however, is comparatively dense — the basis function of level one, for example, is coupled with all other basis functions. The transport of information takes place top-down and bottom-up as well. In the prewavelet basis, different levels are orthogonal with respect to b. On each level, a pentadiagonal matrix arises, but there are no interactions between the levels. Here, the matrix A^P is quite dense (the most convenient way of implementation is the transformation into the hierarchical basis) and causes top-down and bottom-up transport between the levels.

If we put it together for the integrals (3), (4), we note that for each summand we have at most one integral of type A and d or $d-1$ of type B. This causes difficulties for the hierarchical basis, since there are transport processes up and down in more than one coordinate direction and we can no longer apply the standard loop for d-dimensional problems. The algorithm to overcome this — that splits up the upward and the downward data — is relatively complicated, see, for example, [BD98].

In prewavelets, this problem does not arise, since the levels are decoupled in all but possibly one direction. So, we get with the straightforward d-dimensional extension the complete algorithm for the stiffness matrix of the Helmholtz equation on sparse grids. And — after diagonal scaling — the matrix is well conditioned (which was the original reason to introduce prewavelets for sparse grids).

To illustrate this, in table 1 we give some numerically computed condition numbers for the scaled prewavelet stiffness matrix on sparse grids (2). They were computed by simple power iterations with the algorithms above on a 133MHz Intel Pentium personal computer [Nie98].

Summary

The prewavelet basis for sparse grid discretizations results — besides the well conditioned stiffness matrices — in a particularly simple code for the solution of the Helmholtz equation on the unit cube, since the orthogonality properties allow the direct tensor-product type implementation with the number of spatial dimensions simply as a loop parameter.

For more realistic problems, as, for example, on transformed domains, this advantage is lost. Then, it will probably more reasonable to implement the generalizations in an code for the hierarchical basis and to hide the prewavelets in a preconditioner that simply consists of the transformation between prewavelets and the hierarchical basis, its adjoint operator, and a diagonal scaling, and gives the same condition numbers as the direct implementation in prewavelets.

REFERENCES

[BD98] Bungartz H.-J. and Dornseifer T. (1998) Sparse grids: Recent developments for elliptic partial differential equations. In Hackbusch W. and Wittum G. (eds) *Multigrid Methods V*, volume 3 of *LNCSE*. Springer, Berlin/Heidelberg.

[Bun98] Bungartz H.-J. (1998) *Finite Elements of Higher Order on Sparse Grids*. Shaker Verlag, Aachen.

[CW92] Chui C. K. and Wang J. Z. (1992) On compactly supported spline wavelets. *Trans. Amer. Math. Soc.* 330: 903–915.

[GO94] Griebel M. and Oswald P. (1994) On additive schwarz preconditioners for sparse grid discretizations. *Numer. Math.* 66: 449–463.

[GO95] Griebel M. and Oswald P. (1995) Tensor product type subspace splittings and multilevel iterative methods for anisotropic problems. *Adv. Comput. Math.* 4: 171–206.

[Gri94a] Griebel M. (1994) Multilevel algorithms considered as iterative methods on semidefinite systems. *SIAM J. Sci. Stat. Comput.* 15(3): 547–565.

[Gri94b] Griebel M. (1994) *Multilevelmethoden als Iterationsverfahren über Erzeugendensystemen*. B.G.Teubner , Stuttgart.

[Nie98] Niedermeier A. (1998) Lösung von Differentialgleichungen durch Prewavelets. Diplomarbeit, Fakultät für Mathematik, TU München.

[Zen90] Zenger C. (1990) Sparse grids. In Hackbusch W. (ed) *Parallel Algorithms for Partial Differential Equations Proceeings of the 6th GAMM-Seminar, Kiel. Notes on Numerical Fluid Mechanics Vol. 31.* Vieweg, Braunschweig.

35

Comparison of three algorithms for nonlinear metal cutting problems

C.J.Palansuriya[1], C-H.Lai[2], C.S.Ierotheou[3], K.A.Pericleous[4], & D.E.Keyes[5]

Introduction

The properties of a metal can alter and the quality of the cut may degrade considerably due to high temperatures generated in a cutting process. Therefore, the determination of the temperature generated in applying a cutting tool is of industrial interest. The cutting process can be regulated, by actively controlling the speed of the cutter, as well as the application of coolant fluid at the cutter points. This active control may also lengthen the lifetime of the cutter. The active control of the cutting process requires real-time simulation of the temperature distribution. This implies the requirement of fast algorithms.

This paper compares three alternative numerical algorithms applied to a nonlinear metal cutting problem. One algorithm is based on an explicit method [PLIP98] and the other two are implicit. Domain decomposition (DD) is used to break the original domain into subdomains, each containing a properly connected, well-formulated and continuous subproblem. The serial version of the explicit algorithm is implemented in

1 School of Computing & Mathematical Sciences, University of Greenwich, Wellington Street, Woolwich, London SE18 6PF, U.K. email: C.J.Palansuriya@grn.ac.uk
2 Address same as above.
3 Address same as above.
4 Address same as above.
5 Computer Science Department, Old Dominion University, Norfolk, VA 23529-0162 USA and Inst. for Computer Applics. in Science and Engineering, NASA Langley Research Center, MS 403, Hampton, VA 23681-2199 USA
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

FORTRAN and its parallel version uses MPI (Message Passing Interface) calls. One implicit algorithm is implemented by coupling the state-of-the-art PETSc (Portable, Extensible Toolkit for Scientific Computation) [BGLS] software with in-house software in order to solve the subproblems. The second implicit algorithm is implemented completely within PETSc. PETSc uses MPI as the underlying communication library. Finally, a 2D example is used to test the algorithms and various comparisons are made.

The dimensionless 2d nonlinear metal cutting problem

The metal cutting problem considered here is a 2D thin sheet of metal defined in the domain $D = \{(x, y) : 0 < x < 1 \text{ and } 0 < y < 1\}$. The material properties are assumed to be homogeneous across the domain of interest and the following assumptions are made for idealised cutting :- (a) the application of a cutting tool at the cutter points is equivalent to the application of a heat source at these points, (b) no phase changes occur during cutting and (c) the thickness of the cutter is negligible. The cutting is considered to be applied along the y-axis at $x = x_c$. These assumptions lead to the following dimensionless 2D nonlinear, unsteady, parabolic, heat conduction equation,

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}(k(u)\frac{\partial u}{\partial x}) + \frac{\partial}{\partial y}(k(u)\frac{\partial u}{\partial y}) + Q_c(y,t)\delta(x - x_c) \in D, \tag{1}$$

subject to initial condition $u(x, y, 0) = U_i(x, y)$, boundary conditions $u(0, y, t) = B_0(y, t)$, $u(1, y, t) = B_1(y, t)$, $u(x, 0, t) = C_0(x, t)$ and $u(x, 1, t) = C_1(x, t)$. Here $u(x, y, t)$ is a dimensionless temperature distribution, $k(u)$ is the conductivity of the metal, $Q_c(y, t)$ is the unknown source being applied at $x = x_c$, $\delta(x - x_c)$ is the Dirac delta function and U_i, B_0, B_1, C_0 and C_1 are known functions. The unknown source strength may be retrieved by integrating (1) from $x = x_c^-$ to $x = x_c^+$ [PLIP98]. Temperature sensors which are located at $x = x_s$, where $0 < x_c < x_s < 1$, are used to record the temperature variation.

For simulation purposes, the sensor temperatures are modelled by using a periodic function $u(x_s, y, t) = \alpha y(y - 1)^2 sin(\omega t)$. The maximum value is generated by the amplitude α, and the variation with respect to time is generated by the angular frequency ω.

Problem partitioning

Problem partitioning is by a DD method applied at the mathematical/physical problem level [Lai94]. This decomposition provides the primary level of parallelism to the algorithms as discussed in Section 35. In order to solve the inverse problem given in (1) with the additional condition available at $x = x_s$, problem partitioning is carried out to produce three subdomains, such that each subproblem may be solved using a different numerical algorithm. The three subdomains are:

$D_1 = \{(x, y) : 0 < x < x_s \text{ and } 0 < y < 1\}$
$D_2 = \{(x, y) : x_s < x < x_c \text{ and } 0 < y < 1\}$
$D_3 = \{(x, y) : x_c < x < 1 \text{ and } 0 < y < 1\}$

This problem partitioning removes the unknown source term $Q_c(y,t)$ and the Dirac delta function which are associated with the differential equation. The three subproblems (SPs) can be written as follows:

$$SP_1: \quad \frac{\partial u_1}{\partial t} = \frac{\partial}{\partial x}(k(u_1)\frac{\partial u_1}{\partial x}) + \frac{\partial}{\partial y}(k(u_1)\frac{\partial u_1}{\partial y}) \in D_1$$
subject to $u_1(x,y,0) = U_i(x,y)$, $u_1(0,y,t) = B_0(y,t)$,
$u_1(x_s,y,t) = u^*(y,t)$, $u_1(x,0,t) = C_0(x,t)$, $u_1(x,1,t) = C_1(x,t)$.

$$SP_2: \quad \frac{\partial u_2}{\partial t} = \frac{\partial}{\partial x}(k(u_2)\frac{\partial u_2}{\partial x}) + \frac{\partial}{\partial y}(k(u_2)\frac{\partial u_2}{\partial y}) \in D_2$$
subject to $u_2(x,y,0) = U_i(x,y)$, $u_2(x_s,y,t) = u^*(y,t)$,
$\frac{\partial u_2(x_s,y,t)}{\partial x} = \frac{\partial u_1(x_s,y,t)}{\partial x}$, $u_2(x,0,t) = C_0(x,t)$, $u_2(x,1,t) = C_1(x,t)$.

$$SP_3: \quad \frac{\partial u_3}{\partial t} = \frac{\partial}{\partial x}(k(u_3)\frac{\partial u_3}{\partial x}) + \frac{\partial}{\partial y}(k(u_3)\frac{\partial u_3}{\partial y}) \in D_3$$
subject to $u_3(x,y,0) = U_i(x,y)$, $u_3(x_c,y,t) = u_2(x_c,y,t)$,
$u_3(1,y,t) = B_1(y,t)$, $u_3(x,0,t) = C_0(x,t)$, $u_3(x,1,y) = C_1(x,t)$.

Since the temperature values are given at $y = 0$, $y = 1$, $x = 0$ and there are temperature sensors located at $x = x_s$, Dirichlet boundary conditions are defined at the boundary of D_1. The solution of the differential equation provides the required data to calculate the heat flux $\frac{\partial u}{\partial x}(x_s,y,t)$. Therefore, with the knowledge of the temperatures $u(x_s,y,t)$ acquired by the temperature sensors at $x = x_s$, an initial value problem can be formulated in D_2. $u(x_c,y,t)$ values may be obtained by solving this initial value problem. Finally, with the calculated temperatures $u(x_c,y,t)$, another Dirichlet problem can be formulated in D_3. The above three subproblems are well-defined [BBSJ85] [Zwi89]. Hence a unique solution exists for each subproblem and the union of these gives the temperature distribution of the original problem.

Algorithms

Algorithm 1

To solve the probelms in SP_1 and SP_3, a first-order forward difference approximation of the temporal derivative and a second-order Finite Volume (FV) approximation of the spatial derivatives leads to a five-point explicit scheme. Dropping the subscript used in denoting the subdomains, the explicit scheme for the subdomains D_1 and D_3 can be written as,

$$u_{i,j}^{(n+1)} = r_x b_i^{(n)} u_{i-1,j}^{(n)} + r_x a_i^{(n)} u_{i+1,j}^{(n)} + (1 - r_x a_i^{(n)} - r_x b_i^{(n)} - r_y c_j^{(n)} - r_y d_j^{(n)}) u_{i,j}^{(n)} + r_y d_j^{(n)} u_{i,j-1}^{(n)} + r_y c_j^{(n)} u_{i,j+1}^{(n)} \qquad (2)$$

where (i,j) denotes the (i,j)-th grid point, $r_x = \frac{\triangle t}{(\triangle x)^2}$, $r_y = \frac{\triangle t}{(\triangle y)^2}$, $a_i^{(n)} = \frac{k_{i+1,j}^{(n)} + k_{i,j}^{(n)}}{2}$, $b_i^{(n)} = \frac{k_{i-1,j}^{(n)} + k_{i,j}^{(n)}}{2}$, $c_j^{(n)} = \frac{k_{i,j+1}^{(n)} + k_{i,j}^{(n)}}{2}$, $d_j^{(n)} = \frac{k_{i,j-1}^{(n)} + K_{i,j}^{(n)}}{2}$, (n) denotes the time-step, $\triangle t$ is the step size along the temporal axis and $\triangle x$, $\triangle y$ are the grid spacings along the spatial axis x, y, respectively. The initial value problem in SP_2 is solved by employing a second-order Euler Predictor-Corrector (P-C) method along the x-

axis for each time-step. Again, the spatial derivatives are discretised using second-order FV approximations and the time derivative with a first-order finite difference approximation. The two step P-C method can be written as:

$$\begin{pmatrix} u \\ v \end{pmatrix}^* = \begin{pmatrix} u \\ v \end{pmatrix} + \triangle x \underline{f}\ , \quad \begin{pmatrix} u \\ v \end{pmatrix}^{\text{new}} = \begin{pmatrix} u \\ v \end{pmatrix} + \frac{\triangle x}{2}\left\{\underline{f} + \underline{f}^*\right\}\ , \tag{3}$$

where $v = \frac{\partial u}{\partial x}$, $\underline{f} = \underline{f}\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} v \\ \frac{1}{k(u)}(\frac{\partial u}{\partial t} - \frac{\partial}{\partial y}(k(u)\frac{\partial u}{\partial y}) - k'(u)v^2) \end{pmatrix}$ and $\underline{f}^* = \underline{f}\begin{pmatrix} u \\ v \end{pmatrix}^*$. A second order spatially accurate solution may be obtained for each of the three subproblems. Therefore, it is expected to have a second order spatially accurate global solution for the inverse problem (1). The effect of the local truncation error for SP_2 is minimised because of the small size of the subdomain which usually consists of only a few Euler P-C steps. All experiments showed stable results as long as the CFL condition r_x, $r_y \leq 0.25$ was satisfied.

Algorithm 2

Newton's method is applied to D_1 and D_3 and leads to an implicit method. Using Newton's method, the iterative scheme for solving an equation of the type $F(U) = 0$ may be implemented as,

{ Solve
$$F'(U_m)V_m = -F(U_m)$$
Update
$$U_{m+1} = U_m + V_m$$
} Repeat until $||V|| < Tol$

For the metal cutting problem, $F(U) = \frac{\partial}{\partial x}(k(U)\frac{\partial U}{\partial x}) + \frac{\partial}{\partial y}(k(U)\frac{\partial U}{\partial y}) - \frac{\partial U}{\partial t}$ and $F'(U)$ is,

$$\begin{aligned} F'(U) = \; & k''(U)(\tfrac{\partial U}{\partial x})^2 + 2k'(U)\tfrac{\partial U}{\partial x}\tfrac{\partial}{\partial x} + k'(U)\tfrac{\partial^2 U}{\partial x^2} + k(U)\tfrac{\partial^2}{\partial x^2} \\ & + k''(U)(\tfrac{\partial U}{\partial y})^2 + 2k'(U)\tfrac{\partial U}{\partial y}\tfrac{\partial}{\partial y} + k'(U)\tfrac{\partial^2 U}{\partial y^2} + k(U)\tfrac{\partial^2}{\partial y^2} - \tfrac{\partial}{\partial t} \end{aligned}$$

Subdomain D_2 uses the same Euler P-C method as in Algorithm 1. This algorithm is fully implicit and PETSc is used to solve the linearised systems in D_1 and D_3.

Algorithm 3

All three subdomains are solved by using the Newton's method as described in algorithm 2. This algorithm leads to a global linearised system for all three subdomains and they are solved using PETSc.

The implementation of the above three algorithms highlights the fact that the generation of homogeneous and continuous subproblems due to problem partitioning facilitates the use of general purpose libraries, as well as the coupling of subdomain-specialized software solutions.

Exploiting parallelism

The problem partitioning is referred to as "domain parallelism" [ILPP98]; using this concept, each subdomain generated due to the problem partitioning can be mapped directly to a processor and these subproblems may be solved concurrently. However, domain parallelism has an obvious limitation in that it is limited by the number of subdomains and therefore it does not scale with an increasing number of processors. Data partitioning may then be carried out within each subdomain and this is referred to as "domain-data parallelism" [ILPP98].

The parallel implementation of Algorithm 1 is carried out as follows. D_1 is assigned to one group of processors and D_2 and D_3 are assigned to another group of processors, see [PLIP98, ILPP98] for further explanation. Each processor in a group exploits the data-parallelism within the subdomain or subdomains it owns. For Algorithm 2, three groups of processors are created, one subdomain for each group. In order to re-use the in-house software for initial value problem, D_2 is isolated. Again, each processor within a group exploits the data-paralellism in the subdomain it owns, see Figure 1 (dotted lines represent data partitioning). In Algorithms 1 and 2, there is a homogeneous data dependency within each subdomain. Therefore, data partition along x- and y- axises can be carried out arbitrarily within each subdomain.

In Algorithm 3, all three subdomains are solved using one global linearised system. Therefore, data parallelism is used to partition the global linearised system into blocks. However, data partitioning along the x-axis may produce blocks with different sparsity structures, depending upon how D_2 is allocated between the blocks, and therefore with potentially different load balances and interprocessor message patterns. Hence, we only partition along the y-axis.

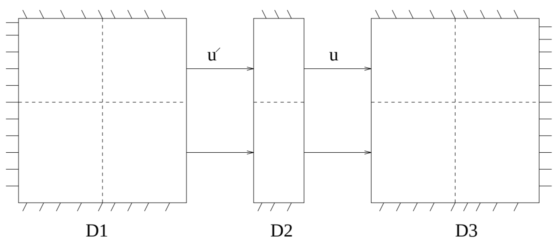


Figure 1 An example of partition for Algorithm 2

Numerical results

Thermal results are obtained for equation (1) with $x_s = 0.5$, $x_c = 0.6$, $U_i(x, y) = 0$, $B_0(y, t) = 0$, $B_1(y, t) = 0$, $C_0(x, t) = 0$ and $C_1(x, t) = 0$. The sensor points are modelled as $u^*(y, t) = \alpha y(y-1)^2 \sin(\omega t)$, with $\alpha = 0.1$ and $\omega = 2\pi$. The nonlinear heat conductivity is given by $k(u) = \frac{1}{1+u^2}$. The temperature fields and the retrival of source/sink strength may be found in [PLIP98]. Figure 2(a) shows the numerical comparison between the three Algorithms, which shows virtually the same numerical results. The sequential implementation of the algorithms was tested for performance in a Sun Sparc 5 workstation. The runtimes are shown in Figure 2(b). Algorithm 1 does not perform very well since it is using a very small $\triangle t$, necessary in order to satisfy the CFL condition. There is no significant difference between the discretizations of Algorithm 2 and Algorithm 3. The latter is more efficient since it solves the subproblems using one system matrix, with a better load balance.

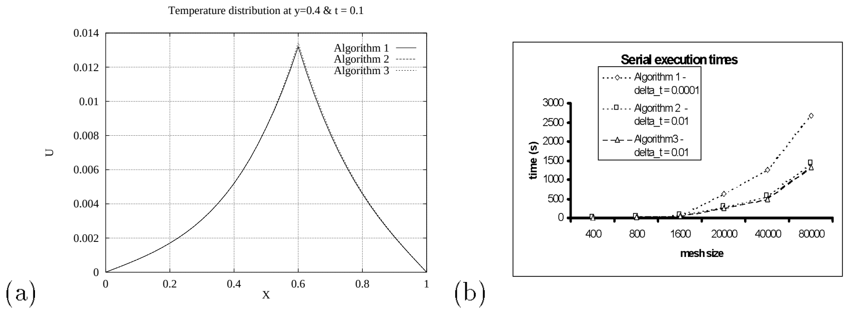


Figure 2 Horizontal solution profile and serial execution times for all three algorithms.

The parallel implementations of the Algorithms were tested on a network of Sun Sparc 5 workstations connected together by an ethernet network. The execution times and the speedups are given in Figures 3, 4 and 5. Algorithm 1 shows very good scalability with increasing number of processors. This is due to the explicit methods in the Algorithm which avoid global synchronisations. Only halo (ghost) points need to be communicated between processors. The other two Algorithms require additional communications in each step of the iterative scheme that lead to more moderate speedups.

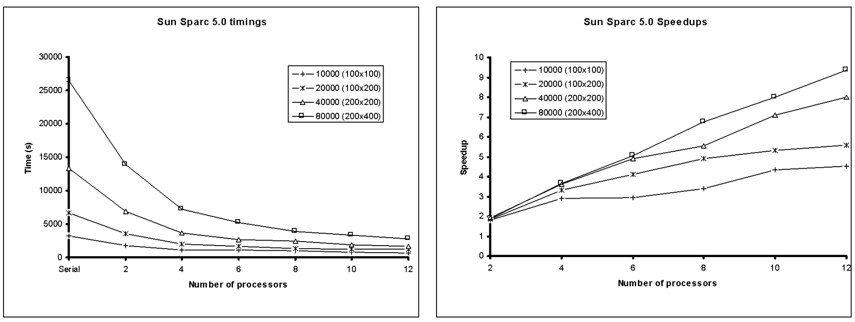


Figure 3 Parallel performance results for Algorithm 1.

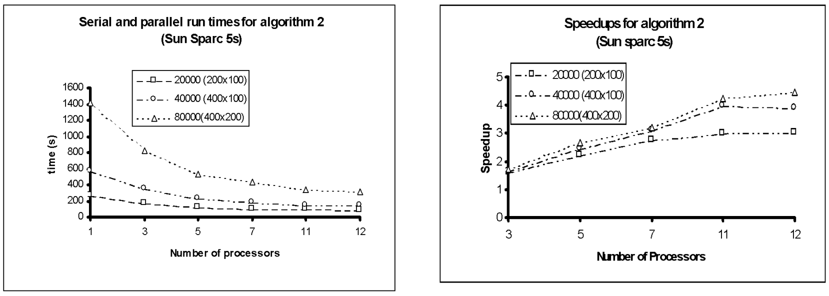


Figure 4 Parallel performance results for Algorithm 2.

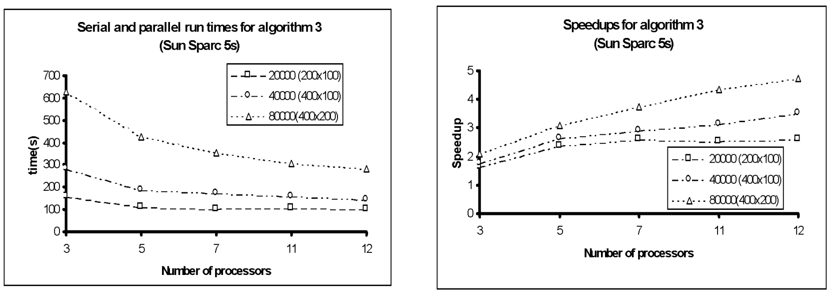


Figure 5 Parallel performance results for Algorithm 3.

Conclusions

The use of alternative numerical algorithms developed by applying DD to the problem domain, in order to calculate the temperature field and to retrieve the unknown source term at the cutter is presented. The three algorithms perform the computation to the same effective accuracy. Algorithms 2 and 3 give better sequential execution times than Algorithm 1. It is shown that good parallelism can be exploited from the DD-based algorithms by using domain-data parallelism and pure data parallelism. An MPI implementation is used to investigate the parallel performance of domain-data parallel versions of the algorithms in a distributed computing environment, that is in a network of Sun Sparc workstations. The parallel performance results show that the two parallelisation strategies can be used effectively in a distributed computing environment.

REFERENCES

[BBSJ85] Beck J. V., Blackwell B., and St.Clair Jr C. R. (1985) *Inverse Heat Conduction: Ill-Posed Problems*. Wiley-Interscience, London.

[BGLS] Balay S., Gropp W., L.C.McInnes, and Smith B.*PETSc 2.0 User Manual*. Argonne National Laboratory. http://www.mcs.anl.gov/petsc/.

[ILPP98] Ierotheou C., Lai C.-H., Palansuriya C., and Pericleous K. (1998) Simulation of 2-d metal cutting by means of a distributed algorithm. *The Computer Journal* 41: 57–63.

[Lai94] Lai C.-H. (1994) Diakoptics, domain decomposition and parallel computing. *The Computer Journal* 37: 840–846.

[PLIP98] Palansuriya C. J., Lai C.-H., Ierotheou C. S., and Pericleous K. A. (1998) A domain decomposition based algorithm for non-linear 2d inverse heat condition problems. In Mandel J., Farhat C., and Cai X.-C. (eds) *Domain Decomposition Methods 10*, volume 218, pages 515–522. American Mathematical Society.

[Zwi89] Zwillinger D. (1989) *Handbook of Differential Equations*. Academic Press Inc., San Diego.

36

Domain Decomposition Methods for the Steady Stokes Equations

EINAR M. RØNQUIST [1]

Introduction

We compare and discuss four different steady Stokes solvers: the Uzawa algorithm, an iterative substructuring method, and two new domain decomposition methods. The methods are tested in the context of high-order spectral element discretizations. The new domain decomposition methods can be categorized as an additive-type preconditioner for the global Stokes system comprising the solution of a coarse Stokes system, the solution of local, non-overlapping pressure systems, and the solution of either local, overlapping viscous systems or a global viscous system. A comparison of the four methods by solving the driven cavity problem in two and three dimensions shows that the new domain decomposition methods applied to the global Stokes system can yield excellent convergence rates. The results for the new methods also indicate that the treatment of the pressure and the velocity degrees-of-freedom in the preconditioner can be quite different; in particular, a non-overlapping treatment suffices for the pressure while an overlapping treatment can be used for the velocity.

Additive Schwarz methods are well established in the context of solving systems of equations derived from the finite element discretization of elliptic problems [DW94, SBG96]. Even though the concept of an overlap is less well-defined for high-order spectral element discretizations, the additive Schwarz methods can also be used in this context. This follows from the fact that the original spectral element system is spectrally equivalent to a finite element system derived from a linear triangulation/tetrahedrazation of the underlying spectral element mesh. Hence, the

1 Nektonics, Inc., 15 Everett Street, Arlington, MA 02474-6901, U.S.A.
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

additive Schwarz preconditioners used for standard, low-order finite element systems can also be used as preconditioners for the spectral element systems. Numerical experiments demonstrating this approach were first presented in [Pah93] in the context of solving the Poisson problem, and later analyzed in [Cas97]. Similar techniques have also been used with success in order to solve the consistent pressure Poisson operator [Fis97, FMTar].

Additive Schwarz methods have only recently been used for the full (indefinite) Stokes system in coupled form. In [KPar], a Krylov space method such as the GMRES method is used in combination with an indefinite preconditioner based upon overlapping Schwarz techniques. The additive version of this preconditioner comprises a coarse (indefinite) Stokes system and local (indefinite) Stokes subproblems associated with the overlapping subdomains. The work in [KPar] was done in the context of stable, mixed, linear finite elements.

In the context of high-order spectral element methods, primarily iterative substructuring methods have been considered for the global, coupled Stokes system in the past [Røn96, Røn98, Cas96, PW97]; the techniques reported in [Røn96, Røn98] have also been extended to solve the steady Navier-Stokes equations in three dimensions. Recently, however, the methodology presented in [KPar] has also been extended to spectral elements; see [Pavar].

In this paper, we consider a new overlapping, additive Schwarz approach for the steady Stokes system in three dimensions. In contrast to the work presented in [KPar] in the context of linear finite elements, and to the work in [Pavar] in the context of spectral elements, we abandon the concept of considering the "full" Stokes operator for the local problems associated with the subdomains. Instead, we consider two new domain decomposition methods where the local pressure solutions are computed based upon non-overlapping techniques, while the velocity degrees-of-freedom are preconditioned either based upon overlapping methods for the local problems or by solving a global problem for the viscous system.

The steady Stokes problem

We consider here the solution of the steady Stokes equations in a domain $\Omega \subset R^d$, $d = 2, 3$: Find a velocity $\mathbf{u} = (u_1, .., u_d) \subset H_0^1(\Omega)^d$ and a pressure $p \subset L_0^2(\Omega)$ such that

$$\begin{aligned} \int_\Omega \mu \nabla \mathbf{u} : \nabla \mathbf{v} \, d\Omega - \int_\Omega p \nabla \cdot \mathbf{v} \, d\Omega &= \int_\Omega \mathbf{f} \cdot \mathbf{v} \, d\Omega \quad \forall \mathbf{v} \in H_0^1(\Omega)^d \ , \\ \int_\Omega q \nabla \cdot \mathbf{u} \, d\Omega &= 0 \quad \forall q \in L_0^2(\Omega) \ . \end{aligned}$$

Here, μ is the viscosity and $\mathbf{f}$ is a body force.

The continuous Stokes problem is discretized using the spectral element method [MP89]. Following this approach, we first decompose the computational domain Ω into K (conforming) spectral elements,

$$\overline{\Omega} \;=\; \cup_{k=1}^K \overline{\Omega}_k .$$

Each spectral element Ω_k is a quadrilateral element in R^2, and a hexahedral element in R^3. Throughout this paper, a single spectral element will also be synonymous with a single subdomain.

On the reference element $\Omega_{\mathtt{ref}} =]-1,1[^d$ associated with each spectral element, the velocity components are approximated as polynomials of degree N in each spatial direction, while the pressure is approximated as a polynomial of degree $N-2$. That is, each spectral element is a $Q_N - Q_{N-2}$ element. More precisely, if we denote the discrete space for the velocity as V and the discrete space for the pressure as W, these spaces are defined as

$$\begin{aligned} V &= \{ v : v_{|\overline{\Omega}_k} \circ F_k \in Q_N(\overline{\Omega}_{\mathtt{ref}})^d \} \cap H_0^1(\Omega)^d \; , \\ W &= \{ w : w_{|\Omega_k} \circ F_k \in Q_{N-2}(\Omega_{\mathtt{ref}}) \} \cap L_0^2(\Omega) \; . \end{aligned}$$

Here, F_k represents the isoparametric (or affine) mapping from the reference domain $\Omega_{\mathtt{ref}}$ onto Ω_k, that is, $\Omega_k = F_k(\Omega_{\mathtt{ref}})$, $k = 1, ..., K$.

In order to derive a set of algebraic equations, we choose (local) tensor-product bases for our discrete spaces V and W, and we evaluate the integrals in the variational formulation by tensor-product (Gauss-Lobatto-Legendre) quadrature. The resulting set of discrete equations can then be written as

$$\underline{\mathbf{S}}\,\underline{\mathbf{x}} = \underline{\mathbf{g}} \tag{1}$$

where

$$\underline{\mathbf{S}} = \begin{bmatrix} \underline{\mathbf{A}} & -\underline{\mathbf{D}}^T \\ \underline{\mathbf{D}} & \underline{0} \end{bmatrix} , \quad \underline{\mathbf{x}} = \begin{bmatrix} \underline{\mathbf{u}} \\ \underline{p} \end{bmatrix} , \quad \underline{\mathbf{g}} = \begin{bmatrix} \underline{\mathbf{f}} \\ \underline{0} \end{bmatrix} .$$

Here, $\underline{\mathbf{A}}$ is the discrete viscous operator (in this case, the vector Laplacian), $\underline{\mathbf{D}}$ is the discrete divergence operator, and its transpose $\underline{\mathbf{D}}^T$ is the discrete gradient operator. The vector $\underline{\mathbf{u}}$ contains the nodal velocity values, $\underline{p}$ represents the nodal pressure values, and the components of $\underline{\mathbf{f}}$ are the nodal forces. Boldface symbols are associated with multi-dimensional quantities, while non-boldface symbols are associated with scalar quantities.

Solution methods

There exist many iterative algorithms for solving the indefinite saddle point problem (1); see for example [MMPR93, Elm96, Cas96, PW97, Pav97, Røn96]. In this section, we will focus our attention on a few iterative methods. In the next section, numerical results will be presented, and the methods will be compared.

Method U: The Uzawa algorithm

A classical solution method, the Uzawa algorithm, consists of first decoupling the Stokes saddle point problem into a positive semi-definite form for the pressure,

$$\underline{\mathbf{D}}\,\underline{\mathbf{A}}^{-1}\underline{\mathbf{D}}^T\,\underline{p} = -\underline{\mathbf{D}}\,\underline{\mathbf{f}} \; . \tag{2}$$

Eliminating the hydrostatic mode, the pressure system can be solved iteratively via a nested conjugate gradient (CG) iteration. Note that each (outer) pressure iteration involves the inversion of the discrete viscous operator, $\underline{\mathbf{A}}$. Since this operator is symmetric, positive definite, the viscous system can also be solved via an (inner) conjugate gradient iteration.

Once the pressure has been computed, the velocity can be found by simply solving the viscous system

$$\underline{\mathbf{A}}\,\underline{\mathbf{u}} = \underline{\mathbf{D}}^T \underline{p} + \underline{\mathbf{f}} \ . \tag{3}$$

An excellent preconditioner for the pressure system (2) is the mass matrix, $\underline{\tilde{M}}$, associated with the pressure degrees-of-freedom. Denoting the Uzawa pressure operator as $\underline{U} = \underline{\mathbf{D}}\,\underline{\mathbf{A}}^{-1}\underline{\mathbf{D}}^T$, it can be shown that the condition number

$$\kappa(\underline{\tilde{M}}^{-1}\underline{U}) \leq c_1/\beta_h^2 \ ,$$

where c_1 is a constant and β_h is the inf-sup parameter [MPR92, PW97].

In order to solve the viscous system (the system (3), as well as the solution of the viscous system associated with each outer pressure iteration), we can choose from a plethora of available preconditioners. For example, for *low-order* finite elements, an overlapping Schwarz method [DW94, SBG96] yields a condition number

$$\kappa(\underline{\mathbf{B}}^{-1}\underline{\mathbf{A}}) \leq c_a\,(1 + H/\delta) \ . \tag{4}$$

Here, $\underline{\mathbf{B}}$ is the additive Schwarz preconditioner corresponding to the viscous operator (in this case, the vector Laplacian), c_a is a constant, H is the diameter of the subdomains, and δ is the overlap. As mentioned in the Introduction, we can also recover the result given in (4) for spectral elements; see [Pah93, Cas97].

Method S: An iterative substructuring algorithm

In this section, we briefly review an iterative substructuring method (i.e., a *non-overlapping* preconditioner) originally proposed in [Røn96]. This method has been extended to solve a variety of steady, incompressible flow problems, including problems described by the steady, three-dimensional Navier-Stokes equations with free surface boundary conditions; see [Røn98]. For comparison reasons, we present this method (here, for the Stokes system only) in a slightly different manner than in [Røn96] and [Røn98]. First, the preconditioned Stokes system is written as

$$\underline{\mathbf{Q}}^{-1}\underline{\mathbf{S}}\,\underline{\mathbf{x}} = \underline{\mathbf{Q}}^{-1}\underline{\mathbf{g}} \ . \tag{5}$$

This preconditioned system is solved using a Krylov space method such as the Generalized Conjugate Residual (GCR) method. The preconditioner can be expressed as

$$\underline{\mathbf{Q}}^{-1} = \underline{\mathbf{Q}}_0^{-1} + \sum_{k=1}^{K} \underline{\mathbf{Q}}_{k,v}^{-1} + \sum_{k=1}^{K} \underline{\mathbf{Q}}_{k,p}^{-1} + \underline{\mathbf{Q}}_{\Gamma}^{-1} \tag{6}$$

where

$$\underline{\mathbf{Q}}_0^{-1} \;\; = \;\; \underline{\mathbf{R}}_0^T\, \underline{\tilde{\mathbf{S}}}_0^{-1}\, \underline{\mathbf{R}}_0 \ ,$$

$$
\begin{aligned}
\underline{\mathbf{Q}}_k^{-1} &= \underline{\mathbf{R}}_k^T\ \tilde{\underline{\mathbf{S}}}_k^{-1}\ \underline{\mathbf{R}}_k\ \ , k=1,...,K\ , \\
\underline{\mathbf{Q}}_{k,v}^{-1} &= \begin{bmatrix} \underline{\mathbf{I}} & \underline{0} \\ \underline{\mathbf{0}} & \underline{0} \end{bmatrix} \underline{\mathbf{Q}}_k^{-1}\ , \\
\underline{\mathbf{Q}}_{k,p}^{-1} &= \begin{bmatrix} \underline{\mathbf{0}} & \underline{0} \\ \underline{\mathbf{0}} & \underline{\mathrm{I}} \end{bmatrix} \underline{\mathbf{Q}}_k^{-1}\ , \\
\underline{\mathbf{Q}}_\Gamma^{-1} &= \underline{\mathbf{R}}_\Gamma^T\ diag(\underline{\mathbf{A}})^{-1}\,\underline{\mathbf{R}}_\Gamma\ [\underline{\mathbf{I}}\ \ + \underline{\mathbf{D}}^T (\sum_{k=1}^{K} \underline{\mathbf{Q}}_{k,p}^{-1})]\ .
\end{aligned}
$$

Briefly explained, the preconditioner consists of a sum of three contributions: (i) a global, coarse Stokes solution ($\underline{\mathbf{Q}}_0^{-1}$); (ii) local Stokes solutions associated with the K subdomains ($\underline{\mathbf{Q}}_k^{-1}$, $k = 1,...,K$); (iii) an approximation to the velocity along the interface Γ between the subdomains ($\underline{\mathbf{Q}}_\Gamma^{-1}$). The operators $\underline{\mathbf{R}}_0$, $\underline{\mathbf{R}}_k$, $k = 1,..,K$ and $\underline{\mathbf{R}}_\Gamma$ denote the restriction operators associated with the coarse problem, the local subdomain problems and the interface problem, respectively; the transpose of these operators denote the corresponding prolongation (or extension) operators.

The coarse solution is obtained by restricting the residual from the original $Q_N - Q_{N-2}$ spectral element mesh to a low-order $Q_2 - P_1$ finite element mesh. In what follows, the coarse mesh has the same number of elements as the fine mesh (i.e., the coarse mesh is induced by the fine mesh). The operator $\tilde{\underline{\mathbf{S}}}_0$ represents the coarse Stokes operator which is inverted by a direct solver. The indefinite Stokes system is made solvable by insisting that the average (global) pressure be zero.

Each local Stokes system assumes homogeneous velocity boundary conditions. Hence, each local Stokes operator, $\tilde{\underline{\mathbf{S}}}_k$, is also indefinite, and the average (local) pressure is assumed to be zero. Each local Stokes system is solved by *explicitly* constructing and factoring the symmetric Uzawa pressure operator *locally* within each element; only back substitution is thus necessary in order to obtain the pressure within each subdomain. Once the local pressure solution has been computed, each local velocity can be found by solving a *local* system of the form (3); see [Røn96] for more details.

Finally, the velocity along the subdomain interfaces is found simply by inverting the diagonal of the viscous operator, $\underline{\mathbf{A}}$ (which is trivial). The associated extension operator $\underline{\mathbf{R}}_\Gamma^T$ represents an extension by zero to all the interior subdomain nodes; again, for more details, see [Røn96].

Method H1: A hybrid Schwarz/substructuring algorithm

In this and the next section, we present two variants of a preconditioner which we will refer to as a hybrid Schwarz/substructuring method. Note that the word "hybrid" is here used to emphasize the fact that part of the preconditioner is of a non-overlapping type, and part of the preconditioner is of an overlapping type. Using the notation from the previous section, the preconditioner can be expressed as

$$\underline{\mathbf{Q}}^{-1} = \underline{\mathbf{Q}}_0^{-1} + \sum_{k=1}^{K} \hat{\underline{\mathbf{Q}}}_{k,v}^{-1} + \sum_{k=1}^{K} \underline{\mathbf{Q}}_{k,p}^{-1} \tag{7}$$

We see that we no longer have an interface system for the velocity degrees-of-freedom. Instead, we solve K (local) *overlapping* systems for the velocity,

$$\hat{\underline{\mathbf{Q}}}_{k,v}^{-1} = \hat{\underline{\mathbf{R}}}_{k,v}^{T} \hat{\underline{\mathbf{A}}}_{k}^{-1} \hat{\underline{\mathbf{R}}}_{k,v} \, [\underline{\mathbf{I}} \ + \underline{\mathbf{D}}^{T} (\sum_{k=1}^{K} \underline{\mathbf{Q}}_{k,p}^{-1})] \quad k = 1, ..., K. \tag{8}$$

Here, $\hat{\underline{\mathbf{R}}}_{k,v}$ is an operator which restricts the residual of the momentum equations to the nodes associated with $\overline{\Omega}_k$ *plus* one layer of nodes outside this subdomain; we denote this extended subdomain as $\overline{\Omega}_k'$. Hence, for this preconditioner, we consider the minimum-overlapping case for the velocity *only*. Next, an approximation of the discrete viscous operator is constructed based upon a triangularization/tetrahedrazation of the Gauss-Lobatto-Legendre nodes associated with the overlapping subdomain $\overline{\Omega}_k'$; linear finite elements are used within each triangle/tetrahedron. The resulting local systems $\hat{\underline{\mathbf{A}}}_k$, $k = 1, .., K$ are then assembled and factored. Hence, only back substitution is required in order to obtain the local velocities during each preconditioning step. (Note that it is sufficient to only assemble and factor the *scalar* Laplacian associated with each extended subdomain.)

The restricted residual for the local velocity problem consists of two contributions: the original residual from the momentum equations plus the gradient operator $\underline{\mathbf{D}}^T$ acting on the local (discontinuous) pressure solutions; see (8). A natural way to motivate this approach is to consider (3) with $\underline{\mathbf{f}}$ playing the role of the original momentum residual, and the pressure $\underline{p}$ playing the role of the local pressure solutions.

An important observation here is the fact that we have abandoned the framework of carrying along the "full" Stokes operator when going from the original, global operator to the operators associated with the local subproblems. Here, a non-overlapping method is used in order to obtain the local pressure solutions, while an overlapping method is used in order to compute the local velocities.

Method H2: A hybrid Schwarz/substructuring algorithm

The next variant of the hybrid Schwarz/substructuring preconditioner can be expressed as

$$\underline{\mathbf{Q}}^{-1} = \underline{\mathbf{Q}}_0^{-1} + \underline{\mathbf{Q}}_v^{-1} + \sum_{k=1}^{K} \underline{\mathbf{Q}}_{k,p}^{-1} \tag{9}$$

where

$$\underline{\mathbf{Q}}_v^{-1} = \underline{\mathbf{A}}^{-1} [\underline{\mathbf{I}} \ + \underline{\mathbf{D}}^{T} (\sum_{k=1}^{K} \underline{\mathbf{Q}}_{k,p}^{-1})] \ . \tag{10}$$

In this case, we abandon the process of computing local velocity solutions associated with the subdomains. Instead, we imagine a process in which we extend each overlapping subdomain to cover the entire domain. One way to look at this variant is as an extreme case of an overlapping method. By comparing Method H1 and H2, we can compare the case of minimum and "maximum" overlap. Any reasonably constructed variant between these two cases could be expected to have a performance (in terms of conditioning) which is between the performance of Method H1 and Method H2.

Similar to the Uzawa method, any good preconditioner for the viscous operator (in this case, the vector Laplacian) can be used to invert $\underline{\mathbf{A}}$.

Numerical Results

In this section, we compare the convergence behavior for the different Stokes solvers by solving the driven cavity problem. The computational domain Ω is the unit square in two dimensions, and the unit cube in three dimensions. On one of the faces, a unit velocity is imposed. On all the other faces, homogeneous velocity boundary conditions are imposed. The extension from homogeneous velocity boundary conditions to non-homogeneous boundary conditions is straight-forward; the previous discussion does not change. All the results were performed in double precision on an SGI Indigo 2 workstation.

For the Uzawa algorithm, a nested conjugate gradient iteration is used. The numerical results reported below indicate the number of *outer* CG iterations required in order to reduce the initial residual in (2) with five orders of magnitude.

For the three domain decomposition solvers, the numerical results indicate the number of GCR iterations necessary in order to reduce the initial residual with five orders of magnitude. For these solvers, the associated coarse grid is based upon Q_2-P_1 finite elements.

In Table 1 and Table 2, we report the two-dimensional results. The total number of degrees-of-freedom, $\mathcal{N}_{d.o.f.}$, is also indicated together with the total number of degrees-of-freedom on the coarse grid, $\mathcal{M}_{d.o.f.}$. Table 1 shows how the number of iterations varies when varying the order N of each spectral element. Here, $K = 4 \times 4 = 16$ equal spectral elements (or subdomains) are used. The results in Table 1 are also plotted in Figure 1.

The results in Table 1 show that the convergence rate for Method U and Method H2 depends very weakly upon the order of each spectral element, in particular for higher values of N. The results are consistent with the results in [MMPR93] for the Uzawa algorithm, where it is found that the inf-sup parameter depends only weakly upon N in two dimensions. Thus, these results indicate that Method H2 also has a very weak dependence upon N.

The results for Method H1 are similar to Method S for $N \leq 10$. However, for $N > 10$, the number of iterations for Method S appears to grow linearly with N, while the growth rate is sublinear for Method H1. Note that Method H1 corresponds to a minimum overlap for the velocity degrees-of-freedom, and that the ratio $H/\delta \sim \mathcal{O}(N^2)$ as $N \Rightarrow \infty$. Hence, based upon the estimate (4), one might expect the iteration count to grow linearly with N for Method H1. It appears that this estimate is somewhat pessimistic, at least for $N \leq 20$.

The results in Table 2 are obtained by keeping the order of the elements fixed to $N = 10$, while using 2 different decompositions, $K = 4 \times 4 = 16$ and $K = 8 \times 8 = 64$. As expected, the convergence rate does not depend upon the number of subdomains, K.

In Table 3, we report the corresponding three-dimensional results. The decomposition we have used corresponds to $K = 4 \times 4 \times 4 = 64$. These results are also plotted in Figure 2. We do not yet have results for Method H1 in R^3. Because of the rapidly increasing problem size with N in R^3, we have not been able to use very high values of N. Hence, we have probably not seen the asymptotic behavior of the methods in R^3; see [MMPR93]. However, even these limited results indicate that the number of iterations does not grow faster than linearly with the polynomial degree N for any of the methods (at least for $N \leq 8$).

TABLE 1

N	U	H2	H1	S	$\mathcal{N}_{d.o.f.}$	$\mathcal{M}_{d.o.f.}$
4	10	14	23	22	593	145
6	11	18	28	27	1,457	145
8	11	20	33	32	2,705	145
10	11	22	36	38	4,337	145
12	12	23	38	43	6,353	145
14	12	23	40	50	8,753	145
16	13	24	44	57	11,537	145
18	14	24	49	61	14,705	145
20	14	25	53	69	18,257	145

TABLE 2

K	U	H2	H1	S	$\mathcal{N}_{d.o.f.}$	$\mathcal{M}_{d.o.f.}$
16	11	22	36	38	4,337	145
64	11	22	35	40	17,665	641

TABLE 3

N	U	H2	S	$\mathcal{N}_{d.o.f.}$	$\mathcal{M}_{d.o.f.}$
3	15	17	26	4,504	1,284
4	15	22	33	11,852	1,284
5	16	25	39	24,672	1,284
6	18	29	46	44,500	1,284
7	19	32	52	72,872	1,284
8	21	35	60	111,324	1,284

Discussion

We now discuss the computational complexity for the Stokes solvers. From the numerical results for the driven cavity problem, we notice that the minimum number

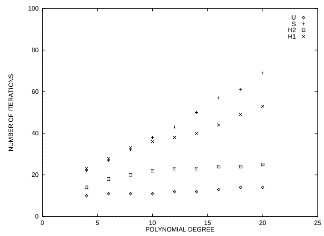


Figure 1 Two-dimensional convergence results

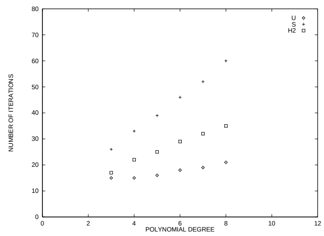


Figure 2 Three-dimensional convergence results

of iterations is achieved for the Uzawa algorithm, both in two and three dimensions. On the other hand, each CG pressure iteration in the Uzawa algorithm involves the inversion of the multi-dimensional viscous operator, as well as global matrix-vector products involving the gradient and divergence operator. The dominant work per iteration is associated with solving the viscous system; the work associated with preconditioning the Uzawa pressure operator is small.

For all the domain decomposition methods, each GCR iteration involves two matrix-vector products with the full Stokes operator. Note that, in contrast to the Uzawa algorithm, the viscous operator here enters as part of the matrix-vector *product*, and not as an operator that needs to be *inverted*. The preconditioning step involves the solution of a coarse Stokes system (effected by a back substitution), as well as the solution of local Uzawa pressure operators (effected by a set of K back substitutions). In addition, for Method S and Method H1, a set of K local viscous systems are solved (effected by a set of $d \cdot K$ back substitutions). For Method S, this corresponds to local, interior solves, while for Method H1, a minimum overlap is used. For Method H2, however, a global viscous system is solved in each preconditioning step.

The number of iterations for Method H2 is almost twice the number of iterations for Method U for larger values of N. Hence, the cost of Method H2 is about 4 times the cost of Method U. Both methods can be regarded as examples of nested iterative methods, with Method U representing a nested CG/CG iteration, and Method H2 representing a nested GCR/CG iteration.

The above comparison of Method U and Method H2 definitely favors the Uzawa method for a standard Stokes problem. However, there are several reasons why Method H2 is interesting. First of all, the results for Method H2 (as well as for Method H1) demonstrates that it is, in fact, possible to separate the treatment of the pressure and the velocity when constructing the Stokes preconditioner. This observation gives rise to an added flexibility.

Second, Method H2 demonstrates that close to optimal convergence rates may be achieved using high-order spectral elements. Even though no theory yet exists for this method, it is doubtful that the convergence rate can be expected to be better than the convergence rate for Method U (i.e., the dependence on the inf-sup parameter may be the limiting factor).

If the number of inner CG iterations in the Uzawa algorithm is higher than about ten, Method S and Method H1 will have the lowest computational cost of all the Stokes solvers tested here; this will, indeed, be the typical case. The computational cost associated with Method S and Method H1 is roughly the same.

In the context of using low-order finite element methods, the results presented in this paper, combined with those in [KPar], offer strong evidence that low-order methods, with a separate velocity and pressure treatment, can yield excellent convergence rates. More numerical results are necessary in order to determine whether a separate treatment of the velocity and the pressure (i.e., an overlapping approach for the velocity and a non-overlapping method for the pressure) is better than associating the "full" Stokes operator on each extended subdomain; see [KPar, KP98].

Finally, we mention that for all of the domain decomposition methods we have considered here, the coarse grid is *essential* in order to obtain the correct solution. This is due to the fact that the local pressure solutions are based upon a *non-overlapping* approach, and a zero mean is enforced in each subdomain. Thus, only the coarse system will provide the correct pressure levels in each subdomain.

REFERENCES

[Cas96] Casarin M. (1996) *Schwarz preconditioners for spectral and mortar finite element methods with applications to incompressible fluids.* PhD thesis, New York University, Courant Institute of Mathematical Sciences.

[Cas97] Casarin M. (1997) Quasi-optimal Schwarz methods for the conforming spectral element discretization. *SIAM J. Numer. Anal.* 34(6): 2482–2502.

[DW94] Dryja M. and Widlund O. B. (May 1994) Domain decomposition algorithms with small overlap. *SIAM J. Sci. Comput.* 15(3): 604–620.

[Elm96] Elman H. C. (1996) Multigrid and Krylov subspace methods for the discrete Stokes equations. *Int. J. Numer. Meth. in Fluids* 22: 755–770.

[Fis97] Fischer P. F. (1997) An overlapping Schwarz method for spectral element solution of the incompressible Navier-Stokes equations. *J. Comp. Phys.* 133: 84–101.

[FMTar] Fischer P. F., Miller N. I., and Tufo H. M. (to appear) An overlapping Schwarz method for spectral element simulation of three-dimensional incompressible flows. In Bjørstad P. and Luskin M. (eds) *Parallel Solution of Partial Differential Equations*, IMA Volumes in Mathematics and its Applications. Springer-Verlag, New York.

[KP98] Klawonn A. and Pavarino L. F. (August 1998) A comparison of overlapping Schwarz methods and block preconditioners for saddle point problems. Technical Report 15/98-N, Westfälische Wilhelms-Universität

Münster, Germany.

[KPar] Klawonn A. and Pavarino L. F. (to appear) Overlapping Schwarz methods for mixed linear elasticity and Stokes problems. *Comp. Meth. Appl. Mech. Engrg.* .

[MMPR93] Maday Y., Meiron D., Patera A., and Rønquist E. (1993) Analysis of iterative methods for the steady and unsteady Stokes problem: Application to spectral element discretizations. *SIAM J. Sci. Stat. Comput.* 14(2): 310–337.

[MP89] Maday Y. and Patera A. T. (1989) Spectral element methods for the Navier-Stokes equations. In Noor A. (ed) *State of the Art Surveys in Computational Mechanics*, pages 71–143. ASME, New York.

[MPR92] Maday Y., Patera A. T., and Rønquist E. M. (1992) The $P_n \times P_{N-2}$ method for the approximation of the Stokes problem. Technical Report 92009, Massachusetts Institute of Technology, Dept. of Mechanical Engineering.

[Pah93] Pahl S. S. (December 1993) Schwarz type domain decomposition methods for spectral element discretizations. Master's thesis, Dept. of Computational and Applied Mathematics, Univ. of the Witwatersrand, Johannesburg, South Africa.

[Pav97] Pavarino L. F. (1997) Preconditioned conjugate residual methods for mixed spectral element discretizations of elasticity and Stokes problems. *Comput. Meth. Appl. Mech. Engrg.* 146: 19–30.

[Pavar] Pavarino L. F. (to appear) Domain decomposition methods with small overlap for Q_n-Q_{n-2} spectral elements. In *Proc. Fourth Int. Conf. on Spectral and High Order Methods.* Conference held in Israel, 1998.

[PW97] Pavarino L. F. and Widlund O. B. (December 1997) Iterative substructuring methods for spectral element discretizations of elliptic systems: Mixed methods for linear elasticity and Stokes flow. Technical Report 755, Dept. of Computer Science, Courant Institute of Mathematical Sciences. To appear in SIAM J. Numer. Anal.

[Røn96] Rønquist E. M. (1996) A domain decomposition solver for the steady Navier-Stokes equations. In Ilin A. V. and Scott L. R. (eds) *Proc. Third Int. Conf. on Spectral and High Order Methods.* Houston Journal of Mathematics. Conference held in Houston, Texas, 1995.

[Røn98] Rønquist E. M. (1998) A domain decomposition solver for three-dimensional steady free surface flows. In Bjørstad P. E., Espedal M. S., and Keyes D. E. (eds) *Proc. Ninth Conf. on Domain Decomposition Methods.* Domain Decomposition Press. Conference held in Ullensvang, Norway, 1996.

[SBG96] Smith B. F., Bjørstad P. E., and Gropp W. (1996) *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations.* Cambridge University Press.

37

Domain Decomposition Methods for Parabolic Problems

A.A.SAMARSKII [1], & P.N.VABISHCHEVICH [2]

INTRODUCTION

Parallel algorithms for multidimensional problems in mathematical physics are designed now on the basis of domain decomposition methods. The original problem is divided into a set of subproblems. Either problem is solved on its own processor (its own elementary computer) and in its own subdomain. The scope for such an approach with an approximate solution of non-stationary problems of mathematical physics is discussed. The main attention is paid for noniterative variants of the domain decomposition method with various of interchanging boundary conditions.

The present investigation is directed to a review and analysis of works on methods of domain decomposition for parabolic initial-boundary value problems. In constructing domain decomposition methods for time-dependent problems there are employed the following approaches.

- The first [Kuz88, Tal94] is based on using classic implicit schemes and involves domain decomposition methods in order to solve an elliptic grid problem at new time level.
- In the second approach peculiarities of transient problems are taken into account in more details. The corresponding region-additive schemes are investigated in various variants in works [Dry90, Lae90, Lae92, SV96,

1 Institute for Mathematical Modeling of Russian Academy of Sciences, 4 Miusskaya Square, 125047 Moscow, Russia email: samarskii@imamod.msu.su
2 Institute for Mathematical Modeling of Russian Academy of Sciences, 4 Miusskaya Square, 125047 Moscow, Russia email: vab@ibrae.ac.ru
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

SV95b, Vab89, Vab94b].

- There are [DD92] constructed decomposition schemes with a special approximation of exchange boundary conditions (inhomogeneous schemes).
- Parallel variants of standard splitting schemes (for example, [JSS87]): schemes of alternating directions, factorized schemes, local one-dimensional schemes of component-wise splitting) are being designed.

The main attention is paid to iterative-free variants of a decomposition method, i.e. regional-additive schemes. These algorithms adequately account for a specific character of non-stationary problems when the transition to a new time level is connected with the solution of a set of separate problems in subdomains. Theoretical analysis of domain decomposition schemes is done with the use of standard and new splitting schemes [Abr90, Vab94a].The study is based on the modern theory of stability and convergence of operation-difference splitting schemes [Sam89, SG73, SV95a].

MODEL PROBLEM

A two-dimensional rectangular domain Ω with parallel to coordinates sides is considered. The solution of the parabolic equations is sought in the domain Ω:

$$\frac{\partial u}{\partial t} - \sum_{\alpha=1}^{2} \frac{\partial}{\partial x_\alpha}(k_\alpha(x)\ \frac{\partial u}{\partial x_\alpha}) = f(x), \quad x = (x_1, x_2) \in \Omega, \quad t > 0. \tag{1}$$

Equation (1) is supplemented by the homogeneous boundary condition (the Dirichlet problem)

$$u(x,t) = 0, \quad x \in \partial\Omega, \quad t > 0 \tag{2}$$

and the initial condition

$$u(x,0) = u_0(x), \quad x \in \Omega. \tag{3}$$

Let us introduce in the domain Ω the uniform grid x_α with the uniform spacing $h_\alpha, \alpha = 1,2$. Let ϖ be the set of the internal points. An approach to more common problems in context of this paper is of editing character. Let us define on the set of functions $y \in H$ such as $y(x) \equiv 0, x \notin \varpi$, a difference operator A:

$$Ay = \sum_{\alpha=1}^{2} \Lambda_\alpha, \tag{4}$$

$$\Lambda_\alpha = -(a_\alpha(x) y_{\bar{x}})_x, \quad x \in \varpi.$$

Here the standard index-free notation of the difference scheme theory [Sam89]–[SV95a] is used, for example:

$$w_{\bar{x}} = \frac{w(x) - w(x-h)}{h}, \quad w_x = \frac{w(x+h) - w(x)}{h},$$

$$a_1(x) = 0,5(k_1(x) + k_1(x_1 - h_1, x_2)),$$

$$a_2(x) = 0,5(k_2(x) + k_2(x_1, x_2 - h_2)).$$

Let us introduce a scalar product and a norm in the Hilbert space H as follows

$$(y, v) = \sum_{x \in \varpi} yvh_1h_2, \quad \|y\| = \sqrt{(y, y)}.$$

Note, that in H operator A is self-adjoint and positive definite [Sam89]–[SV95a], i.e. $A = A^* > 0$.

From equations (1)-(3) for the given $y(x, 0)$, $x \in \varpi$ we pass to the following equation

$$\frac{dy}{dt} + Ay = 0, \quad x \in \varpi. \tag{5}$$

For the last equation difference schemes of domain decomposition schemes are constructed. Numerical implementation of these schemes is based on the solution of problems in separate subdomains of the calculation domain Ω at every time-level.

DOMAIN DECOMPOSITION

Let domain Ω consists from m separate subdomains:

$$\Omega = \Omega_1 \cup \Omega_2 \cup ... \cup \Omega_p.$$

Designing of regional-additive difference schemes is based on a special additive representation of A operator of considered equation (5) and on the application of one or another splitting schemes. The choice of a splitting operator and a splitting scheme corresponds to the choice of a definite scheme of computations in separate subdomains. In particular, to the choice of interchange boundary conditions on the boundary of sub-domains.

Let ϖ_α be the subsets of points ϖ, lying in subdomains Ω_α, $\alpha = 1, 2, ..., p$. Let us construct the decomposition difference schemes similar to presented in [Lae90] on the basis of the unit splitting for domain Ω. Let us define the following functions

$$\chi_\alpha(x) = \begin{cases} > 0, & x \in \Omega_\alpha, \\ 0, & x \notin \Omega_\alpha, \end{cases} \quad \alpha = 1, 2, \tag{6}$$

where

$$\sum_{\alpha=1}^{p} \chi_\alpha(x) = 1, \quad x \in \Omega. \tag{7}$$

We shall consider the class of decomposition schemes, where for the operator A the following additive representation takes place:

$$A = \sum_{\alpha=1}^{p} A_\alpha, \tag{8}$$

where operators A_α, $\alpha = 1, 2, ..., p$ are associated with isolated subdomains and also with splitting (6),(7) and with the solution of the individual subproblems in subdomains Ω_α, $\alpha = 1, 2, ..., p$.

The simplest difference decomposition scheme is defined via the definition of operators $A_\alpha, \alpha = 1, 2, ..., p$ in the following way [Vab89, Vab94b]:

$$A_\alpha = \chi_\alpha A, \quad \alpha = 1, 2, ..., p. \tag{9}$$

The following presentation for the decomposition operators can be used:

$$A_\alpha = A\chi_\alpha, \quad \alpha = 1, 2, ..., p. \tag{10}$$

Clearly, that for such splitting operator A is not selfadjoint, i.e.$A_\alpha \neq A_\alpha^*, \alpha = 1, 2, ..., p$.

It is naturally to use in this case the symmetrical splitting (7),(8) (see, for example, [Dry90, Lae90], where

$$A_\alpha y = -\sum_{\alpha=1}^{2} (a_\beta^\alpha(x) y_{\overline{x}_\beta})_{x_\beta}, \quad x \in \varpi, \quad \alpha = 1, 2, ..., p. \tag{11}$$

Grid operators $A_\alpha, \alpha = 1, 2, ..., p.$ are approximated via difference degenerating elliptic operators

$$-\sum_{\beta=1}^{2} \frac{\partial}{\partial x_\beta}(k_\beta \chi_\alpha(x) \ \frac{\partial u}{\partial x_\beta}), \quad \alpha = 1, 2, ..., p,$$

where coefficients a_β^α are defined as a_β. With expressions (7),(8),(11) we obtain $A_\alpha = A_\alpha^* \geq 0, \quad \alpha = 1, 2, ..., p$.

REGION-ADDITIVE SCHEMES

At first, simplest case of decomposition of domain Ω by means of the straight lines $x_1 =$ const will be demonstrated. In this case while constructing of the difference schemes we can be oriented to the usage of the difference splitting schemes with $p = 2$, where Ω_1 and Ω_2 are defined as a combination of corresponding subdomains. When splitting into two operators, classical alternating direction schemes would be appropriate for transition from the previous time level to the next one.

Unconditionally stable difference schemes for the solution of equation (6) with the corresponding initial condition are easily constructed via splitting (7),(8) with selfadjoint and positive definite operators $A_\alpha, \alpha = 1, 2$ (with decomposition operator (11)). The accuracy problem of the approximate solution, its dependence on the width of subdomain overlapping and also on functions $\chi_\alpha(x), \alpha = 1, 2$ is of special interest. Usage of the schemes with the asymmetric decomposition operators requires individual investigation.

Let us select among the unconditionally steady factorized schemes the scheme of the stabilizing correction similar to the classical Douglas-Rechford scheme. Let y^n be the difference solution at the time moment $t^n = n\tau$, where $\tau > 0$ is the time-step. The transition from the previous time-level to the next one is performed in accordance with the following expressions

$$\frac{y^{n+1/2} - y^n}{\tau} + A_1 y^{n+1/2} + A_2 y^n = 0,$$

$$\frac{y^{n+1} - y^{n+1/2}}{\tau} + A_2(y^{n+1} - y^n) = 0. \qquad (12)$$

For the difference decomposition scheme (8),(12) with the selection of operators $A_\alpha, \alpha = 1,2$ according to (9), (10) and (11) the following estimate of stability in respect to the initial data is valid:

$$\left\|(E + \tau A_2)y^{n+1}\right\|_D \leq \left\|(E + \tau A_2)y^0\right\|_D,$$

where $D = A, A^{-1}$ and E respectively.

Investigation of convergence of decomposition method leads to the following estimates

$$\left\|(E + \tau A_2)z^{n+1}\right\|_D \leq M((1 + \|\chi_2\|_A)\tau + |h|^2). \qquad (13)$$

Moreover, accuracy of decomposition schemes depends on the width of subdomains overlapping Ω_1 and Ω_2 (see term at $\|\chi_2\|_A$ in the estimate (13)).

While constructing difference schemes for the parallel computer we should be oriented to the decomposition with minimum overlapping of domains, i.e. on the minimization of exchange between individual processors. At minimal overlapping of subdomains (width of subdomains overlapping equals to $O(|h|)$) we have from the estimate (13) that the convergence rate is $O(|h|^{-1/2}\tau + |h|^2)$.

Under a more general domain decomposition grid operator A in equation (5) has the form (8) with a number of operators $p > 2$. For such problems difference schemes of summarized approximation [Sam89, SV95a] have some advantages. Investigation of difference schemes of summarized approximation shows that these schemes have low accuracy of spatial approximation.

For instance, for the following fully implicit scheme of multicomponent splitting

$$\frac{y^{n+\alpha/p} - y^{n+(\alpha-1)/p}}{\tau} + A_\alpha y^{n+\alpha/p} = 0, \quad \alpha = 1, 2, ..., p \qquad (14)$$

the error estimate has the form

$$\left\|z^{n+1}\right\| \leq M((1 + \sum_{\alpha=1}^{p} \|D\chi_\alpha\|)\tau + |h|^2). \qquad (15)$$

As for parallel implementation, additive-averaged schemes of domain decomposition [18] should be mentioned separately. For example, the implicit scheme has the following form (compare with (14))

$$\frac{\tilde{y}^{n+\alpha/p} - y^n}{\tau} + \chi_\alpha A\tilde{y}^{n+\alpha/p} = 0, \quad \alpha = 1, 2, ..., p,$$

$$y^{n+1} = \frac{1}{p}\sum_{\alpha=1}^{p} \tilde{y}^{n+\alpha/p}$$

and for the error we have estimate (15).

The principal moment here is concerned with the possibility to calculate $\tilde{y}^{n+\alpha/p}, \alpha = 1, 2, ..., p$ independently (asynchronously).

To construct parallel numerical methods, it seems to be more suitable to use vector additive difference schemes with full approximation, besides these numerical schemes are unconditionally stable at any p [Abr90, Vab94a]. Let's define vector $Y = y_1, y_2, ..., y_p$, the following set of equations for calculation of each component of this vector should be solved:

$$\frac{dy_\alpha}{dt} + \sum_{\beta=1}^{p} A_\beta y_\beta = 0, \tag{16}$$

$$y_\alpha(0) = y_0, \quad \alpha = 1, 2, ..., p. \tag{17}$$

From equations (16),(17) it follows that $y_\alpha(t) = y(t)$, then arbitrary component of vector $Y(t)$ may be chosen as a solution of the main problem for equation (5).

The following scheme is an example of unconditionally stable schemes for a set of equations (16),(17)

$$(E + \sigma\tau A_\alpha)\frac{y_\alpha^{n+1} - y_\alpha^n}{\tau} + \sum_{\beta=1}^{p} A_\beta y_\beta^n = 0,$$

$$\alpha = 1, 2, ..., p$$

for case $\sigma \geq p/2$. Realization of this scheme is connected with the inversion of operators $E + \sigma\tau A_\alpha, \quad \alpha = 1, 2, ..., p$ at every time-level in just the same way as in case with general (scalar) difference additive schemes. These schemes may be considered like difference schemes with weights where the weight σ is larger than unit ($\sigma > 1$).

For the accuracy of these vector schemes of domain decomposition there are the same estimates [SV95b] as for standard schemes of summarized approximation. However, for this class of schemes it is easy to construct schemes with the second order in time.

ACKNOWLEDGEMENT

This work was supported by the Russian Foundation for Fundamental Research, through grant N 98-01-00335.

REFERENCES

[Abr90] Abrashin V. N. (1990) On a variant of alternating direction method for the numerical solution of multidimensional problems of mathematical physics.i. *Differentsial'nye Uravneniya* 26: 314–323. In Russian.

[DD92] Dawson C. N. and Dupont T. F. (1992) Explicit/implicit, conservative, galerkin domain decomposition procedures for parabolic problems. *Math. Comp* 58: 21–34.

[Dry90] Dryja M. (1990) Substructuring methods for parabolic problems. In *IV Intern. Simp. on Domain Decomposition Methods for Partial Differential Equations*, pages 264–271. SIAM.

[JSS87] Johnson S. L., Saad V., and Schultz M. H. (1987) Alternating direction methods on multiprocessors. *SIAM J.Sci. Stat. Comput.* 8: 686–700.

[Kuz88] Kuznetsov Y. A. (1988) New algorithms for approximate realization of implicit difference schemes. *Sov. J. Numer. Anal. and Math. Model.* 3: 99–114.

[Lae90] Laevsky Y. M. (1990) Quadratic elements in splitting methods. *Sov. J. Numer. Anal. and Math. Model* 5: 244–249.

[Lae92] Laevsky Y. M. (1992) On one algorithm of domain decomposition without the overlapping of subdomains for the solution of parabolic equations. *Zh. Vychisl. Mat. and Mat. Fiz.* 32: 1744–1755. In Russian.

[Sam89] Samarskii A. A. (1989) *Theory of Difference Schemes.* Nauka, Moscow, 3 edition. In Russian.

[SG73] Samarskii A. A. and Gulin A. V. (1973) *Stability of Difference Schemes.* Nauka, Moscow. In Russian.

[SV95a] Samarskii A. A. and Vabishchevich P. N. (1995) *Computational Heat Transfer*, volume 1,2. Wiley, Chicheste.

[SV95b] Samarskii A. A. and Vabishchevich P. N. (1995) Vector additive schemes of domain decomposition for parabolic problems. *Differentsial'nye Uravnenija* 31: 1563–1569. In Russian.

[SV96] Samarskii A. A. and Vabishchevich P. N. (1996) Factorized region-additive schemes for problems of convection-diffusiion. *Doklady Akademii Nauk* 336: 742–745. In Russian.

[Tal94] Tallec P. L. (1994) Domain decomposition methods in computational mechanics. *Computational Mechanics Advances* 1: 121–220.

[Vab89] Vabishchevich P. N. (1989) Finite difference schemes with decomposition of the computational domain for the solution of nonstationary problems. *Zh. Vychisl. Mat. and Mat. Fiz* 29: 1822–1829. In Russian.

[Vab94a] Vabishchevich P. N. (1994) On a class of vector additive difference schemes. *Izvestija Vuzov. Matematika* (9): 11–15. In Russian.

[Vab94b] Vabishchevich P. N. (1994) Parallel domain decomposition algorithms for time-dependent problems of mathematical physics. In *Advances in Numerical Methods and Applications*, pages 293–299. World Scientific.

38

An Asynchronous Space Decomposition Method

Xue-Cheng Tai, [1] Paul Tseng [2]

Introduction

With the advent of multiprocessor computing systems, there has been much work in the design and analysis of iterative methods that can take advantage of the parallelism to solve large algebraic problems. In these methods, the computation per iteration is distributed over the processors and each processor communicates the result of its computation to the other processors, possibly subject to communication or computation delays. By using a model of asynchronous computation proposed by Chazan and Miranker [CM69], these methods have been analyzed quite extensively (see [BT89]and references therein). However, aside from the easy case where the algorithmic mapping is a contraction with respect to the L^{∞}-norm, there has been few studies of the convergence rate of these methods.

In this paper, we study the convergence rate of asynchronous block Jacobi and block Gauss-Seidel methods for finite or infinite dimensional convex minimization of

1 Department of Mathematics, University of Bergen, Johannes Brunsgate 12, 5007, Bergen, Norway. Email: Tai@mi.uib.no and URL: http://www.mi.uib.no/˜tai.
2 Department of Mathematics, Box 354350, University of Washington, Seattle, WA 98195, U.S.A. Email: tseng@math.washington.edu and URL: http://www.math.washington.edu/˜tseng.
The work of the first author was supported by the Engineering and Physical Sciences Research Council, United Kingdom, Grant Reference GR/L98862. The work of the second author was supported by the National Science Foundation, Grant No. CCR-9311621.
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

the form

$$\min_{v_i \in K_i, i=1,\ldots,m} F\left(\sum_{i=1}^{m} v_i\right), \tag{1}$$

where each K_i is a nonempty closed convex set in a real reflexive Banach space V and F is a real-valued lower semicontinuous Gâteau-differentiable function that is strongly convex on $\sum_{i=1}^{m} K_i$. Our interest in these methods stems from their close connection to relaxation methods for nonlinear network flow (see [BCEZ95], [BT89], [TBT90] and references therein) and to domain decomposition (DD) and multigrid (MG) methods for solving elliptic partial differential equations (see [CM94], [DH97], [SBG96], [TE98], [TX98], [Xu92] and references therein). For example, the additive and the multiplicative Schwarz methods may be viewed as block Jacobi and block Gauss-Seidel methods applied to linear elliptic partial differential equations reformulated as (1). DD and MG methods are also useful as preconditioners and it can be shown that such preconditioning improves the condition number of the discrete approximation [BZar], [CM94], [SBG96], [Xu92]. In addition, DD and MG methods are well suited for parallel implementation, for which both synchronous and asynchronous versions have been proposed. Of the work on asynchronous methods [BMPSar], [BMPS95], [FSS97], [McC89], we especially mention the numerical tests by Frommer et al. [FSS97] which showed that, through improved load balancing, asynchronous methods can be advantageous in solving even simple linear equations. Although these tests did not use the coarse mesh in its implementation of the DD method, it is plausible that the asynchronous method would still be advantageous when the coarse mesh is used. An important issue concerns the convergence and convergence rate of the above methods. In the case where the equation is linear (corresponding to F being quadratic and $K_1, ..., K_m$ being suitable subspaces of V) or almost linear, this has been much studied for synchronous methods such as block Jacobi and block Gauss-Seidel methods (see [BZar], [CM94], [SBG96], [Xu92] and references therein) but little studied for asynchronous methods [BMPSar], [BMPS95], [McC89]. In the case where the equation is nonlinear (corresponding to $K_1, ..., K_m$ being suitable subspaces of V), there are some convergence studies for synchronous methods [CSar], [DH97], [Sha97], [TE98], [TX98], and none for asynchronous methods. In the case where $K_1, ..., K_m$ are not all subspaces, there are some convergence studies for synchronous methods and, in particular, block Jacobi and Gauss-Seidel methods (see [Kor97], [LT92], [LT93], [Tai92], [Tai95], etc.) but none for asynchronous methods.

Our contributions are two-fold. First, we consider an asynchronous version of block Jacobi and block Gauss-Seidel methods for solving (1), and we show that, under a Lipschitzian assumption on the Gâteau derivative F' and a norm equivalence assumption on the product of $K_1, ..., K_m$ and their sum (see (5) and (6)), this asynchronous method attains global linear rate of convergence with a convergence factor that can be explicitly estimated (see Theorem 1). This provides a unified convergence and convergence rate analysis for such asynchronous methods. Second, we apply the above convergence result to linearly constrained convex programs and, in particular, nonlinear network flow problems. This yields convergence rate results for some asynchronous network relaxation methods (see §38). We also apply the above convergence result to certain nonlinear elliptic partial differential equations. This yields convergence rate results for some parallel DD and MG methods applied to these equations and, in particular, the convergence factor is shown not to depend on the mesh parameters (see §38). Although alternative approaches such as Newton-type methods have been applied to develop synchronous DD and MG methods for nonlinear partial differential equations [BR82], [Bra77], [HR89], [Xu96], these methods use the traditional DD and MG approach or use a special two-grid treatment.

Problem Description and Space Decomposition

Let V be a real reflexive Banach space with norm $\|\cdot\|$ and let V' be its dual space, i.e., the space of all real-valued linear continuous functionals on V. The value of $f \in V'$ at $v \in V$ will be denoted by $\langle f, v\rangle$, i.e., $\langle\cdot,\cdot\rangle$ is the duality pairing of V and V'. We wish to solve the following minimization problem

$$\min_{v \in K} F(v) \ , \tag{2}$$

where K is a nonempty closed (in the strong topology) convex set in V and $F : V \mapsto \Re$ is a lower semicontinuous convex Gâteau-differentiable function. We assume F is strongly convex on K or, equivalently, its Gâteau derivative $\lim_{t\to 0}(F(v+tw) - F(v))/t$, which is a well-defined linear continuous functional of w denoted by $F'(v)$ (so $F' : V \mapsto V'$), is strongly monotone on K, i.e.,

$$\langle F'(u) - F'(v), u - v\rangle \geq \sigma\|u - v\|^2, \quad \forall u, v \in K, \tag{3}$$

where $\sigma > 0$. It is known that, under the above assumptions, (2) has a unique solution $\bar{u}$ [GT89].

We assume that the constraint set K can be decomposed as the Minkowski sum:

$$K = \sum_{i=1}^{m} K_i, \tag{4}$$

for some nonempty closed convex sets K_i in V, $i = 1, ..., m$. This means that, for any $v \in K$, we can find $v_i \in K_i$, not necessarily unique, satisfying $\sum_{i=1}^m v_i = v$ and, conversely, for any $v_i \in K_i,\ i = 1, ..., m$, we have $\sum_{i=1}^m v_i \in K$. Following Xu [Xu92], we call (4) a space decomposition of K, with the term "space" used loosely here. Then we may reformulate (2) as the minimization problem (1), with $(\bar{u}_1, ..., \bar{u}_m)$ being a solution (not necessarily unique) of (1) if and only if $\bar{u}_i \in K_i$ for $i = 1, ..., m$ and $\sum_{i=1}^m \bar{u}_i = \bar{u}$. As was noted earlier, the reformulated problem (1) is of interest because methods such as DD and MG methods may be viewed as block Jacobi and block Gauss-Seidel methods for its solution. The method we study will be an asynchronous version of these methods. The above reformulation was proposed in [Xu92] (for the case where F is quadratic and $K = V$) to give a unified analysis of DD and MG methods for linear elliptic partial differential equations. The general case was treated in [Tai92], [Tai95] (also see [Tai94], [TE98] for the case of $K = V$).

For the above space decomposition, we will assume that there is a constant $C_1 > 0$ such that for any $v_i \in K_i, i = 1, ..., m$, there exists $\bar{u}_i \in K_i$ satisfying

$$\bar{u} = \sum_{i=1}^{m} \bar{u}_i \quad \text{and} \quad \left(\sum_{i=1}^{m} \|\bar{u}_i - v_i\|^2\right)^{\frac{1}{2}} \leq C_1 \left\|\bar{u} - \sum_{i=1}^{m} v_i\right\|. \tag{5}$$

See [CM94], [Tai95], [TE98], [Xu92] for similar assumptions. We will also assume F' has a weak Lipschitzian property in the sense that there is a constant $C_2 > 0$ such that

$$\sum_{i=1}^{m}\sum_{j=1}^{m}\langle F'(w_{ij}+u_{ij}) - F'(w_{ij}), v_i\rangle \le C_2\Big(\sum_{j=1}^{m}\max_{i=1,\dots,m}\|u_{ij}\|^2\Big)^{\frac{1}{2}}\Big(\sum_{i=1}^{m}\|v_i\|^2\Big)^{\frac{1}{2}},$$
$$\forall w_{ij}\in K, u_{ij}\in K_j^{\ominus}, v_i\in K_i^{\ominus},\ i,j=1,\dots,m, \tag{6}$$

where we define the set difference $K_i^{\ominus} = \{u - v : u, v \in K_i\} \subset V$. The above assumption generalizes those in [Tai95], [TE98], [TX98] for the case of K_i being a subspace, for which $K_i^{\ominus} = K_i$.

Furthermore, we will paint each of the sets $K_1, \dots, K_m$ one of c colors, with the colors numbered from 1 up to c, such that sets painted the same color $k \in \{1, \dots, c\}$ are orthogonal in the sense that, for all $u \in K$ and $v_i \in K_i^{\ominus}$, $i \in I(k)$,

$$\Big\|\sum_{i\in I(k)} v_i\Big\|^2 = \sum_{i\in I(k)}\|v_i\|^2, \tag{7}$$

$$\Big\langle F'\Big(u + \sum_{i\in I(k)} v_i\Big), \sum_{i\in I(k)} v_i\Big\rangle \le \sum_{i\in I(k)}\langle F'(u+v_i), v_i\rangle, \tag{8}$$

where $I(k) = \{i \in \{1, \dots, m\} : K_i \text{ is painted color } k\}$. See [CM94], [TX98] for similar orthogonal decompositions in the case K_i is a subspace. Thus $I(1), \dots, I(c)$ are disjoint subsets of $\{1, \dots, m\}$ whose union is $\{1, \dots, m\}$ and $I(k)$ comprises the indexes of the sets painted the color k. Although $c = m$ is always a valid choice, in some of the applications that we will consider, it is essential that c be independent of m. In the context of a network flow problem, each set K_i may correspond to a node of the network and sets are painted different colors if their corresponding nodes are joined by an arc. In the context of a partial differential equation defined on a domain $\Omega \subset \Re^d$, each set K_i may correspond to a subdomain of Ω and sets are painted different colors if their corresponding subdomains intersect (see §38, §38).

Remark 1: It can be seen that condition (6) is implied by the following strengthened Cauchy-Schwarz inequality (also see [SBG96], [Xu92] for the case of quadratic F and subspace K_i):

$$\langle F'(w_{ij}+u_{ij}) - F'(w_{ij}), v_i\rangle \le \epsilon_{ij}\|u_{ij}\|\|v_i\|, \quad \forall w_{ij}\in K, u_{ij}\in K_j^{\ominus}, v_i \in K_i^{\ominus},$$

with C_2 being the spectral radius of the symmetric matrix $\mathcal{E} = [\epsilon_{ij}]_{i,j=1}^m$.

Remark 2: For locally strongly convex problems, the constants σ, C_1, C_2 may depend on $u, v, v_i, w_{ij}, u_{ij}$. In this case, the subsequent convergence estimate should be viewed as being local in nature, i.e., it is valid when the iterated solutions lie in a neighborhood of the true solution (see §38).

An Asynchronous Space Decomposition Method

Since F is lower semicontinuous and strongly convex, for each $(u_1, ..., u_m) \in K_1 \times \cdots \times K_m$ and each $i \in \{1, ..., m\}$, there exists a unique $w_i \in K_i$ satisfying

$$F\Big(\sum_{j \neq i} u_j + w_i\Big) \le F\Big(\sum_{j \neq i} u_j + v_i\Big), \quad \forall v_i \in K_i \tag{9}$$

(see [GT89]). Let $\pi_i(u_1, ..., u_m)$ denote this w_i. Then $(\pi_1, ..., \pi_m)$ may be viewed as the algorithmic mapping associated with the block Jacobi method for solving (1). Consider an asynchronous version of the block Jacobi method, parameterized by a stepsize $\gamma \in (0, 1]$ which for simplicity we assume to be fixed, that generates a sequence of iterates $(u_1(t), ..., u_m(t))$, $t = 0, 1, ...$, with $(u_1(0), ..., u_m(0)) \in K_1 \times \cdots \times K_m$ given, according to the updating formula:

$$u_i(t+1) \;=\; u_i(t) + \gamma s_i(t), \quad i = 1, ..., m, \tag{10}$$

where we define

$$s_i(t) = w_i(t) - u_i(t) \quad \text{if} \quad t \in T^i \quad \text{and} \quad s_i(t) = 0 \quad \text{otherwise}, \tag{11}$$

$$\text{with} \quad w_i(t) = \pi_i\left(u_1(\tau_1^i(t)), ..., u_m(\tau_m^i(t))\right), \tag{12}$$

and T^i is some subset of $\{0, 1, ...\}$ and each $\tau_j^i(t)$ is some nonnegative integer not exceeding t. Since each K_i is convex and $\gamma \in (0, 1]$, an induction argument shows that $(u_1(t), ..., u_m(t)) \in K_1 \times \cdots \times K_m$ for all $t = 0, 1, ...$

We will assume that the iterates are updated in a *partially asynchronous* manner [BT89], i.e., there exists an integer $B \ge 1$ such that

$$\{t, t+1, ..., t+B-1\} \cap T^i \neq \emptyset \qquad t = 0, 1, ..., \ \forall i, \tag{13}$$

$$0 \le t - \tau_j^i(t) \le B - 1 \quad \text{and} \quad \tau_i^i(t) = t \qquad \forall t \in T^i, \ \forall i, j. \tag{14}$$

Remark 3: The above asynchronous method models a situation in which computation is distributed over m processors with the ith processor being responsible for updating u_i and communicating the updated value to the other processors. T^i is the set of "times" at which u_i is updated by processor i (by applying π_i to its current copy of $(u_1, ..., u_m)$); $u_i(t)$ is the value of u_i known to processor i at time t; and $\tau_j^i(t)$ is the time at which the value of u_j used by processor i at time t is generated by processor j, so $t - \tau_j^i(t)$ is the communication delay from processor j to processor i at time t. Thus, the processors need not wait for each other when updating $(u_i)_{i=1}^m$, and the values used in the computation may be out-of-date.

Convergence Rate of the Asynchronous Method

Below is our main convergence result, showing that the iterates $(u_1(t), ..., u_m(t))$ generated by the asynchronous method (10)–(14) attain linear rate of convergence, with a factor that depends on σ, C_1, C_2, c and B, γ only. We refer the proof to [TT98].

Theorem 1 *Consider the minimization problem* (2) *and the space decomposition* (4) *of §38 (see* (3), (5)–(8)*). Let* $(u_1(t), ..., u_m(t))$, $t = 0, 1, ...$, *be generated by the asynchronous space decomposition method of §38 (see* (10)–(12) *and* (13), (14)*). Denote* $u(t) = \sum_{j=1}^{m} u_j(t)$. *Then, there exist* $\gamma_0 \in (0,1)$ *and* $\varrho \in (0,1)$, *depending on* σ, C_1, C_2, c *and* B *only, such that when* $\gamma \leq \gamma_0$, *there holds*

$$F(u(nB)) - F(\bar{u}) \leq \varrho^{n-1} \max\Big\{ F(u(B)) - F(\bar{u}), \sum_{t=0}^{B-1} \sum_{i=1}^{m} \|s_i(t)\|^2 \Big\}, \quad n = 1, 2, ...,$$

where $\bar{u}$ *denotes the unique solution of* (2)*. Moreover,* $u(t)$ *converges strongly to* $\bar{u}$ *and, for each* $i \in \{1, ..., m\}$, $u_i(t)$ *converges strongly as* $t \to \infty$.

Applications to Convex Programming

Primal Applications

Consider the case of the problem (2), where $V = V' = \Re^n$, $F : \Re^n \mapsto \Re$ is a differentiable convex function, and K is a nonempty polyhedral set in $\Re^n$. Then F is continuous [Roc70] and continuously differentiable [Roc70]. We assume that the gradient $F' = (\frac{\partial F}{\partial x_j})_{j=1}^n$ is strongly monotone and Lipschitz continuous on K and we choose a space decomposition (4) such that each K_i is a polyhedral set.

Since each K_i is a polyhedral set, a Lipschitzian property of the solution set of a linear system (see [BT96]) implies that, for any $v_i \in K_i$, $i = 1, ..., m$, there exists $\bar{u}_i \in K_i$ satisfying (5), where C_1 depends on m and certain condition numbers for K_i, $i = 1, ..., m$. In cases where each K_i has a simple structure, C_1 may be estimated explicitly. Also, an analysis similar to that used for (21) shows that (6) holds with $C_2 = L\hat{c}$, where L is the Lipschitz constant for F' and $\hat{c}$ is the maximum number of sets K_j that are not orthogonal to an arbitrary set K_i. To color the sets such that (7)–(8) hold, it suffices to paint K_i and K_j different colors whenever they are not orthogonal, i.e., $(v_i)^T v_j \neq 0$ for some $v_i \in K_i, v_j \in K_j$.

Dual Applications

Consider the linearly constrained convex program

$$\text{minimize} \quad G(x) \quad \text{subject to} \quad Ax = b, \tag{15}$$

where $G : \Re^n \mapsto \Re$ is a strictly convex differentiable function, $b \in \Re^m$, and $A \in \Re^{m \times n}$ has nonzero rows. We assume there exists $\tilde{x} \in \Re^n$ satisfying $A\tilde{x} = b$. By attaching Lagrange multipliers $\lambda \in \Re^m$ to the equations $Ax = b$ in (15), we obtain the Lagrangian dual problem:

$$\min_{\lambda \in \Re^m} \; G^*(A^T\lambda) - b^T\lambda, \tag{16}$$

where G^* is the convex conjugate of G defined by (see [GT89], [Roc70])

$$G^*(u) = \sup_{x \in \Re^n} \left\{ u^T x - G(x) \right\}.$$

The convex programs (15) and (16) are dual in the sense that one has a solution if and only if the other does and these solutions satisfy $G'(x) = A^T\lambda$ [Roc70]. Using $b = A\tilde{x}$, we can rewrite the dual problem (16) in the form of (2) with

$$F(u) = G^*(u) - \tilde{x}^T u, \qquad K = \{u \in \Re^n : u = A^T\lambda \text{ for some } \lambda \in \Re^m\}. \tag{17}$$

We assume that $(G^*)'$ is strongly monotone and Lipschitz continuous on $\Re^n$, so that F satisfies (3) for some $\sigma > 0$.

Let $\bar{u}$ denote the unique solution of (2) and let A_i denote the ith row of A. We decompose K in the form (4) with subspaces

$$K_i = \{u_i \in \Re^n \ : \ u_i = A_i^T\lambda_i \text{ for some } \lambda_i \in \Re\}.$$

It was shown in [TT98] that, for any $v_i \in K_i$, $i = 1, ..., m$, there exists $\bar{u}_i \in K_i$ satisfying (5), where C_1 depends on A only. It was also shown that (6) holds with $C_2 = L\hat{c}$, where L is the Lipschitz constant for $(G^*)'$ and $\hat{c}$ is the maximum number of rows A_j that are not orthogonal to an arbitrary row A_i. Since two subspaces K_i and K_j are orthogonal if and only if $A_iA_j^T = 0$, we can color $K_1, ..., K_m$ as discussed in §38 so that (7)–(8) hold.

Applications to Partial Differential Equations

The first partial differential equation corresponds to the minimization problem (2) with $V = K = H_0^1(\Omega)$ and

$$\langle F'(u), v\rangle = \int_\Omega \left(\sum_{i=1}^d a_i(x, u, \nabla u)\partial_i v + a_0(x, u, \nabla u)v - fv\right)dx, \tag{18}$$

where Ω is a suitable domain of $\Re^d$, $f \in L^2(\Omega)$, and the nonlinear coefficient a_i, $i = 0, 1, \cdots d$ are such that (3) is satisfied for some $\sigma > 0$ (see [TT98]).

The second partial differential equation corresponds to the minimization problem (2) with $V = K = H_0^1(\Omega)$ and

$$F(v) = \int_\Omega \left(\frac{1}{2}|\nabla v|^2 + \frac{1}{4}v^4 - fv\right)dx, \tag{19}$$

with Ω, f as above and with $d \in \{2, 3\}$. The corresponding equation is the simplified Ginzburg-Landau equation for superconductivity:

$$-\Delta u + u^3 = f \quad \text{in} \quad \Omega, \qquad \text{and} \qquad u = 0 \quad \text{on} \quad \partial\Omega, \tag{20}$$

where u is the wave function, which is valid in the absence of internal magnetic field. It can be shown that this F satisfies (3) for some $\sigma > 0$ (see [TT98]).

Domain decomposition methods

In DD methods, the domain Ω is decomposed into the disjoint union of subdomains Ω_i, $i = 1, ..., m$, and their boundary, i.e., $\Omega\cup\partial\Omega = \cup_{i=1}^m(\Omega_i\cup\partial\Omega_i)$ and $\Omega_i\cap\Omega_j = \emptyset$ for $i \neq j$.

The subdomains, which are assumed to form a regular quasi-uniform division (see p. 124 and Eq. (3.2.28) of [Cia78] for definitions) with a specified maximum diameter of H, are the finite elements of the coarse mesh. To form the fine mesh for the finite element approximations, we further divide each Ω_i into finite elements of size h such that all the fine-mesh elements together form a regular finite element division of Ω. We denote this fine division by $\mathcal{T}_h$. For each Ω_i, we consider an enlarged subdomain $\Omega_i^\delta = \{e \in \mathcal{T}_h : \mathrm{dist}(e, \Omega_i) \le \delta\}$, where $\mathrm{dist}(e, \Omega_i) = \min_{x \in e, y \in \Omega_i} |x - y|$. The union of Ω_i^δ, $i = 1, ..., m$, covers Ω with overlap proportional to δ. Let $K_0 \subset H_0^1(\Omega)$ and $K \subset H_0^1(\Omega)$ denote the continuous, piecewise rth-order polynomial ($r \ge 1$) finite element subspaces, with zero trace on $\partial\Omega$, over the H-level and h-level subdivisions of Ω respectively. For $i = 1, ..., m$, let K_i denote the continuous, piecewise rth-order polynomial finite element subspace with zero trace on the boundary $\partial\Omega_i^\delta$ and extended to have zero value outside $\Omega_i^\delta \cup \partial\Omega_i^\delta$. Then $K_i^\ominus = K_i$ for $i = 0, 1, ..., m$, and it can be shown that the space decomposition (4), with summation index from 0 to m, holds. We assume that the overlapping subdomains are chosen such that each subdomain Ω_i^δ and its corresponding finite element subspace K_i can be painted one of n_c colors (numbered from 1 to n_c), with subdomains painted the same color being pairwise non-intersecting. The coarse mesh and its corresponding subspace K_0 are painted the color 0. Moreover, n_c should be independent of h. For general domain Ω, finding overlapping subdomains with such property is nontrivial. If Ω is the Cartesian product of intervals, we can easily find overlapping subdomains with $n_c = 2$ if $d = 1$, and $n_c \le 4$ if $d = 2$, and $n_c \le 6$ if $d = 3$.Then the total number of colors needed for (7) and (8) to hold is $c = n_c + 1$.

Let $\{\theta_i\}_{i=1}^m$ be a smooth partition of unity with respect to $\{\Omega_i\}_{i=1}^m$, i.e., $\theta_i \in C_0^\infty(\Omega)$ with $\theta_i \ge 0$, $\theta_i = 0$ outside of Ω_i, and $\sum_{i=1}^m \theta_i = 1$. Let I_h be the finite element interpolation mapping onto K which uses the function values at the h-level nodes. For any $v \in K$, let v_0 be the projection in the L^2-norm of v onto K_0, i.e., $v_0 \in K_0$ and $\int_\Omega (v_0 - v)\phi \, dx = 0$ for all $\phi \in K_0$, and let $v_i = I_h(\theta_i(v - v_0))$. Then, it can be seen that $v_i \in K_i$ for $i = 0, 1, ..., m$ and satisfy $v = \sum_{i=0}^m v_i$ [SBG96], [Xu92]. By further choosing θ_i so that $|\nabla\theta_i|$ has a certain boundedness property, it was shown in [TX98] that for any $v_i \in K_i$, $i = 0, 1, ..., m$, there exists $\bar{u}_i \in K_i$ satisfying (5) (with summation index from 0 to m), where C is independent of m and the mesh parameters, and

$$C_1 = C\sqrt{c}\left(1 + \left(\frac{H}{\delta}\right)^{\frac{1}{2}}\right).$$

For F given by (18) or (19), it was shown in [TT98] that

$$\begin{aligned}\sum_{i=0}^m \sum_{j=0}^m \langle F'(w_{ij} + u_{ij}) \quad &- \quad F'(w_{ij}), v_i\rangle \le \tilde{C}_2 \left(\sum_{j=0}^m \max_{i=0,1,\dots,m} \|u_{ij}\|^2\right)^{\frac{1}{2}} \left(\sum_{i=0}^m \|v_i\|^2\right)^{\frac{1}{2}} \\ &+ \quad (1 + C)\left(\sum_{i=1}^m \|u_{i0}\|^2_{H^1(\Omega_i^\delta)}\right)^{\frac{1}{2}} \left(\sum_{i=1}^m \|v_i\|^2\right)^{\frac{1}{2}}. \qquad (21)\end{aligned}$$

with $\tilde{C}_2$ a constant depending on $C\alpha, c, \hat{c}$ only. Compared with (6) (with $i, j = 0, 1, ..., m$), we see that (21) has an extra term on the right-hand side. It was shown in [TT98] that this extra term does not affect the convergence rate result of §38.

Multigrid methods

In MG methods, Ω is divided into a finite element triangulation $\mathcal{T}$ by a successive refinement process. More precisely, we have $\mathcal{T} = \mathcal{T}_J$ for some $J > 1$, where $\mathcal{T}_k$, $k = 1, ..., J$, is a nested sequence of regular quasi-uniform triangulation, i.e., $\mathcal{T}_k$ is a collection of simplexes $\mathcal{T}_k = \{\tau_i^k\}$ of size (i.e., maximum diameter) h_k such that $\Omega = \cup_i \tau_i^k$ and for which the quasi-uniformity constants are independent of k [Cia78] and with each simplex in $\mathcal{T}_{k-1}$ being the union of simplexes in $\mathcal{T}_k$. We further assume that there is a constant $r < 1$, independent of k, such that h_k is proportional to r^{2k}.

Corresponding to each triangulation $\mathcal{T}_k$, we define the finite element subspace:

$$\mathcal{M}_k = \{v \in H_0^1(\Omega) \ : \quad v|_\tau \in \mathcal{P}_1(\tau), \quad \forall\, \tau \in \mathcal{T}_k\},$$

where $\mathcal{P}_1(\tau)$ denotes the space of real-valued linear functions of d real variables defined on τ. We associate with $\mathcal{M}_k$ a nodal basis, denoted by $\{\phi_i^k\}_{i=1}^{n_k}$, that satisfies $\phi_i^k \in \mathcal{M}_k$ and

$$\phi_i^k(x_j^k) = \delta_{ij}, \quad \text{the Kronecker function},$$

where $\{x_i^k\}_{i=1}^{n_k}$ is the set of all interior nodes of the triangulation $\mathcal{T}_k$. For each such nodal basis function, we define the one-dimensional subspace: $K_i^k = \text{span } (\phi_i^k)$. Then, $(K_i^k)^\ominus = K_i^k$ and we have the following space decomposition:

$$K = \sum_{k=1}^{J} \sum_{i=1}^{n_k} K_i^k \quad \text{with} \quad K = \mathcal{M}_J.$$

On each level k, we color the nodes of $\mathcal{T}_k$ so that neighboring nodes are always of a different color. The number of colors needed for a regular mesh is a constant independent of the mesh parameters, which we denote by n_c. Then the total number of colors needed for (7) and (8) (with summation indices adjusted accordingly) to hold is $c = n_c J$. Also, it can be shown that (6) holds and that, for any $v_i^k \in K_i^k$, $i = 1, ..., n_k$, $k = 1, ..., J$, there exists $\bar{u}_i^k \in K_i^k$ satisfying (5) (with summation indices adjusted accordingly), where C_1 and C_2 do not depend on h and J.

REFERENCES

[BCEZ95] Bertsekas D. P., Castañon D., Eckstein J., and Zenios S. A. (1995) Parallel computing in network optimization. In Ball M. O., Magnanti T. L., Monma C. L., and Nemhauser G. L. (eds) *Handbooks in Operations Research and Management Science 7: Network Models*, pages 331–399. Elsevier Scientific, Amsterdam.

[BMPS95] Bru R., Migallón V., Penadés J., and Szyld D. B. (1995) Parallel, synchronous and asynchronous two-stage multisplitting methods. *Electronic Trans. Numer. Anal.* 3: 24–38.

[BMPSar] Bai Z., Migallón V., Penadés J., and Szyld D. B. (to appear) Block and asychronous two-stage methods for mildly nonlinear systems. *Numer. Math.* .

[BR82] Bank R. and Rose D. J. (1982) Analysis of a multilevel iterative method for nonlinear finite element equations. *Math. Comp.* 39: 453–465.

[Bra77] Brandt A. (1977) Multilevel adaptive solutions to boundary value problems. *Math. Comp.* 31: 333–309.

[BT89] Bertsekas D. P. and Tsitsiklis J. N. (1989) *Parallel and Distributed Computation: Numerical Methods*. Prentice-Hall, Englewood Cliffs.

[BT96] Burke J. V. and Tseng P. (1996) A unified analysis of Hoffman's bound via Fenchel duality. *SIAM J. Optim.* 6: 265–282.

[BZar] Bramble J. and Zhang X. (to appear) *The Analysis of Multigrid Methods*. North-Holland.

[Cia78] Ciarlet P. G. (1978) *The Finite Element Method for Elliptic Problems*. North-Holland, Amsterdam.

[CM69] Chazan D. and Miranker W. L. (1969) Chaotic relaxation. *Lin. Algeb. Appl.* 2: 199–222.

[CM94] Chan T. F. and Mathew T. P. (1994) Domain decomposition algorithms. *Acta Numerica* pages 61–143.

[CSar] Chan T. F. and Sharapov I. (to appear) Subspace correction multilevel methods for elliptic eigenvalue problems. In Bjørstad P., Espedal M., and Keyes D. (eds) *Proceedings of the 9th International Domain Decomposition Conference*. John Wiley and Sons.

[DH97] Dryja M. and Hackbusch W. (1997) On the nonlinear domain decomposition method. *BIT* 37: 296–311.

[FSS97] Frommer A., Schwardt H., and Szyld D. B. (1997) Asychronous weighted additive Schwarz methods. *Electronic Trans. Numer. Anal.* 5: 48–61.

[GT89] Glowinski R. and Tallec P. L. (1989) *Augmented Lagrangian and Operator-Splitting Methods in Nonlinear Mechanics*. SIAM, Philadelphia.

[HR89] Hackbusch W. and Reusken A. (1989) Analysis of a damped nonlinear multilevel method. *Numer. Math.* 55: 225–246.

[Kor97] Kornhuber R. (1997) *Adaptive monotone multigrid methods for nonlinear variational problems*. Teubner, Stuttgart.

[LT92] Luo Z.-Q. and Tseng P. (1992) On the linear convergence of descent methods for convex essentially smooth minimization. *SIAM J. Control Optim.* 30: 408–425.

[LT93] Luo Z.-Q. and Tseng P. (1993) Error bounds and convergence analysis of feasible descent methods: A general approach. *Ann. Oper. Res.* 46: 157–178.

[McC89] McCormick S. F. (1989) *Multilevel Adaptive Methods for Partial Differential Equations*, volume 6 of *Frontiers in Applied Mathematics*. SIAM, Philadelphia.

[Roc70] Rockafellar R. T. (1970) *Convex Analysis*. Princeton University Press, Princeton.

[SBG96] Smith B. F., Bjørstad P. E., and Gropp W. D. (1996) *Domain Decomposition: Parallel Multilevel Algorithms for Elliptic Partial Differential Equations*. Cambridge University Press, Cambridge.

[Sha97] Sharapov I. A. (1997) Multilevel subspace correction for large scale optimization problems. Technical Report CAM-97-31, UCLA, Computational and Applied Mathematics.

[Tai92] Tai X.-C. (1992) Parallel function decomposition and space decomposition methods with applications to optimisation, splitting and domain decomposition. *Preprint No. 231–1992, Institut für Mathematik, TechnischeUniversität Graz* .

[Tai94] Tai X.-C. (1994) Parallel function and space decomposition methods. In Neittaanmäki P. (ed) *Finite element methods, fifty years of the Courant element, Lecture notes in pure and applied mathematics*, volume 164, pages 421–432. Marcel Dekker, New York.

[Tai95] Tai X.-C. (1995) Parallel function decomposition and space decomposition methods: Part II. Space decomposition. *Beijing Mathematics* 1, part 2: 135–152.

[TBT90] Tseng P., Bertsekas D. P., and Tsitsiklis J. N. (1990) Partially asynchronous, parallel algorithms for network flow and other problems. *SIAM J. Control Optim.* 28: 678–710.

[TE98] Tai X.-C. and Espedal M. (1998) Rate of convergence of some space decomposition method for linear and nonlinear elliptic problems. *SIAM J. Numer. Anal.* 35: 1558–1570.

[TT98] Tai X.-C. and Tseng P. (August, 1998) Convergence rate analysis of an asynchronous space decomposition method for convex minimization. Technical Report 120, Department of Mathematics, University of Bergen.

[TX98] Tai X.-C. and Xu J.-C. (March, 1998) Global convergence of subspace correction methods for convex optimization problems. Technical Report No.-114, Department of Mathematics, University of Bergen, Norway.

[Xu92] Xu J.-C. (1992) Iteration methods by space decomposition and subspace correction. *SIAM Rev.* 34: 581–613.

[Xu96] Xu J.-C. (1996) Two-grid discretization techniques for linear and nonlinear PDEs. *SIAM J. Numer. Anal.* 33: 1759–1777.

39

Viscous-Inviscid Interaction: Domain Decomposition Avant la Lettre

A.E.P. VELDMAN[1], C.-H. LAI[2]

THE VISCOUS BOUNDARY LAYER

The paper describes the history of viscous-inviscid interaction methods, and tries to put them in a modern domain-decomposition context. This history started in 1904, when Ludwig Prandtl presented the 'boundary layer' at the Third International Mathematical Congress in Heidelberg. From that moment on, the flow field around a body was divided in two parts: a thin shear layer where viscosity plays a role, and the remaining outer part where the flow can be considered inviscid (Fig. 1). In the boundary layer the flow equations can be simplified by neglecting the viscous streamwise derivatives, which changes the elliptic character of Navier-Stokes to a much easier handled parabolic character. The latter was very relevant in an era where mainly analytical tools were available for solving differential equations.

The boundary layer is driven by the external pressure distribution p_e and (through Bernoulli's law) its related streamwise velocity u_e. In the boundary layer the streamwise velocity component is reduced to zero, herewith effectively thickening and smoothing the shape of the geometry. The resulting effective shape is called the displacement body δ^*, which becomes a streamline for the inviscid flow.

---

1 Department of Mathematics, University of Groningen, P.O. Box 800, 9700 AV Groningen, The Netherlands. email: veldman@math.rug.nl
2 School of Computing & Mathematical Sciences, University of Greenwich, Wellington Street, Woolwich, London SE18 6PF, U.K. email: C.H.Lai@grn.ac.uk
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

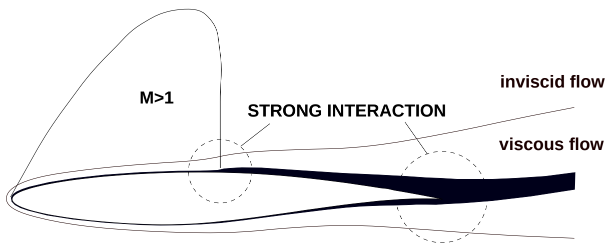


Figure 1 Subdivision of the transonic flow field around an airfoil in an inviscid-flow region and a viscous shear layer (exaggerated in thickness).

The coupled problem can be written as

$$\begin{aligned} \text{external inviscid flow:} \quad & u_e^{(n)} = E[\delta^{*(n-1)}], \\ \text{boundary-layer flow:} \quad & \delta^{*(n)} = B^{-1}[u_e^{(n)}], \end{aligned} \tag{1}$$

where E denotes the external inviscid-flow operator, and B^{-1} is the shear-layer operator; below it will be clear why we denote the latter operator in this way. It is observed that the information exchange between the two regions takes place in terms of the pressure or the streamwise velocity on the one side, and the displacement thickness or the normal velocity on the other side. Introducing a streamfunction ψ, we may correlate

$$u_e \leftrightarrow \frac{\partial \psi}{\partial n}, \qquad \delta^* \leftrightarrow -\psi,$$

hence in domain-decomposition terminology we have a Neumann-Dirichlet coupling.

FLOW SEPARATION AND ALGORITHM BREAK-DOWN

As the convergence rate of the Neumann-Dirichlet problem (1) scales with the thickness of the shear layer [Vel84], a fast convergence is expected. Indeed, for situations with attached flow this fast convergence is found, and the shear layer only provides a small correction to the inviscid-flow solution. However, as soon as the flow wants to separate from the surface the iterative coupling strategy (1) fails, because the shear-layer calculation breaks down, with a solution that tends to infinity.

A number of possible causes can be imagined. Firstly, the assumption of small streamwise derivatives appears to become invalid, and these derivatives would have to be included in the flow equations. Secondly, as the stable parabolic direction of the shear-layer equations is governed by the flow direction, inside separated-flow regions they should have been solved from downstream to upstream. In 1948 Goldstein added another possible cause for the break-down by raising the question "does a singularity always occur except for certain special pressure distributions near separation" [Gol48]. Since then, the singularity at separation bears his name.

It took twenty more years before algorithms and computers were sufficiently powerful to investigate the above options through numerical experiments. One of these was described in 1966 by Catherall and Mangler, who tried to solve the shear-layer equations with prescribed displacement thickness. Indeed, they succeeded to pass the point of flow separation, but ran into difficulties a bit further downstream [CM66]. Not convinced by their success, they stopped further research into this subject.

In the mean time Stewartson formulated the so-called 'triple-deck' theory, describing the flow in the neighbourhood of singular points in the flow field [Ste74]. Near these points the boundary layer is no longer merely providing small corrections to the flow, but instead wants to have an equal say in determining the flow field. In aerodynamical terms, the hierarchy between boundary layer and inviscid flow changes from *weak* interaction into *strong* interaction.

In the late 70-ies – after 75 years of research – it became clear that Goldstein had pointed in the right direction: the problems at flow separation are due to non-existence of a solution of the shear-layer equations at prescribed pressure. This is the reason why we have denoted the boundary layer operator at prescribed pressure by B^{-1}; its existence is not guaranteed, but B does exist.

VISCOUS-INVISCID COUPLING METHODS

From that moment on the path was free to go on, and around 1978 a number of iterative strategies have been brought forward. Two methods have survived [LW87]: the semi-inverse method [LeB78], and the quasi-simultaneous method [Vel79, Vel81].

Semi-inverse The semi-inverse method combines the favourable parts of the direct and the inverse method (Fig. 2, left). It solves both flow domains with prescribed displacement thickness, and could be called a Dirichlet-Dirichlet coupling. In order to obtain convergence some tuning of the relaxation parameter ω is required, and a fair convergence can be obtained:

$$u_e^E = E[\delta^{*(n-1)}], \quad u_e^B = B[\delta^{*(n-1)}], \quad \delta^{*(n)} = \delta^{*(n-1)} + \omega\,(u_e^E - u_e^B). \tag{2}$$

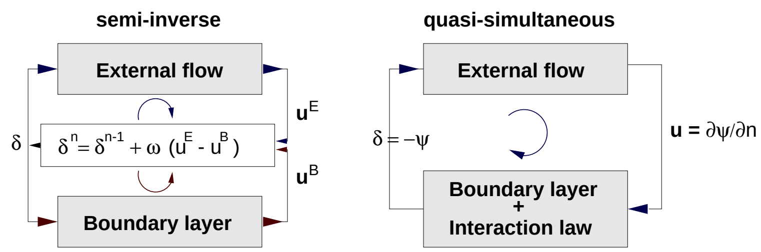


Figure 2 Semi-inverse and quasi-simultaneous VII method.

Quasi-simultaneous The quasi-simultaneous method wants to reflect the lack of hierarchy between the subdomains: in principle, it wants to solve both subdomain problems simultaneously. When the shear layer is modelled by an integral formulation a simultaneous coupling is well feasible, e.g. [HV84]. However, when in both domains a field formulation is chosen, software complexity prevents a practical implementation. Thus the idea was born to solve the shear-layer equations simultaneously with a simple but good *approximation* of the inviscid flow: the *interaction law*. The difference between this approximation and the 'exact' inviscid flow can then be handled iteratively.

Triple-deck theory delivers the flow model that describes the local interaction: thin-airfoil theory. Thus the interaction law

$$\frac{\partial \psi}{\partial n}(s) = -\frac{1}{\pi} \int_{\Gamma} \frac{\partial \psi}{\partial \sigma} \frac{d\sigma}{s-\sigma} \tag{3}$$

appears naturally, where Γ is the boundary between the subdomains. In this way, for aerodynamical applications the quasi-simultaneous method (Fig. 2, right) can be formulated as

$$\begin{aligned} u_e^{(n)} - I[\delta^{*(n)}] &= E[\delta^{*(n-1)}] - I[\delta^{*(n-1)}], \\ u_e^{(n)} - B[\delta^{*(n)}] &= 0, \end{aligned} \tag{4}$$

where the interaction law reads

$$I[\delta^*] = \frac{1}{\pi} \int_{\Gamma} \frac{d\delta^*}{d\sigma} \frac{d\sigma}{s-\sigma}. \tag{5}$$

It is observed in (4) that the interaction law is used in defect formulation, i.e. it does not influence the final converged result; it only enhances the rate of convergence!

Discretization The discretization of the thin-airfoil integral (5) leads to a positive-definite matrix I. E.g. the discretization as presented in [Vel84] yields the following expression on a uniform grid with mesh size h

$$I[\delta^*]|_i \approx \frac{1}{\pi} \sum_{j \neq i-1,i} \left\{ \frac{1}{h}(\delta^*_{j+1} - \delta^*_j) \ln \left| \frac{i-j}{i-j-1} \right| \right\} - \frac{2}{\pi h}(\delta^*_{i+1} - 2\delta^*_i + \delta^*_{i-1}).$$

We recognize in the latter term the discretization of derivative $d^2\delta^*/dx^2$, which describes the local contribution from the two intervals adjacent to the i-th grid point. It follows that I is symmetric and diagonally dominant. The discrete form of (4), setting $E = I$ for convenience, reads $u_e - I\delta^* = R_1$, $u_e - B\delta^* = R_2$, where R_1 contains contributions to the integral (5) from the end points of the computational domain. B now stands for the Jacobian of the shear-layer equations (it is lower triangular in attached flow), and in R_2 we can hide effects from their nonlinearity. After elimination of u_e we are left with

$$(-I + B)\,\delta^* = R_1 - R_2. \tag{6}$$

Experience learns that the diagonal of B is negative in regions of attached flow, but it vanishes in a point of separation after which it is slightly positive. This phenomenon is responsible for the break-down of the classical approach (1) for solving the shear-layer equations. But when the interaction law is added to the formulation, the matrix $I - B$ becomes relevant. With I being diagonally dominant, there is some room for subtracting positive contributions from B without $I - B$ becoming singular. Practice shows that a simple Gauss-Seidel procedure (i.e. a number of traditional boundary-layer sweeps) suffices to solve the system (6). For a more detailed discussion of the numerics involved we refer to [Vel84].

Simplification The interaction law (5) can be simplified even further to

$$I[\delta^*] = -\frac{2h}{\pi}\frac{d^2\delta^*}{ds^2}, \tag{7}$$

after which the first equation in (4) simply becomes

$$u_e^{(n)} + \frac{2h}{\pi}\left(\frac{d^2\delta^*}{ds^2}\right)^{(n)} = u_E^{(n-1)} + \frac{2h}{\pi}\left(\frac{d^2\delta^*}{ds^2}\right)^{(n-1)}. \tag{8}$$

Of course, since the description (7) contains less physics than (5) the convergence rate deteriorates [Coe99]. But (8) is easily implementable in existing boundary-layer codes, and it prevents the fatal break-down that occurs when pressure is prescribed.

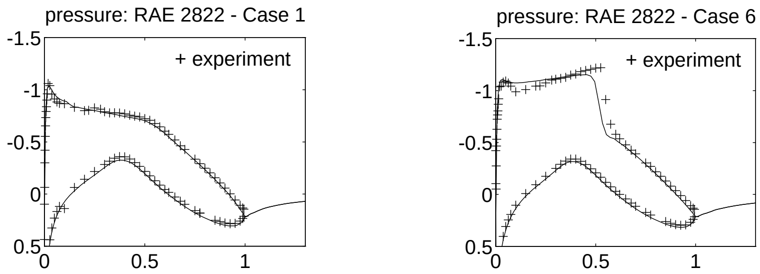
APPLICATIONS

Transonic airfoil flow The performance of the quasi-simultaneous coupling concept will be demonstrated on some calculations of transonic flow past an RAE 2822 airfoil. In these airfoil-flow problems the integral (5) has been used to describe the symmetric displacement effects ('thickness problem'). Its skew-symmetric counterpart has been used to describe the effects of camber ('lift problem'); for details see Veldman *et al.* [VLdBS90].

Fig. 3 (bottom right) shows the inviscid-viscous convergence for a computation of Case 1, subsequently followed by Case 6. It reveals that a handful of quasi-simultaneous iterations suffice to obtain a solution. The convergence is independent of the grids applied in the two subdomains. For completeness we mention that the inviscid flow was computed with an O-type 128x64 grid, whereas the shear layer was covered with a 173x21 C-type grid. Computing time is one minute on a PC. To appreciate the fast convergence even better, one has to realize that the external flow in these examples is transonic, with a significant supersonic flow region in Case 6, whereas the interaction law (5) is based on sub(!)sonic theory.

Also in three dimensions the quasi-simultaneous concept (with the simplified interaction law (7)) has been applied successfully to transonic flow [WM93].

Domain decomposition Viscous-inviscid interaction is an example of a Schwarz non-overlapping domain-decomposition with a Dirichlet-Neumann coupling. Thusfar, VII methods have only been used in situations where the Neumann region is very thin. In [dBV95] the usefullness of the quasi-simultaneous method for differently



	Mach	Reynolds	angle of attack
Case 1	0.676	5.7 mill.	1.93
Case 6	0.725	6.5 mill.	2.44

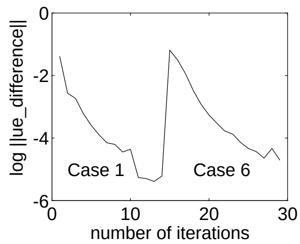


Figure 3 Pressure distributions and convergence for RAE 2822 airfoil.

shaped domains or for different governing equations has been investigated. As an example, it has been applied to a Dirichlet Laplace problem on a domain Ω with two subdomains Ω_1 and Ω_2. In this case the multiplicative Dirichlet-Neumann (D-N) domain-decomposition method reads

$$\Delta\psi_1^{(n)} = 0 \text{ in } \Omega_1, \qquad \psi_1^{(n)} = \psi_2^{(n-1)} \text{ on } \Gamma, \tag{9}$$

$$\Delta\psi_2^{(n)} = 0 \text{ in } \Omega_2, \qquad \frac{\partial\psi_2^{(n)}}{\partial n} = \frac{\partial\psi_1^{(n)}}{\partial n} \text{ on } \Gamma. \tag{10}$$

To the Neumann region Ω_2 an interaction law like (3) has been added, after which the Dirichlet-Neumann Interaction (D-NI) method can be formulated as (9) combined with the following modification of (10)

$$\frac{\partial\psi_2^{(n)}}{\partial n} + \frac{1}{\pi}\int_\Gamma \frac{\partial\psi_2^{(n)}}{\partial\sigma}\frac{d\sigma}{s-\sigma} = \frac{\partial\psi_1^{(n)}}{\partial n} + \frac{1}{\pi}\int_\Gamma \frac{\partial\psi_2^{(n-1)}}{\partial\sigma}\frac{d\sigma}{s-\sigma} \text{ on } \Gamma.$$

We emphasize that the added terms describe how subdomain Ω_1 is reacting on changes in subdomain Ω_2, and hence give an immediate response. Heuristically, this is the mechanism that will lead to a speed-up of the inter-subdomain iterations.

A convergence analysis has been made of the D-N method versus the D-NI method for various shapes of the subdomains Ω_1 and Ω_2; some of these shapes are shown in Fig. 4. For each method relaxation has been added, and the optimal relaxation factor ω_{opt} has been determined by trial-and-error. From Fig. 4 we conclude that the D-NI method converges very fast when the Neumann domain is thin (case A). In the opposite case, with a thick Neumann domain (case B), both D-N and D-NI diverge,

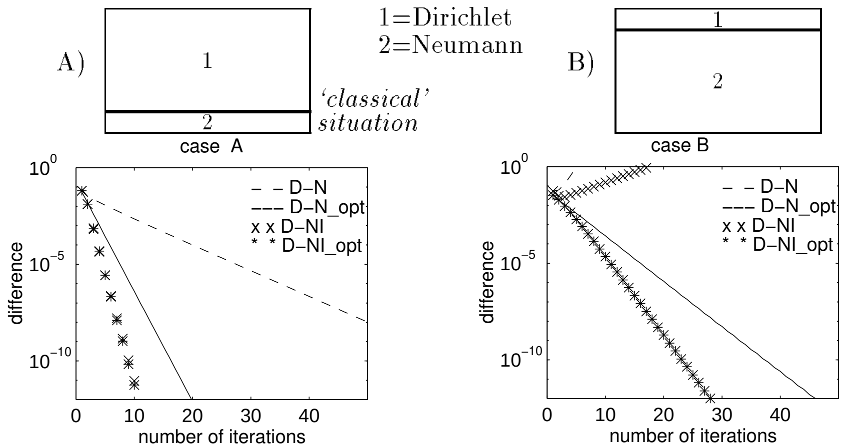


Figure 4 Influence of adding interaction to D-N coupling.

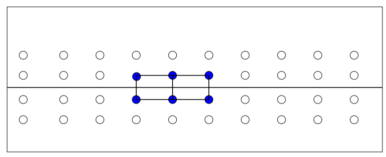


Figure 5 6-point stencil in local coupling by Tan.

but they can be made convergent with appropriate relaxation; again the optimal D-NI method is faster than the optimal D-N method.

In [dBV95] more configurations have been studied. In all of them the convergence behaviour is in line with the above findings, revealing that the (optimum) D-NI method is always faster than the (optimum) D-N method. In many cases the optimum relaxation factor for D-NI is close to $\omega = 1$, whereas the D-N method requires more fine-tuning of the relaxation factor.

Relation with local coupling methods A close relation exists with the class of local coupling methods introduced by Tan [Tan95] which make use of combinations of function values, lateral derivatives and normal derivatives. In fact, his interface condition is a linear combination of ψ, $\frac{\partial\psi}{\partial s}$, $\frac{\partial\psi}{\partial n}$, $\frac{\partial^2\psi}{\partial s^2}$ and $\frac{\partial^2\psi}{\partial s \partial n}$, leading to a 6-point stencil as shown in Fig. 5. Tan has optimized the coefficients in this combination based on mathematical arguments. It appears that his optimum for ellipticity-dominated problems is very close to the interactive coupling condition (8) which is based on physical arguments.

Relation with quasi-Newton methods The present method has also a close relation with the defect equation approach where an equilibrium or a matching condition [Lai98] is required to satisfy. The idea is to set up such a defect equation along the interface of two subproblems which are most likely governed by two different mathematical models. For the present problem, the typical defect equation is given by (compare (6))

$$D(\lambda) \equiv E\lambda - B\lambda = 0,$$

where E and B are defined as in (1). It is natural to solve the above defect equation using the well-known Newton's method

$$\lambda^{(n+1)} = \lambda^{(n)} - J(\lambda^{(n)})^{-1} D(\lambda^{(n)})$$

which has excellent local convergence properties. However, the viscous and inviscid coupling does not allow the Jacobian matrix being evaluated analytically. Therefore quasi-Newton methods obviously suit the present case. There are a lot of different approximations to the Jacobian and its inverse, such as Broyden's method and Schubert's method, etc. These approximate updating methods are applied in the coupling context [Lai98]. In fact, the equivalent interface condition is a suitable linear combination of the field variables deeply hidden in the two subdomains. In the limiting case when the approximation to the Jacobian is chosen as a diagonal matrix with constant diagonal entries, the method is essentially equivalent to the semi-inverse method (2). One can also easily see that the quasi-simultaneous method (4) is equivalent to choose the approximation of the Jacobian to be $I - B$.

CONCLUSIONS

During the last two decades, it has become clear that interaction laws are very beneficial in a viscous-inviscid coupling strategy. We have described their history, and placed them in a modern domain-decomposition setting. Additionally, it has been shown that they can also be beneficial in non-aerodynamic applications.

REFERENCES

[CM66] Catherall D. and Mangler K. W. (1966) The integration of the two-dimensional laminar boundary-layer equations past the point of vanishing skin friction. *J. Fluid Mech.* 26: 163–182.

[Coe99] Coenen E. G. M. (1999) Quasi-simultaneous coupling for wing and aerofoil flow. In *Proceedings 11th Domain-Decomposition Conference.* July 20-24, 1998, Greenwich.

[dBV95] de Boer B. F. A. and Veldman A. E. P. (1995) Interaction laws and domain decomposition. In Morton K. W. and Baines M. J. (eds) *Numerical Methods for Fluid Dynamics V*, pages 385–390. Oxford University Press.

[Gol48] Goldstein S. (1948) On laminar boundary layers flow near a point of separation. *Quart. J. Mech. Appl. Math.* 1: 43–69.

[HV84] Houwink R. and Veldman A. E. P. (1984) Steady and unsteady separated flow computation for transonic airfoils. AIAA paper 84–1618.

[Lai98] Lai C.-H. (1998) A defect equation approach for the coupling of subdomains in domain decomposition methods. *Computers Math. Applic.* 35: 81–94.

[LeB78] LeBalleur J.-C. (1978) Couplage visqueux-non visceux: méthode numérique et applications aux écoulements bidimensionnels transsoniques et supersoniques. *La Recherche Aérosopatiale* 183: 65–76.

[LW87] Lock R. C. and Williams B. R. (1987) Viscous-inviscid interactions in external aerodynamics. *Prog. Aerospace Science* 24: 51–171.

[Ste74] Stewartson K. (1974) Multi-structured boundary layers on flat plates and related bodies. *Adv. Appl. Mech.* 14: 145–239.

[Tan95] Tan K. H. (1995) *Local coupling in domain decomposition.* PhD thesis, University of Utrecht.

[Vel79] Veldman A. E. P. (1979) A numerical method for the calculation of laminar incompressible bounday layers with strong viscous-inviscid interaction. Technical Report NLR TR 79023, National Aerospace Laboratory.

[Vel81] Veldman A. E. P. (1981) New, quasi-simultaneous method to calculate interacting boundary layers. AIAA J. 19: 79–85.

[Vel84] Veldman A. E. P. (1984) A numerical view on strong viscous-inviscid interaction. In Habashi W. G. (ed) *Computational Methods in Viscous Flows*, pages 343–363. Pineridge Press.

[VLdBS90] Veldman A. E. P., Lindhout J. P. F., de Boer E., and Somers M. A. M. (1990) VISTRAFS: a simulation method for strongly-interacting viscous transonic flow. In Cebeci T. (ed) *Numerical and Physical Aspects of Aerodynamic Flow IV*, pages 37–51. Springer Verlag.

[WM93] Wees A. J. v. d. and Muijden J. v. (1993) A fast and robust viscous-inviscid interaction solver for transonic flow about wing/body configurations on the basis of full potential theory. AIAA paper 93–3026.

Part III

Applications

40

Domain decomposition for flow in porous media with fractures

Clarisse Alboin[1], Jérôme Jaffré[2], Jean Roberts[3] &
Christophe Serres[4]

Introduction

We are concerned with flow in a porous medium such as granite or sedimentary rock. Natural fractures occur in the rock and it is necessary to take them into account because they have a profound effect on the transport of the fluid. One can distinguish two types of fractures : numerous small fractures that can be treated with a double porosity model [Arb90], [KPL94], and more important fractures that one may wish to model individually using domain decomposition. The latter is the topic of this article.

The fractures that we are concerned with are filled with debris so we consider them as porous media. The permeability in the fracture is large in comparison with that in the surrounding rock, so the fluid circulates faster in the fracture. Thus we have a highly heterogeneous porous medium. One idea that has been used to take this into account is to treat the fracture as an interface and to assume that the fluid that flows into the fracture stays in the fracture. In fact, in many models the contrast in permeabilities is of such an order that the flow outside of the fracture is neglected. However, here we are concerned with the situation in which the exchange between the fracture and the rest of the domain is significant. To deal with this case we need to model both what happens in the fracture and what happens outside the fracture. One

---

1 INRIA-Rocquencourt, BP 105, 78153 Le Chesnay Cedex
2 Address same as above.
3 Address same as above.
4 IPSN/DES/SESID BP6 92265 Fontenay Aux Roses Cedex.
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

idea is to use domain decomposition treating each fracture as a separate sub-domain. This approach however, leads to well known difficulties due to the small width of a fracture in comparison with the size of the domain. The idea presented here is to use a domain decomposition model with the fractures as natural interfaces, and to model what happens in the interfaces. In this article, we shall consider only the simplest case with two sub-domains separated by one fracture.

This paper is organized as follows. In section 2, we indicate how the model is derived, and in section 3, we show that the corresponding problem has a unique solution. In section 4, we define the mixed finite element formulation and reduce the problem to an interface problem. Numerical results are given in Section 5.

Model

Suppose that Ω is a connected and simply connected domain in R^n, $n = 2$ or 3, with boundary Γ. We assume that flow in Ω is governed by Darcy's equation and for simplicity we suppose that we have a Dirichlet boundary condition on all of Γ:

$$\begin{array}{rcll} \operatorname{div}\mathbf{u} & = & q & \text{in } \Omega \\ \mathbf{u} & = & -K\nabla p & \text{in } \Omega \\ p & = & 0 & \text{on } \Gamma, \end{array} \tag{1}$$

where p is the pressure, $\mathbf{u}$ the Darcy velocity, K the permeability, q an external source. We suppose that K is positive, bounded above and away from 0:

$$0 < K_{min} \le K(x) \le K_{max} \qquad \text{almost everywhere in } \Omega.$$

Suppose that the fracture Ω_f, a connected, simply connected sub-domain of Ω, is the intersection of Ω with a hyperplane times an interval of length d and that Ω_f separates Ω into two connected, simply connected sub-domains:

$$\Omega\backslash\bar{\Omega}_f = \Omega_1 \cup \Omega_2, \qquad \Omega_1 \cap \Omega_2 = \phi.$$

Denote by Γ_i the part of the boundary of Ω_i in common with the boundary of $\Omega,\ i = 1, 2, f$:

$$\Gamma_i = \partial\Omega_i \cap \Gamma, \quad i = 1, 2, f,$$

and by γ_i the part of the boundary of Ω_i in common with the boundary of $\Omega_f,\ i = 1, 2$:

$$\gamma_i = \partial\Omega_i \cap \partial\Omega_f \cap \Omega, \quad i = 1, 2.$$

Let $\mathbf{n}$ denote a unit vector in R^n normal to the hyperplane of Ω_f and directed outward from Ω_1 on γ_1 and η an external, unit vector in R^n normal to Γ (see: figure 1).

If we denote by $p_i, \mathbf{u}_i, K_i$, and q_i the restrictions of $p, \mathbf{u}, K$, and q respectively to $\Omega_i,\ i = 1, 2, f$, we may write the transition problem

$$\begin{array}{rclll} \operatorname{div}\mathbf{u}_i & = & q_i & \text{in } \Omega_i, & i = 1, 2, f, \\ \mathbf{u}_i & = & -K_i\nabla p_i & \text{in } \Omega_i, & i = 1, 2, f, \\ p_i & = & 0 & \text{on } \Gamma_i, & i = 1, 2, f, \\ p_i & = & p_f & \text{on } \gamma_i, & i = 1, 2, \\ \mathbf{u}_i \cdot \mathbf{n} & = & -\mathbf{u}_f \cdot \mathbf{n} & \text{on } \gamma_i, & i = 1, 2, \end{array} \tag{2}$$

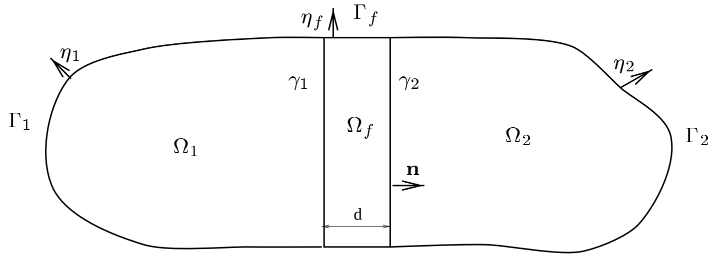


Figure 1 The domain Ω with fracture Ω_f

which is equivalent to (1).

To obtain a formulation in which the fracture is considered as an interface, we use an asymptotic expansion, cf:[Lio73], [GF96] . With an abuse of notation, we shall write Ω_2 for $\Omega_{*,2} = \{x - d\mathbf{n} : x \in \Omega_2\}$, γ for $\gamma_1 = \{x - d\mathbf{n} : x \in \gamma_2\}$, Ω for $\Omega_1 \cup \Omega_2 \cup \gamma$, Γ_1 for $\Gamma_{*,1} = \partial\Omega_1$, Γ_2 for $\Gamma_{*,2} = \partial\Omega_2$ and Γ_f for $\Gamma_{*,f} = \partial\gamma$ (see: figure 2).

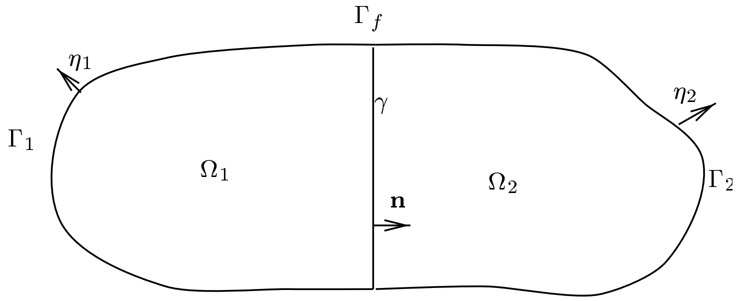


Figure 2 The domain Ω with the fracture γ as an interface

We introduce a real parameter ϵ $(0 < \epsilon < d)$ corresponding to the width of the fracture of the model and destined to tends to 0. We write the solution as an asymptotic expansion

$$u^\epsilon = \sum_{k=0}^{\infty} \epsilon^k u^k, \quad p^\epsilon = \sum_{k=0}^{\infty} \epsilon^k p^k.$$

The equations involving the first terms p^0 and u^0 lead to the problem

$$
\begin{array}{rcll}
\mathrm{div}_i\, \mathbf{u}_i &=& q_i & \text{on } \Omega_i \quad \text{for } i=1,2\\
\mathbf{u}_i &=& -K_i \nabla\, p_i & \text{on } \Omega_i \quad \text{for } i=1,2\\
p_i &=& 0 & \text{on } \Gamma_i \quad \text{for } i=1,2\\
p_i &=& p_f & \text{on } \gamma\\
\mathrm{div}_f\, d\,\mathbf{u}_f &=& d\, q_f + (\mathbf{u}_1\cdot\mathbf{n} - \mathbf{u}_2\cdot\mathbf{n}) & \text{on } \gamma\\
d\,\mathbf{u}_f &=& -d\, K_f \nabla_f\, p_f & \text{on } \gamma\\
p_f &=& 0 & \text{on } \partial\gamma.
\end{array} \tag{3}
$$

We define

$$\mathrm{div}_f \mathbf{v} \;=\; \mathrm{div}\,\mathbf{v} - \nabla(\mathbf{v}\cdot\mathbf{n})\cdot\mathbf{n} \qquad \text{and} \qquad \nabla_f\, r = \nabla r - \nabla r\cdot\mathbf{n}.$$

The model depends on the width d of the fracture and the difference between fluxes from sub-domains is a source term for the fracture cf:[AJRS98]. To obtain a weak formulation of (3), we define the Hilbert spaces

$$
\begin{array}{rcl}
\mathbf{W} &=& \{\mathbf{u}=(\mathbf{u}_1,\mathbf{u}_2,\mathbf{u}_f)\in L^2(\Omega_1)^n\times L^2(\Omega_2)^n\times L^2(\gamma)^{n-1}: \;\; \mathrm{div}\;\mathbf{u}_i\in L^2(\Omega_i) \quad i=1,2\\
&& \;\mathrm{div}_f\; d\,\mathbf{u}f - (\mathbf{u}_1\cdot\mathbf{n}-\mathbf{u}_2\cdot\mathbf{n})\in L^2(\gamma)\}\\[2mm]
M &=& \{p=(p_1,p_2,p_f)\in L^2(\Omega_1)\times L^2(\Omega_2)\times L^2(\gamma)\}.
\end{array} \tag{4}
$$

and their norms

$$
\begin{array}{rcl}
||\mathbf{u}||^2_{\mathbf{W}} &=& \displaystyle\sum_{i=1}^{2}\left(||\mathbf{u}_i||^2_{0,\Omega_i} + ||\mathrm{div}\;\mathbf{u}_i||^2_{0,\Omega_i}\right) + ||\mathbf{u}_f||^2_{0,\gamma} + ||\mathrm{div}_f\, d\,\mathbf{u}_f - (\mathbf{u}_1\cdot\mathbf{n}-\mathbf{u}_2\cdot\mathbf{n})||^2_{0,\gamma}\\
||p||^2_M &=& \displaystyle\sum_{i=1}^{2}||p_i||^2_{0,\Omega_i} + ||p_f||^2_{0,\gamma}.
\end{array}
$$

The weak formulation of (3) is given in terms of the bilinear forms

$$\alpha:\mathbf{W}\times\mathbf{W}\to R \qquad \text{and} \qquad \beta:\mathbf{W}\times M\to R$$

defined by

$$
\begin{array}{rcl}
\alpha(\mathbf{u},\mathbf{v}) &=& \displaystyle\sum_{i=1}^{2}\int_{\Omega_i} K_i^{-1}\mathbf{u}_i\cdot\mathbf{v}_i + \int_\gamma d\; K_f^{-1}\mathbf{u}_f\,\cdot\mathbf{v}_f\\
\beta(\mathbf{u},r) &=& \displaystyle\sum_{i=1}^{2}\int_{\Omega_i}\mathrm{div}\;\mathbf{u}_i\; r_i + \int_\gamma \left(\mathrm{div}_f(d\;\mathbf{u}_f) - (\mathbf{u}_1\cdot\mathbf{n}-\mathbf{u}_2\cdot\mathbf{n})\right) r_f
\end{array}
$$

and the linear forms $L_\sigma(q;\,\cdot\,): M\to R$ and defined by

$$L_\sigma(q;r) = \sum_{i=1}^{2}\int_{\Omega_i} q_i\; r_i + \int_\gamma d\; q_f\;\; r_f,$$

The weak mixed formulation of problem (1) is Find $\mathbf{u}\in\mathbf{W}, p\in M$ such that

$$
\begin{array}{rcll}
\alpha(\mathbf{u},\mathbf{v}) - \beta(\mathbf{v},p) &=& 0 & \forall\mathbf{v}\in\mathbf{W}\\
\beta(\mathbf{u},r) &=& L_\sigma(q;r) & \forall r\in M.
\end{array} \tag{5}
$$

Existence and uniqueness of the solution

We introduce the subspace $\tilde{\mathbf{W}}$ of $\mathbf{W}$ by $\tilde{\mathbf{W}} = \{\mathbf{v} \in \mathbf{W} : \beta(\mathbf{v}, r) = 0 \ \ \forall r \in M\}$. To show the existence and uniqueness of the solution of (1), it is sufficient to show that α is $\tilde{\mathbf{W}}$-elliptic and that β satisfies the inf-sup condition; cf:[BF91],[RT87]; that is there exist constants C_α and C_β such that

$$\inf_{\mathbf{v}\in\tilde{\mathbf{W}}} \frac{\alpha(\mathbf{v},\mathbf{v})}{||\mathbf{v}||^2_{\tilde{\mathbf{W}}}} \geq C_\alpha$$

$$\inf_{r\in M} \sup_{\mathbf{v}\in\mathbf{W}} \frac{\beta(\mathbf{v},r)}{||r||_M \; ||\mathbf{v}||_{\mathbf{W}}} \geq C_\beta .$$

To see that α is $\tilde{\mathbf{W}}$-elliptic, we note that for $\mathbf{u} \in \tilde{\mathbf{W}}$, $||\mathbf{u}||^2_W = \sum_{i=1}^2 ||\mathbf{u}_i||^2_{0,\Omega_i} + ||\mathbf{u}_f||^2_{0,\gamma}$ so that

$$\begin{aligned} \alpha(\mathbf{u},\mathbf{u}) &= \sum_{i=1}^2 \int_{\Omega_i} K_i^{-1}\mathbf{u}_i \cdot \mathbf{u}_i + \int_\gamma K_f^{-1}\mathbf{u}_f \cdot \mathbf{u}_f \\ &\geq K_{max}^{-1}\left(\sum_{i=1}^2 ||\mathbf{u}_i||^2_{0,\Omega_i} + ||\mathbf{u}_f||^2_{0,\gamma}\right) \\ &= K_{max}^{-1}||\mathbf{u}||^2_W . \end{aligned} \tag{6}$$

To see that β satisfies the inf-sup condition, given $r \in M$, using the adjoint equation we construct a $\mathbf{v} \in \mathbf{W}$ such that $\beta(\mathbf{v}, r) = ||r||^2_M$ and $||\mathbf{v}||_{\mathbf{W}} \leq C||r||_M$, where C is the constant of elliptic regularity for the adjoint problem.

For $r = (r_1, r_2, r_f) \in M$, let $(\varphi_1, \varphi_2, \varphi_f) \in H^2(\Omega_1) \times H^2(\Omega_2) \times H^2(\gamma)$ be the solution of

$$\begin{aligned} -\triangle\varphi &= \tilde{r} \quad \text{on } \Omega \\ \varphi &= 0 \quad \text{on } \Gamma, \end{aligned}$$

where $\tilde{r} \in L^2(\Omega)$ is given by $\tilde{r}_{|\Omega_i} = r_i$;and

$$\begin{aligned} -\triangle_f\varphi_f &= r_f \quad \text{on } \gamma \\ \varphi_f &= 0 \quad \partial\gamma. \end{aligned}$$

Pose $\mathbf{v}_i = -\nabla\varphi_{|\Omega_i}, i = 1, 2$, and $\mathbf{v}_f = -\nabla_f \varphi_f$ and note that $\mathrm{div}\,\mathbf{v}_i = r_i \in L^2(\Omega_i), i = 1, 2$, $\mathrm{div}_f\, \mathbf{v}_f = r_f \in L^2(\gamma)$ and $\mathbf{v}_1{\cdot}\mathbf{n}{-}\mathbf{v}_2{\cdot}\mathbf{n} = 0$. Thus $\mathbf{v} = (\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_f) \in \mathbf{W}$ and it is easy to check that $\mathbf{v}$ has the desired properties.

Domain decomposition for the interface problem

In this section we wish to formulate a domain decomposition problem based on mixed finite element methods; see [GW88, CMW95]. Toward this end, we introduce a quasi regular triangulation $\mathcal{T}_($ (of triangles and/or rectangles) of Ω compatible with the decomposition of Ω into the subdomains Ω_i, $i = 1, 2$. Note that in this case a triangulation is induced on the interface γ. We let $M_h = M_{h,1} \times M_{h,2} \times M_{h,f}$ and

$\mathbf{W}_h = \mathbf{W}_{h,1} \times \mathbf{W}_{h,2} \times \mathbf{W}_{h,f}$ be finite dimensional subspaces of $L^2(\Omega_1) \times L^2(\Omega_2) \times L^2(\gamma)$ and $H(\mathrm{div};\Omega_1) \times H(\mathrm{div};\Omega_2) \times H(\mathrm{div}_f;\gamma)$ respectively, such that the pair $(M_{h,i},\, \mathbf{W}_{h,i})$ is a Raviart-Thomas space of order k for Ω_i, $i = 1, 2$, subordinate to the triangulation $\mathcal{T}_{h,i}$ determined by $\mathcal{T}_h$; and the pair $(M_{h,f},\, \mathbf{W}_{h,f})$ is a Raviart-Thomas space of order k for γ associated with the triangulation $\mathcal{T}_{h,f}$ on γ induced by $\mathcal{T}_($. The mixed formulation in the subdomain Ω_i $i = 1, 2$, is

$$\begin{array}{rcll} \mathrm{div}\,\mathbf{u}_i &=& q_i & \text{in } \Omega_i, \\ \mathbf{u}_i &=& -K_i \nabla p_i & \text{in } \Omega_i, \\ p_i &=& 0 & \text{on } \Gamma_i, \\ p_i &=& p_f & \text{on } \gamma \end{array} \tag{7}$$

Introducing for each i, $i = 1, 2$, the bilinear forms

$$\alpha_i(\mathbf{u},\mathbf{v}) = \int_{\Omega_i} K_i{}^{-1}\,\mathbf{u}\cdot\mathbf{v}, \quad \mathbf{u},\mathbf{v} \in \mathbf{W}_{h,i}, \qquad \beta_i(\mathbf{v}, r) = \int_{\Omega_i} \mathrm{div}\,\mathbf{v}\, r, \quad \mathbf{v} \in \mathbf{W}_{h,i},\ r \in M_{h,i};$$

and, for $q_i \in M_{h,i}$ and $p_f \in M_{h,f}$, the linear forms

$$L_{\sigma,i}(q_i; r) = \int_{\Omega_i} q_i\, r, \quad r \in M_{h,i}, \qquad L_{\gamma,i}(p_f;\mathbf{v}) = \int_\gamma p_f\, \mathbf{v}\cdot\mathbf{n}, \quad \mathbf{v} \in \mathbf{W}_{h,i},$$

we may write the weak form of (7):

$$\begin{array}{lcll} \mathbf{u}_i \in \mathbf{W}_{h,i}, \quad p_i \in M_{h,i} & & & \\ \alpha_i(\mathbf{u}_i,\mathbf{v}) - \beta_i(\mathbf{v}, p_i) &=& (-1)^i L_{\gamma,i}(p_f;\mathbf{v}) & \forall \mathbf{v} \in \mathbf{W}_{h,i} \\ \beta_i(\mathbf{u}_i, r) &=& L_{\sigma,i}(q_i; r) & \forall r \in M_{h,i}. \end{array} \tag{8}$$

Following [CMW95] we decompose $\mathbf{u}_i$ and p_i as follows:

$$\mathbf{u}_i = \hat{\mathbf{u}}_i + \mathbf{u}_i^0, \qquad p_i = \hat{p}_i + p_i^0,$$

with $\hat{\mathbf{u}}_i$, $\hat{p}_i$, $\mathbf{u}_i^0$ and p_i^0 determined by

$$\begin{array}{lcll} \hat{\mathbf{u}}_i \in \mathbf{W}_{h,i}, \quad \hat{p}_i \in M_{h,i} & & & \\ \alpha_i(\hat{\mathbf{u}}_i,\mathbf{v}) - \beta_i(\mathbf{v}, \hat{p}_i) &=& (-1)^i L_{\gamma,i}(p_f;\mathbf{v}) & \forall \mathbf{v} \in \mathbf{W}_{h,i} \\ \beta_i(\hat{\mathbf{u}}_i, r) &=& 0 & \forall r \in M_{h,i}. \end{array} \tag{9}$$

and

$$\begin{array}{lcll} \mathbf{u}_i^0 \in \mathbf{W}_{h,i}, \quad p_i^0 \in M_{h,i} & & & \\ \alpha_i(\mathbf{u}_i^0,\mathbf{v}) - \beta_i(\mathbf{v}, p_i^0) &=& 0 & \forall \mathbf{v} \in \mathbf{W}_{h,i} \\ \beta_i(\mathbf{u}_i^0, r) &=& L_{\sigma,i}(q_i; r) & \forall r \in M_{h,i}. \end{array} \tag{10}$$

In the fracture we have the problem

$$\begin{array}{lcll} \mathrm{div}\, d\,\mathbf{u}_f &=& d\,q_f \ + \ \mathbf{u}_1 \cdot \mathbf{n} - \mathbf{u}_2 \cdot \mathbf{n} & \text{in } \gamma, \\ \mathbf{u}_f &=& -K_f \nabla p_f & \text{in } \gamma, \\ p_f &=& 0 & \text{on } \Gamma_f. \end{array} \tag{11}$$

Introducing the bilinear forms

$$\alpha_f(\mathbf{u},\mathbf{v}) = \int_\gamma d\, K_f{}^{-1}\,\mathbf{u}\,\mathbf{v}, \quad \mathbf{u},\mathbf{v} \in \mathbf{W}_{h,f}, \qquad \beta_f(\mathbf{v}, r) = \int_\gamma \mathrm{div}_f\, d\,\mathbf{v}\, r, \quad \mathbf{v} \in \mathbf{W}_{h,f},\ r \in M_{h,f};$$

and, for $q_f \in M_{h,f}$ and $\mathbf{u}_i \in \mathbf{W}_{h,i}$, the linear forms

$$L_{\sigma,f}(q_f; r) = \int_\gamma d\, q_f\, r, \quad r \in M_{h,f}, \qquad L_{\Omega,i}(\mathbf{u}_i; r) = \int_\gamma \mathbf{u}_i \cdot \mathbf{n}\, r, \quad r \in M_{h,f}$$

we may write the weak form of (11):

$$\begin{array}{lcll} \mathbf{u}_f \in \mathbf{W}_{h,f}, \quad p_f \in M_{h,f} & & & \\ \alpha_f(\mathbf{u}_f, \mathbf{v}) - \beta_f(\mathbf{v}, p_f) & = & 0 & \forall \mathbf{v} \in \mathbf{W}_{h,f} \\ \beta_f(\mathbf{u}_f, r) & = & L_{\sigma,f}(q_f; r) + L_{\Omega,1}(\mathbf{u}_1; r) - L_{\Omega,2}(\mathbf{u}_2; r) & \forall r \in M_{h,f}. \end{array} \tag{12}$$

To obtain a problem defined on the interface γ, we also define, for $i = 1, 2$, the following bilinear form on $M_{h,f}$:

$$\mathcal{A}_i(s_f, r_f) = \alpha_i(\hat{\mathbf{u}}_i(s_f), \hat{\mathbf{u}}_i(r_f)),$$

where $(\hat{\mathbf{u}}_i(s_f), \hat{p}_i(s_f))$, respectively $(\hat{\mathbf{u}}_i(r_f), \hat{p}_i(r_f))$ is the solution of (9) for the data $L_{\gamma,i}(s_f;\,\cdot\,)$, respectively $L_{\gamma,i}(r_f;\,\cdot\,)$. Similarly we write $(\mathbf{u}_i(s_f), p_i(s_f))$, respectively $(\mathbf{u}_i(r_f), p_i(r_f))$ for the solution of (8) for the data $L_{\gamma,i}(s_f;\,\cdot\,)$, respectively $L_{\gamma,i}(r_f;\,\cdot\,)$, and we denote by (u_i^0, p_i^0) the solution of (10). Using the symmetry of the forms α_i, both equations of (9), and the decomposition of $\mathbf{u}_i$ into $\hat{\mathbf{u}}_i$ and $\mathbf{u}_i^0$ we have,

$$\begin{aligned} \sum_{i=1}^{2} \mathcal{A}_i(s_f, r_f) &= \sum_{i=1}^{2} \alpha_i(\hat{\mathbf{u}}_i(r_f), \hat{\mathbf{u}}_i(s_f)) \\ &= \sum_{i=1}^{2} (-1)^i L_{\gamma,i}(r_f; \hat{\mathbf{u}}_i(s_f)) \\ &= \sum_{i=1}^{2} (-1)^i L_{\gamma,i}(r_f; \mathbf{u}_i(s_f)) - \sum_{i=1}^{2} (-1)^i L_{\gamma,i}(r_f; \mathbf{u}_i^0) \end{aligned}$$

Then using the definitions of the forms $L_{\gamma,i}$ and $L_{\Omega,i}$ and both of the equations of (12) we have

$$\begin{aligned} \sum_{i=1}^{2} (-1)^i L_{\gamma,i}(r_f; \mathbf{u}_i(s_f)) &= \sum_{i=1}^{2} (-1)^i L_{\Omega,i}(\mathbf{u}_i(s_f); r_f) \\ &= -\beta_f(\mathbf{u}_f(s_f), r_f) + L_{\sigma,f}(q_f; r_f) \\ &= -\delta_f(s_f, r_f) + L_{\sigma,f}(q_f; r_f), \end{aligned}$$

where $\mathbf{u}_f(s_f)$ is the solution of (12) with $L_{\Omega,1}(\mathbf{u}_1(s_f); \cdot) - L_{\Omega,2}(\mathbf{u}_2(s_f); \cdot)$ as data and where the bilinear form $\delta_f(.,.)$ is

$$\delta_f(s_f, r_f) = < B_f A_f^{-1} B_f^T\, s_f, r_f >_{|\gamma}$$

A_f and B_f are the linear mapping respectively associated to the bilinear form $\alpha_f(.,)$ and $\beta_f(.,.)$.

Combining the last two equations we obtain our interface problem:

$$\begin{array}{lcl} s_f \in M_{h,f} & & \\ \displaystyle\sum_{i=1}^{2} \mathcal{A}_i(s_f, r_f) + \delta_f(s_f, r_f) & = & \displaystyle\sum_{i=1}^{2} (-1)^{i+1} L_{\gamma,i}(r_f; \mathbf{u}_i^0) + L_{\sigma,f}(q_f; r_f), \quad r_f \in M_{h,f}. \end{array} \tag{13}$$

It is clear that the left hand side of the above equation determines a symmetric, positive definite form on $M_{h,f}$. Thus for a given source term $q = (q_1, q_2, q_f) \in M_{h,1} \times M_{h,2} \times M_{h,f}$ there is a unique solution $s_f \in M_{h,f}$. We remark that in the absense of a fracture, i.e. when the flux is continuous across the interface γ, $\mathbf{u}_1 \cdot \mathbf{n} = \mathbf{u}_2 \cdot \mathbf{n}$, the second term of each side of (13) vanishes (q_f is of course in this case equal to 0) and we obtain the standard interface problem given in [GW88, CMW95] for the case of two subdomains.

Numerical Results

To illustrate the model we consider an ideal dimensionless problem. The domain is an horizontal rectangular slice of porous medium, of dimensions 2.1×1, with given pressure on the left and right boundaries and no flow conditions on the top and bottom boundaries. In the domain the permeability is equal to one. The domain is divided into two equally large sub-domains by a linear fracture parallel to the x_2 axis. The permeability in the fracture $\times$ the width of the fracture is equal to 2. For example the fracture could be of width 0.1 and could have a permeability equal to 20. Flow in the fracture is driven by a pressure drop of 10 between the two extremities of the fracture for the first example and a pressure drop of 5 for the second example.

Two cases are considered. A symmetric case where pressures on the left and on the right boundaries of the domain are equal. So the flow is driven only by the fracture and is symmetric. In the other case there is a pressure drop from the right boundary to the left one. Then the flow is a combination of the flow in the fracture and that going from left to right in the rest of the porous medium.

Numerical results are shown on figure 3. Arrows represent the flow field with length proportional to the magnitude of the velocity. The gray scale represents also the magnitude of the velocity with the lightest color corresponding to the largest velocity. These results were obtained by solving equation (13) with a direct method. We see that there is actual flow interaction between the fracture and the rest of the porous medium. In particular one can observe that some fluid is coming out of the fracture and then is coming back into it. In the nonsymmetric case we notice also that even though most of the flow is attracted into the fracture, there is still some flow on the left part of the domain pointing toward the left.

Conclusion

A model for flow interaction between a fracture and the rest of the porous medium has been presented. In this model the fracture is an interface dividing the domain of calculation into sub-domains. Existence and uniqueness of the solution has been shown and the model has been reformulated as an interface problem. Simple numerical experiments show actual flow interaction between the fracture and the rest of the porous medium.

Extension to situations with several intersecting fractures is under way as well as the study of efficient preconditionners.

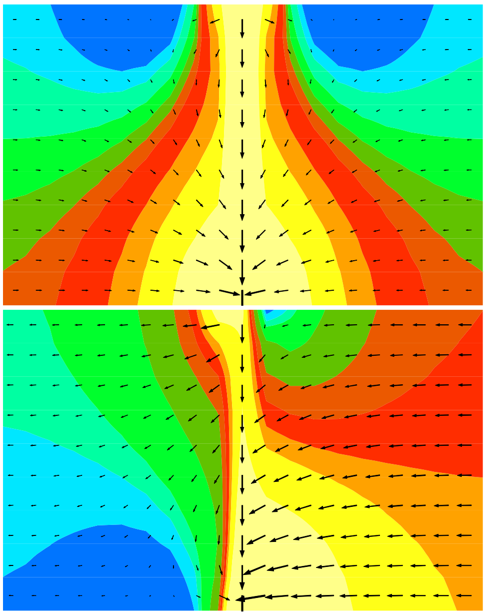


Figure 3 Calculated Darcy's velocity for a symmetric and a nonsymmetric flow pattern

REFERENCES

[AJRS98] Alboin C., Jaffré J., Roberts J., and Serres C. (1998) A model for a porous meduim with fracture. *in preparation* .

[Arb90] Arbogast T. (July 1990) Derivation of the double porosity model of single phase flow via homogenization theory. *Siam J. Math. Anal.* 12(No. 4).

[BF91] Brezzi F. and Fortin M. (1991) *Mixed and hybrid finite element methods*. Springer-Verlag.

[CMW95] Cowsar C. L., Mandel J., and Wheeler M. (July 1995) Balancing domain decomposition for mixed finite elements. *Mathematics of Computation* 64(211): 989–1015.

[GF96] Geymonat G. and F.Krasucki (1996) A limit model of a soft thin joint. *Partial Differential Equations ans Applications* pages 165–173.

[GW88] Glowiski R. and Wheeler M. (1988) Domain decomposition and mixed finite element methods for elliptic problems. In *First International Symposium on Domain Decomposition Methods for Partial Differential Equations (Paris, 1987)*, pages 144–172. SIAM, Philadelphia, PA.

[KPL94] Kischinhevsky M. and Paes-Leme P. J. (1994) Modelling and numerical simulations of contaminant transport in naturally fractured porous media. Technical Report 84, Universidade Federal Fluminense, Niteroi.

[Lio73] Lions J. (1973) *Perturbations singulières dans les problèmes aux limites et en contrôle optimal.* Springer-Verlag.

[RT87] Roberts J. and Thomas J.-M. (1987) *Mixed and hybrid methods,Handbook of numerical analysis*, volume 3 : Finite element methods. North Holland, Amsterdam.

41

Dynamic Load Balancing for Multi-Physical Modelling using Unstructured Meshes

V.Aravinthan[1], S.P.Johnson, K.McManus, C.Walshaw,

M.Cross

Introduction

As the complexity of parallel applications increase, the performance limitations resulting from computational load imbalance become dominant. A mapping that balances the workload of the processors in a parallel machine will typically increase the overall efficiency of a computation and so reduce the run-time. For many cases the computation times associated with a given task cannot be pre-determined even at run-time and so static load balancing returns poor performance. For many classes of problems, such as multi-physics problems involving fluid and solid mechanics with phase changes, the workload may change over the course of a computation and cannot be estimated beforehand. For such applications the mapping of loads to processors is required to change dynamically, at run-time if reasonable efficiency is to be maintained. This paper examines the issues of dynamic load balancing in the context of PHYSICA, a substantial three dimensional unstructured mesh multi-physics continuum mechanics computational modelling code based on finite volumes methods [CCB+96]. PHYSICA is primarily intended for the modelling of the processes involved in metal casting. It provides a range of modelling facilities including compressible Navier Stokes flow, heat,

---

1 Parallel Processing Research Group, Centre for Numerical Modelling and Process Analysis, University of Greenwich, London, UK. email: V.Aravinthan@gre.ac.uk
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

phase change, elastic-visco plastic, solid mechanics, chemical reaction and turbulence. The proposed *generic* strategy should be applicable to any unstructured mesh multi-physics parallel codes regardless of the application.

Parallel Processing

Multi-Physics modelling on a continuum scale brings together established techniques for structural mechanics and Computational Fluid Dynamics (CFD) to address problems which involve many physical phenomena. The significant non-linearity of the differential equations involved leads to a high computational demand from even moderate problem sizes. Parallel computation is required to satisfy this demand. In this respect, Single Program Multiple Data (SPMD) overlapping Domain Decomposition (DD) techniques have been used by authors to successfully parallelise unstructured mesh multi-physics applications e.g. [McM96].

Many computational problems assume a discrete model of a physical system, and calculate a set of values for every domain point in the model. These values are often functions of time, so that it is intuitive to think of the computation as marching through time. DD is used to map such problems onto multiprocessor machines so that regions of the model domain are assigned to each processor. The operational behaviour of such a system is often characterised as a sequence of steps, or iterations. During a step, a processor computes the appropriate values for its domain points. At the end of the step, it communicates any newly computed results required by other processors. Finally, it waits for other processors to complete their computation step and send the data required for the computation of the next step [CCB+96, McM96]. As practical experience is accumulated the focus is directed to the improvement of scalability and consequently load balancing.

Load Balance

Data distribution in an unstructured mesh DD parallel application is ordinarily based on a decomposition of the mesh into P subdomains calculated to balance the computational load on each processor. It is inevitable that the data dependence in a DD parallel application will require punctuation by frequent synchronisation points. A static mesh partition is unlikely to provide a good load balance when solving dynamic non-linear problems in parallel using an unstructured mesh. Prediction of the load associated with each mesh entity (grid point, face, element, etc.) is not simple. Even if the work-load is predicted accurately, the computational work associated with each portion of a problem's subdomain may change over the course of solving the problem. This can occur when the behaviour of the modelled physical system changes with time. For example, during the course of solving a problem, more work may be required to resolve features of the emerging solution. Load variations due to differences, for example, in element shape or perhaps the number of grid point adjacencies may be anticipated, but some effects such as changes in the discretisation or the physics associated with each entity may not be known until the code has run for some time. Cache effects and inhomogeneous architectures further complicate prediction. Adaptive meshing involving refinement and coarsening will inevitably suffer from significant load imbalance. Even with a fixed mesh, multi physical simulations which include the modelling of phase changes such as melting or solidification [CCB+96], can lead to significant imbalance. Here the application of flow calculations are required only for the liquid portion of the problem and similarly the stress calculations are only required for the solid portion. Such load imbalance may only be determined at run time.

Because of the synchronisation between steps, the system execution time during a step is effectively determined by the execution time of the slowest, or most heavily loaded processor. We can then expect system performance to deteriorate in time, as the changing resource demand causes some processor to become proportionally overloaded. One way of dealing with this problem is to periodically redistribute, or remap load among processors. Such Dynamic Load Balancing (DLB) schemes for moderately dynamic load changes have been addressed by many workers [SB94, Wat95] but DLB schemes for large and/or rapid load swings and generic DLB schemes remain a challenge. The presented *generic* algorithm monitors the work load at run time in order to predict the transfer of load between processors that will minimise the overall runtime of the computation.

Methodology

A practical solution of the DLB problem involves [Wat95]:

- Load Evaluation: Some estimators of a processor's load must be required to first determine that a load imbalance exists.
- Profitability Determination: Once the loads of the processors have been measured, the presence of a load imbalance can be detected. If the cost of the imbalance exceeds the cost of load balancing, then load balancing should be initiated.
- Load Transfer Calculation: Based on the measurements taken in the first phase, the ideal work transfers necessary to balance the computation are calculated.
- Load Migration: Workloads are transferred from one processor to another.

By decomposing the DLB process into distinct phases, experiments can be performed with different strategies for each of the above steps, allowing the impact of differing techniques to be investigated.

Load Evaluation

The effect of any load-balancing scheme is directly dependent on the quality of load evaluation. Good load measurement is necessary both to determine that a load imbalance exists and to calculate how much work should be transferred to relieve that imbalance. One way to easily overcome the performance peculiarities of a particular architecture is to measure the load of a task directly. Typical machines provide clocks with millisecond to microsecond level accuracy. These timing facilities can be used to time each task, providing accurate measurements in the categories of execution time, idle time and communication overhead. In fact, the user need not manually time the

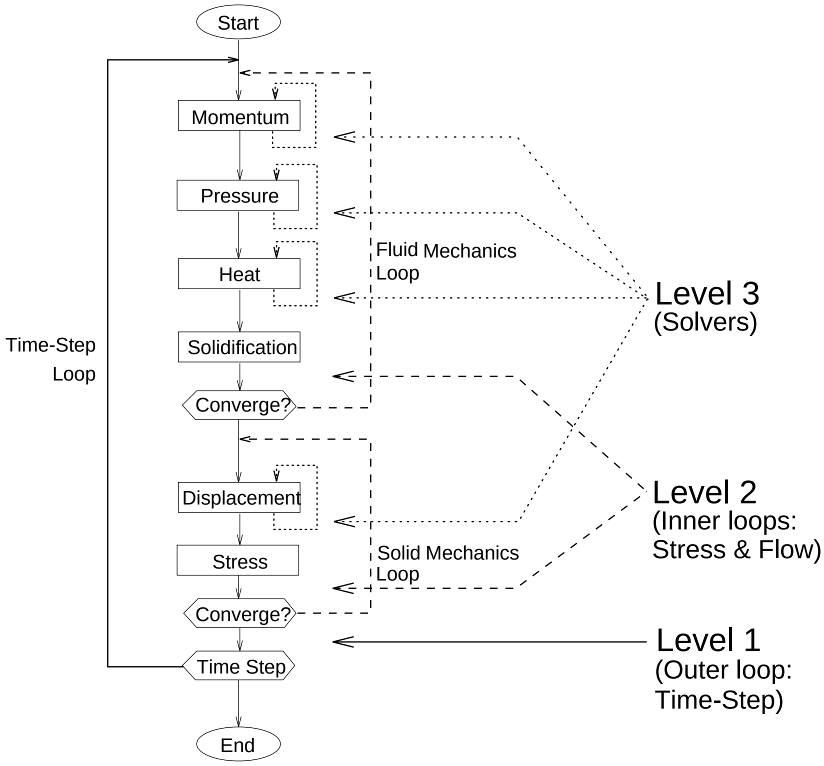


Figure 1 Different levels of loops that can be found in a typical CFD code.

code at all. These timings can be easily taken at the library level. A message-passing library could certainly be instrumented to accumulate time into various categories. Any time between communication operations would be labelled as runtime (execution time or CPU time), any time actually sending or receiving data would be tagged as communication time and any time waiting to receive a message would be accumulated as idle time.

Most scientific codes have different levels of loops within the code, for example a top level loop such as the time step loop and lower level loops such as the solver loops (see Figure 1). Thus the appropriate part(s) of the application code to time can vary widely between different codes. One code may necessitate the timing of the top loop level, but another code may require timing of the lower loop levels. These application codes usually have inherent synchronisation points within the loops. In particular, global norm calculations and other termination detection mechanisms typically involve a global sum, checking of convergence or some other reduction operation, the results of which are checked by each processor involved. These barrier operations provide a natural, clean point at which to initiate load balancing.

Profitability Determination

The DLB model is a run-time overhead and so must not initiate the rebalancing mechanism too frequently as this will waste time on moving the data around. On the other hand, if the rebalancing mechanism is initiated too infrequently, the load between the processors of the parallel machine can become badly balanced and hence the performance will deteriorate in time. Thus, it is important to correctly determine the criteria that will be used to decide when to re-distribute the data. Three inter-linking factors are involved:

- The level of imbalance in each section of the code
- The run-time for each code section
- The time required for calculating and performing a redistribution

These factors must be measured dynamically from the code and used to predict if the reduction in imbalance will compensate for the cost of the DLB algorithm.

When a barrier is initiated, the average load of all of the processors is determined. If the aggregate efficiency is below some user-specified limit, the workload is considered to be imbalanced. Even if a load imbalance exists, it may be better not to load balance, simply because the cost of load balancing would exceed the benefits of a better work distribution. The time required to load balance can be predicted directly by keeping a record of time taken in previous load balance(s). The expected reduction in run time due to load balancing can be estimated loosely by assuming efficiency will be increased to 100 percent or more precisely by maintaining a history of the improvement in past load balancing steps. If the expected improvement exceeds the cost of load balancing, the next stage in the load balancing process should begin.

A re-balancing decision heuristic is proposed here which assumes that the rate of change of imbalance between processors is always linear, that the re-balancing time is constant, and that re-balancing removes all imbalances. Response to large changes in load has the potential to over compensate and lead to instability in the algorithm. Stability of the algorithm also becomes an issue when the communication latency is high compared to the speed of variation of the load; unnecessary migrations should be avoided. It is imperative to avoid oscillation or cycling of the load across the processors and so a damping coefficient is incorporated into the algorithm to relax the movement of entities. The damping coefficient is calculated in response to the rate of change of work and consequently limits the speed of response to load changes.

The presented algorithm forms a cost function, t_{cost}, that models the time for re-distribution and the predicted application code run-time in relation to the rate of increase of imbalance (see equation 1) [AJM$^+$98]. The model is based on an instance in time and predicts what would happen under the model assumptions. It uses the number of iterations (n) between re-distributions to predict run-time. t_{cost} explicitly embodies two of the costs a re-mapping policy must manage: delay cost of re-balancing, and idle-time costs incurred by not re-balancing.

t_{cost} is given by:

$$t_{cost} = \int_0^t \left(\frac{n.i \times B}{2} + \frac{J}{n.i} \right) dt = \frac{Bn.i.t}{2} + \frac{J.t}{n.i} \qquad (1)$$

where i is the time taken for each iteration, B is the rate of increase of imbalance across the processors and J is the re-balancing time. The re-balancing time is then minimised with respect to n (see equation 2). The model (equation 3) predicts an optimal value for n that minimises the run-time prediction function. Re-distribution will be performed n iterations after the previous re-distribution.

$$\frac{dt_{cost}}{dn} = \frac{Bit}{2} - \frac{Jt}{n^2 i} = 0 \tag{2}$$

$$n = \sqrt{\frac{2J}{Bi^2}} \tag{3}$$

Work Transfer Calculation

After determining that it is advantageous to load balance, the amount of work-load that must be transferred from one processor to another must be calculated. Here, the JOSTLE mesh-partitioning tool is used [WCE97]. JOSTLE can be used to rebalance an existing partition in parallel whilst minimising the amount of data migrated. The load balance information is indicated by a weighted graph of the data (e.g. elements) that is passed to JOSTLE together with the current processor ownership array of the data. JOSTLE will then attempt to balance these weights in the resultant partition.

JOSTLE returns the new processor ownership for the local core data, indicating where the data should move. Using this array, processor ownership arrays for the secondary data are updated (e.g. faces and nodes based on an element's partition).

Data Migration

A DLB framework must also provide mechanism for actually moving the data from one processor to another. This includes identifying and updating moved entities data such as an element's temperature, pressure, U-velocity, V-velocity etc. The load balancing algorithm and consequent data movement must be very fast in comparison to the overall run-time. DLB is not merely a pre-processing step such as static partitioning since the algorithm and the consequent load migration may be performed frequently during the run time. Load rebalancing will only provide a performance gain if the time to rebalance is less than the decrease in run time consequent from rebalancing the code, so it is important to relate the overhead cost of remapping with the expected performance gain. Distributed memory systems require that all data are distributed and so data on each processor is locally numbered. Thus the framework must be able to handle entities that are locally numbered, which reduce the memory size by removing the need for any globally sized arrays to be stored. This maximises scalability of memory although locally numbered entities make the moving of data between processors more difficult.

The communication is implemented in two phases. The first phase constructs a movement set (within a processor) and a communication set (between processors) listing the entity numbers to be communicated and the processor to communicate with. The second phase performs the communication using the communication set calculated in the first phase for a particular variable.

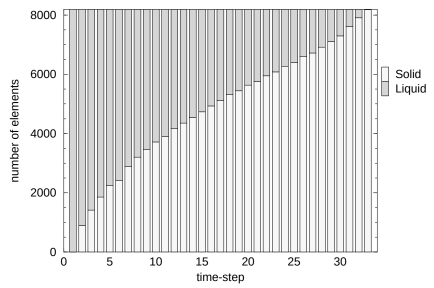


Figure 2 Solidification of a cooling bar

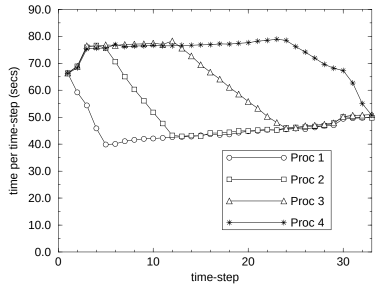


Figure 3 Time-step time without DLB

Some arrays (particularly geometry pointer arrays themselves) have to be converted to global numbers before any movement can take place. The data structures required for the parallel execution and DD consist of the local to global numbering array which stores the original global number of locally numbered entity. Hence, this local to global numbering array can be used to convert all the pointer arrays from local to global numbers. Following the construction of the new partition, the pointer arrays must be renumbered to the new local numbering scheme.

Results

The dynamic load balancing algorithm has been successfully implemented in PHYSICA at the time-step loop level (see Figure 1). It has been tested with a solidification test case which models a cooling metal bar. The bar begins all liquid at a temperature just above solidification and is cooled from one end so that a solidification front moves along the bar. After 30 time-steps, the bar is almost completely solid (see Figure 2).

Figure 3 shows the wall-clock time per time-step for each processor without DLB and Figure 4 shows the times with DLB. It can be seen in Figure 3 that the overall run-time is restricted by the load imbalance leading to the execution time of around 80 seconds per time-step for most of the calculation. Figure 4 shows the initial execution time correspondingly at almost 80 seconds per time-step being steadily reduced to a final figure of less than 60 seconds. A 20% reduction in the overall run-time produced by DLB is shown in Figure 5.

Conclusion

For a given problem then as P increases, the importance of load balance in measured parallel performance becomes increasingly significant. For dynamic inhomogeneous load imbalances, characteristic of multi-physics problems, it may not be possible to obtain a *good* load balance. Nevertheless it is clear that DLB can reduce the

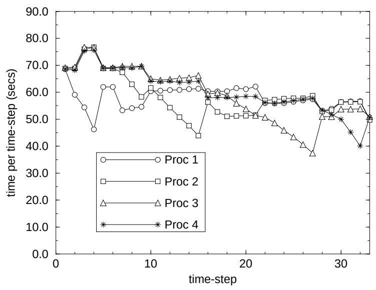


Figure 4 Time-step time with DLB

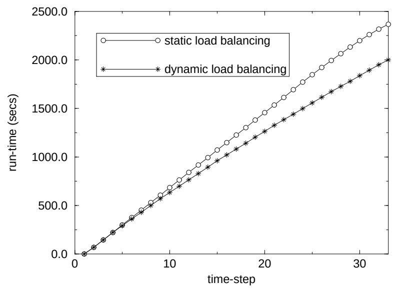


Figure 5 Overall run-time

load imbalance in an initial partition and so provide a worthwhile performance improvement.

The development of parallel JOSTLE has provided an opportunity to advance the state of the art in practical unstructured mesh parallel application and in particular DLB. The DLB scheme in this paper has been developed and tested in PHYSICA using test cases that illustrate the challenging issues in load balancing for dynamic inhomogeneous problems. Information extracted at runtime is used to continuously monitor and migrate the workload as the developing solution causes the workload to move across the problem space. The resulting methodology is not only successful in reducing run-time but should also be sufficiently *generic* to be applicable to a diversity of unstructured mesh based codes.

REFERENCES

[AJM+98] Arulananthan A., Johnson S., McManus K., Walshaw C., and Cross. M. (1998) A generic strategy for dynamic load balancing of distributed memory parallel computational mechanics using unstructured meshes. In *Proc Parallel CFD 1997*, pages 43–50.

[CCB+96] Cross M., Chow P., Bailey C., Croft N., Ewer J., Leggett P., McManus K., and Pericleous K. A. (1996) PHYSICA - a software environment for the modelling of multi-physics phenomena. In *Proc ICIAM 1995*.

[McM96] McManus K. (1996) *A strategy for mapping unstructured mesh computational mechanics programs onto distributed mesh parallel architectures.* PhD thesis, Computing and Mathematical Science, University of Greenwich.

[SB94] Sergent T. L. and Berthomieu B. (1994) Balancing load under large and fast load changes in distributed computing systems - a case study. In *Parallel Processing: CONPAR 94 - VPP VI*, pages 854–865. Springer Verlag.

[Wat95] Watts J. (1995) A practical approach to dynamic load balancing. Master's thesis, Scalable Concurrent Programming Laboratory, California Institute of Technology.

[WCE97] Walshaw C., Cross M., and Everett M. (1997) Parallel Dynamic Graph Partitioning for Adaptive Unstructured Meshes. *J. Par. Dist. Comput.* 47(2): 102–108.

42

Domain Decomposition in High-Level Parallelization of PDE Codes

Xing Cai [1]

INTRODUCTION

We introduce a high-level approach to parallelizing sequential software for solving partial differential equations (PDEs). In short, we use domain decomposition (DD) at a higher level than that of linear algebra. Combined with extensive use of object-oriented (O-O) programming techniques, this approach promotes an efficient, flexible and systematic parallelization process. We present the software engineering aspects of this approach by explaining a generic implementation framework in the O-O scientific computing environment—Diffpack. Finally, a concrete case study will demonstrate the efficiency and flexibility of the parallelization process in this scenario.

Domain decomposition and parallel PDE codes

DD methods (see e.g. [CM94, SBG96]) are not only numerically efficient, but also inherently parallel. The original solution domain is partitioned into many subdomains that can be assigned to different processors of a parallel computer. The global solution is then found by an iterative process in which subdomain solves are invoked under a global administration. Therefore, using DD as the mathematical foundation for writing parallel PDE codes is becoming quite standard nowadays. However, the common

1 Department of Informatics, University of Oslo, P.O. Box 1080, N-0316 Oslo, Norway.
email: xingca@ifi.uio.no
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

practice has been to address the DD methods directly at the level of linear algebra, i.e., in terms of parallel matrix-vector operations. In addition, the implementation of parallel PDE codes is often done using procedural programming languages. These two factors together bring the consequence that parallel PDE codes need often to be written entirely from scratch, thus making little use of existing sequential PDE codes.

Object-oriented programming and Diffpack

In recent years, the application of object-oriented (O-O) programming techniques in developing PDE software has shown its many advantages. The C++ programming language, which is a foremost representative in this respect, has inherent features that are well suited for the complicated process of numerical PDE solution. (We refer to [BN94, Lan99] for an introduction to this topic.) The most attractive features of C++ are modularity, polymorphism and inheritance. So extensive code reuse is a direct advantage of O-O programming. The possibility of a modular implementation of mathematical abstractions gives rise to an application independent PDE kernel, which is reusable in many PDE applications from different disciplines of scientific computing. Moreover, other functionalities such as I/O, visualization and automatic result reporting can also be extracted into libraries, because O-O programming promotes a unified interface of PDE simulators. The application programmer needs only to concentrate on critical numerics, at a high abstraction level if desired. Therefore reliable, flexible and extensible simulators can be developed in an efficient way.

Diffpack is an O-O scientific computing environment (see [Lan99, Dif]). The design of Diffpack has taken numerical efficiency into consideration by confining O-O techniques to high-level administrative tasks, while using low-level C codes and carefully constructed for-loops in CPU intensive numerics. The C++ Diffpack libraries contain, among other things, user-friendly objects for I/O, GUIs, arrays, linear systems and solvers, grids, scalar and vector fields, visualization and grid generation interfaces. This makes it very easy to build prototypical PDE simulators based on reliable and optimized Diffpack components.

The simulator-parallel model

Parallel computing has the potential for not only reducing the computation time, but also concentrating memories belonging to different processors to carry out larger calculations. So the migration of sequential PDE simulators to multiprocessor platforms is well motivated. Basically, we want the time used to solve a PDE on a parallel computer with P homogeneous processors to be about T/P, where T is the solution time needed by a sequential simulator. This requires a good numerical scheme that promotes an optimal parallelization of a sequential PDE simulator.

In this paper, we will focus on the overlapping DD methods of the additive Schwarz type, partly due to their simple algorithmic structure and partly due to the readiness for parallelism. An important observation about the additive Schwarz methods is that the main ingredient is the subdomain solver that carries out local and sequential operations. Furthermore, these sequential operations are of the same type as those needed to solve the original PDE on the whole domain. So if a mechanism can be found to apply an existing sequential simulator in subdomain solves on each subdomain, the parallelization work will reduce to global administration and inter-subdomain data exchange. The O-O programming techniques serve perfectly as such a mechanism. First, they can produce sequential PDE simulators with unified generic interfaces, which easily allow modification and extension in order to be incorporated into a parallel setting. Second, the data encapsulation feature of O-O programming makes it possible to hide the computational details, e.g., the matrix-vector operations. As a result, O-O programming techniques allow code reuse at the level of subdomain simulators, which is higher than the level of linear algebra.

We hence introduce the so-called *simulator-parallel* programming model that was first proposed in [BCLT97]. In this model, each processor of a parallel computer hosts one or several subdomains. Each subdomain is then handled by a subdomain solver, which is easily extended from an existing sequential simulator designed for the whole domain. It is the basic building block of overlapping Schwarz methods—the subdomain solver—which relates our simulator-parallel model to DD methods, because an O-O sequential PDE simulator for the whole domain has all the functionalities needed in the subdomain solves. Of course, running DD methods in parallel needs a global administration and additional communication functionalities. We will show in the next section that O-O programming techniques are the right tool to produce a generic implementation framework where users can plug-and-play ready-built components for global administration and communication, so that parallel PDE simulators can be developed on the basis of existing sequential simulators in an efficient and systematic way.

A GENERIC IMPLEMENTATION FRAMEWORK

Our objective is a generic implementation framework that is flexible, extensible and portable, so that the parallelization work following the simulator-parallel model is relatively straightforward. On the whole, the efficiency and flexibility of the existing sequential simulator will be maintained, while the intrinsic numerical efficiency of DD can enhance the overall efficiency of the resulting parallel simulator. We also allow the user to make run-time decision of whether using additive DD methods as stand-alone iterative methods or preconditioners for Krylov subspace methods.

We consider such a generic framework consisting of three main parts: the sequential subdomain simulators, a communication part and a global administrator. Two class hierarchies, whose base classes are `SubdomainSimulator` and `Communicator`, are built to represent the sequential subdomain simulators and to handle the needed communications, respectively. The different classes in the class hierarchies are designed to be used in different situations. In addition, the design of the global administrator offers great flexibility. It allows the user to choose, among other things, the specific numerical method at run-time. The part inside the global administrator that makes connections with the subdomain simulators and the communication part can also be easily modified by the user.

Subdomain simulators

The base class `SubdomainSimulator` gives a generic representation of any sequential subdomain simulator in our framework. Functionalities of `SubdomainSimulator` include a numerical discretization scheme and an assembly process for building up the local linear system. A linear algebra toolbox also exists in `SubdomainSimulator` to control the choice of the local solution method, preconditioner, stopping criterion etc. We have made most of the member functions of `SubdomainSimulator` to be *pure virtual*, so they need to be overridden in a derived subclass. These member functions constitute a standard interface shared by all the subdomain simulators. It is through this generic interface that the communication part and the global administrator of the implementation framework operate. Adapting an existing sequential simulator also becomes easy, because most of the work consists merely of binding the pure virtual member functions in `SubdomainSimulator` to the concrete member functions in the existing simulator. Building up the class hierarchy, we have derived `SubdomainFEMSolver` for simulators solving a scalar/vector elliptic PDE discretized by finite element methods, and `SubdomainFDMSolver` for simulators using finite difference discretizations. Furthermore, a subclass named `SubdomainMGSolver` is derived from `SubdomainFEMSolver` to represent simulators using multigrid V-cycles in subdomain solves.

Communication

During parallel DD iterations, the concrete communication between processors is in form of exchanging messages and is handled by objects of `Communicator`. On each processor, there is one such object connecting to the local subdomain simulator(s). The reason for separating the communication part from the global administrator is to hide the low-level message passing codes and instead offer convenient high-level communication commands. It is thus possible to change MPI, which is used in the current implementation, to another message passing standard without affecting the other parts of the framework. Different subclasses of `Communicator` are derived to handle different situations. For example, class `CommunicatorFEMSP` specializes in communication between subdomain simulators using the finite element discretization, whereas class `CommunicatorFDMSP` works for subdomain simulators using the finite different method.

The global administrator

The inheritance feature of C++ is reflected in the above two class hierarchies `SubdomainSimulator` and `Communicator`. The user can either take ready-built class objects directly from these two hierarchies and use them in a specific application, or derive new subclasses to incorporate new adaptations. However, the O-O design of the global administrator offers further flexibility and lets the user plug-and-play at run-time. For example, the user can decide whether to use DD as a stand-alone iterative method or a preconditioner for a specific Krylov subspace method. The user is also free to choose between running one-level and two-level DD by removal or addition of a coarse global grid. We have thus devised a main administrator class `PdeFemAdmSP`

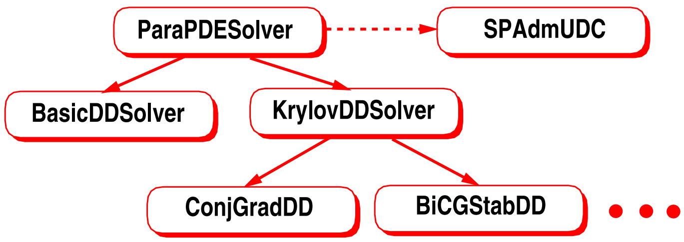


Figure 1 The class hierarchy of `ParaPDESolver` and its connection to `SPAdmUDC`. The solid arrows indicate inheritance ("is-a" relationship), whereas the dashed arrow indicates "has-a" relationship.

whose basic structure is as follows:

```
class PdeFemAdmSP
{
protected:
  ParaPDESolver_prm psolver_prm;
  Handle(ParaPDESolver) psolver;
  SPAdmUDC* udc;
  ...
};
```

In above, `ParaPDESolver_prm` is an object containing many parameters to be chosen by the user at run-time. Among the parameters there are: 1. flag indicating whether the overlapping Schwarz method should be used as a stand-alone iterative method, 2. name of the desired Krylov subspace method (when an overlapping Schwarz method is used as a preconditioner), and 3. number of maximum iterations, prescribed accuracy and type of the convergence monitor etc.

The second component of class `PdeFemAdmSP` is `ParaPDESolver` whose class hierarchy is depicted in Figure 1. In this hierarchy, class `BasicDDSolver` represents an overlapping Schwarz method to be used as a stand-alone iterative solver, and subclasses of `KrylovDDSolver` use one overlapping Schwarz iteration as the preconditioner. At run-time, when the user has chosen the parameters by e.g. filling items on a user-friendly menu, a concrete object of `ParaPDESolver` will be created. (We remark that `Handle(X)` is a safer pointer for class `X` in Diffpack.) The involved subdomain solves will then be undertaken by a sequential subdomain simulator, which `ParaPDESolver` gets connection through the third and last component of `PdeFemAdmSP`, namely `SPAdmUDC`. It is through `SPAdmUDC` that `ParaPDESolver` invokes member functions of the subdomain simulator.

In addition to making connection between `ParaPDESolver` and the subdomain simulators, `SPAdmUDC` is also responsible for getting access to the communication part for the global administrator. We remark that "UDC" stands for "user-defined-codes" and is used here to indicate that the user has the possibility of making modifications of the member functions. In a parallel simulation, one `SPAdmUDC` object resides on each processor and has one `Communicator` object plus one or several `SubdomainSimulator` objects under its control.

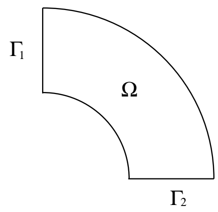


Figure 2 The solution domain for the linear elasticity problem.

A CASE STUDY

We demonstrate the efficiency and flexibility of the generic implementation framework by parallelizing an existing sequential Diffpack simulator for solving a linear elasticity problem in 2D. The mathematical model is the following vector PDE:

$$-\mu\Delta\mathbf{U} - (\mu+\lambda)\nabla\nabla\cdot\mathbf{U} = \mathbf{f},$$

where $\mathbf{U} = (\mathbf{u_1}, \mathbf{u_2})$ is the primary unknown and $\mathbf{f}$ is a given vector function, μ and λ are constants. The 2D domain is a quarter of a hollow disk (see Figure 2). On the boundary the stress vector is prescribed, except on Γ_1, where $u_1 = 0$, and on Γ_2, where $u_2 = 0$.

As stated above, a sequential Diffpack simulator exists for solving the 2D linear elasticity problem. The simulator is represented by class `Elasticity2D`, which has a grid, a finite element field for the unknowns, a member function modelling the integrands in the weak formulation, a linear system toolbox, and some standard functions for prescribing the boundary conditions. To extend `Elasticity2D` with the generic interface required by the implementation framework, and allow multigrid V-cycles as subdomain solvers, we derive a new class `SubdomainELSolver` as the subclass of *both* `Elasticity2D` *and* `SubdomainMGSolver`. A simplified definition of `SubdomainELSolver` is as follows:

```
class SubdomainELSolver : public SubdomainMGSolver,
                          public Elasticity2D
{
  ...
  virtual void markEssBCs ();
  virtual void createHierMatrices ();
};
```

The above code segment shows that the main work of class `SubdomainELSolver` is to give explicit definitions of the pure virtual member functions of class `SubdomainMGSolver`. This is essentially done by binding the virtual functions of `SubdomainMGSolver` to concrete member functions of `SubdomainELSolver`. For example, the member function `markEssBCs` uses a corresponding function of `Elasticity2D` to mark essential boundary conditions on all levels of subgrids, while the member function `createHierMatrices` creates stiffness matrices on all the grid levels and has the following implementation:

```
void SubdomainELSolver:: createHierMatrices () {
  for (int i=no_of_grids; i>=1; i--)
    Elasticity2D::makeSystem(dofs(i)(),Amats(i)(),rhs_vecs(i)());
}
```

The second job a user needs to do is to make a connection between the extended sequential simulator `SubdomainELSolver` and the global administrator. This is achieved by deriving a new class `ELSolverSP` as a subclass of `SPAdmUDC`. Inside `ELSolverSP`, the user basically creates an object of `SubdomainELSolver` for each subdomain and puts them under the control of `SPAdmUDC`. Finally, the Diffpack main program will look as follows:

```
int main (int nargs, const char** args)
{
  initDiffpack (nargs, args);   // MPI_Init etc called inside
  MenuStream* menu = new MenuStream;
  menu->setInputFile ("el_parameters.txt");
  menu->init ("Linear Elasticity Test","DD Approach");

  ELSolverSP udc;
  PdeFemAdmSP adm;
  udc.define (*menu, MAIN); adm.define (*menu, MAIN);
  menu->prompt();
  udc.scan (*menu); adm.scan (*menu);
  adm.attachSPAdmUDC (&udc);
  adm.init ();
  adm.solve ();
}
```

For the numerical experiments we use a structured 241×241 curvilinear global grid. We use one processor per subdomain and denote the number of processors by P. For $P > 1$ the additive Schwarz method is used as the preconditioner for a parallel BiCGStab method for solving the linear system. The stopping criterion requires that the discrete L_2-norm of the global residual is reduced by a factor of 10^6. We introduce a global coarse grid to allow two-level DD iterations and obtain the overlapping subgrids by extending a non-overlapping partition with a factor of $\frac{1}{15}$ in each direction. One local multigrid V-cycle with one pre- and post-SSOR-smoothing is used in subdomain solves. For $P = 1$ a sequential BiCGStab method preconditioned by one global multigrid V-cycle is used. The CPU measurements obtained on an SGI Cray Origin 2000 machine with R10000 processors are listed in Table 1, where I denotes the number of BiCGStab iterations.

CONCLUDING REMARKS

We have shown that addressing DD at the level of subdomain simulators, combined with extensive use of O-O programming techniques, results in an efficient, flexible and systematic process for producing parallel PDE codes. A generic implementation framework is presented in this paper, together with a concrete case study where we

Table 1 Solution of the linear system for the linear elasticity problem.

P	CPU	I	subdomain grid
1	66.01	19	241×241
2	24.64	12	129×241
4	14.97	14	129×129
8	5.96	11	69×129
16	3.58	13	69×69

parallelize an existing sequential Diffpack simulator for a linear elasticity problem. Finally, we mention that the generic implementation framework can be easily extended to parallelize sequential simulators for nonlinear elliptic PDEs and parabolic problems, where the computational kernel of the DD methods is still subdomain solves (see e.g. [Tai98, TE98]).

Acknowledgments. This work has received support from The Research Council of Norway (Programme for Supercomputing) through a grant of computing time. The author also acknowledges the help from Prof. Aslak Tveito, Dr. Hans Petter Langtangen and Dr. Are Magnus Bruaset.

REFERENCES

[BCLT97] Bruaset A. M., Cai X., Langtangen H. P., and Tveito A. (1997) Numerical solution of PDEs on parallel computers utilizing sequential simulators. In et al. Y. I. (ed) *Scientific Computing in Object-Oriented Parallel Environment, Springer-Verlag Lecture Notes in Computer Science 1343*, pages 161–168. Springer-Verlag.

[BN94] Barton J. J. and Nackman L. R. (1994) *Scientific and Engineering C++. An Introduction with Advanced Techniques and Examples.* Addison-Wesley.

[CM94] Chan T. F. and Mathew T. P. (1994) Domain decomposition algorithms. In *Acta Numerica 1994*, pages 61–143. Cambridge University Press.

[Dif] Diffpack Home Page. http://www.nobjects.com/Products/Diffpack.

[Lan99] Langtangen H. P. (1999) *Computational Partial Differential Equations – Numerical Methods and Diffpack Programming.* Springer-Verlag.

[SBG96] Smith B. F., Bjørstad P. E., and Gropp W. (1996) *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations.* Cambridge University Press.

[Tai98] Tai X.-C. (1998) A space decomposition method for parabolic problems. *Numer. Method for Part. Diff. Equat.* 14(1): 27–46.

[TE98] Tai X.-C. and Espedal M. (1998) Rate of convergence of some space decomposition methods for linear and nonlinear problems. *SIAM J. Numer. Anal.* 35(4): 1558–1570.

43

An overview on current multiphysics software strategies for coupled applications with interacting physics on parallel and distributed computers

P. CHOW[1] and C. ADDISON[2]

INTRODUCTION

Computational Multiphysics is generally defined as the combining/coupling of technologies in various developed fields and disciplines, e.g. fluid dynamics, heat transfer, solid and structure mechanics, and electromagnetic, to accurately model the physical phenomena and the interacting processes.

Previously, it was the simplifying assumptions used - because of past numerical algorithms and computing technology - that had constrained the models. Now with advances in computer technology, modelling techniques and numerical methods (e.g. domain decomposition methods - applied with good success in linear and non-linear solvers, exchange of quantities between non-matching grids, and data partitioning in parallel computations), it is no longer the case. Today's engineers and scientists have the tool, in Multiphysics software, to study and tackle tough problems with interacting physics, such as:

1 Fujitsu European Centre for Information Technology Ltd, 2 Longwalk Road, Stockley Park, Uxbridge, Middlesex UB11 1AB, U.K. email: chow@fecit.co.uk
2 Address same as above.
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

- Fluid-structure interaction in bridges and high-rise structures;
- Metal shape casting of high-quality engineering components.

Employing it on parallel computers ensures the solution is deliverable in a practical timeframe.

The big question is how to combine the different disciplines in order to solve and obtain accurate and efficient solutions to these types of multidisciplinary applications. In this paper, we give an overview of current mesh-based computational software technology for Multiphysics simulations. Software strategies such as Single-Code and Multiple-Codes models, coupling libraries, and agents request brokers, are covered.

MODELLING & COMPUTING PERSPECTIVE

Constructing computational "*Analysis-Engines*" for Multiphysics applications is NOT simple. Commonly involving a multidisciplinary team that can include engineers, physicists, chemists, mathematicians, computer and computational scientists, if not all and more. In larger projects it can span to multiple groups and organizations that may geographically be located miles apart. For international ones it could mean different continents of the world.

Often each discipline has their preferred methods. In numerical discretization, the Finite Element Method is commonly associated with Solid and Structure Dynamics. While the Finite Volume Method is popular in Computational Fluid Dynamics. In programming languages, Fortran and C are commonly associated with mature or legacy codes, with C mostly in visualization and Fortran in analysis. Of the new object-oriented languages, C++ is popular with the scientific community. Language inter-operability is difficult in both code integration and reuse of libraries when they are coded in different programming languages. Inter-operability between Fortran and C is included in the next revision of the language standard. But no such standard is near for object-oriented languages, primarily becasue of difficulties with structures and objects.

Multiphysics applications generally involve solving highly non-linear system of equations, commonly in three-dimensional space, time-dependent and complex geometry with unstructured meshes. Meshes into millions of elements and 10 or more unknowns in applications such as transport vehicles, engines, and electronic packaging are anticipated. Parallel processing is a core element to running and to delivering the solution in a practical time.

SOFTWARE STRATEGIES

Today, the software strategies used to put together multiphysics analysis-engines can be categorized into two types of models, Single-Code and Multiple-Codes. In scalar processing settings, Multiple-Codes models would have a number of independent analysis codes with a coordinating program to oversee controls. Often there is a common database and the utilities for mapping the data between the analysis codes and/or the database. In Single-Code models, the analysis-engine is a single analysis code.

The coming subsections go into each model in more detail and the parallel processing paradigms. Also, where possible, giving some examples of the software in each category.

Single-Code Model

The Single-Code model is more common with newer analysis codes for addressing multiphysics problems. This is no surprise as new methods and algorithms are developed for coupled analysis, it is natural to implement them under a single-code model (single program). For example, codes such as SPECTRUM [spe] from Centric Engineering System, Inc. and PHYSICA [phy] from University of Greenwich come under this category.

There are two approaches in which multiphysics are generally achieved under this model:

- *One fully coupled model*: typically using every specific numerical technique to solving the entire coupled non-linear system of equations in a single step. The matrix system is in general ill-conditioned, as a result of the difference in stiffness of the various physical models. Currently, it is very application specific;
- *Build on and combine existing models*: solve each discipline using established methods and using an iterative scheme, such as Newton, to couple the interacting processes between the various phase of the analysis. In most cases some kind of coupling algorithm is required for time integration, solution stability and speed of convergent. The approach shares a lot of the common coupling elements with the Multiple-Code model.

One major advantage of the model is all the data are in one Single Data Space, i.e. no out-of-memory data transfer between various phase of analysis. Consequently, the overall solution procedure is far more efficient than with Multiple-Code models. A possible downside is that these analysis programs are generally a closed system, with only provision for adding boundary conditions, material models, and possibly calling of user routines, in parts of the code. The ability to change internal elements of the code is not possible. One exception is the PHYSICA toolkit. It has an open software analysis-engine framework based on a component-based paradigm to support component exchanges [PCK+98].

Multiple-Codes Model

The Multiple-Codes model is primarily using several independent analysis codes, in an integrated way and frequently with an element of domain decomposition, to solve Multiphysics and coupled problems. Commonly, these codes are well established in a particular field, such as FLUENT [flu] for Computational Fluid Dynamics and NASTRAN [nas] for Computational Structural Dynamics. Often, industries such as automotive and aerospace would have most, if not all, the analysis packages for engineering and design. Running them in a coordinated way and with proper transfer of information between the analysis codes, a framework for solving coupled problems is formed. This is the Multiple-Codes model towards Multiphysics analysis.

The two approaches in which this kind of analysis-engine is usually put together are:

- *From Single Source*: as a fully integrated product supplied by a single software source/vendor, customarily using vendor's own analysis codes. An example is the ANSYS/Multiphysics [ans] suite. Generally these are closed systems.
- *From Multiple Sources*: commonly a purpose built open integration framework based on component paradigm to support a suite of analysis packages from various sources/vendors. Supporting utilities such as data translation, files and databases manager, solution and job control, are essential tools. So is a common application interface for the framework. The ACTS Toolkit [act] and JULIUS [jul] are two examples that come under this category.

These models often have multiple data sets, such as databases, of the same model case for each analysis package (generally not so with single vendor situation). Duplication is common on data such as geometry, material properties and, in some cases the mesh. The flow of data between the packages is frequently done via files or handles. Often there is an extra data translation process for non-matching meshes and variables on different spatial locations (e.g. values from vertex to element). For transient problems with large number of time-steps this can be very inefficient.

With the Multiple-Codes model the integration of the different disciplines' analysis codes is usually done with a loose coupling algorithm [RCJ+95][JR97], as tight coupling is generally viewed not possible without modification to source code of analysis solver. The upside of loose coupling is the parts in the analysis-engine, the analysis codes, can interchanges to cater for application needs.

PARALLEL PROCESSING

Parallel processing add an extra dimension to the analysis-engine software strategy. In the Single-Code model, it very much follows the standard parallel strategies found in conventional analysis codes. The most popular being Single Program Multiple Data (SPMD) paradigm.

The Multiple-Codes approach is generally more complicated but does open up several interesting possibility and models, such as intelligent agent object request broker paradigm [JW98][cor92]. The big question here with the Multiple-Codes model is how to inter-operate with efficient data transfer? For whichever model domain decomposition technology is a key element.

In parallel and distributed environment, some challenging issues for the Multiple-Codes approach are:

- Different codes have different parallel models;
- Communication between codes for data transfer is necessary - a potential bottleneck;
- Dynamic load balancing made even harder in Multiple Instruction Multiple Data (MIMD) setting.

Tools and libraries are being developed to address these points. For example, the COupled COmmunication LIBrary (COCOLIB) being developed in a European Esprit project CISPAR [cis] is addressing these via a library approach. The purpose is to enable matching and exchange of surface interface data on parallel computers, where the analysis codes themselves are parallel codes. The message passing interface (MPI) standard [mpi94] is use in the implementation of COCOLIB, MPI has the facilities to keep communicate separate between the individual components. In the integrated framework side ACTS Toolkit [act], JULIUS [jul] and TOOLSHED [too], are few examples that are in this category.

In the Metacomputing environment, there are several interesting developments and potential models on the horizon. The Internet could potentially be the ultimate compute engine if it can be harnessed and made viable for computational mechanics applications. The inter-operability standard for this kind of environment, such as CORBA and JAVA, are they sufficient for scientific applications such as multiphysics? Are the infrastructure in-placed? What about security matters? These and other infrastructure related questions are being addressed, largely driven by needs of business, commerce, and entertainment sectors. One potential model from this for the Multiphysics analysis-engine is the intelligent agent object broker computing paradigm [JW98][cor92]. One such example is the SciAgents [J.R98][T.T96] from Purdue University. It is agent-based and uses a loose coupling algorithm for distributed components (analysis codes/models) with interface relaxation method for solving composite problems. As the current commercial investments continue to pour heavily into Internet and Web-based solutions for business, commerce, and entertainment, and encouraging an economy of commodity components and services. It is highly that this and related models will develop more rapidly in the near future than the other computing paradigms.

CONCLUDING REMARKS

The demand is increasing in computational mechanics to have the ability to simulate engineering processes with interacting physics. Putting together such an *Analysis-Engine* for Multiphysics simulations is not simple, often requiring a multidiscipline team that commonly involves different numerical methods and codes written in different programming languages. With large models anticipated, meshes into millions of elements and 10 or more unknowns, parallel processing is a core part of the software strategy to running and to delivering the solution in a practical time.

Here, an overview is given on current, and near future, software strategies and models for mesh-based analysis-engines to address Multiphysics problems. The strategies can be categorized into Single-Code and Multiple-Codes models. Each model has a number of approaches depending on the degree of openness, parallel processing, and application needs. Future models maybe more Internet and Web-based related (e.g. intelligent agent object broker computing paradigm) as economy of commodity components (both hardware and software) and services becomes more readily available. But for whichever model domain decomposition technology is a key element.

REFERENCES

[act] ACTS toolkit: Advanced computational testing and simulation toolkit. Department of Energy 2000 Project. U.S.A.

[ans] ANSYS. ANSYS Inc. (http://www.ansys.com). Canonsburg, PA, U.S.A.

[cis] CISPAR - open interface for coupling of industrial simulation codes on parallel computers. ESPRIT Project 20161. E.U.

[cor92] (1992) The common object request broker: Architecture and specification. The Object Management Group and X/Open, John Wiley and Sons.

[flu] FLUENT. Fluent Inc. (http://www.fluent.com). Lebanon, NH, U.S.A.

[JR97] J.R.Cebral and R.Lohner (1997) Fluid-structure coupling: Externsions and improvements. *AIAA-97-0858* .

[J.R98] J.R.Rice (1998) An agent-based architecture for solving partial differential equations. *SIAM news* 31(6).

[jul] JULIUS: Joint industrial interface for end-user simulations. ESPRIT Project 25050. E.U.

[JW98] Jennings N. R. and Wooldridge M. J. (eds) (1998) *Appnt Technology: Foundations, Applications, and Markets.* Springer-Varlag.

[mpi94] (1994) M.P.I. forum, MPI: a message-passing interface standard. Journal of Supercomputer Applications.

[nas] NASTRAN. The MacNeal-Schwendler Corporation (http://www.macsch.com). Los Angeles, California, U.S.A.

[PCK+98] P.Chow, C.Baily, K.McManus, M.Cross, and C.Addison (1998) A single-code software model for multiphysics analysis-engine on parallel and distributed computers with the physica toolkit. In Lai C.-H., rstad P. E. B., Cross M., and Widlund O. B. (eds) *Proceedings of the 11th international domain decomposition conference.* London, U.K. submitted.

[phy] PHYSICA. University of Greenwich (http://physica.gre.ac.uk). London, U.K.

[RCJ+95] R.Lohner, C.Yang, J.R.Cebral, J.D.Baum, H.Luo, D.Pelessone, and C.Carman (1995) Fluid-structure interaction using a loose coupling algorithm and adaptive unstructured grids. In M.Hafez and K.Oshimna (eds) *Computational Fluid Dynamics Review 1995.* John Wiley.

[spe] SPECTRUM. Centric Engineering System Inc. (http://www.centric.com). Sunnyvale, CA, U.S.A.

[too] TOOLSHED: Tools for high productivity engineering design. ESPRIT Project 20231. E.U.

[T.T96] T.T.Drashansky (1996) *An Agent Based Approach to Building Multiple Disciplinary Problem Solving Environments.* PhD thesis, Department of Computer Science, Purdue University.

44

A single-code software model for Multiphysics analysis-engine on parallel and distributed computers with the PHYSICA toolkit

P.Chow[1], C.Bailey[2], K.McManus[2], C.Addison[1] & M.Cross[2]

INTRODUCTION

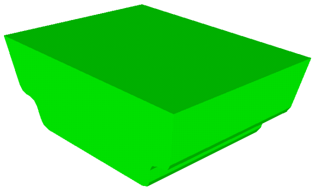
In reality, all engineering processes are multiphysics. It is the simplifying assumptions used previously - because of past numerical algorithms and computing technology - that constrained the models. Now with advances in computer technology, modelling techniques and numerical algorithms (e.g. domain decomposition methods - applied with good success in linear and non-linear solvers, exchange of quantities between non-matching grids, and parallel computations), it is no longer the case. Today's engineers have the tool, in Multiphysics software, to tackle tough engineering problems with interacting physics. Employing it on parallel computers ensures the solution is deliverable in a practical timeframe.

In this paper, we use the PHYSICA toolkit [phy] to provide an overview of a single-code software model strategy of an "*Analysis-Engine*" for Multiphysics simulations on parallel and distributed computers. A mould-ingot casting case that includes analysis of thermal convection, solidification and stress is used to illustrate the approach.

1 Fujitsu European Centre for Information Technology Ltd, 2 Longwalk Road, Stockley Park, Uxbridge, Middlesex UB11 1AB, U.K. email: chow@fecit.co.uk
2 School of Computing & Mathematical Sciences, University of Greenwich, Wellington Street, Woolwich, London SE18 6PF, U.K.
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

MODELLING INTERACTING PROCESSES

Considering a simple geometry sample case from the metal casting process, see Figure 1 for a Lead-Ingot casting. Table 44 gives the physical phenomena and related analyses used in an industry funded mould thickness investigation [C.B]. Individually, there are very good analysis codes to solve each of the physical phenomena. Few have the capability for a number of them together, and fewer still do it all.



Ingot

Mould

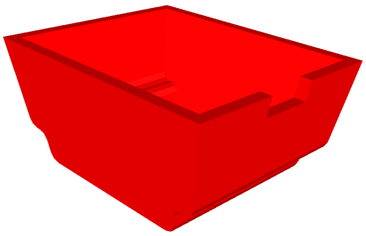


Figure 1 Casting sample case

Table 1 Physics and analysis for the sample case

Physical phenomena	**Type of Analysis**
Thermal convection	Fluid flow analysis
Cooling, solidification (Liquid-Solid)	Thermal and phase change analysis
Residual stress	Solid mechanics analysis
Deformation and gap formation	Contact analysis
Time-dependent	

The analysis elements in Table 44 provide the foundation for a Casting analysis-engine. More complex casting processes could also have:

- Mould filling
- Multiphase (Gas-Liquid-Solid)
- Chemical reactions

- Electromagnetic (mechanism for fluid flow control)
- Microstructures (predicting voids, cracks, etc.)

Putting together a computational analysis-engine for such spectrum of phenomena is not a simple task. It commonly involves a multidisciplinary team or groups using different numerical methods and techniques. The next section gives the PHYSICA toolkit's open software framework that can help with such engine construction under a single-code paradigm.

OPEN SOFTWARE FRAMEWORK

Apart from software related component such as input-output, error handling, etc., a common component in all computational mechanics codes is some kind of a mesh or computational lattice representing the domain for which the unknown variables are solved and holding of relevant information (e.g. material properties, boundary regions). For 3-dimensional space, all these quantities commonly relate to one of four mesh entities: nodes (points), edges, faces and elements. And with the size of models anticipated, meshes into millions of elements and 10 or more unknowns, parallel processing needs to be part of the strategy for such a computational software framework.

Multiphysics modelling projects commonly requires a multidisciplinary team or collaborating groups that may geographically located miles apart. For international projects this can mean participating organizations from different continents of the world. So it is easy to understand the advantages of a common and open software framework for such programs. Also for some collaborative programs code ownership is an important issue. In such cases, open software frameworks that have no provision to address this can have the opposite effects i.e. they can hinder collaboration.

From the above points the following elements have been identified as core and essential components to achieving a good open software framework:

- Data repository system
- Common integration interface
- Safeguard developers' investments

The following subsections give an overview of these matters for a computational mechanics analysis-engine toolkit called PHYSICA [phy].

Software Model

The model for this mesh-based open analysis-engine software framework can be referred to as a Single-Code Component-Based paradigm (single executable). Much of the object-oriented approach has been included in the software design, and for addressing maintainability and extendibility. Common programming language inter-operability is made possible with a Data Repository System (DRS) and a Component Integration Interface (CII), within the framework - DRS and CII are covered in the subsections below.

Table 2 Level of abstraction

Level	**Likely Components**
L4 (Model)	Fluid Flow, Residual Stress, Heat Transfer, Solidification, Contacts, etc., & Coupled Models.
L3 (Advanced)	General Equation for Finite Volume and Finite Element, Solution Procedures, Linear & Non-linear Solvers, Domain Coupling library, etc.
L2 (Standard)	System Matrices, Material Properties, Math Libraries, etc.
L1 (System)	Management Systems (Memory, IO, Database), Parser, Parallel Library, Geometry/Mesh Tools and Libraries, etc.

Table 44 shows the abstraction levels of the present framework. Level-1, the System level, is where system related components are found and most computer experts or system developers contribute, such as inter-communication in parallel processing. Level-2, the Standard level, sees the input from numerical analysis and math library developers. Level-3, the Advanced level, sees the components for numerical methods such as Finite Element and Finite Volume methods, and domain decomposition methods such as in solvers for systems of equations and domain coupling/mapping algorithms - predominately of computational scientists and engineers. Finally Level-4, the Models level, sees the various analysis engines/modules at the component level. Most can perform analysis individually but the potential power comes when model developers couple them together to solve challenging problems. If generic robust "Gluing" technology (coupling) is available, and has been applied to the model components, then Coupled-Model construction with existing component is straightforward. Customized models can be put together and tuned for a particular application.

Data Repository System

Provide the mechanism for storing and accessing of data within the analysis-engine framework. This is where software components would normally take data for processing and put the resulting data on the DRS.

With majority of the data (e.g. field variables) relating to some mesh entities - Points, Edges, Faces and Elements - it is convenient to treat them as mesh-entity objects, and for the DRS to manage them accordingly. Access to the DRS needs to be efficient and inter-operable for popular scientific programming languages. Meeting these objectives limit the options. Therefore, the mesh-entity objects are arrays with each array index location holding the relevant data, e.g. element material type (element object), nodal temperature (point object).

The allocation (and de-allocation) of memory areas for objects (and other data) is also manage by the DRS. A name (text string) is required for allocation, this provide the mean for software components to query objects and data in the DRS.

Component Integration Interface

An N-pin connector model is used to support a generic way of connecting components to the analysis-engine framework, akin to the common 25-pin sockets for connecting external devices to computers. Like the 25-pin sockets not all the pins need to be present, only the ones that are needed by the device. Here the pin equates to function or subroutine calls in the software. The "Pin-function" is given in Table 44 with name association for each pin in the N-pin connector. The last two arguments of the Pin-function are pretty obvious but the Data Repository Arrays (DRA) require explanation.

Depending on the implementation language for the DRS, such as C or Fortran, the arguments for DRA can be different. For a C implementation it can be argument free. This is not possible with standard Fortran77, which lacks dynamic memory allocation (such feature is in some vendors' extensions to the standard but not all and commonly not compatible). Therefore, the seven standard data type arrays (Real, Integer, Character, Logical, Complex Number, Double Real and Double Complex) are on the argument list, such as the example given below.

¡Pin-function name¿ (Ra, Ia, CHa, La, Xa, DRa, DXa, Error Code, Failed Flag)

Table 3 An N-pin connector

`<Pin-function name> ( [DRA], Error Code, Failed Flag )`
⇔ Banner
⇔ Default Settings
⇔ Error Handling
⇔ Read Command Script
⇔ Initialisation
⇔ Before Solution Procedure
⇔ Standard Procedure
⇔ Start of Time-Step
⇔ Sweep Entry (Start, Middle, End)
⇔ Convergence Status
⇔ End of Time-Step
⇔ Users' Own Procedure
⇔ After Solution Procedure
⇔ Output Data
⇔ General Equation Source Terms

Safeguarding Developers' Investments

Using a Component paradigm with both DRS and CII, the code ownership issue is addressed. A "Component Template" that corresponds to the CII's N-pin connector with private and public sections for data and functions is where all components are based. The template is a component itself in the software framework and is referred to as the User-Template component. The syntax for the Pin-function in the User-Template component is

USER_¡Pin-function name¿ ([DRA], Error Code, Failed Flag)

Hence, by changing the name USER to some other, for example name of the component, a new component is created. Together with a "Component Socket Definition" file (a simple text file with the needed Pin-functions) accompanying the binary file, integration to the analysis-engine framework is straightforward and can be automated in a plug-and-go fashion.

Like any software packages the technology or know-how of the process is encapsulated in binary form, with clear instruction of input/output data, purpose of processing and any limitations. The software component approach is no different from this paradigm, only now is at the component level. Like libraries, it is aim towards integration. But unlike conventional scientific libraries the component can be an entire analysis program under the analysis-engine framework. Thereby, authorship is at the component level but maintain all the rights of a library.

Parallel and Distributed Model

The PHYSICA toolkit use the Single Program Multiple Data (SPMD) paradigm for parallel and distributed processing - each processing element runs the same program operating on a local subset of the model domain. For more details see the reference by K.McManus et al [KMS97].

Domain Decomposition with Message Passing

The partitioning of the model domain (i.e. the mesh) to sub-meshes is done with an overlapping domain decomposition procedure. A local-global numbering scheme is used to map between the local (sub-partition) and global (undivided) meshes. Communication is required between the processing elements to exchange information, predominately for the overlapping regions, to solving the model.

The mesh partitioning is done using a version of JOSTLE [jos], a graph partition code, to divide the mesh. And for the inter-processors message passing (e.g. overlapped regions) is with the portable communication library, CAPlib of CAPTools [cap]. These elements plus others (construct the mesh-graph for JOSTLE, local-global renumbering, etc.) forms the Parallel component.

The objective with parallel processing is to obtain the same solution (within round-off errors) much faster than processing in scalar, else there is no advantage. The total execution time will tell you if you have gained or not. Genuine scalable parallel solvers/codes will guarantee speed-ups over the scalar, but bottlenecks commonly occurring at Input/Output can have a big impact on the overall execution time. This is very much system dependent and until parallel I/O is standardized and commonly available, this bottleneck remains.

The I/O bottleneck is one of two challenges that are actively being researched. The other is dynamic load balancing. In the case of Casting applications, the computational load imbalance can come from, moving surfaces, contact areas, adaptive meshing, solidification, and possibly all simultaneously. In the case of solidification with thermal convection and residual stress analysis, the workload is totally imbalance even with a balance mesh partition. Because of the physical states - Liquid, Solid, and phase transition - the workload for each is different. The heavy loads are with partitions having the phase transition, these needs to compute both the liquid and solid states and the phase transition.

A Simple Programming Model

With the parallel and distributed model used, a Simple Programming Model (SPM) within the analysis-engine framework is becoming possible. A SPM component can be created building on the Parallel, Geometry and DRS components.

Initially providing functions for performing overlap updates for all mesh-entity objects and global operations (such as summations). These functions takes the object name (those in the DRS), access the DRS and do the inter-communication using functions of the Parallel and Geometry components, to complete the operation. With these simple functions developers can put together codes/components to integrate with the framework that can take advantage of parallel processing. The component is at an advanced stage for inclusion with the analysis-engine framework.

CONCLUDING REMARKS

Demand is increasing in computational mechanics to have the ability to simulate engineering processes with interacting physics - going beyond the simplifying assumptions used in previous models. Putting together such an Analysis-Engine for Multiphysics simulations is not simple, often requiring a multidiscipline team that commonly involves different numerical methods and codes written in different programming languages. Domain decomposition technology is one key component in such analysis-engine.

With large models anticipated parallel processing need to be part of the software strategy for such analysis-engine. An open software framework using a single-code component-based software model can help with the analysis-engine construction. Two essential elements for such open framework are Data Repository System and Component Integration Interface. In the case of multiple working groups, code authorship is at the component level.

REFERENCES

[cap] CAPTOOLS. University of Greenwich (http://www.gre.ac.uk/ captools). London, U.K.
[C.B] C.BaileyComputational modelling of mould design in lead ingot casting. in press.
[jos] JOSTLE. University of Greenwich (http://www.gre.ac.uk/ wc06/jostle). London, U.K.
[KMS97] K.McManus, M.Cross, and S.Johnson (1997) Issues and strategies in the

parallelisation of unstructured multiphysics codes. In *Proceedings of Parallel and Distributed Computing for Computational Mechanics*.

[phy] PHYSICA. University of Greenwich (http://physica.gre.ac.uk). London, U.K.

45

Adaptive Multilevel FEM as Decisive Tools in the Clinical Cancer Therapy Hyperthermia

P. Deuflhard[1], M. Seebass[2]

INTRODUCTION

The paper surveys recent progress in a joint mathematical-medical project on cancer therapy planning. Within so-called *regional hyperthermia* the computational task is to tune a set of coupled radiofrequency antennas such that a carefully measured tumor is locally heated, but any outside *hot spots* are avoided. A mathematical model of the whole clinical system – air, applicator with antennas, water bolus, individual patient body – involves Maxwell's equations in inhomogeneous media and a parabolic bioheat transfer equation, which represents a simplified model of heat transfer in the human body (ignoring strong blood vessel heat transport). Both PDEs need to be computed fast and to medical reliability (!) on a workstation within a clinical environment. This requirement triggered a series of new algorithmic developments to be reported here, among which is an adaptive multilevel FEM for Maxwell's equations, which dominates the numerical simulation time. In total, however, the main bulk of computation time still goes into segmentation – a necessary preprocessing step in the construction of a 3D *virtual patient* from the input of a stack of 2D computed tomograms (left out here).

1 Invited speaker, DD 11, Greenwich, July 20–24, 1998
Konrad-Zuse-Zentrum für Informationstechnik Berlin (ZIB), Takustr. 7, D-14195 Berlin, Germany, email: deuflhard@zib.de
2 Address same as above, email: seebass@zib.de. This author is supported by Deutsche Forschungsgemeinschaft, Sonderforschungsbereich 273.
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

ELECTROMAGNETIC FIELD COMPUTATIONS

The electromagnetic field computations for the whole system (exterior domain, antennas, water bolus, patient body) dominate the numerical simulation work.

Finite Element Formulation. As the hyperthermia applicator is working with a fixed angular frequency ω, our field computations are based on the time-harmonic Maxwell's equations. The electric field $\mathcal{E}$ has the representation $\mathcal{E}(\mathbf{x},t) = Re\,\mathbf{E}(\mathbf{x})e^{i\omega t}$ where $\mathbf{E}(\mathbf{x})$ is a complex amplitude defined on a computational domain $\Omega \subset \mathcal{R}^3$.

In our application we are dealing with linear isotropic dielectric media without free charges. We start with the well-known double-curl equations for the electric field $\mathbf{E}$

$$\mathbf{curl}\,\frac{1}{\mu}\mathbf{curl}\,\mathbf{E} - \omega^2\epsilon\mathbf{E} \;=\; 0\;, \tag{1}$$

where μ is the permeability and $\epsilon = \epsilon' - i\sigma/\omega$ represents a complex dielectric constant related to the generic dielectric constant ϵ' and the conductivity σ. For the current density $\mathbf{j}$, Ohm's law $\mathbf{j} = \sigma\mathbf{E}$ can be assumed.

Homogeneous Dirichlet boundary conditions for the tangential components of $\mathbf{E}$ define the metallic surfaces of the antennas, where a perfectly conducting material is assumed. Nonhomogeneous Dirichlet conditions are applied at the central antenna gaps, representing a prescribed voltage of the power generators. The computational domain of interest consists of a sphere covering patient model, water bolus and antenna array. On the surface of this sphere Sommerfeld-type conditions according to [KW93] are applied. They preserve both sparsity and symmetry properties of the finite element matrices.

According to the differential operator in (1), an appropriate space for the desired solution, which accommodates the Dirichlet boundary conditions (stated as $\mathbf{E}_t = \mathbf{E}_t^0$ on Γ_D) is given by

$$H_{\Gamma_D}(\mathbf{curl}\,,\Omega) := \{\mathbf{w} \in (L^2(\Omega))^3\;;\;\; \mathbf{curl}\,\mathbf{w} \in (L^2(\Omega))^3,\;\mathbf{w}_t = \mathbf{E}_t^0 \text{ on } \Gamma_D\}\;.$$

The related space for homogeneous boundary conditions $\mathbf{w}_t = 0$ will be denoted by $H_{\Gamma_D;0}(\mathbf{curl}\,,\Omega)$. In this space, a variational formulation for the desired field $\mathbf{E}$ reads: Find $\mathbf{E} \in H_{\Gamma_D}(\mathbf{curl}\,,\Omega)$ such that for all $\mathbf{w} \in H_{\Gamma_D;0}(\mathbf{curl}\,,\Omega)$

$$\int\limits_{\Omega}\{\frac{1}{\mu}\mathbf{curl}\,\mathbf{E}\,\mathbf{curl}\,\mathbf{w} \;-\; \omega^2\epsilon\mathbf{E}\,\mathbf{w}\}\;d\Omega \;-\; \int\limits_{\Gamma_{ext}}\beta(\mathbf{n}\times\mathbf{E})\,(\mathbf{n}\times\mathbf{w})\;d\Gamma \;=\; 0\;. \tag{2}$$

The second integral describes a contribution on the exterior surface, where $\beta > 0$ is related to the wave vector of the outgoing field (see [KW93]). Observe that for non-vanishing σ the bilinear form occurring in (2) is coercive and the problem has a unique solution (cf. e.g. [CR96]).

To obtain a finite element discretization for (2), we generate a tetrahedral triangulation $\mathcal{T}_h$ of our domain and employ Nédélec's **curl**–conforming finite elements of lowest order [N80], also called Whitney 1-forms or edge elements [Bos88]. For a vector field $\mathbf{E}$ the degree of freedom E_e associated with each edge e in $\mathcal{T}_h$ is given by the path integral

$$E_e = \int_e \mathbf{E}\cdot\mathbf{t}\,ds \tag{3}$$

along the edge e with tangent vector $\mathbf{t}$. The convergence behaviour of such discretizations for (2) is studied in [Mon92].

Edge elements possess several distinct advantages: they guarantee the desired continuity of the tangential components of the electric field and are well-suited to suppress unwanted spurious modes [Bos90]. Concave metallic edges and internal boundaries between materials with jumps in the coefficients can be incorporated in agreement with the physical continuity relations of the fields.

An outstanding feature becomes apparent with regard to the nullspace of the **curl**-operator. In terms of physics, this nullspace comprises irrotational fields which can be written as gradients of potentials. Whenever a vector field $\mathbf{E}_h$ is given in the Nédélec space ND_h on the triangulation $\mathcal{T}_h$, then its irrotational part $\mathbf{E}_p$ is the gradient of a discrete potential Φ_h, *with* Φ_h *lying in the space of piecewise linear continuous finite element functions* S_h *on* $\mathcal{T}_h$. Thus for $\mathbf{E}_h \in ND_h$ we obtain a discrete Helmholtz decomposition

$$\mathbf{E}_h = \mathbf{E}_{h_p} + \mathbf{E}_{h_s}\ ,\ \ \mathbf{E}_{h_p} = \mathbf{grad}\,\Phi_h \text{ with } \Phi_h \in S_h\ ,$$

where $\mathbf{E}_{h_s}$ denotes the solenoidal part of $\mathbf{E}_h$ (for a detailed description we refer to Hiptmair [Hip96] and Beck et al. [BDH+97]). This decomposition with directly accessible potentials Φ_h is of crucial importance for the construction of efficient solvers for the arising linear systems.

Linear System Solution and Multilevel Preconditioning. The variational formulation (2) yields a sparse linear equation system $A\mathbf{u} = \mathbf{b}$ with complex symmetric and indefinite matrix A. Due to Nyquist's theorem, typical "coarse" grids in our virtual patient models already comprise approximately 100,000 degrees of freedom.

Considering the bilinear form in (2), it is apparent that the nullspace $\mathbf{grad}\,S_h$ of the **curl**-operator is shifted to eigenvalues with negative real part. This is also the case for a certain (typically much smaller) number of modes in its orthogonal complement $ND_h^{\perp}$ with respect to ND_h. Since the dimension of the nullspace is quite "large", an iterative solver should provide means for tackling the modes within this space, which are inaccessible to standard smoothers.

As a basic solver we use the conjugate residual (CR) method, which is similar to the well-known conjugate gradient algorithm, but adjusted to symmetric indefinite systems [Hac93]. For preconditioning we set up a hybrid smoothing procedure, whose prerequisites are analyzed in detail in [BH96, BDH+97]. Its basic operations are Gauss-Seidel sweeps both in the Nédélec space ND_h, coping with the elliptic part of A, and in the nullspace. Within this framework, efficient transfer operators between field representations in S_h and ND_h are essential.

If we represent a vector field $\mathbf{E}_{h_p}$ in the nullspace by $\mathbf{E}_{h_p} = \mathbf{grad}\,\Phi_h$, then the representation in ND_h can be obtained easily from (3):

$$E_e = \int_{P_1}^{P_2} \mathbf{grad}\,\Phi_h \cdot \mathbf{t}\, ds \ = \ \Phi_h(P_2) - \Phi_h(P_1)\ .$$

Here P_1 and P_2 denote the positions of the endpoints of the edge e.

In the following P_{S_h} denotes the transfer operator from the potential space S_h into the Nédélec space ND_h on $\mathcal{T}_h$. $P^*_{S_h}$ will denote the adjoint operator, defining the canonical restriction. Then the smoother A_S for the nullspace is constructed by the Galerkin product $A_S = P^*_{S_h} A P_{S_h}$. The following hybrid smoothing algorithm for a given right hand side vector $\mathbf{r} \in ND_h$ will provide an updated solution vector $\mathbf{u} \in ND_h$ via the following steps:

(1) One Gauss-Seidel step for $A\mathbf{u} = \mathbf{r}$
(2) $\tilde{\mathbf{r}} \leftarrow \mathbf{r} - A\mathbf{u}$
(3) $\mathbf{r}_S \leftarrow P^*_{S_h}\tilde{\mathbf{r}}$
(4) $\mathbf{u}_S \leftarrow 0$
(5) One Gauss-Seidel step for $A_S\mathbf{u}_S = \mathbf{r}_S$
(6) $\mathbf{u} \leftarrow \mathbf{u} + P_{S_h}\mathbf{u}_S$

In order to obtain a symmetric preconditioner, a subsequent step of the adjoint procedure has to applied. Note that both spaces are treated in a "multiplicative" fashion (speaking in terms of domain decomposition methods). As an alternative, we propose a symmetric additive version:

(1) One symmetric Gauss-Seidel step for $A\mathbf{u} = \mathbf{r}$
(2) $\mathbf{r}_S \leftarrow P^*_{S_h}\mathbf{r}$
(3) $\mathbf{u}_S \leftarrow 0$
(4) One symmetric Gauss-Seidel step for $A_S\mathbf{u}_S = \mathbf{r}_S$
(5) $\mathbf{u} \leftarrow \mathbf{u} + P_{S_h}\mathbf{u}_S$

If we employ adaptive mesh refinement, thus creating a sequence of nested triangulations $\mathcal{T}_0 \subset \mathcal{T}_1 \subset \ldots \subset \mathcal{T}_h$, the extension to a multilevel solver is quite straightforward. Taking into consideration that the associated sequence of Nédélec spaces is nested, i.e. $ND_0 \subset ND_1 \subset \ldots \subset ND_h$, we may adopt the classical multigrid idea [Hac85] by using canonical grid transfer operations between these spaces, but employing a hybrid smoother on each level. Now the basic CR algorithm is preconditioned by one multiplicative V-cycle within each iteration. For detailed examinations of its convergence behaviour we refer to [BDH$^+$97].

To assess the efficacy of our algorithmic concept, we present an example from our applications. As Table 1 shows, the solvers with hybrid smoothing exhibit a superior performance and do not deteriorate with increasing refinement level. Note that we have no direct factorization available on the coarse grid, thus facing comparatively large iterations counts. A proof of optimal multigrid complexity of this type of algorithm can be found in [BDH$^+$97, Hip96].

Adaptive Mesh Refinement. Due to the complex geometrical structure and strongly varying material properties of our finite element models, we consider adaptive mesh refinement to be of crucial importance to ensure both efficiency of our algorithm and reliability of our computational results.

For steering the local refinement, we employ a hierarchical local error indicator in the spirit of [DLY89] which renders asymptotically correct error estimates in the case of self-adjoint problems [BDH$^+$97]. As we are dealing with equations of partly hyperbolic

Table 1 Convergence history for multilevel solvers with standard Gauss-Seidel (Std), multiplicative (M-Hyb), and additive (A-Hyb) hybrid Gauss-Seidel smoothing on each level. The iteration is terminated if the ratio of the euclidian norms of residual and right hand side is below 10^5.

Ref. Depth	Nodes	#Iter Std	#Iter M-Hyb	#Iter A-Hyb	CPU [min] Std	CPU [min] M-Hyb	CPU [min] A-Hyb
0	128 365	4250	354	413	150	24	20
1	373 084	4832	265	277	800	76	60
2	1 085 269	$>$ 10000	186	194	$>$ 2000	215	160

nature in this context, it is clear that any local estimator will be unable to capture far-field contributions so that we prefer here to use the name error indicator in the sense introduced by Babuška [BDR92]. Surprisingly, the efficiency of this indicator in certain simple problems with known analytic solution appears to be quite close to 1 – compare section 8.2 in [BDH+97].

In order to obtain a hierarchical extension of the lowest-order edge element, we use basis functions as described in [SP96], including polynomials up to second order. If λ_i denotes the barycentric (i.e. linear Lagrangian) shape function associated with vertex i, then the vector basis function of the lowest order Nédélec space associated with edge $\{ij\}$ may be written as

$$\mathbf{w}^L_{\{ij\}} \;=\; \lambda_i \mathbf{grad}\,\lambda_j - \lambda_j \mathbf{grad}\,\lambda_i \;.$$

When adding the hierarchical surplus, we have one more degree of freedom on each edge

$$\mathbf{w}^H_{\{ij\}} \;=\; \lambda_i \mathbf{grad}\,\lambda_j + \lambda_j \mathbf{grad}\,\lambda_i$$

and two additional ones on each face (with vertices $i < j < k$) of the triangulation:

$$\mathbf{w}^{H,1}_{\{ijk\}} = \lambda_i\lambda_j \mathbf{grad}\,\lambda_k - \lambda_i\lambda_k \mathbf{grad}\,\lambda_j \;, \quad \mathbf{w}^{H,2}_{\{ijk\}} = \lambda_i\lambda_j \mathbf{grad}\,\lambda_k - \lambda_j\lambda_k \mathbf{grad}\,\lambda_i$$

For the discrete global problem we obtain an extended system

$$\begin{pmatrix} A^{LL} & A^{LH} \\ A^{HL} & A^{HH} \end{pmatrix} \begin{pmatrix} \mathbf{E}^L \\ \mathbf{E}^H \end{pmatrix} \;=\; \begin{pmatrix} \mathbf{b}^L \\ \mathbf{b}^H \end{pmatrix} , \tag{4}$$

where the superscript ^{H} denotes components of the hierarchical surplus. For the mere purpose of error indication, we solve (4) only approximately via a defect equation for the hierarchical components:

$$A^{HH}\,\mathbf{E}^H \;=\; \mathbf{b}^H \;-\; A^{HL}\,\mathbf{E}^L \;. \tag{5}$$

An approximate solution of (5) may be obtained via one block-Jacobi step only (!), where it is essential to keep the 2×2-block-entries of both functions $\mathbf{w}^{H,l}_{\{ijk\}}$, $l = 1, 2$, attached to each face of the triangulation (see [BDH+97]) for details). At first glance, this block-Jacobi sweep appears critical, as the matrix contains negative eigenvalues and the related eigenmodes may be amplified. However, the modulus of all negative eigenvalues being comparatively small, these modes can be expected to give only minor contributions.

For the purpose of measuring field energy and discretization error, we define the following norm (asterisks * denote complex conjugates):

$$\|\mathbf{v}\|_{\mathbf{E}}^{2} \;:=\; \int_{\mathbf{\Omega}} \{\frac{\mathbf{1}}{\mu}\mathbf{curl\,v\,curl\,v}^{*} \;+\; \omega^{\mathbf{2}}\epsilon\mathbf{v}\,\mathbf{v}^{*}\}\;\mathbf{d\Omega} + \int_{\mathbf{\Gamma_{ext}}} \beta(\mathbf{n}\times\mathbf{v})\,(\mathbf{n}\times\mathbf{v}^{*})\;. \qquad (6)$$

Table 2 below gives the percentage of the relative discretization error

$$\eta \;=\; 100\cdot\frac{\|\mathbf{E}^{H}\|_{E}^{2}}{\|\mathbf{E}^{L}\|_{E}^{2}}$$

for the different refinement levels.

Table 2 Estimated discretization errors.

Ref. Level	Nodes	η
0	128 365	10.1 %
1	373 084	4.96 %
2	1 085 269	2.54 %

LINEAR VERSUS NONLINEAR HEAT TRANSFER MODEL

In a phased array applicator the antennas are grouped into k channels that can be independently controlled. For each channel j the amplitude a_j and phase delay θ_j can be set. For convenience we define complex amplitudes

$$z_j = a_j\;exp(-i\theta_j) \qquad j = 1,\ldots,k \qquad (7)$$

The aim is to determine an optimal set of complex amplitudes for each individual patient.

Bio–Heat–Transfer Equation. The heat transfer model used in our simulations is the bio–heat–transfer (BHT) equation originally proposed by PENNES [Pen48]

$$\rho c\,\frac{\partial T}{\partial t} = \mathrm{div}\,(\kappa\,\mathrm{grad}\,T) - c_b W(T - T_b) + Q_e\;, \qquad (8)$$

where ρ is the density, c and c_b are specific heat of tissue and blood, κ is the thermal conductivity, T_b is the blood temperature, W is the mass flow rate of blood per unit volume of tissue (the so-called perfusion), $Q_e = \frac{1}{2}\,\sigma\mid\mathbf{E}\mid^2$ is the electromagnetic power density deposited in tissue, σ is the electric conductivity of tissue and $\mathbf{E}$ is the electric field.

The BHT equation describes the heat transport by blood in a potential flow under the additional assumption that heat is exchanged between blood and tissue only through capillaries [WNF+89]. In reality, a major part of heat exchange will occur through stronger blood vessels – a topic, which is presently under investigation.

Another, more realistic, model is based on a *nonlinear* version of the BHT equation taking into account that blood flow depends on tissue temperature. Experiments [SLR+84] have shown that the blood flow in normal tissues, e.g., skin and muscle, increases significantly when heated up to 41–43°C, whereas in the tumor zone the blood flow decreases with temperature. On this experimental basis, we chose $W = W(T)$ monotonically increasing in muscle and fat tissue, but monotonically decreasing in tumor tissue (for details see [LES97]).

Optimization Algorithm. In the following we focus on the stationary BHT equation. In a hyperthermia treatment the steady state is typically reached after 20 - 30 minutes and maintained for 40 - 60 minutes.

Our goal is to control the complex amplitudes z_j such that an effective hyperthermia therapy is achieved. In medical terms, an optimal temperature distribution is characterized as follows:

- within the tumor a therapeutic temperature level $T_t \approx 43°$C is maintained,
- regions of healthy tissue are not heated above $T_h \approx 42°$C.

For most patients both requirements cannot be fulfilled simultaneously. In searching for a compromise we avoid destruction of healthy tissue by the additional constraint that temperature in healthy tissue must not exceed certain limits which depend on the tissue type: 42°C for more sensitive tissue compartments (like bladder, intestine) and 44°C otherwise. From this we arrive at the objective function

$$f(p) = \int\limits_{\substack{x \in V_{tumor} \\ T(x,p) < T_t}} (T_t - T(x,p))^2 \, dx + \int\limits_{\substack{x \notin V_{tumor} \\ T(x,p) > T_h}} (T(x,p) - T_h)^2 \, dx \tag{9}$$

to be minimized subject to the constraint

$$T(x,p) \le T_{lim}(x) \,, \quad x \notin V_{tumor} \,.$$

where $p = \{\Re z_j, \Im z_j\}$ denote the control parameters.

In the *linear* heat transfer model superposition of the electric field $\mathbf{E}$ into k modes can be employed, which in $Q_e \sim |\ \mathbf{E}\ |^2$ leads to k^2 basic modes to be computed in advance, plus one further mode for the basal temperature T_{bas}.

In the *nonlinear* case, we presently apply some (cheap) fixed point iteration in terms of perfusion iterates W^m corresponding to iterates p^m.

optimization algorithm for the nonlinear BHT equation:

k **Maxwell solves:** $\mathbf{E}^j(x)$ per channel $j = 1, \ldots, k$

superposition $\mathbf{E}(x) = \sum_{j=1}^{k} z_j \mathbf{E}^j(x)$, $z_j \in \mathbb{C}$

control parameters $p = \{\Re z_j, \Im z_j\}$

$Q_e(x) = \frac{1}{2}\sigma(x) \mid \mathbf{E}(x) \mid^2 = \frac{1}{2}\sigma(x) \sum_{j,l=1}^{k} \gamma_{jl}(p) \overline{\mathbf{E}}^j(x) \mathbf{E}^l(x)$

$\gamma_{jl}(p) \in \mathcal{P}_2(p)$

for $m = 0, 1, \ldots, n$: given p^m

one *nonlinear* **BHT solve** for $\mathbf{E}(p^m)$

supplies iterates T^m and $W^m = W(T^m)$

k^2 **+ 1** *linear* **BHT solves** with W^m:

$T^m_{bas}(x)$, $T^m_{jl}(x)$ from $\frac{1}{2}\sigma(x)\overline{\mathbf{E}}^j(x)\mathbf{E}^l(x)$ $j, l = 1, \ldots, k$

superposition $T(x, p, W^m) = T^m_{bas}(x) + \sum_{j,l=1}^{k} \gamma_{jl}(p) T^m_{jl}(x)$

$T(x, p, W^m) \in \mathcal{P}_2(p)$

$f(p) \in \mathcal{P}_2(T) \in \mathcal{P}_4(p)$

optimization $f(p) = min$ supplies p^{m+1}

The above fixed point iteration exploits the fact that the Maxwell solves are considerably more expensive than the BHT solves. It converges quite fast at an observed contraction rate $\theta \approx 0.3$. In the *nonlinear* case the total computational cost is then

$$\begin{aligned} cost_{total} &= k * cost_{Maxwell} + \\ & \quad n * (cost_{nlBHT} + (k^2 + 1) * cost_{lBHT} + cost_{Opt}) \end{aligned} \tag{10}$$

where the notation $cost_{Maxwell}$, $cost_{nlBHT}$, $cost_{lBHT}$ and $cost_{Opt}$ is self-explaining. In typical computations we need about $n \approx 6$ optimization iterations.
In the *linear* case the total cost is reduced to

$$cost_{total} = k * cost_{Maxwell} + (k^2 + 1) * cost_{lBHT} + cost_{Opt} \tag{11}$$

For the linear BHT equation we apply the additive multilevel algorithm KASKADE [DLY89, BEK93]. For the nonlinear BHT equation, the algorithm KARDOS [LES97] has been selected up to now.

Model Comparison. Our simulations show significant qualitative differences between the temperature distributions predicted by the linear and the nonlinear heat

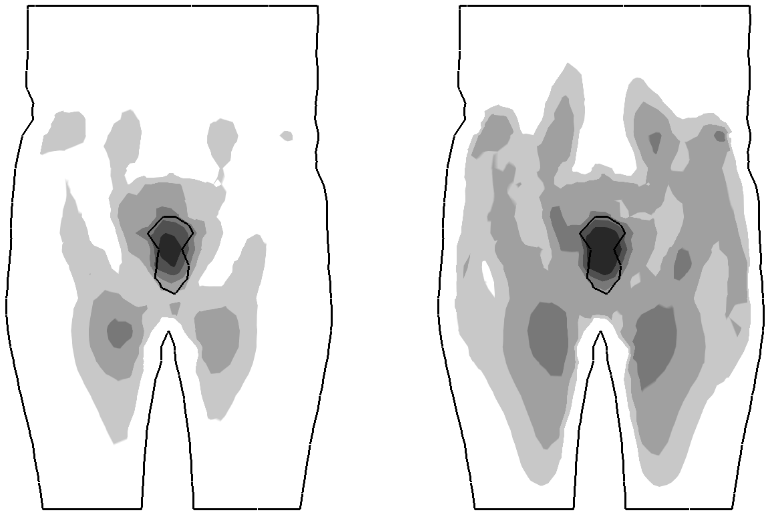


Figure 1 Optimized temperature distributions in a frontal section of the pelvic region, based on the linear model (left) and the nonlinear model (right). Black lines: body outline and tumor contour. Light grey to dark grey: regions heated above 39°C to 43°C.

transfer model, as illustrated in Fig. 1. Generally speaking, the self–regulation of healthy tissue reflected by the nonlinear model reduces "hot spots" caused by local maxima of the absorbed electromagnetic fields. This is one reason for a slightly better tumor heating (ca. 0.5°C) predicted by the nonlinear model. An analogous result is reported in [TVK+94] for ferromagnetic thermoseed hyperthermia. An important finding is that the nonlinear model has an impact on the optimal treatment parameters as well. Maximal discrepancies turned out to be 22° for the phases θ_j and 0.22 for the relative amplitudes a_j. See [LES97] for a more detailed discussion.

OLD VERSUS NEW HYPERTHERMIA APPLICATOR

During the past ten years the applicator most frequently used for regional hyperthermia was the *Sigma-60* applicator of *BSD Medical Corp.*, Salt Lake City, Utah (see Fig. 2 left). This applicator consists of eight antennas arranged on a ring with 60 cm diameter. The antennas are grouped into $k = 4$ channels which can be independently controlled. The space between the antenna ring and the patient's body is filled with a so-called *water bolus* containing de-ionized water.

The new *Sigma-Eye* applicator (see Fig. 2 right), partly based on results of our simulations, has been introduced in early 1998. It has 24 antennas (hence $k = 12$ channels) arranged in three parallel rings thus allowing for an additional power steering

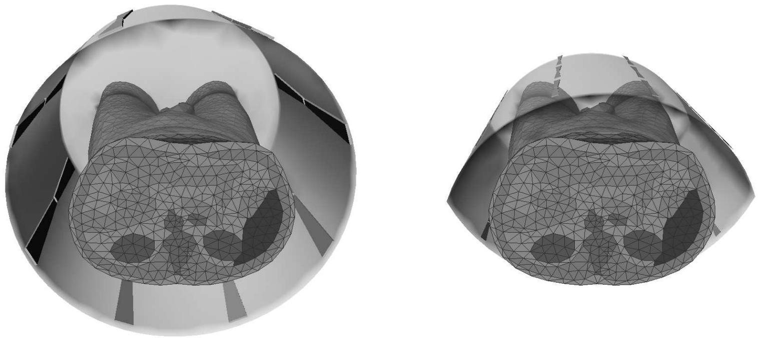


Figure 2 Sigma-60 applicator (left) and Sigma-Eye applicator (right).

along the patient's axis. The much smaller bolus volume has an "eye"-shaped cross-section.

In what follows we document some comparative results for both the old and the new applicator. Table 3 shows the computation times for the whole simulation process, based on the linear BHT as heat transfer model. Note that the field calculation time per channel of the Sigma-Eye is about 10 minutes compared to about 20 minutes for the Sigma-60, essentially due to the smaller bolus volume. As explained in the description of the optimization algorithm, the temperature calculation times roughly increase with k^2, whereas the Maxwell solves enter with k.

Table 3 Computation times for the whole simulation process.

	Sigma-60 $k = 4$	Sigma-Eye $k = 12$
Segmentation	2 - 4 hours*	
Grid Generation	15 min**	
Field Calculation	80 min**	120 min**
Temperature Calculation	2 min**	20 min**
Optimization	6 sec**	1 min**
	* interactive ** CPU time (SUN UltraSparc)	

Table 4 summarizes simulation results for three patients with typical tumor locations. Obviously, the new applicator significantly improves tumor heating in all

Table 4 Comparison of old and new applicator for three patients with different tumor locations.

	part of tumor volume heated to above 43°C	
	Sigma-60 (old)	Sigma-Eye (new)
distal (supraanal) rectal carcinoma	17.5%	62.5%
highly presacral rectal carcinoma	0.7%	18.4%
cervical carcinoma at pelvic wall	24.8%	49.1%

cases while keeping "hot spots" in healthy tissue at a constant level or slightly reducing them. The second location is the most difficult one since the tumor is to a large extent enclosed by bony structures. A similar comparison would show up for the nonlinear heat transfer model – details of medical interest will be published elsewhere.

CONCLUSIONS

The simulations reported herein are based on a *virtual patient* model that involves detailed individual geometry (patient, applicator, antennas) and correct electrical material properties, but only average values for the biological parameter perfusion. Even with this restriction, our simulations seem to be already helpful in the *decision*, whether regional hyperthermia is a promising modality for an individual patient. Beyond that, the developed methods also support the design of new applicator geometries and of new special purpose antennas.

Acknowledgments. The authors wish to thank Peter Wust from Virchow-Klinikum Berlin for his inspiring cooperation and Rudolf Beck for his input concerning the Maxwell multilevel solver. This work was supported by Deutsche Forschungsgemeinschaft, Sonderforschungsbereich 273 "Hyperthermie: Methodik und Klinik".

REFERENCES

[BDH+97] Beck R., Deuflhard P., Hiptmair R., Hoppe R., and Wohlmuth B. (1997) Adaptive multilevel methods for edge element discretizations of Maxwell's equations. *ZIB Preprint SC 97-66.* To appear in *Surveys for Mathematics in Industry* .

[BDR92] Babuška I., Durán R., and Rodriguez R. (1992) Analysis of the efficiency of an a posteriori error estimator for linear triangular finite elements. *SIAM J. Numer. Anal.* 29, pp. 947-964.

[BEK93] Bornemann F., Erdmann B., and Kornhuber R. (1993) Adaptive multilevel methods in three space dimensions. *Int. J. Numer. Meth. Engrg.* 36, pp. 3187-3202.

[BH96] Beck R. and Hiptmair R. (1996) Multilevel solution of the time-harmonic Maxwell equations based on edge elements. *ZIB Preprint SC 96-51.* To appear in *Int. J. Num. Meth. Eng.* .

[Bos88] Bossavit A. (1988) Whitney forms: a class of finite elements for three-dimensional computation in electromagnetism. *Inst. Elec. Eng. Proc., Part A* 135, pp. 493-500(8).

[Bos90] Bossavit A. (1990) Solving Maxwell's equations in a closed cavity and the question of spurious modes. *IEEE Trans. Magn.* 26(2, pp. 702–705).

[CR96] Caorsi S. and Raffetto M. (1996) Electromagnetic boundary value problems in the presence of a partly lossy dielectric: considerations about the uniqueness of the solution. *IEEE Trans. Microwave Theory Tech.* 44(8, pp. 1511-1513).

[DLY89] Deuflhard P., Leinen P., and Yserentant H. (1989) Concepts of an adaptive hierarchical finite element code. *IMPACT Comput. Sci. Engrg.* 1, pp. 3-35.

[Hac85] Hackbusch W. (1985) *Multi-Grid Methods and Applications.* Springer, Berlin.

[Hac93] Hackbusch W. (1993) *Iterative Solution of Large Sparse Linear Systems of Equations.* Springer, Berlin.

[Hip96] Hiptmair R. (1996) *Multilevel Preconditioning for Mixed Problems in Three Dimensions.* Ph.D. Thesis, Wissner, Augsburg.

[KW93] Kanellopoulos V. and Webb J. (1993) 3D finite element analysis of a metallic sphere scatterer: comparison of first and second order vector absorbing boundary conditions. *J. Phys. III France* 3, pp. 563-572.

[LES97] Lang J., Erdmann B., and Seebass M. (1997) Impact of nonlinear heat transfer on temperature control in regional hyperthermia. *ZIB Preprint SC 97-73.* To appear in *IEEE Trans. Biomed. Engrg.* .

[Mon92] Monk P. (1992) A finite element method for approximating the time–harmonic Maxwell equations. *Numer. Math.* 63: 243–261.

[N8́0] Nédélec J. (1980) Mixed finite elements in $\mathcal{R}^3$. *Numer. Math.* 35, pp. 315-341.

[Pen48] Pennes H. (1948) Analysis of tissue and arterial blood temperatures in the resting human forearm. *J. Appl. Phys.* 1, pp. 93-122.

[SLR+84] Song C., Lokshina A., Rhee J., Patten M., and Levitt S. (1984) Implication of blood flow in hyperthermic treatment of tumors. *IEEE Trans. Biomed. Engrg.* 31, pp. 9-16.

[SP96] Savage J. and Peterson A. (1996) High-order vector finite elements for tetrahedral cells. *IEEE Trans. Microwave Theory Tech.* 44(6, pp. 874-879).

[TVK+94] Tompkins D., Vanderby R., Klein S., Beckman W., Steeves R., Frey D., and Palival B. (1994) Temperature–dependent versus constant–rate blood perfusion modelling in ferromagnetic thermoseed hyperthermia: results with a model of the human prostate. *Int. J. Hyperthermia* 10, pp. 517-536.

[WNF+89] Wust P., Nadobny J., Felix R., Deuflhard P., John W., and Louis A. (1989) Numerical approaches to treatment planning in deep rf-hyperthermia. *Strahlenther. Onkol.* 165, pp. 751-757.

46

Sound Generation by Vortex-Blade Interactions

G.S. DJAMBAZOV[1], C.-H. LAI[2], & K.A. PERICLEOUS[3]

Introduction

Aerodynamic sound generation is a result of the interactions of vortex structures that arise in viscous flows. A full aeroacoustic simulation therefore should include the generation of these vortex structures themselves. The present work concentrates on the second stage of the sound generation process, namely the actual production of acoustic waves by vortices hitting a solid object.

Computational Fluid Dynamics (CFD) codes can be employed to resolve the aerodynamic *sources* of sound if a proper software coupling is implemented with an acoustic (linearised Euler) solver. This coupling is necessary because the fast and robust numerical schemes that make CFD codes valuable are not suitable for wave propagation simulations [DLP98a]. It is most natural that the Domain Decomposition ideas should be applied in such circumstances.

The physical problem of aerodynamic sound can be decomposed in two sub-problems: airflow with sound generation, and sound propagation in non-uniformly moving media. When external noise problems are considered, there is no possibility of acoustic resonance, and any feedback from the propagating waves to the flow can be completely ignored because of the extremely small magnitude of the acoustic perturbations [Lig52].

Mathematically, decomposition is applied to the variables of the fluid motion:

1 School of Computing & Mathematical Sciences, University of Greenwich, Wellington Street, Woolwich, London SE18 6PF, U.K. Email: G.Djambazov@gre.ac.uk
2 Address same as above. Email: C.H.Lai@gre.ac.uk
3 Address same as above. Email: K.Pericleous@gre.ac.uk
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

density, pressure and velocity as they are separated into flow and acoustic parts [DLP97a, Har93]. At each time step the flow part is resolved first. Information about the generated sound is extracted from the time-dependent CFD solution and is transfered into the acoustic solution via the source terms of the linearised Euler equations. This method was validated against analytical one-dimensional solutions, and two-dimensional results with simple geometry were shown to be physically correct [DLP98a].

In this paper the coupling technique employing the linearised Euler source term is considered for sound generation at solid surfaces with realistic geometry.

The PHYSICA Package as a CFD Code

The coupling between this software package and an acoustic software module will be used for the simulation of aerodynamic sound generated by vortex-blade interactions in *realistic* geometries. PHYSICA [CPC95] was selected for its flexibility with complex shapes and different numerical algorithms, and also, because it is being developed at the University of Greenwich.

The PHYSICA package has a flexible modular structure which allows various modelling procedures of various physical phenomena to be accessed in a single numerical simulation. New modules or new features of existing modules can be added to the package at any time. Currently, the following modules are available: heat transfer, fluid flow, solidification, and elastic/visco-plastic solid mechanics module. In the present study only the fluid flow module is used.

The solution algorithms in PHYSICA are based on *unstructured* meshes which can be comprised of cells of various types and shapes. This makes the modelling of curved solid boundaries very easy. With the fluid flow variables there is no staggering of the grids: the velocity vector components are stored at the centres of the cells together with the pressure and density values. In this way only one computational mesh is used during the whole simulation rather than four separate meshes needed with the staggered approach.

Second order schemes with the flow solution

Most CFD implementations provide as a default option the stable upwind scheme [VM95]. It ensures that during the iterative solution, an increase of a quantity at a given location will *always* be followed by an increase and not by a decrease at the neighbouring points. Unfortunately, this restricts the approximation of the variables to piecewise constant in both space and time.

For the *flow perturbations* which generate aerodynamic sound the QUICK differencing scheme [Leo79] is an alternative to increase the accuracy of the solution. With it the cell-face values of fluxes are calculated by second order interpolation between the two neighbouring nodes and an upstream node. The scheme can be formulated in a standard way and in several alternative ways [VM95]. The alternative formulations where troublesome negative coefficients are placed in the source term (right-hand side) of the discretised equations are usually preferred for stability reasons [Cro98].

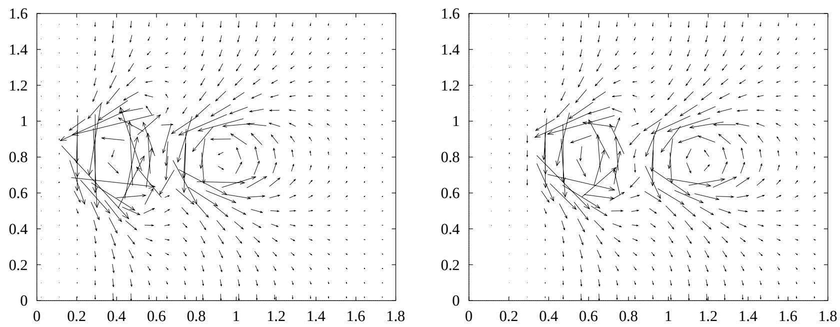


Figure 1 PHYSICA: Vortex convection with 2nd order (QUICK) scheme. Perturbation velocity vectors shown after subtracting the mean velocity from left to right (scale: 4 m/s to 0.1 m)

For the PHYSICA unstructured meshes a special procedure has been developed for finding the second upwind node [Cro98]. This implementation was tested with the inviscid example of vortices carried by a mean flow of 160 m/s (Figure 1).

The Acoustic Module

Since most CFD codes with which the acoustic module has to communicate, use finite volume methods, initially a finite volume algorithm of extended accuracy (to third order) was developed for the numerical solution of the linearised Euler equations [DLP97b]. However, optimised *finite difference* numerical schemes of higher order turned out to be less complex to implement in three dimensions and also exhibited better accuracy, so they were finally selected as a basis of the acoustic code.

The acoustic algorithm implemented solves the 3D linearised Euler equations (1) and (2) in a time-accurate way and exhibits the following features:

- Fully-staggered storage, with respect to pressure, of the velocity components in the three spatial directions and in time;
- Optimised forth order finite difference schemes:
  - — non-staggered convection terms [TW93]
  - — staggered propagation terms [DLP98b]
- Regular Cartesian grids with stepwise representation of solid boundaries;
- Mirroring of variables at solid walls for the missing values of the numerical scheme;
- Acoustic radiation boundary conditions.

The code can make direct numerical simulation of the *sound field* in an efficient way by solving the linearised Euler equations (1) and (2) given the sound sources (S, F_i) and the mean flow quantities: $\overline{v_j}$, $\overline{\rho}$, $c^2 = 1.4\overline{p}/\overline{\rho}$.

$$\frac{\partial p}{\partial t} + \overline{v_j}\frac{\partial p}{\partial x_j} \quad + \quad \overline{\rho} c^2 \frac{\partial v_j}{\partial x_j} = S \tag{1}$$

$$\frac{\partial v_i}{\partial t} + \overline{v_j}\frac{\partial v_i}{\partial x_j} \quad + \quad \frac{1}{\overline{\rho}}\frac{\partial p}{\partial x_i} = F_i \tag{2}$$

The acoustic perturbation parts of the pressure and the velocity components are denoted by p and v_i respectively. The right-hand sides S and F_i accommodate, along with any external sources and forces, all the small nonlinear terms that arise when the equations of motion for the full variables $(\overline{\rho}+\rho)$, $(\overline{v_i}+v_i)$, and $(\overline{p}+p)$ are expanded [DLP97a]. Within the acoustic solver the right-hand sides are considered as known functions of x_i and time t. In some rare cases when long-distance or resonant non-linear sound propagation effects have to be taken into account, the terms S and F_i can be evaluated iteratively.

The acoustic software module was validated against benchmark solutions [DLP98b]. It has two aspects of application. First, it can be used on its own with known mean flow and sound sources. Second, the module can be coupled with a CFD package to study the time-dependent noise generation by oscillating formations in the flow.

Sound sources on overlapping meshes

With the CFD simulation finite volume meshes are used, and the flow domain is divided into computational cells. If the assumption is made that the cells which become sources of sound are known in advance, the source term S (1) can be used to transfer the information about the generation of sound from the CFD code to the acoustic solver.

Closer examinations of the time history of test solutions obtained from *CFD* codes showed that the pressure at the *first* node next to the source of sound is resolved with sufficient accuracy. This suggests that when the source nodes are known, the time dependent CFD pressure at these nodes may be used to calculate the necessary source term S of the acoustic equations.

The following *assumption* has to be made: the CFD code resolves the *full* physical pressure (comprised of mean flow and acoustic components) in the first layer of computational cells next to a solid surface or in any other cells that have been identified as sound sources. The term ‘resolves’ is used here to denote that the CFD pressure is a good approximation of the true pressure signal in these selected cells.

Since the CFD pressure signal contains a mean-flow component, it cannot be fed directly into the acoustic code; the time dependent component has to be separated first. This can be done if a preliminary *steady* CFD solution is obtained in the same geometry, and the time dependent simulation is started with this initial condition. Then the difference between the time dependent and the steady pressure is the signal that has to enter the acoustic simulation at the prescribed source nodes.

One way of inserting this signal into the linearised Euler solver is to specify it as a fixed-value internal condition at the selected nodes. However, this will preclude the possibility of other acoustic waves (reflected from solid boundaries or generated by neighbouring source nodes) to propagate through the prescribed source layers.

The other option is to calculate the contribution of the CFD source to the local increment of pressure at the selected nodes over each time step. Since any transients associated with the establishment of the mean flow have been eliminated by starting

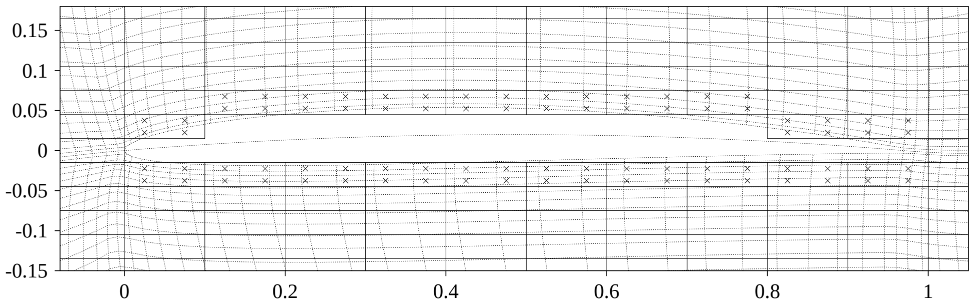


Figure 2 Overlapping CFD and CAA meshes and interpolation locations of the acoustic source cells

from a steady solution, this CFD contribution is simply the difference $\overline{p}(t) - \overline{p}(t - \Delta t)$ between the new and the old CFD pressure values.

If the CFD mesh and the acoustic mesh are the same, it is enough to add this difference to the other terms forming the acoustic pressure increment $p(t) - p(t - \Delta t)$ (see equation 1). However, most often this will not be the case because the CFD mesh is refined in the boundary layer while the acoustic mesh has to be coarse in order to obey the Courant limit. Therefore, it is best to express the CFD pressure contribution in terms of continuous quantities:

$$\overline{p}(t) - \overline{p}(t - \Delta t) = \frac{\partial \overline{p}}{\partial t} \Delta t. \tag{3}$$

Then the temporal derivative of the local pressure at the source nodes, calculated from the CFD solution, can be added to the source term S of the acoustic continuity equation (1):

$$S = \frac{\partial \overline{p}}{\partial t} + S_{vib}\,. \tag{4}$$

Here S_{vib} denotes sources external to the flow like vibrating solid objects.

The algorithm of the sequential coupling between the CFD code and the Computational Aeroacoustics (CAA) code based on the above definition of the linearised Euler source term was outlined in our previous communication [DLP98a].

The two codes (CFD and CAA) have separate meshes in overlapping domains. The CFD mesh must be body-fitted to represent smooth solid boundaries. The acoustic mesh is regular Cartesian to ensure high accuracy of the wave simulation, and the CAA domain can be larger than the flow domain because typically the acoustic wavelength is larger that the size of corresponding oscillating structures in the flow. In the latter case uniform mean flow is assumed outside the region of the CFD simulation.

Prior to the introduction into the acoustic simulation the flow quantities ($\overline{v_j}, \overline{\rho}, \overline{p}$, and c^2) have to be interpolated from the irregular CFD mesh. This mesh is usually finer than the acoustic mesh (in order to resolve vortices and boundary layers), and therefore, piecewise constant functions can be used for the interpolation. Also, averaging of the above flow quantities over each of the big acoustic cells has to be performed to ensure consistency of the communicated values. In Figure 2 the two overlapping meshes are shown, and for the designated acoustic source cells only, the interpolation locations are marked. The cell average values are then arithmetic averages of the interpolated values in these locations of each acoustic cell.

Simulation Results

As an example, the technique described above is applied to the sound generation due to vortex-blade interactions. The CFD body-fitted mesh around the section of the blade (aerofoil) can be seen in Figure 2. The production of sound by vortices hitting a blade is essentially an inviscid phenomenon since it is due to the inertial forces, and for this reason the boundary layer close to the aerofoil is not modelled. If the exact lift and drag are needed the mesh can be refined next to the aerofoil; this will not change the coupling technique in principle.

According to the algorithm outlined [DLP98a], first a **steady** solution of the airflow around the aerofoil was obtained. The aerofoil chord formed an angle of attack 7^0 with the free-stream velocity vector. No turbulence model was used, and inviscid flow was assumed instead, as explained above.

At the second stage of the aeroacoustic computation, the **time dependent** simulation of the flow is initialised with the steady solution, and with the inflow boundary conditions at the left end of the domain, a time dependent perturbation of the mean flow is specified in the perpendicular direction. It is sinusoidal with amplitude 7.5% of the mean velocity of 160 m/s and is applied to the inflow momentum in the middle part of the inflow boundary. In this way a series of vortex perturbations of the mean flow is generated. In a real aeroacoustic computation the flow perturbations (vortex structures) should not be prescribed but resolved within the CFD code. Due to the finer meshes and the use of turbulence models, the CFD part of the simulation is expected to be computationally much more expensive than the acoustic part.

The evolution of the flow field due to the vortex convection is illustrated by two snapshots in Figure 3 with arrows representing velocity vectors (scale: 4 m/s to 0.1 m). These vectors depict only the perturbation of the mean flow due to the passing vortices (the mean velocity vector has been subtracted before plotting).

At the third stage of this aerodynamic noise problem the special **acoustic** module (Section 46) is used. In order for the aerofoil to be better discretised on the rectangular acoustic mesh, the CFD mesh and velocity vectors have been rotated to the angle of attack around the leading edge of the aerofoil. The blocked cells forming the solid boundaries in the acoustic simulation have been omitted from the plot in Figure 2.

Although the flow solution does not contain acoustic waves, the sources of sound can be calculated from the pressure variations on the surface of the aerofoil. The pressure fluctuations (temporal derivatives) of the flow solution next to the solid surface are averaged over the rectangular cells which are neighbouring the blocked cells, and are imposed as the source term S of the linearised Euler equations. The aerofoil surface is assumed stationary ($S_{vib} = 0$). Since the linearised Euler solver uses explicit schemes, for stability reasons several acoustic time steps are needed to cover one flow time step.

In order to compare the mean convection and the sound propagation times, each of the instantaneous plots (Figure 3) shows the flow perturbation field and, superimposed on it, the resulting acoustic waves propagating away from the aerofoil. It can be

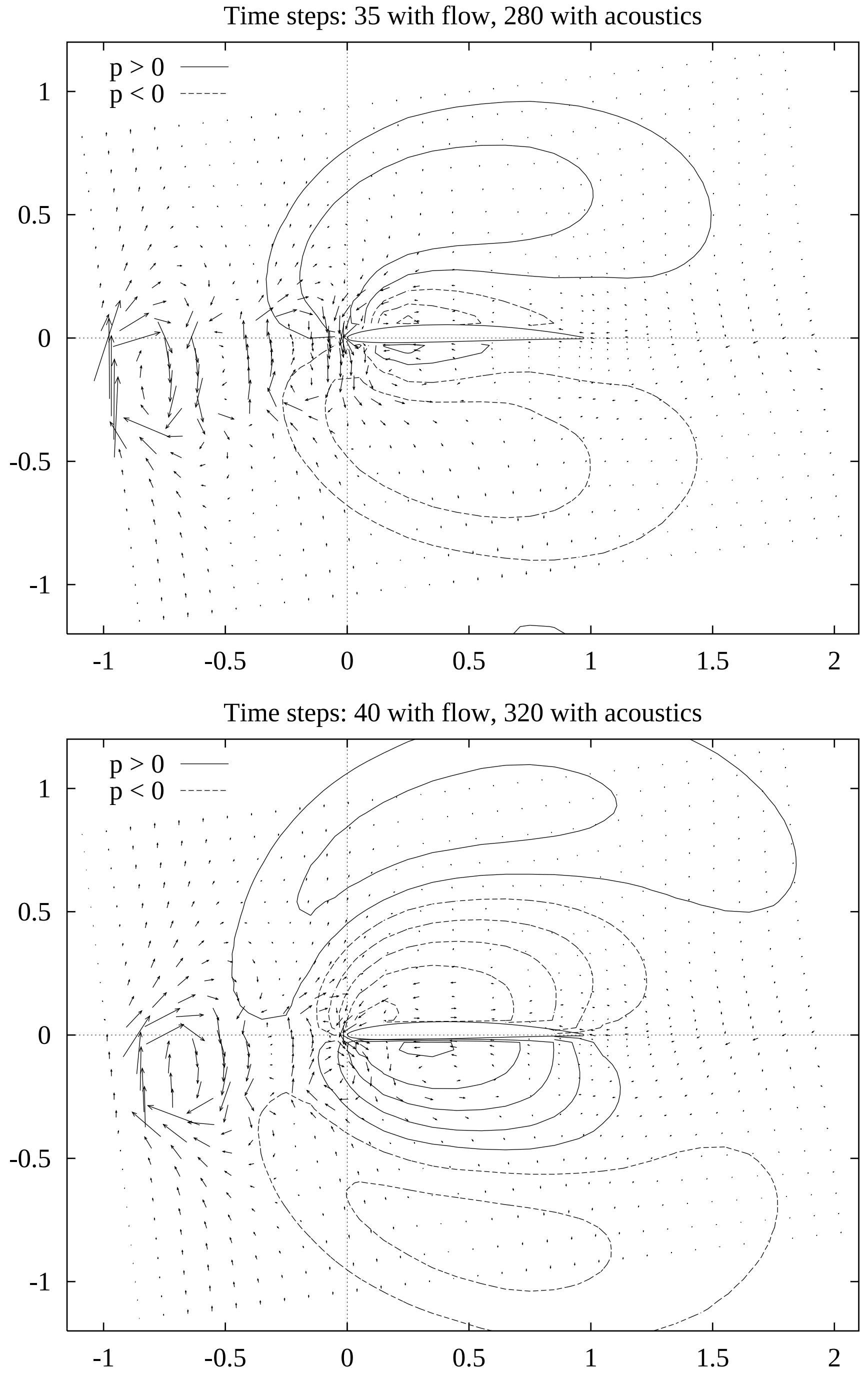


Figure 3 Flow and acoustic perturbation fields (showing superposition of mean-flow and acoustic domains)

seen that the mechanism of inviscid sound generation by perturbations in the flow has been captured by the combined simulation. The present implementation of the coupling technique does not account for sound generated in the wake downstream of the aerofoil.

Conclusions

The physical decomposition of the aerodynamic sound problem was implemented in two dimensions with realistic geometry using overlapping body-fitted and Cartesian meshes and two separate codes coupled in a mono-directional way. The technique was applied to vortex-generated sound at the surface of an aerofoil in sub-sonic conditions. There is no analytic validation but the results obtained are physically correct.

REFERENCES

[CPC95] Croft N., Pericleous K., and Cross M. (1995) PHYSICA: A multiphysics environment for complex flow processes. In Taylor C. *et al.* (eds) *Num. Meth. Laminar & Turbulent Flow '95*, volume 9, part 2, page 1269. Pineridge Press, U. K.

[Cro98] Croft N. (1998) *Unstructured Mesh - Finite Volume Algorithms for Swirling, Turbulent, Reacting Flows.* PhD thesis, University of Greenwich.

[DLP97a] Djambazov G., Lai C.-H., and Pericleous K. (1997) Development of a domain decomposition method for computational aeroacoustics. In *DD9 Proceedings.* John Wiley & Sons.

[DLP97b] Djambazov G., Lai C.-H., and Pericleous K. (1997) Testing a linear propagation module on some acoustic scattering problems. In Tam C. and Hardin J. (eds) *Second Computational Aeroacoustics (CAA) Workshop on Benchmark Problems*, number 3352 in Conference Publications, pages 221–229. NASA.

[DLP98a] Djambazov G., Lai C.-H., and Pericleous K. (1998) Efficient computation of aerodynamic noise. In *Contemporary Mathematics*, volume 218, pages 506–512. American Mathematical Society.

[DLP98b] Djambazov G., Lai C.-H., and Pericleous K. (1998) Staggered-mesh computation for aerodynamic sound. In *4th AIAA/CEAS Aeroacoustics Conference, part 1*, number 98-2219 in AIAA Papers, pages 97–103.

[Har93] Hardin J. C. (1993) Recent insights into computational aeroacoustics. In Mankbadi R. *et al.* (eds) *Computational Aero- and Hydro-acoustics*, volume FED-147, page 1. ASME.

[Leo79] Leonard B. (1979) A stable and accurate convective modelling procedure based on quadratic upstream interpolation. *Computer Methods in Applied Mechanics and Engineering* 19: 59–98.

[Lig52] Lighthill J. M. (1952) On sound generated aerodynamically: I. General theory. *Proceedings of the Royal Society* 211 A: 564–587.

[TW93] Tam C. and Webb J. (1993) Dispersion-relation-preserving finite difference schemes for computational acoustics. *Journal of Computational Physics* 107: 262–281.

[VM95] Versteeg H. and Malalasekera W. (1995) *An Introducion to Computational Fluid Dynamics. The finite volume method.* Longman.

47

Analysis of Substructuring in a Metal Forming Process

A.Müller[1], P.Adamidis[2]

Introduction

Many metal forming processes are solved by the Finite Element method. Due to the high demands in computing time and storage these methods should be parallelized especially for nonlinear use. The substructure method is an integrated approach which allows one to parallelize on the one hand the element loop and on the other hand the solution process. By decomposition of the element graph, the work is distributed homogeneously between the processors. The use of a proper numbering scheme for the nodes gives the system matrix the shape of an arrow matrix which is well suited for parallel solution.

For this representation we regard the process of twin–roll casting [ABBL+98]. Getting results in a reasonable response time is essential to make progress in understanding complex processes which are, for example, needed in reducing the number of production steps of manufacturing steel strips [Jes95].

Sequential Algorithm

The metal forming process is analysed by using the Finite Element system Larstran. This solution system has originally been developed at the University of Stuttgart

1 University of Stuttgart, Institute of Parallel and Distributed Systems, Breitwiesenstrasse 20–22, D–70565 Stuttgart, Germany, current affiliation: LBNL, NERSC, One Cyclotron Road; Berkeley, 94720 California, USA, email: Amueller@lbl.gov
2 same as above, current affiliation: HLRS, Allmandring 30, 70550 Stuttgart, Germany
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

especially for the simulation of nonlinear behaviour of material (in strain–stress and strain–displacement relations) and geometry (i.e. major displacements and rotations). The system is now maintained and distributed by Lasso Consulting Engineers. Further development is also conducted by the Institute for Metal Forming at the RWTH Aachen. The system has proved successful in practical use and is installed at a number of industrial customer locations.

Larstran is an implicit code. The calculations are executed in small increments. Equilibrium of loads and (nonlinear) material reaction is traced by an iteration in each increment. For the simulation of rolling, a thermomechanical algorithm for slow viscous motion problems, coupled with transient heat flow processes is used. Both the mechanical and the thermal problems are iterated to equilibrium in a staggered solution for each increment. The flow–chart is shown in figure 3.

In this application, the two–dimensional elements are quadrilateral plane strain elements. For the constitutive law, a superplastic approach is applied. The contact problem is solved by penalty techniques, i.e. boundary nodes of the workpiece are traced against penetration into the roll's surface, and in the case of penetration, correction terms of high order are introduced into the stiffness matrix and the force vector. The direct solution of the linear equation systems is performed by an LU-decomposition with a skyline storage scheme of the coefficient matrix [Ort88]. All involved matrices are symmetric and positive definite.

Parallelization

Parallelization Strategies

An analysis of the computation time of the Finite Element package shows that most of the computational time is spent in the calculation of the entries in the system matrices and in the solving modules (more than 95%). It is clearly seen, by Amdahl's law, that the parallelization of the solver of the linear system is not sufficient even for large problems. For a parallelization strategy, the element calculation for the system matrices has to be accounted for.

Two elementary strategies can be regarded to parallelize the Finite Element system. One strategy is to parallelize the modules independently from one another, offering the advantage of a possibly independent modular software development for parallelization. Another approach is the coordinated parallelization of the two computationally time consuming parts. The coordination of the element–loop and the solving of the linear system can be realized by the substructuring method [SBG96].

Substructuring

We start with an unstructured grid and a global numbering as usually used in finite element programs. Such grids provide information about the coordinates of the nodes, the topology of the nodes and the assignment of the nodes to elements. A principle example is illustrated in figure 4.

From this information, an element graph is created whose nodes are the elements and whose edges are between nodes of the element graph iff the elements have one or more nodes in common in the original grid of nodes.

In the second step, each processor decomposes the element graph in the same way, using a partitioning method, as given by the software package *metis* [KK97] for example. The partitioning tools aim at a homogeneous distribution of the given graph with the constraint of a heuristic minimization of the number of interface grid points (boundary points) between the different partitions. Because of the homogeneous distribution, the load is evenly balanced, and the minimization aims at a possibly low cost for the explicit communication using a parallel machine with distributed memory. After this, all the information about the element graph and its decomposition between different processes is available to each processor. The chosen number of partitions is the same as the number of involved processors. With this element partitioning, we can easily assign the nodes belonging to a processor. One node can belong to more than one partition.

Each processor now separates the grid points belonging to its partition into inner grid points – lying only in its partition – and into interface (boundary) grid points – belonging also to another partition. After this, each processor defines its local numbering for its inner grid points, to which it has been assigned to, starting from local number 1. The interface grid points are numbered globally starting from the maximum of all inner grid points of all partitions plus 1 up to the number of the interface grid points. The numbering of the boundary grid points is done globally because of the common use of these grid points by different processors.

Next, the calculation of the contributions of the elements is done in parallel. Each processor calculates the elements it is responsible for. We get the following local parts for the system matrix for each processor:

$$\begin{pmatrix} A^i_{II} & A^i_{IB} \\ A^i_{BI} & A^i_{BB} \end{pmatrix}, i = 1, \ldots, n. \tag{1}$$

In (1) I stands for inner and B for boundary, and n is the number of partitions, which is also the number of processors.

The global system matrices have the form of arrow matrices given in (2)

$$A = \begin{pmatrix} A^1_{II} & & & & A^1_{IB} \\ & A^2_{II} & & & A^2_{IB} \\ \ldots & & \ddots & & \vdots \\ & & & A^n_{II} & A^n_{IB} \\ A^1_{BI} & \ldots & \ldots & A^n_{BI} & A_{BB} \end{pmatrix} \tag{2}$$

with

$$A_{BB} = \sum_{i=1}^{n} A^i_{BB}. \tag{3}$$

This global form does not exist explicitly on the parallel system, but rather in parts, as given in (1).

The compilation of the matrices A^i_{II}, A^i_{IB}, A^i_{BI} and A^i_{BB} can be calculated completely independent of each other in parallel. So, with the exception of A_{BB}, the other submatrices of (1) are calculated locally and are, consequently, available in the local memory – no communication is needed. The sum in (3) has to be generated by a reduce–operation.

The system (2) can be also written in the form

$$\begin{pmatrix} A_{II} & A_{IB} \\ A_{BI} & A_{BB} \end{pmatrix}.$$

So the solution of the system

$$A \begin{pmatrix} x_I \\ x_B \end{pmatrix} = \begin{pmatrix} b_I \\ b_B \end{pmatrix} \tag{4}$$

decomposes into the two parts

$$A_{II} x_I + A_{IB} x_B = b_I \tag{5}$$

$$A_{BI} x_I + A_{BB} x_B = b_B. \tag{6}$$

Under the assumption of regularity of the matrices A_{II}, we form (6) - $A_{BI} A_{II}^{-1}$ (5) and get

$$\hat{A} x_B = \hat{b} \tag{7}$$

with the Schur complement $\hat{A}$

$$\hat{A} = A_{BB} - A_{BI} A_{II}^{-1} A_{IB} \tag{8}$$

$$\hat{b} = b_{BB} - A_{BI} A_{II}^{-1} b_I. \tag{9}$$

Due to the shape of the arrow matrix the calculation of the inner degrees of freedoms can be parallelized in a straight forward way, which is described in [MA98]. Here we will present comparisons of sequential and parallel solution of the Schur complement system. For the parallel solution of the Schur complement we have used the general direct solver of the public domain package ScaLAPACK [EEE+97]. The matrices have high conditioning numbers. For the solution of such systems the use of direct methods is more suitable.

Results

The described substructuring method applied to the nonlinear Finite Element Method has been implemented on the parallel computer IBM RS/6000 SP with local memories. A software–technical demand was a far reaching use of existing software of Larstran and the use of existing standard libraries. For basic linear algebra operations, the BLAS libraries, which are available on all major platforms, have been used. The standard message passing library MPI [SHLWD96] has been used. The LU–decomposition and the serial solution of the Schur complement have been executed by the skyline–solver of Larstran. For the parallel solution of the Schur complement the solver for general matrices of ScaLAPACK with a block cyclic distribution has been chosen [EEE+97].

A comparably low degree of freedom is established by the nonlinearity of the simulated process of metal forming. Within each increment, several thermomechanical coupled iterations are executed. Each thermomechanically coupled iteration consists of up to several hundreds of thermal and mechanical iterations. Real world simulation studies need days of computation time at a workstation.

For the analysis of the parallelization, a two–dimensional model has been chosen. In the thermal calculation, to each node one degree of freedom is assigned. In the mechanical calculation, two degrees of freedom are assigned to most of the nodes. Some nodes at the boundary use suppressed values in the mechanical part, i.e. the number of degrees of freedom for the mechanical calculation is lower than twice the number of nodes. Due to the vertical symmetry of the problem, one half of the strip in figure (2) is modeled. So no horizontal displacement of the grid points laying at the symmetric axis is allowed. The model size is given in table 1.

Table 1 model size

elements	nodes	degree of freedom mechanical/thermal
3600	4207	7813/4207

Due to the fixed topology of the grid, the computation of both, the mechanical and the thermal iteration is constant. Therefore we only compare single iterations between different runs parameterized by the number of processors.

The main computational portions are the calculation of the entries for the system matrix (element calculation and assembly) and the solution of the linear systems. In table 2, time portions of a sequential calculated coupled iteration are listed.

Table 2 Sequential time portions (in seconds) of the time consuming parts

mechanical iteration		thermal iteration	
solver	0.17	solver	0.05
element	4.7	element	4.3

In the two–dimensional example under consideration, most of the time is spent in the element loops. In figure 5, the reduction of time to build up the system matrices is shown. Because of faster access to smaller data sets a super–linear speed–up was reached.

Results of time–measurement for solving the linear systems in a mechanical, and thermal, iteration are demonstrated in figure 6 for different numbers of processors. In figure 7 the total computational time of the simulation is shown. The comparison of a sequential and a parallel solver show that the use of a parallel solver for the Schur complement solution is for a moderate number of processors, slightly better than the sequential solution.

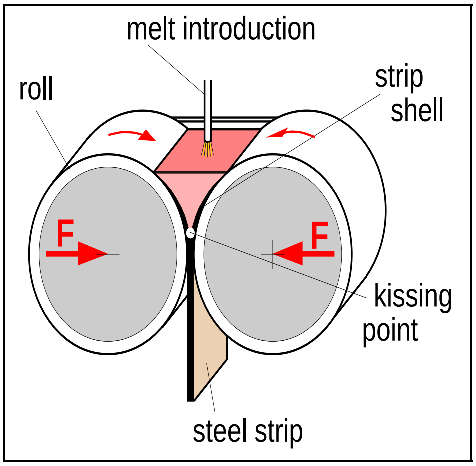


Figure 1 Principle of twin–roll casting process

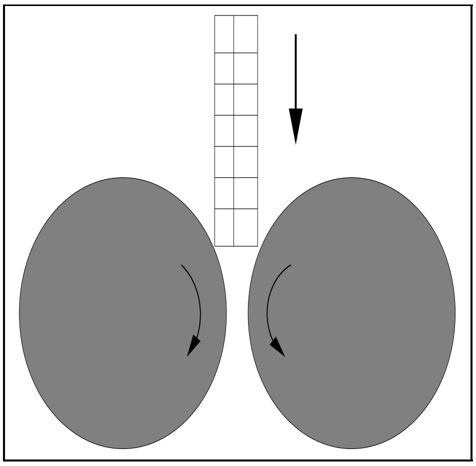


Figure 2 Initial grid

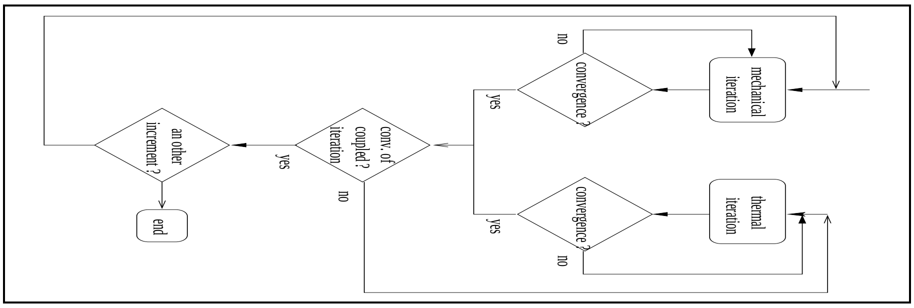


Figure 3 Flow chart of the coupled algorithm

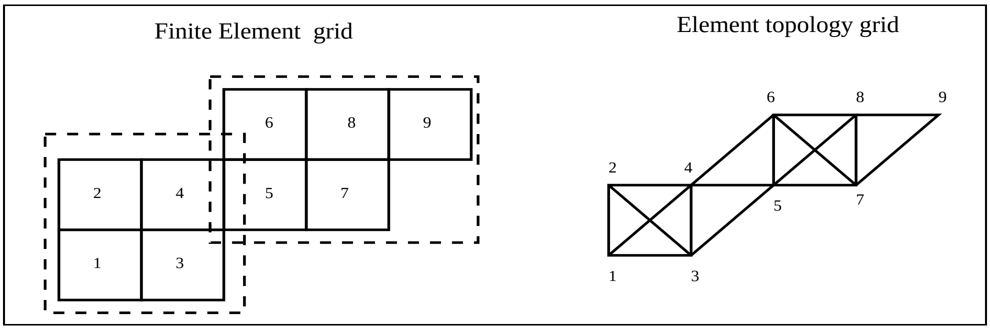


Figure 4 Unstructured element grid and associated element topology grid

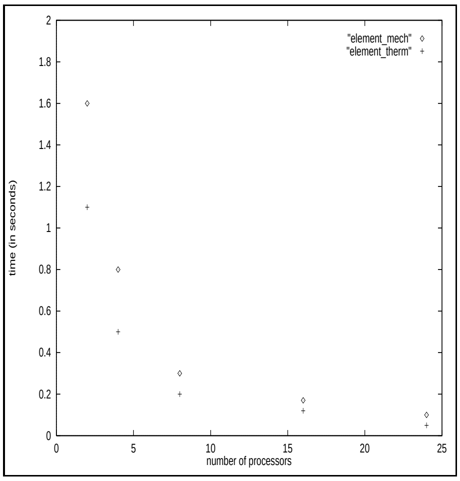


Figure 5 Time for calculation of the entries in the mechanical/thermal system matrices

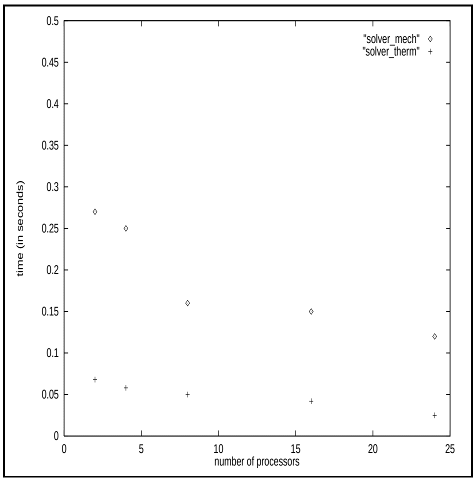


Figure 6 Time for the solution of linear systems in a mechanical/thermal iteration

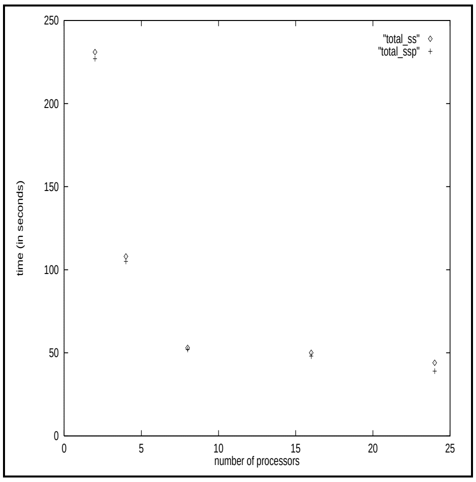


Figure 7 Comparison of the total runtime with a sequential and parallel solution of the Schur complement system

Further Work

Parallelization of three–dimensional modeling of this problem is currently under consideration, for this and other modeling tasks. Furthermore, a comparison with overlapping domain decomposition (Schwarz–methods) is in progress.

Acknowledgements

We are grateful to our colleagues Achim Beck, Yun Ding, Manfred Münch, University of Stuttgart; Matthias Laux, Christoph Pospiech, IBM and Artur Kurz, Lasso Consulting Engineers for their kind support. We thank the Scientific Supercomputing Center Karlsruhe for the possibility of using the IBM RS/6000 SP. The work has been performed in the GRISSLi project, which was sponsored by the German Federal Ministry of Education, Science, Research and Technology under contract No. 01 IS 512 A-C/GRISSLi.

REFERENCES

[ABBL+98] Adamidis P., Beck A., Becker-Lemgau U., Ding Y., Franzke M., Müller H. H. M. L. A., Münch M., Reuter A., Steckel B., and Tilch R. (September 1–3 1998) Steel strip production - a pilot application for coupled simulation with several calculation systems. *Metal Forming 98* .

[EEE+97] (Editor) L. S. B., (Editor) G. H., (Editor) D. W., (Editor) J. D., (Editor) J. C., (Editor) A. C., (Editor) E. D., (Editor) J. D., (Editor) I. D., , (Editor) S. H., (Editor) A. P., (Editor) K. S., and (Editor) R. C. W. (1997) *ScaLAPACK User's Guide.*

[Jes95] Jestrabek J. (1995) *Stahlbandherstellung nach dem Zweirollenverfahren – Modellierung des Strömungs- und Temperaturfeldes.* PhD thesis, RWTH Aachen.

[KK97] Karypis G. and Kumar V. (1997) METIS:A Software Package for Partitioning Unstructured Graphs, Partitioning Meshes, and Computing Fill-Reducing Orderings of Sparse Matrices. Technical report, University of Minnesota,Department of Computer Science / Army HPC Research Center.

[MA98] Müller A. and Adamidis P. (July 13-16 1998) Parallelization of a metal forming process by substructuring. *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'98)* .

[Ort88] Ortega J. M. (1988) *Introduction to parallel and vector solution of linear systems.* Plenum Press.

[SBG96] Smith B., Bjørstad P., and Gropp W. (1996) *Domain decomposition : parallel multilevel methods for elliptic partial differential equations.* Cambridge University Press.

[SHLWD96] Snir S. W., Huss-Ledermann S., Walker D. W., and Dongarra J. (1996) *MPI The Complete Reference.* The MIT Press.

48

The Parallel Solution of Early-exercise Asian Options with Stochastic Volatility

A.K. Parrott [1], N.A.L. Clarke [2]

Introduction

This paper describes an parallel semi-Lagrangian finite difference approach to the pricing of early exercise Asian Options on assets with a stochastic volatility. A multigrid procedure is described for the fast iterative solution of the discrete linear complementarity problems that results. The accuracy and performance of this approach is improved considerably by a strike-price related analytic transformation of asset prices.

Asian options are contingent claims with payoffs that depend on the average price of an asset over some time interval. The payoff may depend on this average and a fixed strike price (Fixed Strike Asians) or it may depend on the average and the asset price (Floating Strike Asians). The option may also permit early exercise (American contract) or confine the holder to a fixed exercise date (European contract). The Fixed Strike Asian with early exercise is considered here where continuous arithmetic averaging has been used. Pricing such an option where the asset price has a stochastic volatility leads to the requirement to solve a tri-variate partial differential inequation in the three state variables of asset price, average price and volatility (or equivalently, variance). The similarity transformations [WDH93] used with Floating Strike Asian options to reduce the dimensionality of the problem are not applicable to Fixed Strikes

1 School of Computing & Mathematical Sciences, University of Greenwich, 30 Park Road, London SE10 9LS, U.K., email: pa10@gre.ac.uk
2 Oxford University Computing Laboratory, now with Goldman Sachs, London
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

and so the numerical solution of a tri-variate problem is necessary. The computational challenge is to provide accurate solutions sufficiently quickly to support real-time trading activities at a reasonable cost in terms of hardware requirements.

American options with stochastic volatility

American Asian options with a stochastic volatility contain the vanilla American stochastic volatility option as a sub-problem; consequently the solution of this sub-problem (see [CP98] for more details) is summarised here. A standard American option with a stochastic volatility has a share price process S_t and its variance process Y_t (the variance has been used rather than the volatility $\sqrt{Y_t}$) which satisfy

$$\begin{aligned} dS_t &= \mu S_t dt + \sqrt{Y_t} S_t dB_t^1 && (1)\\ dY_t &= \alpha(\beta - Y_t)dt + \gamma\sqrt{Y_t} dB_t^2 && (2)\end{aligned}$$

where $B_t = (B_t^1, B_t^2)$ is a two-dimensional standard Brownian motion with correlation coefficient $\rho \in [-1, 1]$. The square-root mean-reverting stochastic volatility model described in [BR94] has been used where the reversion is to some mean value β at a rate determined by α, and $\gamma > 0$ governs the volatility of the variance process.

For an American option the following pricing in-equation must hold [WDH93] for the price u

$$\mathcal{D}u(s, y, t) \leq 0 \;\; \forall (s, y) \in \Omega, \quad t \in [0, T), \tag{3}$$

$$u(s, y, T) = g(s, T) \tag{4}$$

in the infinite quarter-plane $\Omega = \{(s, y) | s \geq 0, y \geq 0\}$, where s and y are the asset price and volatility variables respectively, g is the payoff and the pricing operator $\mathcal{D}$ (see [BR94]) is given by

$$\begin{aligned}\mathcal{D} = \frac{\partial}{\partial t} + \frac{1}{2}[s^2 y \frac{\partial^2}{\partial s^2} + 2\rho\gamma y s \frac{\partial^2}{\partial s \partial y} + \gamma^2 y \frac{\partial^2}{\partial y^2}] \\ + rs\frac{\partial}{\partial s} + [\alpha(\beta - y) - \lambda\gamma\sqrt{y}]\frac{\partial}{\partial y} - r,\end{aligned} \tag{5}$$

where λ is the market price of risk and r is the risk-free interest rate. This leads to the following linear complementarity problem,

$$\left.\begin{array}{rcl} u(s, y, t) & \geq & g(s, t), \\ \mathcal{D}u(s, y, t) & \leq & 0, \\ (u(s, y, t) - g(s, t))\mathcal{D}u(s, y, t) & = & 0, \end{array}\right\} \; \forall (s, y, t) \in \Omega \times [0, T] \tag{6}$$

with initial data $u(s, y, T) = g(s, T)$. The boundary conditions for this problem are:

$$\begin{aligned}
u(0,y,t) &= g(0,t)\ \forall y \geq 0 \ \text{ and } \ t \in [0,T]. && (7)\\
u(s,y,t) &\rightarrow 0 \text{ as } s \rightarrow \infty\ \forall y \geq 0 \text{ and } t \in [0,T], && (8)\\
\frac{\partial u(s,y,t)}{\partial y} &\rightarrow 0 \text{ as } y \rightarrow \infty\ \forall s \geq 0 \text{ and } t \in [0,T], && (9)
\end{aligned}$$

A boundary condition is not required on $y = 0$ since the differential operator (5) becomes hyperbolic here. The domain is truncated at finite values $[s_{max}, y_{max}]$ for numerical solution, chosen sufficiently removed from the region of pricing interest for the solution to be unaffected.

The numerical procedure described in [CP98] uses an adaptive–upwind finite difference approximation together with a projected full approximation scheme (PFAS) Multigrid method [BC83]. Co-ordinate stretching transformations were also used for optimal efficiency,

$$\begin{aligned}
\tilde{s} &= sinh^{-1}(s-k) - sinh^{-1}(-k);\ \forall s \in [0, s_{max}], && (10)\\
\tilde{y} &= ln(y') = ln(y/\beta)\ \ \forall y \in [y_0, y_{max}], && (11)
\end{aligned}$$

where $y_0/\beta \ll 1$ and k is the strike price for the option. Both transformations induce rapid coarsening at large values of $\tilde{s}$ and $\tilde{y}$ enabling the truncation boundary to be placed far enough away for the asymptotic conditions (8), (9) to hold approximately. The co-ordinate stretching produced by both these transformations is equivalent to the use of a smooth and highly refined mesh about the strike.

A theta-method approximation of (5) on a uniform set of mesh points $(\tilde{s}_i, \tilde{y}_j) \in \Omega'_h \subseteq \Omega' = [0, s_{max}] \times [0, y_{max}]$, is then

$$\mathcal{D}u(\tilde{s}_i, \tilde{y}_j) \simeq \mathcal{D}_h U^n_{ij} = (U^{n+1}_{ij} - U^n_{ij})/\Delta t^n + \theta \mathcal{L}_h U^{n+1}_{ij} + (1-\theta)\mathcal{L}_h U^n_{ij} \qquad (12)$$

where $(0 \leq \theta \leq 1)$, U^n_{ij} is the discrete option price and $\mathcal{L}_h$ is an adaptive-upwind approximation to the drift-diffusion terms in (5). This leads to the following discrete linear complementarity problem, with initial data $U^N_{ij} = g(s_i, T)$, to be solved at a backwards sequence of times $t^n, n = N-1, N-2, ...0$,

$$\left.\begin{array}{rcl}
U^n_{ij} & \geq & g(s_i, t^n),\\
U^n_{ij} - (1-\theta)\Delta t^n \mathcal{L}_h U^n_{ij} & \geq & U^{n+1}_{ij} + \theta \Delta t^n \mathcal{L}_h U^{n+1}_{ij}\\
(U^n_{ij} - g(s_i, t^n))\mathcal{D}_h) U^n_{ij} & = & 0
\end{array}\right\} \ \forall (i,j) \in \Omega'_h, \qquad (13)$$

The value $\theta = 0.5 + \epsilon, \epsilon > 0$ was used since it gives unconditional stability (essential because of the coordinate stretching) and close to second order accuracy in the timestep $\Delta t^n = t^{n+1} - t^n$, provided $\epsilon \simeq O(h)$.

The conventional solution method for (13) is Projected Successive Over-Relaxation (PSOR) however the PFAS Multigrid method described in [CP98] is much superior as can be seen from in Table 1. Grid-independent convergence required the use of an adapted $\tilde{s}$-line smoother.

Table 1 Comparison of Multigrid with PSOR

	Avg. Iterations		CPU (s)	
Grid(Ω_h)	MG	PSOR	MG	PSOR
33 × 13	1.52	22.0	0.59	1.44
65 × 25	2.04	42.7	2.62	5.25
129 × 49	2.08	72.6	9.76	42.1
257 × 97	2.52	176.4	49.3	631.6

American Asian Options

The value of Asian options depend additionally on the average price; the continuously sampled average A_t is defined as

$$A_t = \frac{1}{t}\int_0^t S_\tau d\tau, \quad \text{i.e.} \quad dA_t = \frac{1}{t}(S_t - A_t)dt$$

where $A_0 = S_0$, leading to the following pricing operator

$$\begin{aligned}\mathcal{D}_A u(s,a,y,t) &= \frac{\partial u}{\partial t} + \frac{1}{2}[s^2 y \frac{\partial^2 u}{\partial s^2} + 2\rho\gamma y s \frac{\partial^2 u}{\partial s \partial y} + \gamma^2 y \frac{\partial^2 u}{\partial y^2}] \\ &\quad + rs\frac{\partial u}{\partial s} + [\alpha(\beta - y) - \lambda\gamma\sqrt{y}]\frac{\partial u}{\partial y} + \frac{1}{t}(s-a)\frac{\partial u}{\partial a} - ru\end{aligned} \tag{14}$$

for the option price u on $\Omega = \{s,a,y : s \geq 0, a \geq 0, y \geq 0\} \times [0,T]$, with $u(s,a,y,T) = g(a,T)$ at $t = T$, for some payoff g and where a is the average-price variable. A fixed-strike Asian has a payoff of the form $g(s,a,t) = max(a-k,0)$. Early exercise means that the option price satisfies the following LCP

$$u \geq g(a,t), \tag{15}$$
$$\mathcal{D}_A u \leq 0 \tag{16}$$
$$(u-g)\mathcal{D}_A u = 0 \tag{17}$$

with appropriate boundary conditions [Cla98]. It can be seen that the pricing operator (14) is hyperbolic in the average price direction. Accurate approximation in this direction requires special care so a semi-Lagrangian approach was used. The Lagrangian derivative of u along any trajectory $\mathcal{T}(a,t)$ is

$$\frac{du}{dt} = \frac{\partial u}{\partial t} + \frac{\partial u}{\partial a}\frac{da}{dt}$$

and $\mathcal{D}_A$ can be simplified by choosing the trajectory to satisfy

$$\frac{da}{dt} = \frac{1}{t}(s-a) \tag{18}$$

Consider a uniform 3-D set of meshpoints $(s_i, a_j, y_k) \in \Omega_h$. Let $\mathcal{T}(\bar{a}_j, t^n + \Delta t; a_j, t^n)$ be a trajectory satisfying (18) where (a_j, t^n) is the departure point and $(\bar{a}_j, t^n + \Delta t)$

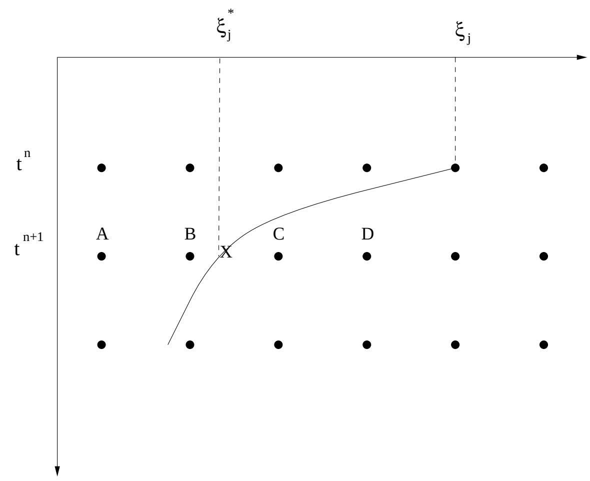


Figure 1 Lagrangian trajectory from (a_j, t^n) to $(\bar{a}_j, t^n + \Delta t)$.

is the arrival point of the trajectory (see Figure 1). Note that equation (18) can be solved exactly. Integrating (16) along this trajectory gives

$$u(s_i, \bar{a}_j, y_k, t^n + \Delta t) - u(s_i, a_j, y_k, t^n) \leq -\int_{t^n}^{t^n+\Delta t} \mathcal{L} u d\tau,$$

where $\mathcal{L}$ contains only asset-price/volatility drift diffusion terms. Approximating this integral with a weighted average of the end-point values leads to the discrete approximation

$$\tilde{U}_{ijk}^{n+1} - U_{ijk}^n \leq -\Delta t((1-\theta)\mathcal{L}_h U_{ijk}^n + \theta \mathcal{L}_h \tilde{U}_{ijk}^{n+1})$$

where $\mathcal{L}_h$ is the adaptive-upwind approximation described in the previous section of the asset price - volatility component of the pricing operator, and $\tilde{U}_{ijk}^{n+1}$ is obtained via cubic interpolation.

This semi-Lagrangian approach leads to the following discrete linear complementarity problems, with initial data $U_{ijk}^N = g(a_j, T)$, to be solved at a backwards sequence of times $t^n, n = N-1, N-2, ..., 0$,

$$\left.\begin{array}{ccc} U_{ijk}^n & \geq & g(a_j, t^n), \\ U_{ijk}^n - (1-\theta)\Delta t^n \mathcal{L}_h U_{ijk}^n & \geq & \tilde{U}_{ijk}^{n+1} + \theta \Delta t^n \mathcal{L}_h \tilde{U}_{ijk}^{n+1} \\ (U_{ijk}^n - g(a_j, t^n)\mathcal{D}_h U_{ijk}^n & = & 0 \end{array}\right\} \forall (i,j,k) \in \Omega_h,$$

The system of equations has a block-diagonal structure so that each $s - y$ plane can be solved independently using a modified form of the PFAS procedure described in Section 48. Co-ordinate stretching transformations are again essential for optimal efficiency. The results in Table 2 demonstrate the convergence of the method for the stochastic volatility American Asian and indicate that acceptable accuracy can be obtained on the $65 \times 65 \times 25$ mesh.

Table 2 Fixed Strike Asian Put Convergence results, $K = 10$

Asset Price (s)	Grid Size	Volatility ($\sqrt{y}$) 0.1	0.2	0.3	0.4	0.5
9.5	$33 \times 33 \times 13$	0.7733	0.7870	0.8104	0.8353	0.8840
	$65 \times 65 \times 25$	0.6372	0.6702	0.7300	0.8036	0.8869
	$97 \times 97 \times 33$	0.6216	0.6646	0.7269	0.8038	0.8881
10	$33 \times 33 \times 13$	0.2281	0.2762	0.3569	0.4477	0.5444
	$65 \times 65 \times 25$	0.2136	0.2762	0.3599	0.4535	0.5535
	$97 \times 97 \times 33$	0.2129	0.2756	0.3594	0.4538	0.5531
10.5	$33 \times 33 \times 13$	0.0638	0.1039	0.1686	0.2453	0.3284
	$65 \times 65 \times 25$	0.0592	0.0978	0.1604	0.2388	0.3287
	$97 \times 97 \times 33$	0.0583	0.0972	0.1602	0.2386	0.3290

Parallelization

The algebraic structure of the semi-Lagrangian approach makes it very suitable for parallelization. An s-line smoother is used in the PFAS multigrid method so it is advantageous on small-scale parallel systems to partition in the y-direction only, since this will not affect the smoother performance (a zebra ordering was used), non-local memory requests can be arranged to be for contiguous data, and this choice will keep trajectories within the partition. The surface to volume ratio for a 1-D partition degrades fairly quickly with the number of processors p and grid-coarsening so it was important to model the performance of this approach to establish its efficiency on different type of parallel systems.

The American Asian algorithm described above can be described in terms of the performance cost of a single iteration decomposed into its floating point cost, its remote memory access cost G, and some multiple of the processor network synchronization cost L . G and L are expressed in units of flop using a representative megaflop rate s. G can be written as gh where h is the maximum number of bytes read and written by any one processor during the iteration (termed an *h–relation*) so that g is the average cost in flop of accessing one remote word. This communication model is conservative, emphasizing global access; however low-level parallel library optimizations (which aggregate remote access between synchonisations to minimise latency and maximise bandwidth) make this a practical approach. The parallelisation relied on the BSP Library [Hil95] which uses such optimisations and for which estimates of L and g are available for many systems (see Table 3).

The iteration cost model presented assumes that the partitioning is load balanced and ignores the costs of restriction and prolongation operations within the multigrid iteration (they are small in comparison to the smoothing costs). These assumptions lead to the following

$$\text{iteration cost} = \sum_{i=1}^{N_{grids}} \left[4B_1 \left(\frac{M r^{i-1}}{p} \right) + \left(32(p-1)\tilde{M}\tilde{r}^{i-1} \right) g + 16L \right].$$

Table 3 BSP Parameters

Machine	s (Mflops)	p	L	g
SGI Power-Challenge	55	2	627	7.6
		4	1248	7.4
Pentium (400MhZ)	40	2	20500	51
		4	29500	65
		8	39500	66

Table 4 Parallel efficiency (%) for the SGI Power Challenge

Grid Size		No. of Processors (p)		
		p=2	p=3	p=4
13 × 33 × 33	predicted	93	82	–
13 × 33 × 33	measured	92	83	–
25 × 65 × 65	predicted	97	90	83
25 × 65 × 65	measured	90	86	82

where r and $\tilde{r}$ are the grid/interface coarsening factors, M and $\tilde{M}$ are the total number of grid and interface mesh points and $B_1 \approx 300$ is the flop count. The cost of the semi-Lagrangian integration is an additional $B_2 M/p$ flop where $B_2 \approx 100$.

The parallel performance of the algorithm is shown in Table 4; the trend is comparable to those predicted by the performance model. The 1-D partitioning and grid coarsening limitations are visible in the trend but do not affect performance significantly on the 4-processor system. The predicted efficiency (not shown) for the 13×33×33 is 77%, signalling the expected turndown due to 1-D partitioning. Work is in progress to repeat these results to other parallel systems. The fine mesh results can be obtained in 1.5 minutes (see Table 5) for a short-dated options, demonstrating that the semi-Lagrange algorithm combined with the PFAS multigrid approach has real-time performance on modestly parallel systems. Furthermore the SGI system performance can now be reproduced much more cheaply using Pentium based multiprocessor systems making this a widely affordable approach. The longer dated options are still rather slow to compute however the algorithm could be made more efficient with the use of adaptive time-stepping in these cases [CP98].

Conclusions

The semi-Lagrangian time integration has been shown to be well-suited to American Asian-options providing accurate integration of the hyperbolic terms in the pricing operator and making two-factor early-exercise solution methods applicable. Stochastic

Table 5 CPU times (in seconds) required for solution.

Time to Expiry	$33 \times 33 \times 13$			$65 \times 65 \times 25$			
	p=1	p=2	p=3	p=1	p=2	p=3	p=4
0.25	20	11	8	325	183	126	89
0.50	40	22	16	630	351	242	195
1.00	80	43	32	1264	695	498	382

volatility Asian options with early exercise can be solved in parallel by partitioning in the volatility direction and using a zebra-ordering; good performance was obtained on small parallel systems giving real-time computation of option prices for a reasonable cost. Finally the BSP performance model gave a useful indication of the portability of this parallel application.

REFERENCES

[BC83] Brandt C. and Cryer C. W. (1983) Multigrid algorithms for the solution of linear complementarity problems arising from free boundary problems. *SIAM Journal of Scientific and Statistical Computing* 4: 655–684.

[BR94] Ball C. A. and Roma A. (1994) Stochastic volatility option pricing. *Journal of Financial and Quantitative Analysis* 29: 589–607.

[Cla98] Clarke N. A. L. (1998) *The Numerical Solution of Financial Derivatives.* PhD thesis, University of Oxford.

[CP98] Clarke N. A. L. and Parrott A. K. (1998) Multigrid for American option pricing with stochastic volatility. *To be published in Applied Mathematical Finance* .

[Hil95] Hill J. M. D. (November 1995) The Oxford BSP toolset users guide. *Oxford University Computing Laboratory* .

[WDH93] Wilmott P., Dewynne J., and Howison S. (1993) *Options Pricing: Models and Computation.* Oxford Financial Press.

49

Efficient Mesh Partitioning For Adaptive HP Finite Element Meshes

A. K. Patra [1] & D. W. Kim[2]

Introduction

The use of domain decomposition solvers presumes the existence of a partitioning of the domain that distributes the computational effort equitably. Adaptive *hp* finite elements which are capable of delivering solution accuracies far superior to classical $h-$ or $p-$version finite element methods, for a given discretization size [BS94] create special difficulties in generating such load balanced partitions. Two major difficulties that arise in partitioning such meshes are a) the choice of a good *a priori* measure of computational effort, which can be equidistributed among the processors and b) minimizing the data migration among processors as the mesh changes and is repartitioned. In uniform meshes using simple solvers, computational effort is directly related to the degrees of freedom in each sub-domain. In schemes using *hp* meshes, and domain decomposition solvers (e.g. preconditioned iterative substructuring [JTOF97]), the computational effort is not simply function of the degree of freedom. The distribution of polynomial orders, cost of constructing the preconditioner, connectivity have important effects that are difficult to quantify. In *hp* methods, especially when using adaptive strategies like the "Texas 3 step" [JTOF97] the mesh changes in size very rapidly (often an order of magnitude more degrees

1 Corresponding Author, Department of Mechanical Engineering, University at Buffalo, Buffalo NY 14260, email: abani@eng.buffalo.edu
2 Address same as above. Authors were supported by National Science Foundation Award # ASC-9702947, Computations were performed on NPACI computers
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

of freedom in two solution cycles). Repartitioning to balance the load again usually results in migration of large data sets.

We introduce here a new measure of computational effort and load balancing strategy designed to mitigate these problems. The load measure introduced is error on a coarser mesh. We will study its use for several complex adaptive *hp* meshes and compare its effectivity to the degrees of freedom(DOF) normally used. Our use of the error on a coarse mesh as a measure of computational effort has motivated a simple strategy whereby we repartition before we refine/enrich the mesh. Thus new elements are created on the processors on which they will eventually be used greatly alleviating the data migration problem. Again, we will test this strategy on several complex meshes and compare it to the traditional repartitioning after refinement. Such a strategy will greatly reduce the need for complex dynamic rebalancing procedures.

Mesh Partitioning And Partitioners

The mesh partitioning problem may be viewed as a weighted graph partitioning problem with either the nodes/elements representing vertices of a graph and the edges being either the edges of the finite element mesh or the element connectivity respectively. Vertex and edge weights can be assigned based on the computational effort associated with different nodes and the the need for communication between them. The abstract problem may then be stated as:

Given a graph G with n weighted vertices and m weighted edges, partition the vertices into p sets in such a way that the sum of vertex weights is equidistributed and that of edges crossing between sets is minimized.

This is a problem that is provably “NP hard” and hence solved with heuristics [KK97]. Several tools are available to implement the partitioning once the weighted graph is defined. The primary tools we use here are the Metis (Karypis and Kumar [KK97]) and Chaco packages (Hendrickson et al.[HL93]. These tools are based on multilevel partitioning ideas. At the coarsest level of the graph the Fiedler vector is used to define an ordering of the nodes of the graph and this is then recursively bisected. While broadly similar in their underlying ideas, Chaco and Metis differ significantly in their implementation of these ideas. Another partitioner we make use of in these studies is based on using space filling curves to order the nodes/elements of the finite element mesh (Patra and Oden [OP95]). These curves are mappings to n dimensional unit hypercubes ($[0,1] \times [0,1] ... \times [0,1]$) from the unit interval $[0,1]$. The inverse of these mappings defines an ordering of the elements which can then be used in a k-way split or a recursive bisection to obtain the partitioning.

For adaptive meshes in which the mesh changes as the computation procedes the partitioning must also change to reflect the changing grid. The changing of partitioning causes data to be migrated to reflect the new partitioning. This migration can be a serious bottleneck in obtaining good parallel efficiencies. We can now define the three goals of partitioning such meshes as obtaining a good balance of computational effort, minimizing communications among processors and data migration as the mesh and partitioning change. While the basic mesh partitioning problem has been explored by many researchers, the problem of minimizing data migration when dealing with changing meshes(dynamic graphs) has not received much attention. Together, with the problem of estimating computational effort in *hp* meshes we thus have a challenging task in formulating appropriate algorithms and strategies for partitioning adaptive *hp* meshes.

A priori Measures Of Computational Effort

Load Measures

Any problem decomposition strategy for efficient parallel computing requires an estimate of computational effort on the different tasks that are a part of the computation. In mesh partitioning for data parallel computations on finite element meshes, this translates to a measure of computational effort linked to the elements/nodes that comprise the mesh. This implies that a usable measure of computational effort must also posses the local additivity property. The total load measure of should be the sum of the load measure associated with each element.

Difficulties With DOF based measures

In simpler uniform or h-adaptive meshes and solution strategies using direct solvers, the number of simultaneous equations to be solved also known as degrees of freedom (DOF) in the model is an efficient measure of computational effort in the problem. This follows from well known results relating the operation count in the equation solver to $O(N^{\alpha})$ where α is determined by the choice of solver and the sparsity of the matrix, and N is the number of unknowns. This has motivated the traditional choice of DOF as a computational load measure. Communication costs are usually measured by counting the number of unknowns on the interfaces generated by a partitioning of the mesh.

However, when using higher order elements and iterative solvers (preconditioned conjugate gradient type in our applications [JTOF97]), such a simple and straightforward count of the number of unknowns is not adequate. The spatial distribution of computational effort for these meshes is highly non-uniform. The element stiffness matrix corresponding to a bilinear quadrilateral element for solving the equations of elasticity in two dimensions comprises of only 8 DOF whereas the same matrix for a seventh order element has 64 DOF! In adaptive *hp* meshes where the mix of element orders is highly non-homogeneous and determined at run time the non-uniformity of the distribution of computational effort can thus be very severe.

We also note that degrees of freedom associated with higher orders do not involve the same amount of computation as those associated with lower orders especially for the degrees of freedom internal to an element as these are usually eliminated at the element level using a substructuring technique. The cost associated with other degrees of freedom like those on the side of the element is also significantly lower as they are shared by only two elements. The connectivity (number of other DOF that any DOF interacts with) thus has great influence on the computation cost. The matrix sparsity is also dictated by this and thus this has great implications for storage and memory requirements. The conditioning of the system (usually defined as the ratio of the maximum and minimum eigen values of the system) will also influence the computational effort greatly since it controls the number of iterations required for an iterative solver to converge. While both connectivity and conditioning appear to be good measures of computational effort they fail the local additivity test. Local connectivities do not add up. Conditioning cannot even be defined at the local level. Thus neither of these may be used as a suitable primary computational measure.

Numerical Error Based Measure

In the solution adaptive meshing process distribution of h and p is determined by estimates of the error distribution in a previous coarser mesh [OP95]. It is thus reasonable to assume that the local distribution of computational effort on a refined/enriched mesh would also be determined by the error distribution in a previous mesh. A variety of such error estimates are widely available now in the research literature. For our study here we use an error estimator of the type introduced by Bank and Smith [BS93] and further developed by Patra and Oden[PO97]. In this type of error estimator the coarse mesh problem is used to define a local boundary value problem on each element using the underlying operator and a hierarchical discretization. For example if the mesh uses quadratic elements then the local problem for the error in each element uses the additional shape functions corresponding to a cubic element. We now propose that the **square of the norm of the error in a coarser mesh** be used as a measure of computational effort for an adapted mesh and compare this with the more classical degrees of freedom in the mesh (DOF) measure. Communication costs will be factored into either measure by minimizing the number of DOF on the interfaces.

Use of Load Measures and Weighted Graph Partitioning

The weighted graph partitioners (Metis and Chaco) can account for different computational measures through the use of edge and vertex weights for the graph. The partitioners then attempt to equidistribute the weights among the different partitioners while keeping the weights associated with the edges cut by the partitioning minimum. The conventional weighting schemes are all designed for h adaptive meshes. We introduce here a modified weighting scheme for hp adaptive meshes. Since each element in the mesh constitutes a node in the graph either the square of the norm of the error or the square of the polynomial order associated with the element is used as the vertex weight. The purpose of the edge weight is to minimize the number of unknowns on the inter-partition boundary thus minimizing communications. For the edge weight we thus use the higher polynomial order of the two elements that the edge connects. This provides a good edge weight since the number of unknowns on the inter-partition boundary is governed by the number of elements on the boundary and their polynomial orders.

Partitioning and Repartitioning Strategies

Basic Parallel Adaptive Algorithm

This type of error based problem decomposition can be easily embedded in an overall adaptive algorithm using for example the "Texas-three step" *hp* adaptive strategy developed by Oden and Patra [OP95]. In this strategy the adaptive process is started using a uniform mesh. Errors, convergence rates and other constants are then estimated on this simple mesh. *a priori* estimates of the form $||e||_1 \leq C\frac{h^\mu}{p^\nu}||u||_1$ where $||e||_1$ is the norm of the error, C is a constant, μ and ν are convergence rates, and $||u||_1$ is the H^1 norm of the solution, are then used to predict the level of mesh modification required for a target error, first using only h refinements (the second step of three), and then using p enrichment (the third step of three). When performed in a parallel environment each step of mesh modification is followed by a repartitioning of the mesh to balance the loads again. We call this strategy **Partition After Refinement (PAR)**. This type of repartitioning from scratch often results in high data migration that can cripple the computations as repartitioning takes place. The use of the coarse mesh error as a partitioning measure motivates an alternate strategy that promises to greatly reduce data migration.

Partition Before Refinement (PBR)

In this strategy we take advantage of our *a priori* knowledge of the computational effort in the next mesh via the error estimate to compute a repartitioning before modifying the mesh. The new/modified elements are then generated on the processor that will use them – thus greatly reducing mesh migration. It is obvious that the cost of moving a few parent elements will be significantly less than migrating a host of child elements. We name this strategy **Partition Before Refinement (PBR)**.

The above parallel adaptive process can be modified to incorporate this strategy. After the first step the mesh can be repartitioned and redistributed among the processors based on the error in the solution. Then, the mesh modification is carried out. Again following the solution of this step the mesh is repartitioned and redistributed among the processors based on the error in the solution. We have implemented both of the load measures and partitioning strategies outlined here in an MPI (message passing interface) based parallel *hp* adaptive code. Results and conclusions will be discussed in the next section.

Results and Discussion

Measures of partition quality

We define two quantities from [PO95] to measure the quality of partitioning;

Imbalance Fraction(IF): This is designed to measure load imbalance and is defined as:. $IF_i = \frac{N_i - N_{av}}{N_{max}}$ where N_i is the DOF associated with the subdomain i, N_{av} and N_{max} are the average and maximum DOF associated with the different subdomains.

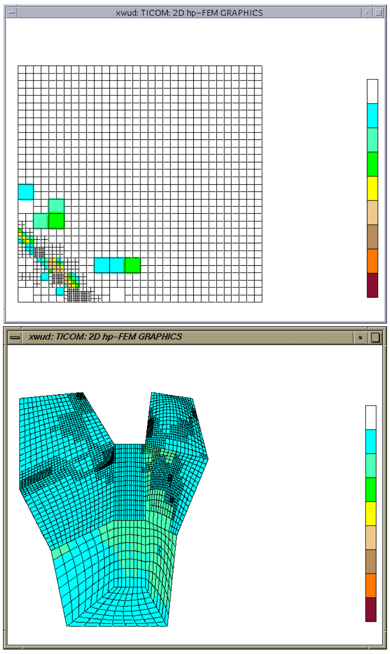


Figure 1 Test Mesh1 and 4

Shape Fraction(SF): This is defined as the percentage of interface DOF in a particular partition with respect to the total DOF associated that partition. For the whole domain all DOF on interfaces are compared to the total DOF in the problem. This is a measure of interprocessor communication. $SF = \frac{N_{int}}{N_i}$ where N_{int} is the number of DOF on all the interfaces associated with subdomain i.

Test Problem

We used the model problem of heat conduction with a variety of geometries and boundaries to test our problem. This choice of test problem helps keep the solution to the problem used in this study simple and allows us to focus on the partitioning. Four adaptive *hp* meshes are used this study (See Fig. 1 for samples) They have different sizes ranging from 2364 to 11242 degrees of freedom. We use all the partitioners and partitioning strategies described in the previous sections.

Error Based Partitioning

We have used error in a coarser mesh to partition the grid. New elements inherit error from parent elements. Figure 2 show the corresponding mesh partitions for mesh 4. We recorded the 2 measures IF and SF defined previously for all meshes. We average their measures using different partitioners and present the results in Figure 3.

We observe here some general trends when partitioning the above test meshes using the strategies described above. We note that the shape fractions of the partitions

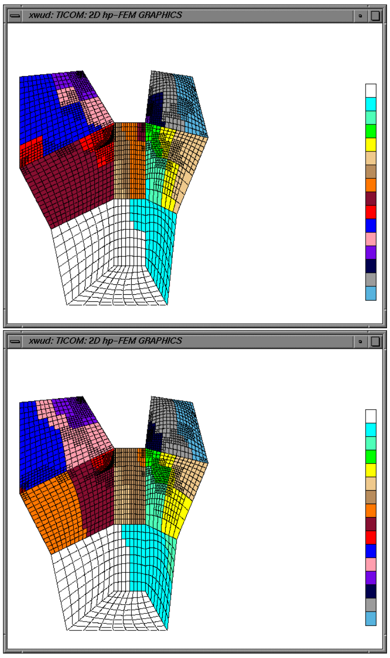


Figure 2 Error Based Partition & DOF Based Partition Using Linear Method and No local Refinement on Test Mesh4

when using error as the measure are comparable (usually within 10-15%) with those obtained from using the traditional DOF. The imbalance fractions when using error are somewhat higher but still usually within 20%. This can be partially explained by the limitations of the adaptive algorithm which can only refine 2 levels i.e. one element may have no more than 16 children. However, on problems with singularties, the error in some element may be larger by 2 orders of magnitude. In this case, the error would not be a very good predictor of the mesh refinement and hence a bad measure of computational load too. Further, we note that both IF and SF are DOF based measures and hence are limited measures of partitioning effort. Again, as we show below – error based partitioning will enable us to greatly reduce data migration during repartitioning.

Repartitioning and Data Migration

In our studies here, each run includes 3 steps of partitioning corresponding to the 3 steps of the adaptive algorithm. We test the PBR and PAR strategies on four test meshes and measure number of elements that have to migrate between processors. Table 1 shows the number of elements that migrated using different partitioners for meshes 1 and 4. We observe drastic reductions in the data migration when using the PBR based strategy as opposed to the classical PAR.In phase 1 (repartitioning after the *h*-refinement stage), we observe that PBR strategy requires 93% less data migration than PAR for our test meshes. In phase 2(repartitioning after the *hp*

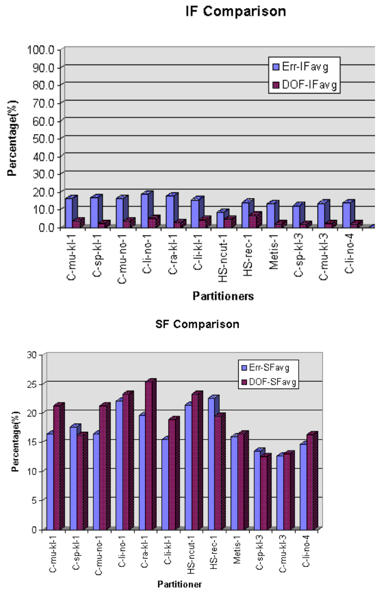


Figure 3 Average imbalance and shape fractions for different partitioning algorithms. Data averaged over 4 test meshes. Different partitioning algorithsm on abscissa.(e.g.C-mu-kl-1:Chaco, Multi-level, Kernigan-Lin on Test Mesh 1, HS-ncut-1: Hilbert space filling curve, k-way partition on Mesh 1)

–	–	PBR	PAR	PBR	PAR
Partitioner	Mesh	P1-P2	P1-P2	P2-P3	P2-P3
Ch-mu-no	1	253	345	400	1221
Ch-sp-kl	1	246	427	422	1261
Ch-ra-kl	1	210	335	284	1074
Ch-li-kl	1	218	349	291	733
Ch-mu-kl	1	253	345	400	1221
Ch-sp-no	1	240	412	432	1264
AVG	1	236.7	368.8	371.5	1129.0
Ch-mu-kl	4	570	1278	1278	2621
Ch-sp-kl	4	572	879	1263	2901
Ch-sp-no	4	576	1250	1242	2901
Ch-mu-no	4	566	1228	1278	2745
AVG	4	571.0	1156.8	1267.3	2792.0

Table 1 Comparison between PBR and PAR in Mesh 4. Rows show results for different choices of partitioning algorithms.(e.g. Ch-sp-kl: Chaco, spectral, with Kernighan-Lin)

refinement), this advantage goes up to 171% . Repartitioning data before refinement can thus greatly reduce the data migration. In conclusion, we can thus state that a) we have introduced a new measure of computational effort namely, the error in a coarser mesh, and, b) our use of the error in a corse mesh as a measure of computational effort has motivated a novel strategy whereby we repartition before we refine/enrich the mesh.

REFERENCES

[BS93] Bank R. E. and Smith R. K. (July 1993) *A Posteriori* error estimates based on hierarchical bases. Technical report, Institute for Mathematics and its Applications.

[BS94] Babuska I. and Suri M. (December 1994) The p and $h-p$ versions of the finite element method, basic principles and properties. *SIAM Review* 36(4).

[HL93] Hendrickson B. and Leland R. (1993) A multilevel algorithm for partitioning for graphs. Technical Report SAND93-1301, Sandia National Laboratories.

[JTOF97] J. T. Oden A. P. and Feng Y. S. (December 1997) Parallel domain decomposition solver for adaptive *hp* finite element methods. *SIAM Journal on Numerical Analysis* pages 2090–2118.

[KK97] Karypis G. and Kumar V. (1997) Multilevel k-way partitioning scheme for irregular graphs. *Journal of Parallel and Distributed Computing* .

[OP95] Oden J. T. and Patra A. (1995) A parallel adaptive strategy for *hp* finite element computations. *Comput. Methods. Appl. Mech. and Engg.* 121.

[PO95] Patra A. and Oden J. T. (1995) Problem decomposition for adaptive *hp* finite element methods. *Computing Systems in Engineering* 6(2): 97–109.

[PO97] Patra A. and Oden J. (1997) Computational techniques for adaptive *hp* finite element methods. *Finite Elements in Analysis and Design* 25: 27–39.

50

A FETI Solver for Corotational Nonlinear Problems

Kendall H. Pierson , Michel Lesoinne [1]

Introduction

Corotational Finite Element Methods (CFEM) are a way of approaching structural problems where geometric nonlinearities are involved due to large rotations and displacements but small deformations. Such problems are of particular importance to the aerospace industry where structures may exhibit large rotations and flexions, but all components remain within the realm of small deformations. CFEM is nonlinear by design and solution of a single problem requires solving many linear systems with an evolving tangent stiffness matrix. Large systems lead naturally to using the Finite Element Tearing and Interconnecting (FETI) method to solve them. The FETI method is a robust and efficient domain decomposition method for linear structural problems and its use for the resolution of some other classes of nonlinear problems has already been examined by several investigators [Rou95, RR98].

Specific issues to CFEM are related to an unsymmetric tangent stiffness matrix and associated null space which evolves during the nonlinear analysis. We have approached this problem by symmetrizing the tangent stiffness matrix as it was shown to not destroy convergence [Hau94]. Investigations into efficient preconditioning strategies have dealt with reuse of previous Krylov spaces and freezing the preconditioner from one outer iteration to another. Finally, we present a large scale wing type structure undergoing large rotations and flexions.

1 Department of Aerospace Engineering and Sciences and Center for Aerospace Structures University of Colorado at Boulder Boulder, CO 80309-0429, U.S.A.
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

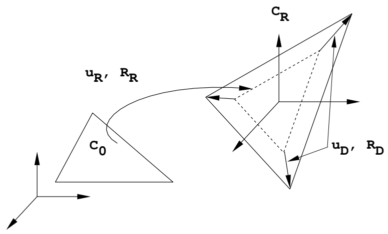


Figure 1 Base and corotated configurations

The Corotational Formulation

The corotational formulation (CR) of geometrical nonlinear structural problems separates rigid body motions from purely deformational motions. Conceptually, in a finite element framework, CR decomposes total displacements into pure rigid body motions and deformational motions at each configuration of nonlinear structural analysis. This can be expressed in vector form for translation degrees of freedom (dofs) and in rotation matrix form for rotation dofs

$$v = v_D + v_R \qquad R = R_D R_R \tag{1}$$

where v_D is deformational motion, v_R is rigid body motion, R_D is the deformational rotation matrix, and R_R is the rigid body rotation matrix. The non-additive behavior of rotation dofs when the rotation axis changes and other geometrical nonlinearities associated with large displacements leads to the nonlinear equilibrium equation

$$F_{int}(x) + F_{ext} = 0 \tag{2}$$

where x contains both v and R, $F_{int}(x)$ is the internal force vector which is dependent on the state of the structure x, and F_{ext} is the external force vector. Eq. (1) are written on an element by element basis. For each element, a reference configuration C_0 is created, and the v_R, R_R displacements express the passage from this configuration to a *corotated* configuration C_R (also called *shadow element*) [dV76]. The element corotated configuration is calculated as a rigid body motion of the element base configuration C_0. From this shadow configuration, the deformational displacements v_d, R_d are used to evaluate the elemental contribution to the internal force vector. As these deformations are small, the usual linear elemental matrices can be used to compute the forces in a frame attached to C_R. After obtaining these local forces, they are transformed back to the global frame (to which C_0 is attached) before summation (see Figure 1). Benefits of using CR over other nonlinear structural analysis description such as the Total Lagrangian method are

- Effective for large rotation/small-strain problems
- Re-use of existing small-strain finite element libraries
- Ability to decouple material nonlinearities from geometric nonlinearities

Maneuvering aircraft undergo large rotations while experiencing small deformational motions. Structural material properties remain linear while nonlinearities are geometric in nature. Adopting CR to model geometrical nonlinear behavior of maneuvering aircraft offers the extra advantage of correctly modeling control surfaces.

Overview of FETI

To keep this paper self-contained, we begin with an overview of the original FETI method [FR94, Far91, FR92]. The general problem to be solved is

$$Ku = F \tag{3}$$

where K is an an $n \times n$ symmetric positive semi-definite sparse matrix arising from the finite element discretization of a second- or fourth-order elastostatic (or elastodynamic) problem defined over a domain Ω, and F is a right hand side n-long vector representing generalized forces. Let Ω be partitioned into a set of N_s *disconnected* subdomains $\Omega^{(s)}$, then the FETI method replaces Eq. (3) with the following equivalent system of subdomain equations

$$\begin{aligned} K^{(s)}u^{(s)} &= F^{(s)} - B^{(s)^T}\lambda \qquad s = 1, \dots, N_s \\ \Delta &= \sum_{s=1}^{N_s} B^{(s)}u^{(s)} = 0 \end{aligned} \tag{4}$$

where $K^{(s)}$ and $F^{(s)}$ are the unassembled restrictions of K and F to subdomain $\Omega^{(s)}$, λ is a vector of Lagrange multipliers introduced to enforce the constraint $\Delta = 0$ on the subdomain interface boundary $\Gamma_I^{(s)}$, $u^{(s)}$ is the local solution vector, and $B^{(s)}$ is a signed Boolean matrix that describes the subdomain interconnectivity. A detailed derivation of (4) can be found in [FR94, FMR94]. An arbitrary mesh partition may contain $N_f \leq N_s$ floating subdomains — that is, subdomains lacking the necessary number of essential boundary conditions needed to prevent the subdomain matrices $K^{(s)}$ from being singular. Therefore N_f of the local Neumann problems

$$K^{(s)}u^{(s)} = F^{(s)} - B^{(s)^T}\lambda \quad s = 1, \dots, N_f \tag{5}$$

are ill-posed. Solvability is guaranteed based on the following condition

$$R^{(s)^T}(F^{(s)} - B^{(s)^T}\lambda) = 0 \qquad s = 1, \dots, N_f \tag{6}$$

where $R^{(s)}$ is the null space of $K^{(s)}$. The solution of Eq. (5) is then given by

$$u^{(s)} = K^{(s)^+}(F^{(s)} - B^{(s)^T}\lambda) + R^{(s)}\alpha^{(s)} \tag{7}$$

where $K^{(s)^+}$ is a generalized inverse of $K^{(s)}$ that need not be explicitly computed [FR92] and $\alpha^{(s)}$ is a vector of six or fewer constants. The extra unknowns $\alpha^{(s)}$ are compensated by additional equations resulting from Eqs. (6) Substituting Eq. (7) into Eq. (4) and using Eq. (6) leads to the FETI interface problem

$$\begin{bmatrix} F_I & -G_I \\ -G_I{}^T & 0 \end{bmatrix} \begin{bmatrix} \lambda \\ \alpha \end{bmatrix} = \begin{bmatrix} d \\ -e \end{bmatrix} \tag{8}$$

where

$$\begin{aligned}
F_I &= \sum_{s=1}^{N_s} B^{(s)} K^{(s)^+} B^{(s)^T}; \qquad G_I \;=\; \left[\; B^{(1)} R^{(1)} \quad \ldots \quad B^{(N_f)} B^{(N_f)} \;\right]; \\
\alpha^T &= \left[\; \alpha^{(1)^T} \quad \ldots \quad \alpha^{(N_f)^T} \;\right]; \qquad d \;=\; \sum_{s=1}^{N_s} B^{(s)} K^{(s)^+} F^{(s)}; \\
e^T &= \left[\; F^{(1)^T} B^{(1)} \quad \ldots \quad F^{(N_f)^T} B^{(N_f)} \;\right]
\end{aligned} \tag{9}$$

$$\begin{aligned}
K^{(s)^+} &= K^{(s)^{-1}} \quad \text{if } \Omega^{(s)} \text{ is not a floating subdomain} \\
K^{(s)^+} &= \text{a generalized inverse of } K^{(s)}, \text{ if } \Omega^{(s)} \text{ is a floating subdomain}
\end{aligned} \tag{10}$$

For structural mechanics and structural dynamics problems, F_I is symmetric since the subdomain stiffness matrices $K^{(s)}$ are symmetric. The objective is to solve by a Preconditioned Conjugate Gradient (PCG) algorithm the interface problem (8) instead of the original problem (3). The PCG algorithm is modified by a projection that enforces iterates λ^k satisfy Eq. (6). The projector P is defined as

$$P = I - G_I (G_I^T G_I)^{-1} G_I^T \tag{11}$$

and the FETI algorithm can be written as

$$\begin{array}{rrl}
1. & \text{Initialize} & \\
& \lambda^0 &= G_I \; ({G_I}^T G_I)^{-1} e \\
& r^0 &= d - F_I \lambda^0 \\
2. & \text{Iterate } k = 1,\; 2,\; \ldots \text{ until convergence} & \\
& w^{k-1} &= P^T \; r^{k-1} \\
& z^{k-1} &= \overline{F_I^{-1}} \bar{w}^{k-1} \\
& y^{k-1} &= P \; \bar{z}^{k-1} \\
& \zeta^k &= y^{k-1^T} w^{k-1} / y^{k-2^T} w^{k-2} \quad (\zeta^1 = 0) \\
& p^k &= y^{k-1} + \zeta^k p^{k-1} \quad (p^1 = y^0) \\
& \nu^k &= y^{k-1^T} w^{k-1} / p^{k^T} F_I p^k \\
& \lambda^k &= \lambda^{k-1} + \nu^k p^k \\
& r^k &= r^{k-1} - \nu^k F_I p^k
\end{array} \tag{12}$$

The reader can check that the FETI algorithm results in applying PCG to:

$$A\lambda = d \tag{13}$$

where $A = P^T F_I P$.

Use of FETI for Corotational Problems

We wish to solve Eq. (4) using a Newton-Raphson approach which leads us to solving the following set of successive linear systems

$$\begin{aligned}
K_T(x_0)\Delta u_1 &= f_1 \\
K_T(x_{i-1})\Delta u_i &= f_i \\
K_T(x_{n-1})\Delta u_n &= f_n
\end{aligned} \tag{14}$$

where $K_T(x_{i-1})$ is the i^{th} configurations tangent stiffness matrix, x_{i-1} is the previous structural state, Δu_i is the i^{th} incremental displacement vector, f_i is the i^{th} right hand side vector which depends on the solution strategy [Rik72], and n is the number of iterations for convergence. After each linear solve, x_i is updated using the incremental displacement. Convergence of Newton-Raphson is based on reaching these criteria

$$||\Delta u||_2 < tolerance_{disp} \qquad ||f||_2 < tolerance_{residual} \tag{15}$$

The tangent stiffness matrices are generally unsymmetric before convergence is reached. In this work we used a symmetrized version of the tangent matrices, as it is required by the FETI method and has been shown not to harm convergence [Hau94].

Solution of Nearby Linear Systems

Multiple Left Hand Sides (MLHS)

We will refer to the method described in the previous paragraph as the Full Newton (FN) method. The tangent stiffness matrix is rebuilt at each nonlinear iteration, resulting in a robust geometrical nonlinear structural solution algorithm.

The use of FETI to solve Eqs. (14) creates a set of systems similar to Eq. (13). Solving each system by PCG can be viewed as a minimization problem of the form:

$$\min_{\lambda}(\Phi_i) = 1/2\lambda^T A_i \lambda - d_i^T \lambda \qquad i = 1, \ldots, n \tag{16}$$

To solve each system, a Krylov space is generated by the PCG algorithm. This space is spanned by the set of search directions. Let us denote the set of search directions for the linear system i by

$$S_i = \{s_i^1, s_i^2, \ldots, s_i^k, \ldots, s_i^{r_i}\} \qquad i = 1, \ldots, n \tag{17}$$

The search directions computed within a Newton iteration are orthogonal with respect to A_i. However, since the tangent stiffness matrix is changing over successive Newton iterations, the search directions are not orthogonal across Newton iterations. Nonetheless, the successive A_i matrices can be expected to be spectrally close one to another. This observation suggests that at step i we can make use of the previous sets S_i to define a preconditioner [Rou95]. The approach is to replace the usual preconditioned residual vector Pr by a modified vector $\hat{Pr}$ given by:

$$\hat{Pr} = Pr + \sum_{j=1}^{i-1} S_j y_j \tag{18}$$

where the vectors y_j are successively chosen to satisfy the minimization problem

$$\min_{y_j}(\Phi_j) = 1/2\lambda^T A_j \lambda - r^T \lambda \tag{19}$$

with $\lambda = Pr + \sum_{l=1}^{j} S_l y_l$. Note that the function being minimized in this case is similar to that given in Eq. (16) when A_i is replaced with A_j and d_i is replaced by the current

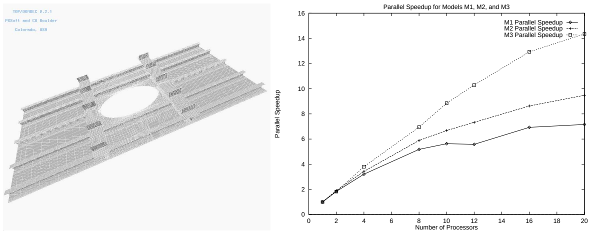


Figure 2 V-22 Wing Panel, Parallel Speedup for M1, M2 and M3

PCG residual r. All computations performed, we obtain the expression for y_j:

$$y_j = (S_j^T A_j S_j)^{-1}(S_j^T r - S_j^T A_j(Pr + \sum_{l=1}^{j-1} S_l y_l)) \qquad j = 1, \ldots, i-1 \tag{20}$$

The assumption is that successive tangent stiffness matrices are *similar* in some sense and thus the preconditioner will be of benefit. Due to the corotational formulation, the null space of the subdomain tangent matrices $K_T^{(s)}$ evolves in time. This implies that the projector must be correctly recomputed at each iteration and the space of search directions must contain projected vectors.

Multiple Right Hand Sides (MRHS)

An alternative approach to solve Eq. (14) is to use a Newton Like (NL) Method where the tangent stiffness matrix may be "frozen" and reused for subsequent nonlinear iterations. In the extreme case, the initial tangent stiffness matrix may be used for the entire simulation but then convergence of the outer Newton iterations is not guaranteed. The effect of using a NL method versus a FN method is to slow down convergence and possibly increase the number of linear solutions.

In contrast to the FN method, we now solve with a fixed A_f matrix for a set of varying right hand sides. Therefore, as has been done for linear dynamic problems [Far95], the set of search directions S_i can be kept A_f orthogonal to all previous sets. The initial λ_i^0 must now be modified to include the contribution of all previous sets and then orthogonality of the search directions with respect to A_f enforced at each FETI iteration.

We can of course combine the MRHS approach with the Krylov Space preconditioning of MLHS. In this case, the matrix A_f is changed every m iteration, and we can apply the preconditioning of Eqs. (18) in which case the sets S_i are the union of all sets generated by the same tangent matrix A_f.

Numerical Example: Composite Wing Panel

The composite wing panel from a V-22 tilt-rotor aircraft [Dav91] considered here contains design features such as ply drop-offs, ply interleaves, axial stiffeners, transverse ribs, clips, brackets and a large elliptical access hole. The panel is clamped at one end and loaded on the opposite end by prescribed displacements. The finite element models, designated M1, M2, and M3 have 56916, 223620, and 885924 d.o.f. respectively (see Figure 2). Each model is composed of 47 composite materials and discretized with three node ANS shell elements.

The results in Table 1 were generated with the 2-Level FETI Method [FM98] for fourth order elastostatic problems on a SGI Onyx 2000 using 16 processors. FETI convergence was monitored using a tolerance of 1.0E-3 while Newton's convergence was checked using 1.0E-5. These values were chosen due to quadratic convergence properties of Newton's method. The optimal Dirichlet FETI preconditioner was selected for this problem. Using a NL method, this problem requires two load steps and eight Newton iterations while FN requires one load step and four iterations to converge. Both methods were tested with and without Krylov acceleration methods to compare iteration counts, CPU, and memory requirements. For FN, K_T, and FETI preconditioner F_D^{-1} are rebuilt at each Newton iteration while for NL method, rebuilding occurs at the start of a load step. The most evident conclusion is the

Table 1 Composite Panel Model, M3 (885924 d.o.f., 250 Subdomains)

Load Step	Newton Itr.	NL-KLR	NL-KR	FN-KL	FN
1	1	143 itr.	143 itr.	143 itr.	143 itr.
	2	72 itr.	72 itr.	171 itr.	198 itr.
	3	36 itr.	36 itr.	108 itr.	198 itr.
	4	19 itr.	19 itr.	46 itr.	198 itr.
2	5	124 itr.	168 itr.	- itr.	- itr.
	6	57 itr.	64 itr.	- itr.	- itr.
	7	36 itr.	33 itr.	- itr.	- itr.
	8	16 itr.	14 itr.	- itr.	- itr.
	Total	503 itr.	549 itr.	**468 itr.**	737 itr.
	FETI CPU	**940 sec.**	1078 sec.	979 sec.	1513 sec.
	FETI Mem.	3710 Mb.	3417 Mb.	3657 Mb.	**3307 Mb.**

approximately 40% decrease in CPU when a Krylov acceleration technique is applied. The other conclusion is a small 10% increase in memory requirements associated with the Krylov acceleration methods over the FN method. This can be attributed to the number of stored search direction vectors for each algorithm. The FN with Krylov left hand side acceleration (KL) had the lowest total iteration count and was roughly the same CPU time as NL with Krylov right hand side acceleration (KR). When both Krylov accelerators (KLR) are applied to the NL method, there is a slight improvement in total iteration count and CPU time. Figure 2 shows almost linear parallel speedup for each model. This trend suggests that larger problems will gain greater speedups with increasing numbers of processors. Optimal mesh decompositions of 40 (M1), 140 (M2), and 250 (M3) subdomains were used to compute these speedups [LP98].

Conclusion

We have shown the FETI method in conjunction with CFEM concept to be efficient in solving large scale geometrical nonlinear problems undergoing large rotations and small deformations. Future research will involve implementing the FETI solver and CFEM in a nonlinear dynamic algorithm to solve time-dependent geometrical nonlinear problems such as maneuvering aircraft.

REFERENCES

[Dav91] Davis D. D. (1991) Detailed analysis and test correlation of a stiffened composite wing panel. Technical report and memo, U.S. Army, Aerostructures Directorate, NASA Langley Research Center, Hampton VA.

[dV76] de Veubeke F. (1976) The dynamics of flexible bodies. *Int. Jnl. Engrg. Sci* 1: 895–913.

[Far91] Farhat C. (1991) A lagrange multiplier based divide and conquer finite element algorithm. *J. Comput. Sys. Engrg.* 2: 149–156.

[Far95] Farhat C. (1995) Optimizing substructuring methods for multiple right hand sides. *SIAM, Domain-Based Parallelism and Problem Decomposition Methods in Computational Science and Engineering* 1: 141–160.

[FM98] Farhat C. and Mandel J. (1998) The two-level feti method for static and dynamic plate problems - part i: an optimal iterative solver for biharmonic systems. *Computer Methods in Applied Mechanics and Engineering* 155: 129–152.

[FMR94] Farhat C., Mandel J., and Roux F. (1994) Optimal convergence properties of the feti domain decomposition method. *Comput. Meths. Appl. Mech. Engrg.* 115: 367–388.

[FR92] Farhat C. and Roux F. (1992) An unconventional domain decomposition method for an efficient parallel solution of large-scale finite element systems. *SIAM J. Sc. Stat. Comput.* 13: 379–396.

[FR94] Farhat C. and Roux F. (1994) Implicit parallel processing in structural mechanics. *Computational Mechanics Advances* 2: 1–124.

[Hau94] Haugen B. (1994) *Buckling and Stability Problems for Thin Shell Structures using High-Performance Finite Elements.* PhD dissertation, University of Colorado, Boulder, CO.

[LP98] Lesoinne M. and Pierson K. (1998) An efficient feti implementation on distributed shared memory machines with independent numbers of subdomains and processors. *Contemporary Mathematics* 218: 318–324.

[Rik72] Riks E. (1972) The application of newton's method to the problem of elastic stability. *Journal of Applied Mechanics* 39: 1060–1066.

[Rou95] Roux F. (1995) Parallel implementation of a domain decomposition method for nonlinear elasticity problems. *SIAM, Domain-Based Parallelism and Problem Decomposition Methods in Computational Science and Engineering* 1: 161–175.

[RR98] Risler F. and Rey C. (1998) On the reuse of ritz vectors for the solution of nonlinear elasticity problems by domain decomposition methods. *Contemporary Mathematics* 218: 334–340.

51

Parallel Solution of General Sparse Linear Systems using PSPARSLIB

Yousef Saad[1], Sergey Kuznetsov[2], Gen-Ching Lo[3]

Distributed Sparse Linear Systems

PSPARSLIB [SM95] is a portable package for solving sparse linear systems on parallel platforms. The package adapts a number of methods based on domain decomposition concepts to general sparse irregularly structured linear systems. It is implemented in FORTRAN 77 with a few C functions and uses the MPI communication library for message passing. This paper describes some of the methods implemented in the package and examines a number of strategies used to improve their performance.

Given a large sparse nonsymmetric real matrix A of size n, we consider the linear system

$$Ax = b, \tag{1}$$

that can arise, for example, from a finite element discretization of a partial differential equation. To solve this system on a distributed memory computer, it is common to partition the finite element mesh and assign a cluster of elements representing a physical subdomain to each processor. Each processor assembles only the local equations associated with the elements assigned to it. When the system is already in

1 University of Minnesota, Department of Computer Science and Engineering, Minneapolis, MN. Work supported in part by ARPA under grant NIST 60NANB2D1272 and in part by NSF under grant CCR-9618827. Additional computational resources were provided by the Minnesota Supercomputer Institute and by the University of Minnesota IBM Shared Project.
2 Institute of Mathematics, Novosibirsk, Russia. This work was carried out while the author was on leave at the University of Minnesota, Department of Computer Science and Engineering, Minneapolis, MN.
3 Morgan-Stanley, New-York
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

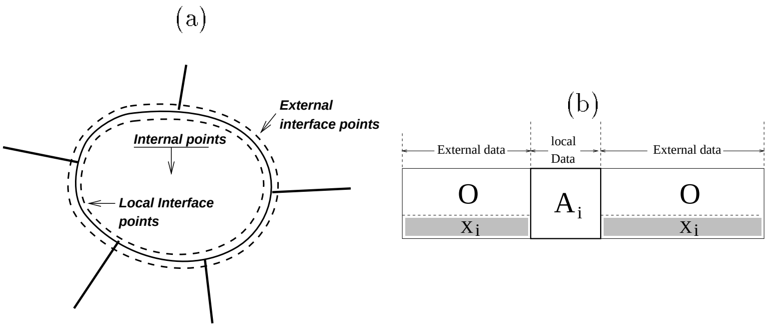


Figure 1 A local view of a distributed sparse matrix.

assembled form, subsets of equations-unknowns are assigned to processors. In either case, each processor will wind up with a set of equations (rows of the linear system) and a vector of the variables associated with these rows. A distributed linear system is a set of linear equations that are assigned in this fashion to a distributed memory computer.

This paper presents the general principles used in a parallel iterative sparse linear system solver with an emphasis on defining preconditioners for distributed sparse linear systems. These (global) linear systems of equations are regarded as distributed objects and solution methods must be developed by exploiting the local data structures associated with the distributed linear system. The first step of any parallel sparse iterative solver is to set up these local data structures. In this preprocessing phase, the local equations are represented along with the dependencies between the local variables and external variables, and any other information needed during the iteration phase. We will give a brief overview of the local representation of the linear system and discuss how the solution package is built around this representation. A number of variations of some known preconditioning techniques will then be presented.

The Local System.

Figure 1-(a) shows a 'physical domain' viewpoint of a sparse linear system which is akin to that used in the domain decomposition literature. A point (node) in a 'subdomain' is actually a pair representing an equation along with an associated unknown. It is common to distinguish between three types of unknowns: (1) Interior variables which are coupled only with local variables by the equations; (2) Local interface variables which are those coupled with non-local (external) variables as well as local variables; and (3) External interface variables which are variables in other processors that are coupled with local variables. We can also represent the related local equations as shown in Figure 1-(b). Note that these equations are not contiguous in the original system. The matrix represented in the figure can be viewed as a reordered version of the equations associated with a local numbering of the equations/unknowns pairs.

As can be seen in Figure 1-(b), the rows of the matrix assigned to a certain processor have been split into two parts: a *local* matrix A_i which acts on the local variables and an *interface* matrix X_i which acts on remote variables. These remote variables must be received from other processor(s) each time that a matrix-vector product with the matrix A is performed. It is common to list the interface nodes last, after the interior nodes. This local ordering of the data leads to efficient interprocessor communication, and reduces local indirect addressing during matrix-vector products. The zero blocks shown are due to the fact that local internal nodes are not coupled with external nodes.

Each local vector of unknowns x_i consists of the subvector u_i of internal nodes followed by the subvector y_i of local interface variables. The right-hand side b_i is conformally split into the subvectors f_i and g_i, and the local matrix A_i residing in processor i as defined above is block-partitioned according to this splitting. The local equations can therefore be written as follows.

$$\begin{pmatrix} B_i & E_i \\ F_i & C_i \end{pmatrix} \begin{pmatrix} u_i \\ y_i \end{pmatrix} + \begin{pmatrix} 0 \\ \sum_{j \in N_i} E_{ij} y_j \end{pmatrix} = \begin{pmatrix} f_i \\ g_i \end{pmatrix} \tag{2}$$

The block matrix on the left side is the local matrix A_i. Here, N_i is the set of subdomains that are neighbors to subdomain i. Each term $E_{ij} y_j$ in the sum in the left-hand side is the contribution to the local equation from neighboring subdomain number j.

The subvectors of external interface variables are grouped into one vector called $y_{i,ext}$ and the notation

$$\sum_{j \in N_i} E_{ij} y_j \equiv X_i y_{i,ext}$$

will be used to denote the contributions from external variables to the local system (2). In effect, this represents a local ordering of the external variables. With this notation, the left-hand side of (2) becomes

$$w_i = A_i x_i + X_i y_{i,ext} \tag{3}$$

The vector w_i is the local part the matrix-by vector product Ax in which x is a vector which has the local vector components x_i, $i = 1, \ldots, s$.

To facilitate matrix operations and communication, an important task is to gather the data structure representing the local part of the linear matrix as was just described. In this preprocessing phase it is also important to form any additional data structures required to prepare for the intensive communication that will take place during the solution phase. In particular, each processor needs to know (1) the processors with which it must communicate, (2) the list of interface points and (3) a break-up of this list into pieces of data that must be sent and received to/from the "neighboring processors". A complete description of the data structure associated with this boundary information is given in [SM95] along with additional implementation details.

Matrix-vector operations.

The matrix-vector product is carried out according to equation (3). First, the external data $y_{i,ext}$ needed in each processor is obtained. The matrix-vector product with the matrix A_i on the local data x_i can be carried out at the same time that this communication step is being performed. Then the matrix-vector product with the matrix X_i on the external data $y_{i,ext}$ can be carried out and the result is added to the result obtained from $A_i x_i$. The following is a code segment for performing this operation as it is implemented in PSPARSLIB.

```
c    send local interface data to neighbors
     call MSG_bdx_send(nloc,x,y,nproc,proc,ix,ipr,ptrn,ierr)
c    do local matrix-vector product for local points
     call amux(nloc,x,y,aloc,jaloc,ialoc)
c    receive remote interface date from neihgbors
     call MSG_bdx_receive(nloc,x,y,nproc,proc,ix,ipr,ptrn,ierr)
c    peform matrix-vector product with external data and add it
c    to current result vector y
     nrow = nloc - nbnd + 1
     call amux1(nrow,x,y(nbnd),aloc,jaloc,ialoc(nloc+1))
    return
```

Here `MSG_send` and `MSG_receive` are subroutines that invoke the proper MPI calls for sending and receiving the interface data to and from neighboring processors. The send is non-blocking. The set of parameters `nproc`, `proc`, `ix`, `ipr`, `ptrn`, contains precisely the data structure needed for the interface variables.

Distributed Krylov Subspace Solvers

The main operations in a standard Krylov subspace acceleration are (1) vector updates, (2) dot-products, (3) matrix-vector products and (4) preconditioning operations. If we exclude the matrix-vector products (discussed above) and preconditioning steps (to be seen later), the rest of the operations in an algorithm such as CG or GMRES are mainly dot-products and vector updates. Since vector quantities are split in the same fashion, a global SAXPY of two vectors across p processors, consists of p independent SAXPYs. In contrast, a global dot product requires a global sum of the separate dot products of the subvectors in each processor. The dot products are mainly used in orthogonalizing sets of Krylov vectors and this constitutes one of the potential bottlenecks in a parallel implementation of Krylov subspace procedures. Experience shows however, that in most practical situations the loss of efficiency due to inner products is usually minor unless the sub-problems become relatively small[KLS97].

One of the main Krylov accelerator used in PSPARSLIB is the flexible variant of GMRES [SS86] known as FGMRES [Saa93]. This is a right-preconditioned variant that allows the preconditioning to vary at each step. Since the preconditioning operations require solving systems associated with entire subdomains it becomes important to allow the preconditioner itself to be an iterative solver. This means that the GMRES iteration should allow the preconditioner to vary from step to step within the inner GMRES process. A variant of GMRES which allows this is called the flexible variant of GMRES (FGMRES), see [Saa93] for details.

It is important to implement the accelerators (e.g. FGMRES) with “reverse communication”, a mechanism whose goal is to avoid passing data structures to the accelerator. When calling a standard FORTRAN subroutine implementation of an

D_1	$\vdots$	D_2	D_3	$\vdots$	D_4
$\ldots$	S_1	$\ldots$	$\ldots$	S_2	$\ldots$
D_5	$\vdots$	D_6	D_7	$\vdots$	D_8

Table 1 An example of splitting eight domains on the plane into two groups

iterative solver, we normally need to pass a list of arguments related to the matrix A and to the preconditioner. This can be a burden on the programmer because of the rich variety of existing data structures. The solution is not to pass the matrices in any form. When a matrix – vector product or a preconditioning operation is needed, the subroutine exits and the calling routine performs the desired operation and then calls the subroutine again, after placing the desired result in one of the vector arguments of the subroutine. For details, see [SW95, Saa95, Saa96].

Variants of Additive Schwarz Preconditioners.

The additive Schwarz procedure is a form of block Jacobi iteration, in which the blocks refer to systems associated with entire domains. Each iteration of the process consists of the following steps: (1) Obtain external data $y_{i,ext}$; (2) Compute (update) local residual $r_i = (b - Ax)_i = b_i - A_i x_i - X_i y_{i,ext}$ (3) Solve $A_i \delta_i = r_i$; and (4) Update solution $x_i = x_i + \delta_i$. To solve the systems in step 3, a standard (sequential) ILUT preconditioner [Saa96] combined with GMRES or one step of an ILU preconditioner is used. Of particular interest in this context are the overlapping additive Schwarz methods. In the domain decomposition literature, see e.g., [Bjø89, BW89, SBG96] among others it is known that overlapping is a good strategy to reduce the number of steps. There are however several different ways of implementing overlapping block Jacobi iterations, for example, we can replace the data in the overlapping subregions by its external version or use some average of the data.

It is sometimes possible to reduce the number of outer iterations required by block Jacobi preconditioners by using a 2-level clustering of the subdomains. The subdomains are split into groups $S_1, S_2, \ldots, S_p$. The assignment of the subdomains to groups can be determined from knowledge of neighboring subdomains. Figure 1 shows an example in which 8 subdomains are split into two groups. Consider a certain group of subdomains, say S_k, and the set of equations-unknowns corresponding to all the subdomains belonging to S_k. The local system for subdomain $i \in S_k$ can be rewritten in the form:

$$A_i x_i + Z_i z_{i,ext} + (X_i y_{i,ext} - Z_i z_{i,ext}) = w_i$$

where $z_{i,ext}$ is a part of the vector $y_{i,ext}$ corresponding to remote variables in the same group as subdomain i, Z_i is a part of the interface matrix X_i which acts on the remote variables $z_{i,ext}$. Thus the solution of the linear system

$$A_i x_i + Z_i z_{i,ext} = w_i$$

will provide a preconditioner for the (outer) block Jacobi iteration. This nested two-level form of the block Jacobi iteration, can be described as follows.

ALGORITHM **1** *Hierarchical Block Jacobi Iteration*
1. *Obtain external data* $y_{i,ext}$
2. *Compute (update) local residual* $r_i = (b - Ax)_i = b_i - A_i x_i - X_i y_{i,ext}$
3. *Do* $i = 1, \ldots, i_1$
4. *Obtain external data* $z_{i,ext}$
5. *Compute (update) local residual* $s_i = r_i - Z_i z_{i,ext}$
6. *Solve* $A_i \delta_i = s_i$
7. *End*
8. *Update solution* $x_i = x_i + \delta_i$

The number of inner iteration i_1 depends on the problem. Often, the choice $i_1 = 2$ achieves a good compromise between accuracy for the local block solver and overall performance. The hierarchical block Jacobi iteration presents several advantages over the traditional block Jacobi iteration. It allows to reduce the number of outer iterations due to increased accuracy in the preconditioning step. On the other hand, this approach increases communication costs which can be reduced by using small groups.

Variants of Multiplicative Schwarz Preconditioners.

The multiplicative Schwarz preconditioner uses the same extended domains as the additive Schwarz method, but the subdomain solves are sequential: every processor uses interface variables defined by preceding local solves. The simplest form of the multiplicative Schwarz is the block Gauss-Seidel algorithm used in domain decomposition techniques [BW86, SBG96, CM94].

A partial remedy to the sequential nature of the multiplicative Schwarz technique is to use multi-coloring. Thus, if the domains are colored, the multiplicative Schwarz as executed in each processor would involve a color loop through all the colors - and execution of the loop is performed only when the color variable equals the color of the processor, see [SBG96, Saa96]. The rest of the iteration is identical with that of Additive Schwarz. A problem with multicoloring is that as the domain associated with the given color is active, all other colors will be inactive. As a result it is typical to obtain only $1/numcol$ efficiency where numcol is the number of colors. To reduce this effect, one can further block the local variables into two blocks: interior and interface variables. Then the global SOR iteration is performed with this additional blocking. In effect, each local matrix A_i is split as

$$A_i = \begin{pmatrix} B_i & E_i \\ F_i & C_i \end{pmatrix} = \begin{pmatrix} B_i & 0 \\ 0 & C_i \end{pmatrix} + \begin{pmatrix} 0 & E_i \\ F_i & 0 \end{pmatrix} \tag{4}$$

where the B_i part corresponds to internal nodes. The resulting segregated multiplicative Schwarz is as follows,

ALGORITHM **2** *Segregated multiplicative Schwarz*
1. *Solve* $B_i \delta_{i,x} = r_{i,x}$
2. $x_i := x_i + \delta_{i,x}$
3. *Do* $col = 1, \ldots, numcols$
4. *If* $(col.eq.mycol)$ *then*

5. *Obtain external data* $y_{i,ext}$
6. *Update y-part of residual* $r_{i,y}$
7. *Solve* $C_i \delta_{i,y} = r_{i,y}$
8. *Update interface unknowns* $y_i = y_i + \delta_{i,y}$
9. *EndIf*

The advantage of this procedure is that the bulk of the computational work in each domain is done in parallel. Loss of parallelism comes from the color loop which involves only solves with interface variables, which are of lower complexity.

Numerical Experiments

In this section, we report on some results obtained when solving distributed sparse linear systems on an IBM SP2 with 14 nodes, an IBM cluster of 8 workstations, and an SGI Challenge cluster and a 64 processor CRAY-T3E. The SGI challenge workstation cluster consists of three 4-processor Challenge L servers and one 8-processor Challenge XL server. The processors on the Challenge L and XL are the same (R10,000) but the memory sizes are different. Communication between different SGI cluster workstations (respectively, IBM RS/6000 workstations) can be performed via a HiPPI (High Performance Parallel Interface) switch or a Fibre-Channel switch. On the IBM cluster, an ATM switch is used. The MPI communication library is used for all communication calls. The ATM and HiPPI are high speed interfaces and can transfer data at 155 Mbps and 800 Mbps, respectively. The IBM RS/6000 Model 590 workstations are based on the Power2 architecture. The SP2 nodes communicate with an internal switch which achieves a bandwidth of 320 Mbps bandwidth and has a latency of about 40 microseconds. The CRAY-T3E has 64 nodes and 128 Megabytes of memory per node. The processors are connected in a 3-D torus by a network capable of 480 Mbps in each direction for each link.

Experiments with block Jacobi preconditioning. First, we compare the three different overlapping options mentioned at the end of Section 51, when they are used in conjunction with the block Jacobi iteration using. The results are summarized in Figure 2. In the figures, *jacno* stands for block Jacobi with no overlapping, *jaco_av* for block Jacobi with overlapping and averaging of the overlapping data, *jac* for block Jacobi with overlapping and exchange of overlapped data.

A standard ILUT preconditioner combined with GMRES was used as a local solver and the number of inner iterations was equal to $its = 11$. We select $its = 11$ because it minimizes the execution time for a small number of processors if preconditioned GMRES is used as a local solver. Note that the optimal set of parameters depends on the number of processors and the problem considered. It is sometimes reasonable to use a smaller number for the level of fill or a smaller number of inner iterations if the number of processors is large. The results are for the matrix VENKAT01 of dimension 62,424 with 1,717,792 non-zero entries on the CRAY-T3E, using a relative tolerance of $\varepsilon = 10^{-6}$, a Krylov subspace dimension of $m = 50$ and a fill-in of 25 for the ILUT preconditioner. The comparison shows that overlapping can reduce the number of outer iterations and *jac* usually requires fewer iterations.

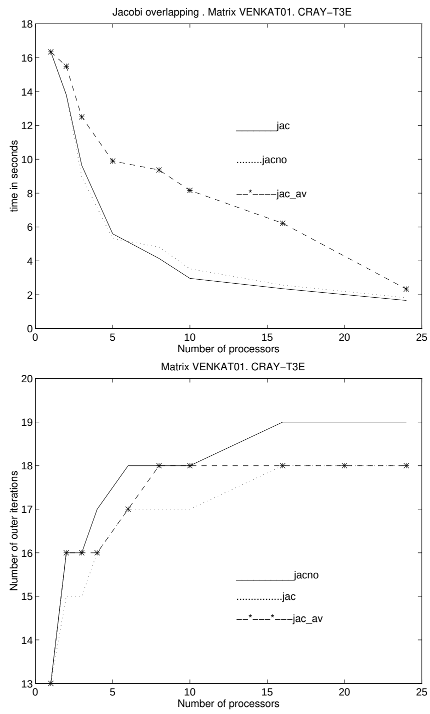


Figure 2 Comparison of FGMRES with distributed block Jacobi preconditioner for three different overlapping strategies on the CRAY-T3E. Left plot: execution times. Right plot: Iterations

Number of processors		1	3	5	10	16	24
preconditioned GMRES	time	16.34	9.64	5.59	2.96	2.35	1.66
	speed-up		1.69	2.92	5.52	6.95	9.84
	its	13	16	17	18	19	19
ILU solver	time	8.85	3.78	2.49	1.39	1.04	0.77
	speed-up		2.34	3.55	6.37	8.51	11.49
	its	15	18	20	21	21	21

Table 2 Execution times in seconds, speed-up and number of outer iterations for the block Jacobi preconditioner with two different local solvers

PEs	8	10	16	20
Matvecs	399	493	589	727
Seconds	308.21	254.38	170.85	165.66

Table 3 Number of matrix-vector multiplications and execution times (seconds) for the BARTH1S matrix.

The ILUT factorization is often more economical. Table 2 gives timing results, speed-ups and iteration counts for the block Jacobi preconditioner with overlapping using two different local solvers. Table *its* shows the number of FGMRES steps. These results are given for the VENKAT01 matrix on the CRAY-T3E, using a relative tolerance of $\varepsilon = 10^{-6}$, a Krylov subspace dimension of $m = 50$ and a level of fill of 25 for the ILUT preconditioner. These tests and others, indicate that using ILUT as a local solver (without any acceleration) is faster than using preconditioned GMRES preconditioned with ILUT.

Several approaches can be used to improve the load balance among processors. One approach is to adapt the number of inner iterations for each processor based on the time consumed by the preconditioning step for a few first iterations. Numerical tests on the SP2 and on the IBM RS6000 show that this 'forced load balancing' approach can yield a 10% to 25 % reduction in the execution time when the number of processors is small, see [KLS97] for details.

The next matrix, referred to as BARTH1S, was supplied by T. Barth of NASA Ames, see [CSW96]. It models a 2D high Reynolds number airfoil problem, with turbulence. The matrix has a 5x5 block structure and has 189,370 rows and 6,260,236 nonzero entries. This linear system is very ill-conditioned (see [CSW96]) and very hard to solve iteratively. We had to use a deflated version of GMRES [CS97]. The local systems are solved using one step of an ILU solve, where the LU factors are obtained from a block ILU(k) preconditioner (see [CSW96]) where k denotes a level of fill. Table 3 gives the timing results, and the number of matrix-vector multiplications. The following parameters have been used: the level of fill was 4, the number of deflated eigenvectors was 8, the Krylov subspace dimension was 50, and the tolerance threshold was 1.E-5.

Method	its = 1	its = 2	its = 3	its = 5	ILU sol
Matvecs	238	108	127	106	238
Seconds	15.23	9.88	15.08	18.02	15.147

Table 4 Performance of the hierarchical Jacobi preconditioner with different number of inner steps and for the ILU solve.

Experiments with hierarchical block Jacobi. We now compare the overlapping Jacobi preconditioner with the forward-backward LU solver and the hierarchical Jacobi preconditioner for 2-D elliptic problems. It is assumed that all processors have been split into groups by means of the following function $f_b(j)$ defined by: $f_b(j) = \lfloor (j-1)/b \rfloor + 1$, where b is a blocking factor for processors. Table 4 shows performance results for the hierarchical block Jacobi preconditioner using four different numbers of inner iterations. The results are for a Poisson equation on the unit square. The number of processors is 8, the problem size is 90,000. A standard forward-backward local ILU solver was used. The blocking factor for processors is 2, the tolerance is 10^{-6}, the level of fill is 5, and the Krylov subspace dimension is 30.

An increase in the number of inner iterations leads to a significant reduction in the number of outer iterations. However, the execution time increases as the number of inner iterations increases from 2 to 5 due to higher computational costs of each step. The number of inner iterations of 2 seems to give the best compromise in this case.

Experiments with Multiplicative Schwarz. Figure 3 shows a comparison of multiplicative Schwarz for four different types of local solution strategies on the CRAY-T3E for the VENKAT01 matrix. As was already explained, the Multicolor SOR scheme used here involves substantial idle time for each processor since only one color is active at any given time. It is therefore much less efficient than the other methods. The figure shows once again that requiring more accuracy for the ILU local solves leads to a reduction in the total time. Also the use of a single ILU solve leads to good savings in computational time.

REFERENCES

[Bjø89] Bjørstad P. E. (1989) Multiplicative and Additive Schwarz Methods: Convergence in the 2 domain case. In Chan T., Glowinski R., Périaux J., and Widlund O. (eds) *Domain Decomposition Methods*. SIAM, Philadelphia, PA.

[BW86] Bjørstad P. E. and Widlund O. B. (1986) Iterative methods for the solution of elliptic problems on regions partitioned into substructures. *SIAM Journal on Numerical Analysis* 23(6): 1093–1120.

[BW89] Bjørstad P. E. and Widlund O. B. (1989) To overlap or not to overlap: A note on a domain decomposition method for elliptic problems. *SIAM Journal on Scientific and Statistical Computing* 10(5): 1053–1061.

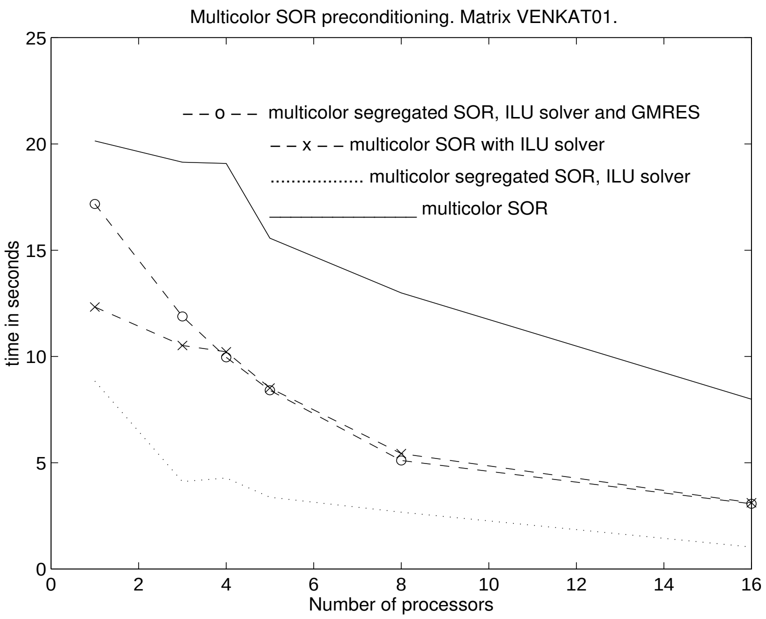


Figure 3 Comparison of 4 different multiplicative Schwarz algorithms.

[CM94] Chan T. F. and Mathew T. P. (1994) Domain decomposition algorithms. *Acta Numerica* pages 61–143.

[CS97] Chapman A. and Saad Y. (1997) Deflated and augmented Krylov subspace techniques. *Numerical Linear Algebra with Applications* 4: 43–66.

[CSW96] Chapman A., Saad Y., and Wigton L. (1996) High-order ILU preconditioners for CFD problems. Technical Report UMSI 96/14, Minnesota Supercomputer Institute.

[KLS97] Kuznetsov S., Lo G. C., and Saad Y. (1997) Parallel solution of general sparse linear systems. Technical Report UMSI 97/98, Minnesota Supercomputer Institute, University of Minnesota, Minneapolis, MN.

[Saa93] Saad Y. (1993) A flexible inner-outer preconditioned GMRES algorithm. *SIAM Journal on Scientific and Statistical Computing* 14: 461–469.

[Saa95] Saad Y. (1995) Krylov subspace methods in distributed computing environments. In Hafez M. (ed) *State of the Art in CFD*, pages 741–755.

[Saa96] Saad Y. (1996) *Iterative Methods for Sparse Linear Systems.* PWS publishing, New York.

[SBG96] Smith B., Bjørstad P., and Gropp W. (1996) *Domain decomposition: Parallel multilevel methods for elliptic partial differential equations.* Cambridge University Press, New-York, NY.

[SM95] Saad Y. and Malevsky A. (1995) PSPARSLIB: A portable library of distributed memory sparse iterative solvers. In et al. V. E. M. (ed) *Proceedings of Parallel Computing Technologies (PaCT-95), 3-rd international conference, St. Petersburg, Russia, Sept. 1995.*

[SS86] Saad Y. and Schultz M. H. (1986) GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing* 7: 856–869.

[SW95] Saad Y. and Wu K. (1995) Design of an iterative solution module for a parallel sparse matrix library (P_SPARSLIB). In Schonauer W. (ed)

Proceedings of IMACS conference, Georgia, 1994.

52

Parallel Solvers for the Two-Group Neutron Diffusion Equations of Reactor Kinetics

ROBERT SCHEICHL[1]

INTRODUCTION

We are concerned with the solution of the transient two-group neutron diffusion equations arising in the simulation process of nuclear reactor cores [DH76]:

$$\begin{aligned} \frac{\partial \phi_g}{\partial t} &= \vec{\nabla} \cdot (D_g \vec{\nabla} \phi_g) + f_g\left(\phi_1, \phi_2, (C_i)_{i=1,\ldots,6}\right), \qquad g = 1,2 & (1)\\ \frac{\partial C_i}{\partial t} &= h_i(\phi_1, \phi_2, C_i), \qquad i = 1, ..., 6 & (2) \end{aligned}$$

The unknowns in this system are the flux densities ϕ_1 and ϕ_2 of the "fast" ($g = 1$) and "thermic" ($g = 2$) neutrons, as well as the precursor concentrations C_i in each precursor group $i = 1, ..., 6$. The linear operators f_g and h_i, and the diffusion coefficients D_g are determined by the underlying physics. We consider this system on a polyhedral domain $\Omega \subset R^3$, with given initial data at $t = t_0$ and the following vacuum boundary condition on $\partial\Omega$:

$$\left(\frac{\partial \phi_g}{\partial \vec{n}}\right)_{in} = \vec{0}, \qquad g = 1, 2. \tag{3}$$

In this paper we investigate the excellent parallel performance of preconditioned Bi-CGStab and Multi-Grid applied to this system of equations. Other than reported

1 Dept. Math. Sci., University of Bath, UK. Email: R.Scheichl@maths.bath.ac.uk.
This research was supported by the Siemens AG, and a Leonardo da Vinci scholarship.
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

elsewhere in the literature (e.g. [FB$^+$88, HP$^+$96, HDB96, UTK95, YN93]), we are able to achieve optimal (linear) speedup for all tested examples up to 12 processors on all but the coarsest grid. Moreover, the parallel execution times are better than the parallel execution times of the established reactor simulation code PANBOX of the Siemens AG Operating Division KWU, described in [FB$^+$88] and [HP$^+$96]. This is of practical interest, especially since the multilevel method implemented in PANBOX is one of the most efficient methods for the sequential case to date.

We use a special finite volume method (NEM-M0), and a method that combines Crank-Nicholson and the BDF(2)-method to discretise Equations (1-3) in space and time, respectively. The parallelisation is based on a simple grid partitioning. The most important features of our methods are firstly the spatial discretisation that assigns each degree of freedom to exactly one subdomain and therefore needs no assembling at the subdomain interfaces, and secondly an effective parallel hybrid Block-Jacobi - Block-SOR smoother / preconditioner that makes use of the particular structure of the linear equation systems. The implementation of the methods was carried out on a shared memory 12-processor machine (SGI Power Challenge) using "threads". For more details about the methods and the implementation, and for more numerical results see [Sch97].

DISCRETISATION

On a uniform rectangular grid $\mathcal{T} = \{T_m : m = 1...M\}$ we discretise Equations (1-3) by using the mixed cell-centred finite volume method NEM-M0, especially proposed in [FBW77] for the multigroup neutron diffusion equations. It is closely related to lowest-order Raviart–Thomas–Nédélec mixed-hybrid finite elements [HDV93]. To simplify the presentation let $T_0 = R^3 \backslash \Omega$. As for mixed finite elements, we introduce the physically important neutron current density

$$\vec{j}_g = -D_g \vec{\nabla} \phi_g, \qquad g = 1, 2 \qquad (\text{Fick's Law}). \tag{4}$$

To obtain semi-discrete (time dependent) approximations of Equations (1-2) we introduce zeroth momenta ϕ_g^m and C_i^m of the unknown functions ϕ_g and C_i on each element T_m, like in standard finite volume approximations. The vector-valued Equation (4) on the other hand, is integrated over each face $F_{m_1,m_2} = \bar{T}_{m_1} \cap \bar{T}_{m_2} \neq \emptyset$ of the triangulation $\mathcal{T}$, leaving us with an equation for the average current density $j_g^{m_1,m_2}$ on face F_{m_1,m_2}. The significant step is the separation of this average current density into a current $\vec{j}_g^{m_1+}$ from the left element T_{m_1} and a current $\vec{j}_g^{m_2-}$ from the right element T_{m_2}. Let $u \in \{x, y, z\}$ denote one of the coordinate directions, and let F_{m_1,m_2} be perpendicular to u. Then

$$j_g^{m_1,m_2} = \left(j_g^{m_1+}\right)_u - \left(j_g^{m_2-}\right)_u . \tag{5}$$

The boundary condition (3) can then be easily fulfilled by setting $\vec{j}_g^{0\pm} = \vec{0}$. After a simple linear transformation of the resulting system we get the following differential-algebraic equations:

$$\frac{d\phi_g^m}{dt} = f_g^m\left(\phi_1^m, \phi_2^m, \left(j_g^{l_u+}\right)_{u\in\{x,y,z\}}, \left(j_g^{r_u-}\right)_{u\in\{x,y,z\}}, \left(C_i^m\right)_{i=1,\ldots,6}\right) \quad (6)$$

$$\frac{dC_i^m}{dt} = h_i^m(\phi_1^m, \phi_2^m, C_i^m) \quad (7)$$

$$\left(j_g^{m\pm}\right)_u = k_{gu}^{m\pm}\left(\phi_g^m, \left(j_g^{l_u+}\right)_u, \left(j_g^{r_u-}\right)_u\right) \quad (8)$$

The operators f_g^m, h_i^m, and $k_{gu}^{m\pm}$ are all linear again, and l_u (resp. r_u) denotes the left (resp. right) neighbour of m in u-direction.

The system of differential Equations (6-7) is stiff and it is necessary to use stable implicit methods for its integration. As a first-order method we use implicit backward–Euler, as a second-order method we use TR-BDF(2), a composition of Crank-Nicholson with the second-order backward differentiation formula proposed in [BC$^+$85]. This method combines the desirable properties of both methods. It is as stable as BDF(2), but it has a smaller truncation error.

The integration of (7) leads to an explicit equation for C_i^m, and C_i^m can be substituted in (6). Integrating (6) for all energy groups g and elements m and coupling it with (8) finally leads to a $(14\,M)$–dimensional non-symmetric linear equation system

$$A\,\mathbf{x} = \mathbf{b} \quad (9)$$

for $\mathbf{x} := \left(\phi_\mathbf{1}^\mathbf{1}, \phi_\mathbf{2}^\mathbf{1}, (\mathbf{j_1^{1-}})_\mathbf{x}, \ldots, (\mathbf{j_2^{1+}})_\mathbf{z}, \ldots\ldots, \phi_\mathbf{1}^\mathbf{M}, \phi_\mathbf{2}^\mathbf{M}, (\mathbf{j_1^{M-}})_\mathbf{x}, \ldots, (\mathbf{j_2^{M+}})_\mathbf{z}\right)$ in each time step.

PARALLEL SOLVERS

We solve these non-symmetric linear equation Systems (9) using standard Multi-Grid [Hac85] and left-preconditioned Bi-CGStab (P-BiCGStab) [VdV92] on grids of different fineness, where the fine grids are obtained by uniform refinement of the coarse-grid in each coordinate direction. The key components of Multi-Grid are as follows: the coarse-grid correction is obtained using a W-cycle ($\gamma = 2$); the systems on the coarsest grids are solved iteratively using P-BiCGStab; the prolongation P is chosen to be piecewise constant for the fluxes ϕ_g^m, and piecewise linear for the currents $\vec{j}_g^{m\pm}$; the restriction $R = 1/8\; P^T$ using the Galerkin approach; the smoother is a hybrid Block-Jacobi - Block-SOR method that will be discussed in more detail at the end of this section. If we use μ (resp. ν) pre- (resp. post-) smoothing steps in the Multi-Grid method, we will denote the method by MG(μ, ν). We will also use this hybrid method for the preconditioning of Bi-CGStab.

All these solution methods can be fully described in terms of vector operations and matrix-vector multiplications. To parallelise these operations we have to partition the matrices and vectors. We partition the grid(s) $\mathcal{T} = \{T_m : m = 1, \ldots, M\}$ into evenly sized parts $\mathcal{T}_i$ (homogeneous parallelisation), and assign each part and all unknowns and equations that are defined thereon to one processor. The advantage that the NEM-M0 discretisation has over other discretisations is that each component of a vector can be assigned to exactly one element T_m and therefore to one part $\mathcal{T}_i$ of the grid. An important consequence of this property is that vectors do not have to be assembled on the interfaces between two parts of the grid. In the following we will talk of local calculations, if we are concerned only with calculations on one part $\mathcal{T}_i$ of the grid. The local parts of a matrix A and a vector $\mathbf{x}$ on $\mathcal{T}_i$ will be denoted by A_i and $\mathbf{x_i}$ respectively.

For the basic vector operations the individual computations for each component are independent and can be carried out entirely in parallel. In scalar products and matrix-vector multiplications on the other hand, data dependencies arise, and communication or synchronisation is needed. Scalar products $\sigma = \langle \mathbf{x}, \mathbf{y} \rangle$ can be parallelised by reduction. In a first step local scalar products $\sigma_i = \langle \mathbf{x_i}, \mathbf{y_i} \rangle$ are calculated on each processor. In a "critical section" in which the calling processor has sole access to the statements contained within it, they are added to the global scalar product σ. Any other processor, on encountering the occupied section, is forced to wait until the section is cleared. For the matrix-vector multiplications on the other hand, in particular for $A \cdot \mathbf{x}$ and the restriction $R \cdot \mathbf{x}$, we need synchronisation. Each processor needs components of the vector $\mathbf{x}$ that are assigned to its neighbouring processors. Since the processors have shared memory access, we only have to synchronise them at the beginning of each matrix-vector multiplication to guarantee that.

Let us finally discuss the smoother in $\mathrm{MG}(\mu, \nu)$ and the preconditioner for Bi-CGStab. The matrix A in (9) is built up by 14×14 diagonal blocks A_{mm} and a small number of off-diagonal entries. The diagonal blocks have the following structure

$$A_{mm} = \left(\begin{array}{c|c} \begin{matrix} * & * \\ * & * \end{matrix} & \mathbf{0} \\ \hline \begin{matrix} * & 0 \\ \vdots & \\ 0 & * \end{matrix} & I_{12} \end{array} \right), \tag{10}$$

and can be inverted fast and easily. In the sequential case this fact can be exploited in an effective Block-SOR smoother/preconditioner. However, the solution of each block system in the Block-SOR method always depends on the solution of the previous blocks. A straightforward parallelisation of the Block-SOR method is therefore not effective, since the processors would have to wait for each other. The natural way to avoid this, is the application of a hybrid Block-Jacobi - Block-SOR method. Locally each processor uses one iteration of the original Block-SOR method for the approximate inversion of the diagonal block of his part A_i of A, and calculates the local components of the next iterate using Block-Jacobi relaxation. These operations to find the next iterate are independent of the calculations on all other processors, and can be done entirely in parallel. It is only necessary to synchronise the processors prior to each iteration. If the number of elements T_m per part $\mathcal{T}_i$ is reasonably high, the hybrid method only differs in very few components from the sequential method.

NUMERICAL RESULTS

We applied both solution methods in the simulation process of real three-dimensional reactor problems to test their robustness, speedup, and efficiency on three grids of different fineness. The numbers of unknowns N of the systems that arise in each time step are about $5.5 \cdot 10^4$, $4.5 \cdot 10^5$, and $3.5 \cdot 10^6$ on the three different grids. Finer grids were not possible because of a lack of computer memory. Coarser grids are hard to introduce because of the difficult geometry of a reactor. It was therefore only possible to run MG(μ,ν) on the two finer grids. If not otherwise stated, the results have all been obtained using TR-BDF(2) as the time discretisation. One composite time step therefore involves the solution of two linear equation systems.

Robustness and comparison of MG(3/2) and P-Bi-CGStab

The methods converge robustly and the parallelisation has no effect on the convergence, for all the tested problems. In Figures 1 and 2 the iterations and execution times for a short transient of one of the examples on the second grid are given. The time steps are determined adaptively. To check for convergence we use the residual test with $\varepsilon = 10^{-6}$. The iterations and the execution times are almost constant after a short start-up period. In comparison, MG(3/2) is about 2.5 times faster than P-Bi-CGStab throughout the simulation for this problem. It is important to note that the modification of the Block-SOR method in the parallel case does not effect the convergence of the algorithms significantly, and the iterations are in average not higher than in the sequential case.

More generally (see [Sch97]), we can conclude that for the tested examples MG(3/2) is about 2 to 3 times faster than P-BiCGStab on the second grid, and about 3 to 4 times faster on the finest grid. The convergence of both methods does not depend on the partitioning. For MG(3/2) the rate of convergence lies between 0.3 and 0.5 for all examples, and does not depend strongly on the number of unknowns N. However, for a more thorough investigation of the asymptotics we would need more results for other values of N. For P-Bi-CGStab, on the other hand, the rate of convergence increases from 0.5 on the coarsest grid to 0.85 on the finest. Since one iteration of MG(3/2) is only 20 % more expensive than P-Bi-CGStab on all grids we can conclude as expected: the finer the grid becomes the more MG(3/2) outperforms P-Bi-CGStab.

Speedup

The speedup of both methods is optimal (linear) for all tested examples on all but the coarsest grid. On the coarsest grid the speedup for P-Bi-CGStab is also optimal, if less than 10 processors are used. For a higher number of processors the speedup curve starts to level off slightly. In Figure 3 the speedup graphs for one of the examples are shown. We can see the claimed excellent behaviour. The reasons for the superlinear speedup on the second grid are different numbers of iterations and cache-effects. Moreover, these results suggest a good performance even for a much higher number of processors which were unfortunately not available.

Comparison to other parallel reactor simulation codes

An exact comparison of the speedup results with the results reported in the literature [FB$^+$88, HDB96, UTK95, YN93] is difficult, since they are all obtained on different computer systems, for different examples, and for different problem sizes. Nevertheless, none of them obtains optimal speedup for full three-dimensional problems. The best comparison are the results in [HP$^+$96]. They were obtained on exactly the same computer for the same examples, also using threads for the parallelisation of the

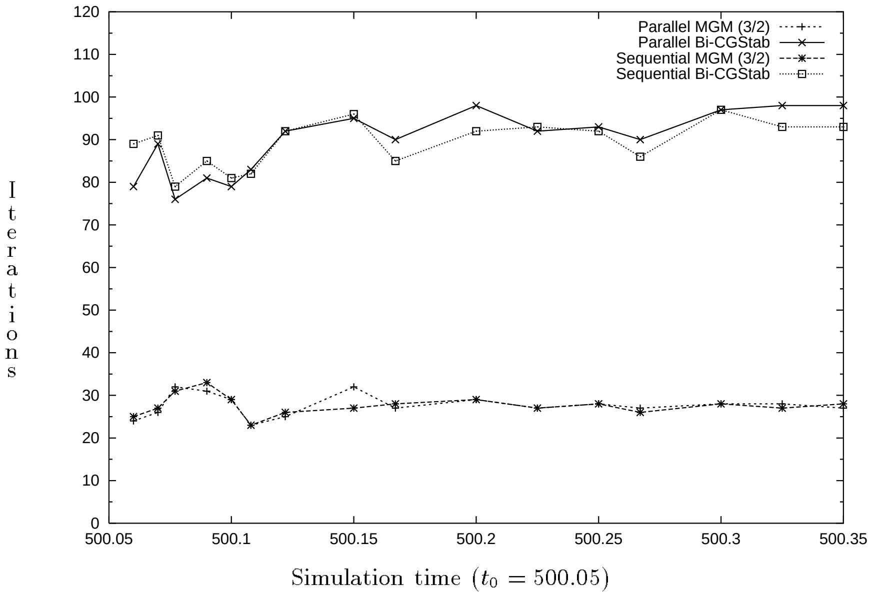


Figure 1 Number of iterations (parallel: 9 processors)

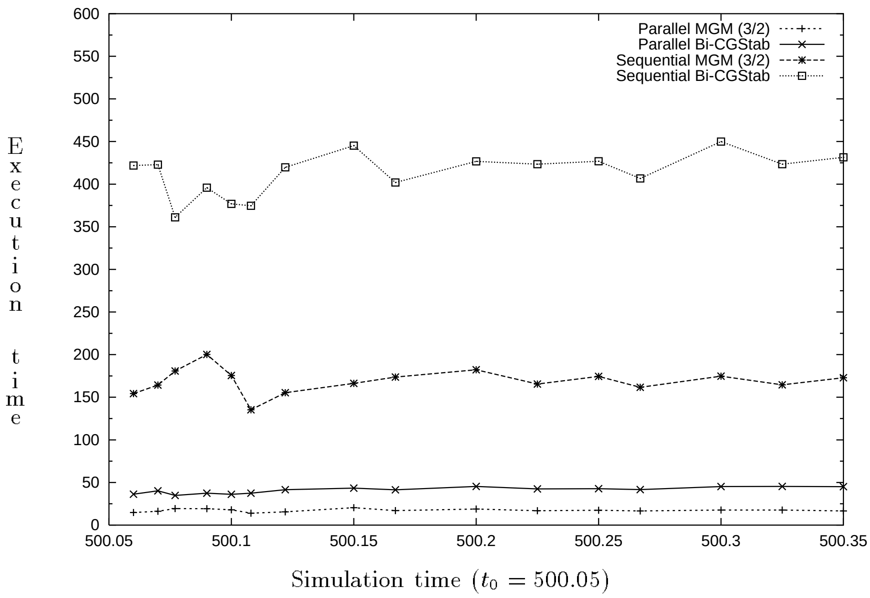


Figure 2 Execution time (in seconds) (parallel: 9 processors)

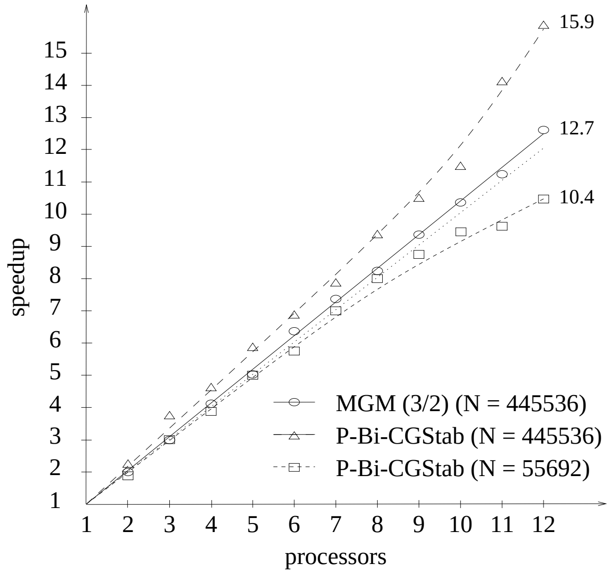


Figure 3 Speedup for one time step ($dt = 0.01$ s)

established reactor simulation code PANBOX of the Siemens AG Operating Division KWU. On the coarsest grid they report a speedup of 2.7, 3.5, and 3.7 for 4, 6, and 8 processors respectively (see Fig. 7 in [HP$^+$96]). Even for higher numbers of processors the speedup will not be much higher than 4. The corresponding speedup factors of P-Bi-CGStab are 3.8, 5.7 and 8.0, and the speedup can be increased to more than 10 for higher numbers of processors. On a finer grid ($N = 2.2 \cdot 10^5$) the speedup factors of PANBOX are 4.1, 5.8, and 7.3 for 4, 6, and 8 processors respectively. In comparison to the speedup obtained with MG(3/2) and P-Bi-CGStab on the second grid, the speedup curve of PANBOX seems to start to level off already at 8 processors.

For the comparison of the execution times we calculated 10 time steps of a control rod ejection benchmark problem on the coarsest grid using PANBOX and P-Bi-CGStab. This is of practical interest, since PANBOX is one of the most efficient methods for the sequential case to date. The solution of 10 time steps of the benchmark problem takes 53 seconds using PANBOX on the SGI Power Challenge. In PANBOX, Equations (1-3) are discretised using NEM-M0 and a special first-order heuristic exponential integration scheme related to implicit backward-Euler. Using P-Bi-CGStab to calculate the same 10 time steps of the benchmark problem with the implicit backward-Euler method takes 115 seconds on the same machine. For a fair comparison we also used the same convergence test as in PANBOX. We see that the sequential version of P-Bi-CGStab is about two times slower than PANBOX. Nevertheless, the far better speedup of P-Bi-CGStab makes it possible to catch up with PANBOX already for 8 processors. For higher numbers of processors, P-Bi-CGStab is therefore going to be faster than the established reactor simulation code PANBOX.

ACKNOWLEDGEMENTS

The author would like to thank Prof. U. Langer from the University of Linz, Dr. M. Paffrath from Siemens ZT Munich, and Dr. I. Graham from the University of Bath for their helpful advice and comments.

REFERENCES

[BC+85] Bank R., Coughran W. M., *et al.* (1985) Transient simulation of silicon devices and circuits. *IEEE Transactions on Computer Aided Design of integrated circuits* 4(4): 436–451.

[DH76] Duderstadt J. J. and Hamilton L. J. (1976) *Nuclear reactor analysis.* J. Wiley & Sons, New York.

[FB+88] Finnemann H., Brehm J., *et al.* (1988) Mulitgrid solution of diffusion equations on distributed memory multiprocessor systems. *Atomkernenergie / Kerntechnik* 52(3): 169–174.

[FBW77] Finnemann H., Bennewitz F., and Wagner M. R. (1977) Interface current techniques for multidimensional reactor calculations. *Atomkernenergie / Kerntechnik* 30: 123–128.

[Hac85] Hackbusch W. (1985) *Multi-grid methods and applications.* Springer, Berlin.

[HDB96] Han G. Joo, Downar T. J., and Barber D. A. (1996) Methods and performance of a parallel reactor kinetics code PARCS. *Proceedings Int. Conf. Physics of Reactors* pages J42–J51.

[HDV93] Hennart J. P. and Del Valle E. (1993) On the relationship between nodal schemes and mixed-hybrid finite elements. *Numerical Methods for Partial Differential Equations* 9: 411–430.

[HP+96] Hammer C., Paffrath M., *et al.* (1996) Performance of the coupled thermalhydraulics/neutron kinetics code R/P/C on workstation clusters and multiprocessor systems. *Proceedings Int. Conf. Physics of Reactors* .

[Sch97] Scheichl R. (1997) *Parallel solution of the transient multigroup neutron diffusion equations with multi-grid and preconditioned Krylov-subspace methods.* (Master's Thesis), volume C 21 of *Schriften JKUL.* Trauner, Linz.

[UTK95] Uematsu M., Tsuiki M., and Kubota K. (1995) Parallel processing of boiling reactor core analysis. *Journal Nuclear Science and Technology* 32(6).

[VdV92] Van der Vorst H. A. (1992) Bi-CGSTAB: A more smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing* 13: 631–644.

[YN93] Yong H. Kim and Nam Z. Cho (1993) Parallel solution of the neutron diffusion equation with the domain decomposition method on a transputer network. *Nuclear Science and Engineering* 114: 252–270.

53

Remarks on the implementation of the Generalized Neumann-Neumann algorithm

Marina Vidrascu [1]

INTRODUCTION

In the past decade significant research efforts have been applied to develop robust algorithms using domain decomposition methods. From both a theoretical and a numerical point of view these methods can now be considered mature. For widespread use, these methods must be accessible to the non-specialist and thus must be available as black boxes. This paper addresses the issues faced when trying to maintain an open software platform suitable for ongoing research while simultaneously meeting commercial code requirements as black boxes. The characteristics of the generalized Neumann-Neumann algorithm, a good black box domain decomposition algorithm candidate, are defined and illustrated with specific examples.

First the general methodology used to practically solve a given problem is described. The next sections review the main characteristics of the Neumann-Neumann preconditioner and discusses the key points of its practical implementation and use. This algorithm is implemented whitin the framework of the general purpose finite element library Modulef. For message passing the PVM library is used. Thus the software is portable and can be run on different platforms including standard parallel computers and clusters of workstations.

1 INRIA-Rocquencourt, B.P 105, F78153 Le Chesnay Cedex, e-mail : Marina.Vidrascu@inria.fr
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

GENERAL METHODOLOGY

To achieve efficiency and reliability modern software must be open and modular. An additional important feature is that it should be easlly reused in different situations. On the other hand, in order to promote the use of numerical methods for non-specialists, it is important to develop black boxes. A black box is a specialized software which is easy to use and minimizes the user's decisions. In addition, it has to be robust and efficient. The question is how to fulfill the apparent conflicting requirements of both modern software and black boxes.

For numerical analysts who use an open software, the solution of a problem is built using a selection of appropriate modules. In general several solution methods are available and it is the user's responsibility to choose the best ones depending, for instance, on the problem type or size. Modules for direct or iterative solutions of linear systems are typical examples of software used in all such constructions. On the contrary, a black box contains all steps involved in the modeling process and corresponds to a particular choice at each step.

The approach proposed to derive a black box from an open software for a given problem is the following:

- Select the optimal solution method, i.e. one whith serious theoretical background and proved to be robust and reliable. In the context of black boxes the best choice is not always the most efficient one. It is interesting to note that, in commercial codes, a direct solution method is often preferred to an iterative one even if in the research community it is well established that iterative methods are more efficient than direct ones especially for large three dimensional problems.
- Design specific pre- and post-processing tools well adapted to the problem to be solved. In particular, check the validity of the model as far as possible and use the specific vocabulary of the actual application.

The advantage of this approach is that the solution of a new problem requires only a few software developments which increase both productivity and robustness.

An overview of existing methods available to solve linear systems shows why it is interesting to consider domain decomposition algorithms. On the one hand direct methods are robust but too expensive for large three dimensional problems. On the other hand, iterative methods are efficient but it is difficult to have robust preconditioners in particular when considering highly non-homogeneous materials. Domain decomposition methods use direct solvers on each subdomain and iterative one on the interface. The choice of the preconditioner is a key point to achieve robustness and performance. The objective of the following sections is to show that the balanced Neumann Neumann algorithm is a good candidate to build a black box for linear and non linear structural mechanics problems.

GENERALIZED NEUMANN-NEUMANN PRECONDITIONER

Definition

The generalized Neumann-Neumann preconditioner described in [LV98], [LV97] is a particular case of the additive Schwarz method applied to the primal Schur complement.

To define the interface problem, first split the original domain of calculation Ω into non-overlapping sub-domains $\Omega = \bigcup_{i=1}^{N} \Omega_i$ with interfaces $\Gamma_i = \partial\Omega_i \backslash \partial\Omega \;\; \Gamma = \cup_i \Gamma_i$.

Then consider a decomposition of each local stiffness matrix in internal and interface nodes contributions :

$$\mathbf{K}^i = \begin{bmatrix} \mathring{\mathbf{K}}^{\mathbf{i}} & \mathbf{B}^i \\ \mathbf{B}^{i^t} & \bar{\mathbf{K}}^i \end{bmatrix},$$

Denoting by $\mathbf{R}^i\bar{X}$ the restriction of the set of interface nodal values $\bar{X}$ to the interface Γ_i, the entire interface problem takes the form of the classical Schur complement system :

$$\mathbf{S}\bar{X} := \sum_i \mathbf{R}^{i^t}\left(\bar{\mathbf{K}}^i - \mathbf{B}^{i^t}(\mathring{\mathbf{K}}^i)^{-1}\mathbf{B}^i\right)\mathbf{R}^i\bar{X} = \bar{F} - \sum_i \mathbf{R}^{i^t}\mathbf{B}^{i^t}(\mathring{\mathbf{K}}^i)^{-1}\mathring{F}^i. \tag{1}$$

To define the generalized Neumann Neumann preconditioner choose :

1. a partition of unity $\mathbf{D}^i : \mathbf{V}_{|\mathbf{\Gamma_i}} \to \mathbf{V}$ satisfying $(\mathbf{V} = \{\bar{\mathbf{v}} = \mathbf{Tr}\ \mathbf{v}|_{\mathbf{\Gamma}}\})$
$$\sum_{i=1}^{N} \mathbf{D}^i\mathbf{R}^i = \mathbf{Id}|_{\mathbf{V}}. \tag{2}$$
2. an approximate local operator $\tilde{\mathbf{S}}^i$ such that
$$\tilde{\mathbf{S}} = \sum \mathbf{R}^{i^t}\tilde{\mathbf{S}}^i\mathbf{R}^i \approx \mathbf{S}. \tag{3}$$
In linear elasticity this is generally the exact local Schur complement, in non linear elasticity it may be the local Schur complement of the linearized elasticity operator.
3. an $\tilde{\mathbf{S}}^i$-orthogonal decomposition of each local space $\mathbf{V}_{|\mathbf{\Gamma_i}}, \mathbf{i} = \mathbf{1}, \mathbf{N}$, into
$$\mathbf{V}_{|\mathbf{\Gamma_i}} = \mathbf{V_i} \oplus \mathbf{Z_i}. \tag{4}$$
The local coarse space $\mathbf{Z}_i$ contains all potential local singularities.

The generalized Neumann-Neumann domain decomposition technique is then the additive Schwarz algorithm solving $\mathbf{S}$ on the space $\mathbf{V}$ of restrictions with

1. coarse space $\mathbf{V_0} = \sum_{\mathbf{i=1}}^{\mathbf{N}} \mathbf{D^i Z_i} \subset \mathbf{V}$, endowed with the scalar product $\tilde{\mathbf{S}}$,
2. local spaces $\mathbf{V_i}, \mathbf{i} = \mathbf{1}, \mathbf{N}$ endowed with the scalar product $\mathbf{B_i} = \tilde{\mathbf{S}}^i$,
3. extensions $I_i = (I - \mathbf{P})\mathbf{D}^i$, with $\mathbf{P}$ the $\tilde{\mathbf{S}}$ orthogonal projection of $\mathbf{V} \to \mathbf{V_0}$.

This algorithm corresponds to the following preconditioning operator

$$\mathbf{M}^{-1} = \tilde{\mathbf{S}}_0^{-1} + \sum_i (I - \mathbf{P})\mathbf{D}^i\tilde{\mathbf{S}}_i^{-1}\mathbf{D}^{i^t}(I - \mathbf{P})^t. \tag{5}$$

General properties

The purpose of this section is to explain why the Neumann-Neumann preconditioner is a good candidate for a black box system. The relevant features in this context include the fact that one can use arbitrary unstructured subdomains and meshes and various different finite elements (2d, 3d, shells).

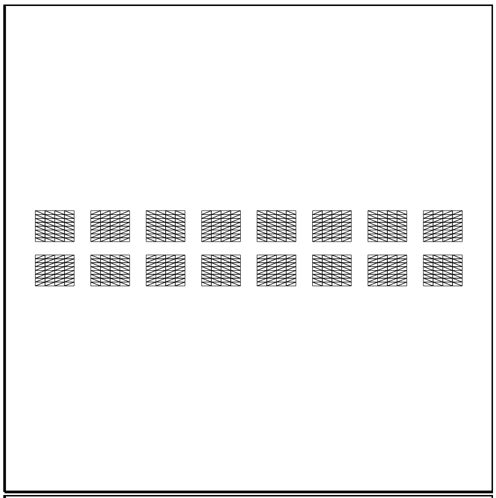
This preconditioner has been extensively studied both from a theoretical and numerical point of view ([Man93], [Le 94], [LV97]). It is robust and can handle non-homogeneous materials and jumps in coefficients. The algorithmic scalability of the preconditioner was proved, i.e. the number of iterations to reach convergence is asymptotically independent on the mesh size and on the number of subdomains. In early versions of the preconditioner (Original Neumann-Neumann in Table 1) the independence on the number of subdomains was not achieved. This improvement, due to J. Mandel [Man93], obtained by the introduction of the coarse space is mandatory for the robustness (Generalized Neumann Neumann in Table 1). There still is a drawback, the aspect ratio of the subdomains has an important influence in the convergence. In practice this means that the way the domain is split into subdomains is very important.

Both the importance of algorithmic scalability and aspect ratio of the subdomains are illustrated with a very simple example. We consider a two dimensional elasticity problem. The domain is clamped on the left side and a uniform traction is imposed on the right side. The domain is decomposed into 2, 4, 8, 16 subdomains. In this last case several decompositions are considered, see Figures 1, 2, 3. The convergence behavior is presented in Figure 4 and clearly shows that the original Neumann-Neumann is not a good candidate for a black-box as the number of iterations increases with the number of subdomains. In addition, the generalized Neumann-Neumann algorithm is stable with respect to the number of subdomains. The number of iterations is given in Table 1. In Figure 4 the best result for the decomposition in 16 subdomains is used for the graphic representation. This example is not used to prove the performance of the method (for this purpose a small 2d problem is irrelevant), but to illustrate that the algorithmic scalability is an issue. The comparison between the three decompositions used for the case of 16 subdomains shows the importance of the aspect ratio of the subdomains and of the regularity of the boundary. This is even worth in 3d real life problems where the algorithm may not converge for an inappropriate decomposition.

In addition this preconditioner is easy to implement and several extensions, to non-matching grids, to non-linear elasticity or to low frequency vibration problems are possible in a straightforward manner.

PARALLEL IMPLEMENTATION FOR THE SOLUTION OF LARGE SIGNIFICANT PROBLEMS

One motivation for the design of domain decomposition algorithms was that they are suitable for coarse-grained parallel computers with distributed memory or for clusters of workstations. The actual implementation is done within the finite element library Modulef with no restrictions on the shapes of the domains and on the choice of finite elements. To preserve the portability, the PVM library is used to exchange messages



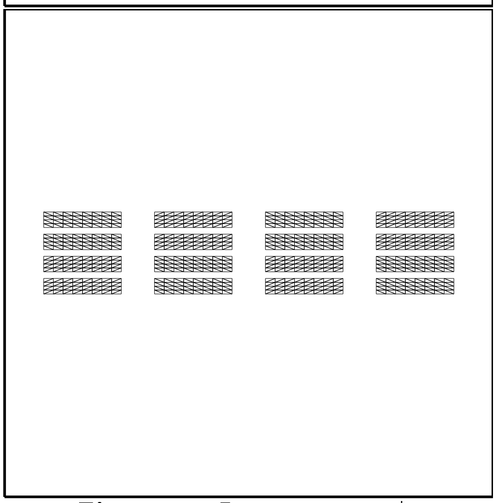
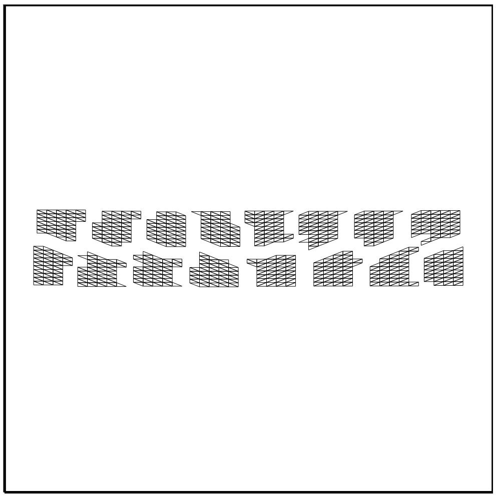


Figure 1 *Decomposition 8*2*

Figure 2 *Decomposition 4*4*



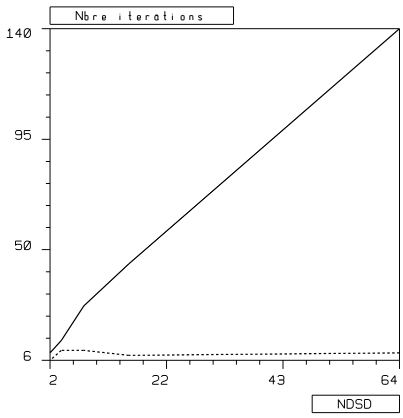


Figure 3 *Irregular decomposition*

Figure 4 *Number of iteration vs number of subdomains*

Table 1 *Number of iterations for the two versions of the preconditioner*

# subdomains	Orig. Neumann Neumann	Generalized Neumann Neumann
2	9	6
4	14	10
8	28	10
16 (4*4) (Fig: 2)	65	21
16 (8*2) (Fig: 1)	45	8
16 (irregular) (Fig: 3)	70	17
64	140	9

and data between processors.

There are two different ways to tackle the practical implementation of this domain decomposition algorithm : either consider the entire domain or a given decomposition into non-overlapping subdomains as an input.

The choice used here is the second one because:

- It is more amenable to a modular approach. Indeed, from an algorithmic point of view the only link between the decomposition of the domain into subdomains and the solution is that regular domains with nice aspect ratio are needed. In addition several tools can be used for mesh partitioning (see section 53) and the re-usability of algorithms is increased.
- It allows the solutions of very large systems where the memory of one processor is not sufficient for the entire mesh.
- Each subdomain is considered as a standard mesh data structure. Thus, the data locality, an important problem in parallel processing, is obtained with no additional work and this also ensures a very low degree of data migration.

There is, of course a drawback: a particular data structure to describe the interface between subdomains is needed. We will concentrate here on the solution algorithm, the mesh partitioning being a completely different problem.

To summarize section 53, the problem is to solve $\mathbf{S}\bar{X} = F$ where $\mathbf{S}$ is defined by (1), using a preconditioned conjugate gradient algorithm where the preconditioner M is defined by (5).

The software is parallel and uses the message passing paradigm and a **master-slave** approach. Here, the **master** pilots the PCG algorithm for the interface problem and each **slave** is in charge of the local computations by subdomain. As for classical problems the boundary conditions, material characteristics and loads are described globally, in a standard way.

The user starts the master process which will start as many slave processes as subdomains. A modular approach is used which make it easy to program the extensions described in section 53.

The **master** process has the following tasks:

i) spread data to all processors, collect data to compute the data structure describing the interface,
ii) pilot construction of the coarse problem,
iii) pilot the conjugate gradient iterations. The only non-standard step is the solution of the coarse problem in the preconditioning step. As this is a problem of small size (less then $6 \times subdomains$) a direct method is convenient.

The main programming effort is contained in step ii). Step iii) is standard and uses numerical kernels such as BLAS. The specific steps are the matrix vector product and the preconditioning step. Notice that $\mathbf{S}$ is never computed but, building the matrix-vector product $\mathbf{S}\bar{X}$ involves local solutions of Dirichlet problems on each subdomain. This will be done by the **slave** processes. Similarly, at the preconditioning step, computation of $M^{-1}\bar{r}$, involves solution of local Neumann problems. In both situations the **master** process send to the **slave** processes a vector containing data on the boundary, the **slave** processes perform the local computations and send back a vector containing local solutions and, finally, the **master** process add up all contributions. This description shows that the communications are very simple, so the choice of the message passing software is not an issue.

The **slave** processes have the following tasks:

i) receive data from master and use it to compute matrix $\mathring{\mathbf{K}}^i$ in (1) and $\tilde{\mathbf{S}}$ in (5). These matrices will be used to solve local Dirichlet problems (for the computation of $\mathbf{Sx}$), respectively Neumann problems (for the preconditioner). They are computed and factorised once for all as direct methods are used to solve local problems. Also compute matrix $\mathbf{B}^i$ in (1).
ii) construction of the coarse problem. The key point is the choice of the local coarse space $\mathbf{Z}_i$ in (4). For elasticity problems, $\mathbf{Z}_i$ is the space of local rigid-body motions. To compute it, introduce the independent degrees of freedom characterizing the rigid body motions (six or less of them). The simplest way to identify these modes is to stop the factorization process when a "null pivot" is found. This procedure is not very robust as the value of the "null pivot" depends on the conditionning of the matrix which is unknown. A reliable procedure, such as that described in [FG96], is requiered for a black box.
iii) conjugate gradient iteration. The local contribution of vector $\bar{X}$ is $X_i = \mathbf{R}^i\bar{X}$ and contribution to solution of $\mathbf{S}\bar{X}$ implies the solution of the Dirichlet problem $(\mathring{\mathbf{K}}^i)^{-1}(\mathbf{B}^i X_i)$. Similar computations for the preconditioning step, once the $\mathbf{D}^i$ in (5) are choosed as indicated in (2). A good choice which allows to consider jumps in coefficients is to define $\mathbf{D}^i$ at each interface point P_l as the relative value of the average stiffness ρ_i of subdomain Ω_i to the sum $\sum_{P_l \in \Omega_j} \rho_j$ of stiffness values of all the subdomains Ω_j containing P_l.

Notice that the computations performed by the **slaves** are done in parallel.

REQUIREMENTS FOR CORRECT USE OF DOMAIN DECOMPOSITION

To use a domain decomposition algorithm as a black box it is essential to use an automatic mesh partitioning tool. The Modulef library has a partitioner based on K-means techniques. Other popular partitioning tools well-adapted to this algorithm are described in ([SF95], [Sim91]). As noted in section 53 the quality of the partition is very important from an algorithmic point of view, which is true for an entire class of domain decomposition methods [FMB94]. For this method a good aspect ratio of the domains is required, as already mentioned in the example of section 53 (compare results obtained with the decompositions in figures 1 and 2). It is almost impossible, for general meshes to obtain a regular boundary with an automatic mesh partitioner. This has an impact on convergence behavior (again, see in section 53 results obtained with meshes of figures 1 and 3). Nevertheless, in realistic computations it is mandatory to use an automatic tool partitioning.

To obtain parallel performance the decomposition must also achieve a good load balancing. How to achieve a good decomposition is still an active research area. This is an essential requirement for a correct use of domain decomposition methods.

CONCLUSIONS

Domain decomposition methods when used on parallel computers provide a very powerful tool to solve large real-life elliptic problems. The generalized Neumann-Neumann algorithm is robust and can easily be implemented using existing software. In order to use it as a black box it is necessary to combine the use ot this algorithm with an automatic mesh partitioning tool, further research should address the issue of the impact of the size of subdomains to achieve parallel performance as well as the choice of the best strategy to partition the mesh.

REFERENCES

[FG96] Farhat C. and Gradin M. (may 1996) On the computation of the null space and generalized inverse of a large matrix, and the zero energy modes of a structure. Technical Report CU-CAS-96-13, Center for aerospace structures, University of Colorado, Boulder.

[FMB94] Farhat C., Maman N., and Brown G. (january 1994) Mesh partitioning for implicit computations via domain decomposition: Impact and optimization of the subdomain aspect ratio. Technical Report CU-CAS-94-02, Center for aerospace structures.

[Le 94] Le Tallec P. (1994) Domain decomposition methods in computational mechanics. In *Computational Mechanics Advances*, number 1 in Computational Mechanics Advances, pages 121–220. North-Holland.

[LV97] Le Tallec P. and Vidrascu M. (1997) *Solving Large Scale Structural Problems on Parallel Computers using Domain Decomposition*, chapter 2, pages 49–82. John Wiley & Sons.

[LV98] Le Tallec P. and Vidrascu M. (1998) Generalized neumann-neumann preconditioners for iterative substructuring. In Petter Bjørstad M. E. and

Keyes D. (eds) *Proceedings of the ninth international symposium on Domain Decomposition Methods for Partial Differential Equations, Bergen, June 96*, pages 413–425.
[Man93] Mandel J. (1993) Balancing domain decomposition. *Communications in numerical methods in engineering* 9: 233–241.
[SF95] Sharp M. and Farhat C. (1995) TOP/DOMDEC A Totally Object Oriented Program for Visualisation, Domain Decomposition and Parallel Processing. User's manual, PGSoft and The University of Colorado, Boulder.
[Sim91] Simon H. (1991) Partitioning of unstructured problems for parallel processing. *Comput. Sys. Engrg.* 2: 135–148.

54

Scalability and load imbalance for domain decomposition based transport

P. Wilders[1]

Introduction

Transport phenomena play an important role in reservoir engineering and surface or subsurface environmental studies. From the application viewpoint, there is a need to resolve large computational models with fine grids (for example, to study fingering). Scaled distributed computations are useful at this point and, as a consequence, scalability is of some concern in this field. We study a simple 2D tracer flow in a porous medium in order to reveal some basic properties at this point.

The domain decomposition method under consideration is a one-level Krylov-Schwarz method, formulated in terms of interface variables [BW97], [WB99]. This method is combined with ILU-preconditioned BiCGSTAB for the subdomain inversion. For the parallel implementation the focus is on distributed MIMD platforms with explicit parallel programming and a gather/scatter as the basic communication scheme. The parallel performance is studied, both analytically and experimentally on an IBM-SP2. It is shown that load balancing effects are of major importance and the impact of this observation on the scalability will be examined. Related studies on parallel performance modelling of domain decomposition methods have been undertaken in [GK89], [HZ93], [CS95], [BSK95]. In these studies load balancing effects are either absent or have assumed to be absent.

1 Delft University, Dept. of Math., P.O. Box 5031, 2600 GA Delft, NL, p.wilders@math.tudelft.nl
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

Following [HZ93], [CSSY94], our study deals with relative measures (relative efficiency, overhead) and concerns, speaking in terms of [Qui94], the parallelizability. This can be seen as a basic step in a full parallel performance analysis. The absolute efficiency may be written as a product of the numerical efficiency and the relative (or parallel) efficiency. This enables a separate study of the two. Of course, the application of one-level methods leads to a numerical efficiency that does not scale perfectly, e.g. [AN95], and in some cases degradation occurs. However, for linearly scaled transport problems the degradation at this point has been found to be moderate [Cai91], [WF98]. Moreover, it is known that in a two-level method the coarse grid correction leads to a significant degradation of the parallel efficiency, e.g. [HZ93].

Physical and numerical issues

We consider a scalar conservation law of the form

$$\varphi \frac{\partial c}{\partial t} + \nabla \cdot [\mathbf{v} f(c) - D \nabla c] = 0 \quad , \quad \mathbf{x} \in \Omega \in I\!R^2 \quad , \quad t > 0 \quad . \tag{1}$$

In this paper only tracer flows are considered. The coefficients $\varphi, \mathbf{v}$ and D are assumed to be time-independent and $f(c) = c$. Our interests are on the advection-dominated case, i.e. the diffusion tensor D depends on small parameters. For the spatial discretization we employ cell-centered triangular finite volumes, based upon a 10-point molecule [WF97]. This results in the semi-discrete system

$$L \frac{dc}{dt} = F(c, \mathbf{x}) \quad , \tag{2}$$

for the centroid values of the concentration. L is a diagonal matrix, containing the cell values of the coefficient φ multiplied with the area of the cell. Moreover, F is a nonlinear differentiable function of c. We set

$$J = \frac{\partial F}{\partial c} \quad . \tag{3}$$

The Jacobian J represents a sparse matrix with a maximum of 10 nonzero elements in each row.

The linearly implicit trapezoidal rule is used for the time integration:

$$(\frac{L}{\tau_n} - \frac{1}{2} J^n) c^{n+1} = (\frac{L}{\tau_n} - \frac{1}{2} J^n) c^n + F^n \quad . \tag{4}$$

Here, τ_n denotes the time step. The scheme is second-order accurate in time.

The linear system (4) is solved iteratively by means of a one-level Krylov-Schwarz domain decomposition method, in our case a straightforward nonoverlapping additive Schwarz preconditioner with GMRES as the Krylov subspace method. ILU-preconditioned BiCGSTAB is used for the approximate inversion of the subdomain problems.

Experimental studies have been conducted for tracer flows in reservoirs. The velocity $\mathbf{v}$

is obtained from the pressure equation, using strongly heterogeneous permeability data provided by Agip S.p.A (Italian oil company). In this paper we consider the quarter of five spots, a well-known test problem for porous medium applications. A fully balanced blockwise decomposition into Q subdomains is introduced. The total number of unknows is N. Extensive data with regard to timing and parallel performance (on a SP2) are available for linearly scaled problems, i.e. $N = N_0 Q$ (with $N_0 = 3200$) and $Q = 4, 9, 16$. As soon as N varies, either in the analysis or in experiments, it is important to take application parameters into account [SHG93]. From the theory of hyperbolic difference schemes it is known that the Courant number is the vital similarity parameter. Therefore, we fix the Courant number. This means that both the spatial grid size h ($N = O(1/h^2)$) and the time step τ vary with τ/h constant. For linearly scaled problems this results in $h, H, \tau = O(1/\sqrt{Q})$ with H measuring the subdomain size.

Timing and load imbalance

The computational domain has been divided into Q subdomains. Most of the computation within subdomain q can be done independently of the other subdomains; this part of the computation is referred to as *subtask q*. The subtasks may be executed in parallel. The computation is divided into $(p+1)$ tasks, numbered $j = 0, ..., p$. Task 0 is the *sequential task* and the tasks $1, ..., p$ are the *distributed tasks*. $Q(j)$ of the subtasks are assigned to the distributed task j. We distinguish p *processes*. Task 0 and task 1 are taken together to form process 1. For $j > 1$, process j is identical to task j. Each process is executed on its own processor. The processes are connected via explicit parallel programming, using message passing.

Our concern is on the computational heart of the code, i.e. the time stepping loop. For the ease of presentation we consider a single time step. There are three phases in the program, i.e. communication and computation either as a part of the sequential task 0 or as a part of the distributed tasks $1, ..., p$. In this paper we consider the fully synchronized case only. The different processes are synchronized explicitly each time the program switches between two of these three phases. The elapsed time in a synchronized p processor run is denoted with T_p. It follows that

$$T_p = T^{(s)} + T_p^{(d)} + T_p^{(c)} \quad , \tag{5}$$

with $T^{(s)}$ the computational time spent in the sequential task 0, $T_p^{(d)}$ the maximal time spent in the distributed tasks and $T_p^{(c)}$ the communication time. Of course, $T_1^{(c)} = 0$.

Let us define

$$T_1(p) = T^{(s)} + pT_p^{(d)} \quad . \tag{6}$$

For $p = 1$, there holds $T_1(p) = T_1$. For $p > 1$, $T_1(p)$ is the elapsed time of a single processor shadow run, carrying out all tasks in serial, while forcing the distributed tasks to consume the same amount of time. It is clear that $T_1(p) - T_1$ presents a measure of idleness in the parallel run due to load balancing effects. It easily follows that

$$T_1(p) - T_1 = pT_p^{(d)} - T_1^{(d)} \quad . \tag{7}$$

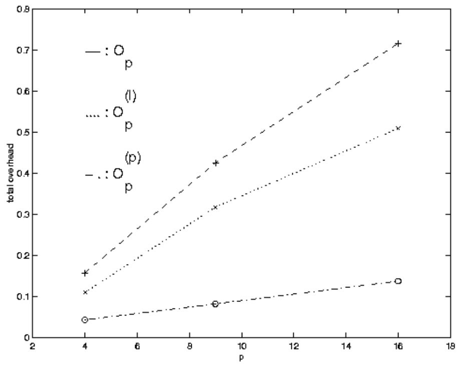


Figure 1 Overhead O_p and its components $O_p^{(l)}, O_p^{(p)}$.

Performance measures; factorization and approximation

The (relative) *efficiency* E_p, the (relative) *costs* C_p and the (relative) *overhead* O_p are defined by

$$E_p = \frac{T_1}{pT_p} \quad , \quad C_p = \frac{1}{E_p} = \frac{pT_p}{T_1} \quad , \quad O_p = C_p - 1 \quad . \tag{8}$$

Note that in (8) we compare the parallel execution of the algorithm (solving a multi-domain problem with Q subdomains) with the serial execution of the <u>same</u> algorithm (*relative*).

We introduce the following factorization of the costs C_p:

$$C_p = C_p^{(l)} C_p^{(p)} \quad , \quad C_p^{(l)} = 1 + O_p^{(l)} \quad , \quad C_p^{(p)} = 1 + O_p^{(p)} \quad , \tag{9}$$

where

$$O_p^{(l)} = \frac{T_1(p) - T_1}{T_1} \quad , \quad O_p^{(p)} = \frac{pT_p - T_1(p)}{T_1(p)} \quad . \tag{10}$$

It follows that

$$O_p = O_p^{(l)} + O_p^{(p)} + O_p^{(l)} O_p^{(p)} \quad . \tag{11}$$

The term $O_p^{(l)}$ is associated with load balancing effects. $O_p^{(p)}$ is referred to as the parallel overhead. $O_p^{(p)}$ combines sequential overhead and communication costs. Figure 1 presents the overhead O_p and its components $O_p^{(l)}, O_p^{(p)}$ such as measured in the linearly scaled experiments, mentioned in Section 54. The importance of load balancing effects can be observed. Let us introduce

$$\tilde{O}_p^{(l)} = \frac{pT_p^{(d)} - T_1^{(d)}}{T_1^{(d)}} \quad . \tag{12}$$

From (7), (10) it follows that

$$\frac{\widetilde{O}_p^{(l)} - O_p^{(l)}}{O_p^{(l)}} = \varepsilon \quad , \quad \varepsilon = \frac{T^{(s)}}{T_1^{(d)}} \quad . \tag{13}$$

Here, ε measures the fraction of the work done in the sequential task and this fraction is small ($< 1\%$ in our experiments). Therefore, we expect $\widetilde{O}_p^{(l)}$ to present a good approximation of $O_p^{(l)}$.

Performance model for a square domain

As before, a fully balanced blockwise decomposition of the square domain into Q subdomains is employed; the total number of unknowns is N, N/Q per subdomain. The total number of edges on internal boundaries is B, on the average B/Q per subdomain. The number of interface variables depends linearly on B. B is given by

$$B = \sqrt{8N}(\sqrt{Q} - 1) \quad . \tag{14}$$

We assume that either $p = 1$, i.e. a serial execution of all subtasks, or $p = Q$, i.e. the subtasks coincide with the distributed tasks. In first approximation there holds

$$\frac{p}{Q} T_p^{(d)} = \alpha_1 \frac{N}{Q} + M(\alpha_2 \frac{B}{Q} + \alpha_3 \frac{N}{Q} I_p) \quad , \tag{15}$$

with

$$I_p = \begin{cases} \frac{1}{M} \sum_{m=1}^{M} \frac{1}{Q} \sum_{q=1}^{Q} I(m,q) & , \quad p = 1 \\ \frac{1}{M} \sum_{m=1}^{M} \max_{q=1,\ldots,Q} I(m,q) & , \quad p = Q \quad . \end{cases} \tag{16}$$

Here, $m = 1, ..., M$ counts the number of matrix-vector multiplications in the domain decomposition iteration (outer iteration) and $I(m,q)$ denotes the number of matvec calls associated with the inner iteration in subdomain q for the m-th outer matvec call. The first term on the rhs of (15) corresponds with building processes (subdomain matrix, etc.). The third term on the rhs of (15) reflects the inner iteration and the second term is due to correcting the subdomain rhs.

A least-squares estimation of the coefficients in (15) has been carried out, using the linearly scaled experiments mentioned in Section 54 (six measurements available for fitting (15)). This leads to

$$\alpha_1 = .72 * 10^{-4} \quad , \quad \alpha_2 = .59 * 10^{-4} \quad , \quad \alpha_3 = .26 * 10^{-5} \quad . \tag{17}$$

Using (12), (15), it is now straightforward to obtain the analytical approximation $\widehat{O}_p^{(l)}$ of the overhead function $O_p^{(l)}$:

$$\widehat{O}_p^{(l)} = \frac{\alpha_3 N M (I_p - I_1)}{\alpha_1 N + M(\alpha_2 B + \alpha_3 I_1)} \quad . \tag{18}$$

This shows that the load imbalance is due to variations of the number of BiCGSTAB iterations over the subdomains. The load balancing effects seems to be that strong, because we are dealing with an advection-dominated transport problem in which a front, connecting two constants states, is moving through the domain. This means that there are subdomains in which the solution is constant and little happens, besides subdomains close to the front with a lot of activity.

We set

$$\text{Dev} = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{\widehat{O}_p^{(l)} - O_p^{(l)}}{O_p^{(l)}} \right| * 100\% \quad , \tag{19}$$

with n denoting the number of measurements for $O_p^{(l)} (n = 3;\ p = Q = 4, 9, 16)$. It is found that Dev $= 5.8$, showing that the analytical approximation is quite reliable.

Scalability

Using the isoefficiency metric a parallel algorithm is called scalable if it is possible to find curves in the (p, N)-plane on which the efficiency E_p is constant [GK93], [CSSY94]. A sufficient condition for the existence of isoefficiency curves is

$$\lim_{\substack{N \to \infty \\ p \text{ fixed}}} E_p > 0 \quad \text{or} \quad \lim_{\substack{N \to \infty \\ p \text{ fixed}}} O_p < \infty \quad . \tag{20}$$

The condition regarding the overhead function is equivalent with (see (11))

$$\lim_{\substack{N \to \infty \\ p \text{ fixed}}} O_p^{(l)} < \infty \quad , \quad \lim_{\substack{N \to \infty \\ p \text{ fixed}}} O_p^{(p)} < \infty \quad . \tag{21}$$

A discussion of the second limit in (21) is beyond the scope of the present paper. We only remark that numerical experiments indicate that $O_p^{(p)}$ approaches zero in the limit.

In Section 54 an analytical approximation of $O_p^{(l)}$ for square domain has been given together with arguments showing that this approximation is valuable, at least for linearly scaled problems. Here, we use this approximation to investigate the limiting process in (21). From (14), (18) it follows that

$$\widehat{O}_p^{(l)} \sim F_1 F_2 \quad , \quad N \to \infty \ , \ p \text{ fixed} \tag{22}$$

with

$$F_1 = \frac{M I_1}{\frac{\alpha_1}{\alpha_3} + M I_1} \quad , \quad F_2 = \frac{I_p - I_1}{I_1} \quad . \tag{23}$$

Note that (17) leads to $\alpha_1/\alpha_3 \approx 28$.

The numbers F_1 and F_2 have been computed, as an average over multiple time steps, for $p = Q = 4$ (4 subdomains) and $N = 3200, 12800, 51200$. Table 1 presents all relevant information (note that these data can be obtained from a serial run). The

Table 1 Iterative properties for 4 subdomains ($p = Q = 4$) and F_1F_2, the approximation of load imbalance.

N	M	I_1	I_p	F_1	F_2	F_1F_2
3200	11.2	5.8	7.0	.70	.21	.15
12800	11.5	6.2	8.0	.72	.29	.21
51200	12.0	6.8	8.6	.74	.26	.19

data in Table 1 indicate that the parallel algorithm meets the first condition of (21). However, the overhead due to load balancing effects does not approach zero in the limit and some degradation of the efficiency has to be accepted. The values found for F_1F_2 in Table 1 are in agreement with some early experiments done on a SP2. In fact, these experiments gave $O_p = .18, .19, .20$ for, respectively, $N = 3200, 12800, 51200$.

Concluding remarks

We have presented a parallel performance model for domain decomposition based transport. The model was used to investigate load balancing effects. Load imbalance due to variable iterative properties of the inner iteration over the subdomains presents a major factor. The analysis shows that load imbalance excludes efficiencies close to one. However, it was observed that load balancing effects do not prevent scalability in the isoefficiency metric. Experimental results indicate that isoefficiency curves with an efficiency around 0.8 exist.

The values of M, I_1 and I_p found in Table 1 indicate that the iterative procedures depend only weakly on the mesh size h for a fixed number of subdomains. With regards to the domain decomposition iteration a similar observation was done in [BW97]. A further study of the iterative procedures is necessary in order to present theoretical arguments for this behaviour.

Acknowledgment

The research was supported by the Dutch Ministery of Economic Affairs as part of the HPCN-TASC program.

REFERENCES

[AN95] Axelsson O. and Neytcheva M. (1995) Scalable parallel algorithms in CFD computations. In Hafez M. and Oshima K. (eds) *Computational Fluid Dynamics Review 1995*, pages 837–857. Wiley.

[BSK95] Brakkee E., Segal A., and Kassels C. (1995) A parallel domain decomposition algorithm for the incompressible Navier-Stokes equations. *Simulation Practice and Theory* 3: 185–205.

[BW97] Brakkee E. and Wilders P. (1997) The influence of interface conditions on convergence of Krylov-Schwarz domain decomposition for the advection-diffusion equation. *J. Scientific Computing* 12: 11–30.

[Cai91] Cai X. (1991) Additive Schwarz algorithms for parabolic convection-diffusion equations. *Numer. Math* 60: 41–61.

[CS95] Chan T. and Shao J. (1995) Parallel complexity of domain decomposition methods and optimal coarse grid size. *Parallel Computing* 21: 1033–1049.

[CSSY94] Carey G., Schmidt J., Singh V., and Yelton D. (1994) A prototype scalable, object-oriented finite element solver on multicomputers. *J. Par. Distr. Comp.* 20: 357–379.

[GK89] Gropp W. and Keyes D. (1989) Domain decompositions on parallel computers. *Impact of Computing in Science and Eng.* 1: 421–439.

[GK93] Gupta A. and Kumar V. (1993) Performance properties of large scale parallel systems. *J. Par. Distr. Comp.* 19: 234–244.

[HZ93] Hoffmann K. and Zou J. (1993) Parallel efficiency of domain decomposition methods. *Parallel Computing* 19: 1375–1391.

[Qui94] Quinn M. (1994) *Parallel computing: theory and practice*. Mc Graw-Hill. second edition.

[SHG93] Singh J., Hennessy J., and Gupta A. (1993) Scaling parallel programs for multiprocessors: methodology and examples. *IEEE Computer* July: 42–50.

[WB99] Wilders P. and Brakkee E. (1999) Schwarz and Schur: a algebraical note on equivalence properties. *SIAM J. Sci. Comp.* to be published.

[WF97] Wilders P. and Fotia G. (1997) Implicit time stepping with unstructured finite volumes for 2D transport. *J. Comp. Appl. Math.* 82: 433–446.

[WF98] Wilders P. and Fotia G. (1998) One-level Krylov-Schwarz domain decomposition for finite volume advection-diffusion. In Bjørstad P., Espedal M., and Keyes D. (eds) *Domain Decomposition Methods 9*, pages 558–565. Domain Decomposition Press, Bergen.

Part IV

Student Papers

Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

55

Parallel Implementation of the Spectral Element Method with Nonconforming Mesh

H.Feng[1], C.Mavriplis[2]

The Nonconforming Spectral Element Method (NSEM) solves PDEs in complex geometries with high accuracy, however, it is an expensive method. Since parallel computation is effective in decreasing CPU time, a parallel algorithm for the NSEM is presented. Implementations on SGI Power Challenge using MPI are evaluated in terms of measured speedup and parallel efficiency for schemes of one element and multiple elements per processor.

Introduction

The Spectral Element Method (SEM) [Pat84] is a high order discretization scheme for the solution of nonlinear PDEs. The method draws its strengths from the finite element methods e.g.[SF73] for geometrical flexibility and the spectral methods e.g.[GO77] for high accuracy. The NSEM [MMP89], which allows nonconforming matching at element interfaces, provides increased geometrical flexibility. The flexibility in turn provides computational efficiency by allowing finer resolution in areas where physical variables are changing rapidly.

1 Civil, Mechanical and Environmental Engineering Department, School of Engineering and Applied Science, George Washington University, Washington, DC, 20052, USA . email: fhy@seas.gwu.edu
2 Address same as above.
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

The NSEM is capable of solving complicated PDEs such as the Navier-Stokes equations for fluid flow with high accuracy. However, high accuracy is costly, i.e. it requires long CPU times. Parallel computation is an effective way to decrease CPU time by using multiprocessors to do computations simultaneously. Due to the heterogeneous nature of the discretization, the SEM is naturally suited to concurrent implementation [FHK+88, FP91]: the high accuracy within each element provides sufficient loading of individual processors and the finite element approach decouples elemental work which can be assigned to independent processors.

This paper presents a parallel algorithm implementation of the NSEM. First, the implementation is based on a one element per processor scheme (OEPP). This work is based on a previous investigation of Ebrat et al [EMD97]. From execution time evaluations, we observe that the speedup and parallel efficiency are poor due to the processor load being low and the communication taking a big part of the total execution time. This paper therefore focuses on a scheme of multiple elements per processor (MEPP). Mappings of two and four elements per processor are tested for the algorithm, using message passing interface (MPI) on a SGI Power Challenge.

Spectral Element Method

SEMs are high-order weighted residual domain decomposition techniques based on spectral expansions for solution of PDEs such as the Navier-Stokes equations for direct simulation of fluid flow. The computational domain is broken up into macro-elements within which variables and geometry are represented by high-order tensor product polynomial expansions. The weak coupling between dependent variables for adjacent elements result in symmetric, and relatively sparse assembled matrices, which help to decrease the computational complexity in terms of memory requirements and processing time.

A simple test Poisson problem is presented to illustrate the spectral element discretization technique: $-\nabla^2 u = f$ in Ω, $u = 0$ on $\partial\Omega$ which has the following equivalent variational form: $a(u, v) = (f, v)$, $\forall v \in H_0^1(\Omega)$. The spectral element method proceeds by discretizing the domain Ω into $k = 1, \ldots, K$ subdomains Ω^k. The discretized equation is obtained from the Galerkin approximation with a finite set of functions. Legendre polynomials are introduced as the basis functions and Gauss-Lobatto quadrature is used to obtain the matrix equation

$$Au = Bf \tag{1}$$

where A is the discrete Laplacian operator and B is the mass matrix [MP89].

Mortar Method for Nonconforming Mesh

While the SEM, in comparison with spectral methods, is better suited to modeling complex geometries, the conforming geometric matching required between neighboring high-order elements leads to complications and inefficiencies in mesh generation, dynamic mesh refinement, and the treatment of moving boundaries. Allowing nonconforming matching at the element interfaces, the Nonforming Mortar Element Method [MMP89, BMP94] greatly increases the flexibility of the spectral element domain decomposition.

The method introduces a new mortar trace space which contributes to decoupling the local residual evaluation and the transmission of elemental boundary, i.e., continuity conditions. The configuration of the discretization of a nonconforming mesh is described in Figure 1(b) where each element is surrounded by a mortar structure γ with vertices ν at each corner. In the conforming case, the discretization space $X_h \subset H_0^1(\Omega)$; in nonconforming approximations, X_h is not a subspace of H_0^1, thereby introducing additional "consistency" errors. However, a carefully constructed discretization space retains spectral accuracy and with corresponding basis functions [MMP89], we obtain the fully discrete equations for the Poisson problem for the nonconforming mesh

$$Q^T A Q u = Q^T B f \tag{2}$$

where A and B represent the elemental conforming discrete Laplace operator and mass matrix respectively and Q is a projection operator from the mortar to the elemental structure (as shown in Figure 1(c)). The global Laplace operator is generated by local operators "mortared" together by Q and Q^T operations, where the Q^T operator is the algebraic form of the standard direct stiffness summation procedure.

The Nonconforming Spectral or Mortar Element method lends itself well to efficient implementation on parallel supercomputers: it allows for the classification and decoupling of the computational work into loosely coupled subdomains; it lays the foundation for sparse inter-domain communication and preserves local structure for fast evaluation procedures.

Parallel Implementation

The implementation is accomplished on a SGI Power Challenge with the intent of testing and future implementation on a larger more powerful parallel platform. Message passing between processors is achieved by MPI.

One Element per Processor Scheme (OEPP)

The simplest mapping of elements to processors is to assign each processor one element. Figure 2(b) shows a simple form of element-processor mapping for a model nonconforming mesh. Since each processor can only access the data of its own element, a neighborhood investigation operation is performed first to establish an array for each element recording the information about the neighborhood. After decomposition, a local mortar is established based on the global mortar configuration. Each processor retains only its corresponding portion of the global mortar. Figure 2 shows the relation between global mortar and local mortar for both the sequential and parallel implementation. All basic operators involved in spectral element discretization are

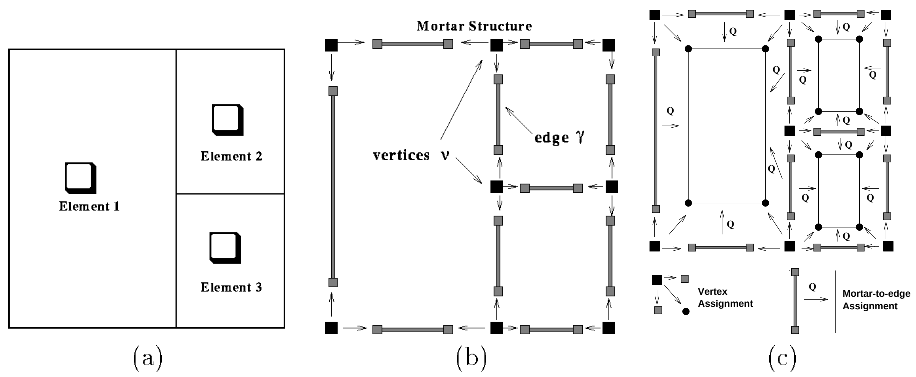


Figure 1 (a)A typical nonconforming mesh, (b) the structure of its corresponding mortar, (c) projection operation Q on nonconforming mesh.

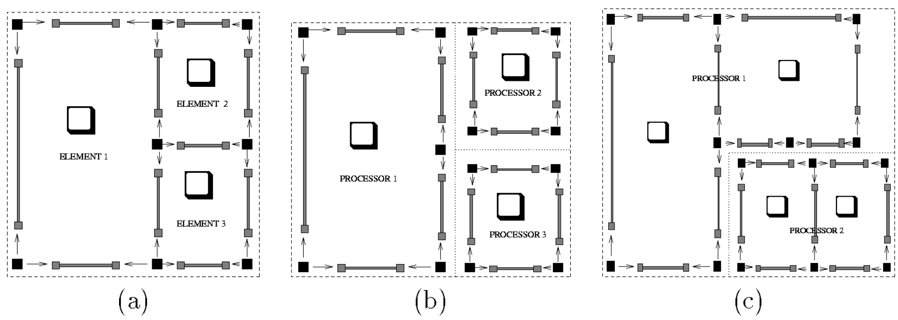


Figure 2 The structure of the mortar: (a) the global mortar for sequential implementation, (b) the local mortars for OEPP, (c) the local mortars for MEPP

formed in an initialization part of the program by all processors concurrently. The projection operators Q and Q^T are then created separately. For edges which have nonconforming matching, short communication with adjacent elements is necessary in order to obtain information about the size and coordinates of the nonconforming interfaces. Since the SEM generates a sparse linear positive definite system [MP89], the use of a conjugate gradient iterative solver is effective. After each iteration, neighboring elements exchange their edge information [EMD97]. Finally, a parallel I/O operation is performed by all processors concurrently to report the solution.

Multiple Elements per Processor Scheme (MEPP)

From the execution time analysis (in section 55), it is clear that OEPP has some weak points. Firstly, it is not practical since it requires the target machine to have the same number of processors as the number of elements. For a practical problem with a large number of elements, it is usually difficult to access that large number of processors. And if the number of processors is less than specified, then some processors will have

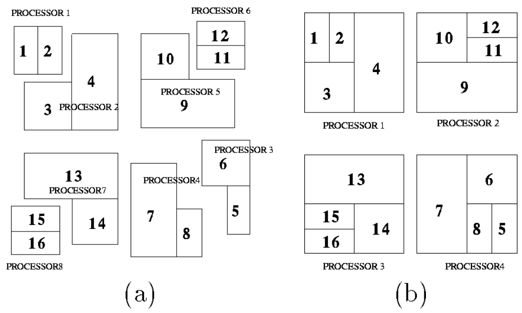


Figure 3 (a)Two and (b)four elements per processor mappings

twice or more load than others. Load-balancing is then upset and the operating system will spend all the time synchronizing MPI threads. Secondly, the OEPP scheme spends more time on communication than computation. To achieve good speedup and parallel efficiency, the computational load for each processor needs to be increased to get an optimal balance between computation and communication. The MEPP scheme is therefore developed as follows.

In the MEPP scheme, the same number of elements are assigned to each processor. Figure 3 shows the two and four elements per processor mappings for a 16 elements case.

As in the OEPP scheme, each element has an array recording neighborhood information. Using the information from element-processor mapping, it is known whether the adjacent elements are in the same processor or not. So in comparison with OEPP, MEPP has an additional array recording the information indicating whether the adjacent elements are located in the same processor or not. Thus, communication will only be performed between processors, not elements. After decomposition, a local mortar is established based on the global mortar configuration. Each processor retains only its corresponding portion of the global mortar. The mortar is shared by elements having adjacent interfaces in the same processor as shown in Figure 2(c). All processors do the initialization concurrently. Within each processor, every element does it sequentially. On each processor, the computation of Q and Q^T is performed sequentially for each element if needed. The short communication necessary to obtain information about size and coordinates of the nonconforming interface may be eliminated if the adjacent elements are on the same processor. The size of the linear system solved by conjugate gradient is proportional to NEP (number of elements per processor) times that of OEPP. In that case, the local computational load is increased NEP times (for a fixed number of C.G. iterations). After each iteration, only adjacent elements located in different processors need to exchange their mortar information. Otherwise, the mortar is updated locally, no message passing is needed. Thus communication cost is decreased.

Elements	Processors	Order	T_s(sec)	T_p(sec)	$\frac{T_c}{T_p}$	S	η
4	4	4	0.338	0.418	44.7%	0.809	20.2%
4	4	6	0.820	0.564	34.0%	1.454	36.3%
4	4	8	1.624	0.860	27.1%	1.888	47.2%
4	4	10	2.865	1.182	21.0%	2.424	60.6%
4	4	12	4.671	1.694	18.9%	2.757	68.9%
8	8	4	0.625	0.751	68.1%	0.832	10.4%
8	8	6	1.587	0.951	54.3%	1.669	20.9%
8	8	8	3.234	1.179	43.6%	2.743	34.3%
8	8	10	5.664	1.513	37.9%	3.744	46.8%
8	8	12	9.267	2.003	24.7%	4.627	57.8%

Table 1 Timing results for 250 conjugate gradient iterations for different problem sizes and order of approximation with the OEPP scheme.

Elements	Processors	Order	T_s(sec)	T_p(sec)	Speedup	Efficiency
16	8	4	1.254	0.843	1.488	18.6%
16	8	6	3.242	1.246	2.602	32.5%
16	8	8	6.528	1.900	3.436	42.9%
16	4	4	1.254	0.935	1.342	33.5%
16	4	6	3.242	1.585	2.045	51.1%
16	4	8	6.528	2.538	2.572	64.3%

Table 2 Timing results for 250 conjugate gradient iterations of 16 elements problem performed on 4 and 8 processors using MEPP scheme

Results and Evaluation

To test the correctness and effectiveness of the algorithm, error analysis and execution timings are performed on a test 2D Poisson problem with solution $u = sin(2\pi x)sin(2\pi y)$ on a square domain of side 4. Three cases of 4, 8, and 16 elements for the same computational domain decomposition are tested. Tables 1 and 2 show the speedup S and how parallel efficiency η varies with the number of elements and the order of approximation, where T_p and T_s refer to the parallel and sequential running time respectively and $\frac{T_c}{T_p}$ refers to the percentage of the communication time in the total running time. A fixed number of conjugate gradient iterations is used for time comparisons.

From these results, it can be observed that the speedup is not exactly proportional to the number of processors allocated to the problem. For a fixed problem, using more processors results in lower efficiency. Increasing the order of approximation for a fixed number of processors increases efficiency due to a decreasing percentage of the communication time in the overall running time.

Table 2 presents the speedup and parallel efficiency on 8 and 4 processors with two and four elements per processor respectively using the MEPP scheme. Obviously, the more processors used, the higher speedup should be attained. However, if a large part of the running time is spent on communication rather than computation, it will lead to lower speedup. Thus for the 4th order cases, the speedup does not change much even though twice as many as processors are used.

Conclusion and Future Research Direction

By the measured running time of the parallel algorithm, it can be observed that speedup increases with an increasing number of processors. However, the increase is not proportional to the number of processors. There is a limit beyond which speedup will not continue growing by adding more processors, because the communication time will dominate the execution time due to machine limitations.

Parallel efficiency decreases when more processors are used due to the increase of the interprocessor message exchange, but increases with increasing order of approximation. This means, to achieve good parallel efficiency, we need to load each processor with more computational work while minimizing the total interprocessor communication. Based on this observation, as well as the machine limitations (the number of processors available), the MEPP scheme is developed and comparatively high parallel efficiency is achieved. MEPP shows better performance than OEPP and will be the practical algorithm used in future research.

To increase computational load while decreasing communication work, an advanced data structure to deal with the connectivity information of the domain decomposition is required. This new structure should lead to an optimal element-processor mapping assigning adjacent elements to the same processor. To make the SEM more powerful and increase computational efficiency, dynamic mesh refinement and adaptive order of approximation assignment are needed. An adaptive sequential algorithm has been developed in [HM97]. In parallel, decomposition of the computational domain will be a dynamic problem which requires that the element-processor mapping also to be dynamic during execution. A parallel algorithm associated with dynamic mesh refinement and adaptive order of approximation will be the subject of future research.

REFERENCES

[BMP94] Bernardi C., Maday Y., and Patera A. T. (1994) A new nonconforming approach to domain decompositions: The mortar element method. *Pitman Research Notes in Mathematics* series 1, No. 299: 13.

[EMD97] Ebrat O., Mavriplis C., and Deane A. E. (1997) A parallel nonconforming spectral element algorithm on the Paragon. In *Proceedings of ISUG'97*.

[FHK+88] Fischer P. F., Ho L., Karniadakis G. E., Rønquist E. M., and Patera A. T. (1988) Recent advances in parallel spectral element simulation of unsteady incompressible flows. In *Proceedings of the Symposium on Advances and Trends in Computational Structural Mechanics and Computational Fluid Dynamics*. Washington, D.C.

[FP91] Fischer P. F. and Patera A. T. (1991) Parallel spectral element solution of the Stokes problem. *Journal of Computational Physics* 92: 380–421.

[GO77] Gottlieb D. O. and Orszag S. (1977) *Numerical Analysis of Spectral Methods: Theory and Applications*. No.26, SIAM. NSF-CBMS Monograph, Philadelphia.

[HM97] Hsu L. and Mavriplis C. (1997) A two-dimensional adaptive spectral element method. In *13th AIAA Computational Fluid Dynamics Conference*.

[MMP89] Maday Y., Mavriplis C., and Patera A. T. (1989) Non-conforming mortar element methods: Application to spectral discretizations. In Chan T. F., Glowinski R., Periaux J., and Widlund O. B. (eds) *Domain Decomposition Methods*, pages 392–418. SIAM, Philadelphia.

[MP89] Maday Y. and Patera A. T. (1989) Spectral element method for the Navier-Stokes equations. In Noor A. K. (ed) *State-of-the-Art Surveys in Computational Mechanics*, pages 71–143. New York.

[Pat84] Patera A. T. (1984) A spectral element method for fluid dynamics: Laminar flow in a channel expansion. *Journal of Computation Physics* 54(4): 468–488.

[SF73] Strang G. and Fix G. (1973) *An Analysis of The Finite Element Method*. Prentice-Hall.

56

NKS Methods for Compressible and Incompressible Flows on Unstructured Grids

D. K. Kaushik[1], D. E. Keyes[2], & B. F. Smith[3]

Introduction and Motivation

We review and extend to the compressible regime an earlier parallelization of an implicit incompressible unstructured Euler code [KKS98], and solve for flow over an M6 wing in subsonic, transonic, and supersonic regimes. While the parallelization philosophy of the compressible case is identical to the incompressible, we focus here on the nonlinear and linear convergence rates, which vary in different physical regimes, and on comparing the performance of currently important computational platforms.

Multiple-scale problems should be marched out at desired accuracy limits, and not held hostage to often more stringent explicit stability limits. In the context of inviscid aerodynamics, this means evolving transient computations on the scale of the convective transit time, rather than the acoustic transit time, or solving steady-

1 Computer Science Department, Old Dominion University, Norfolk, VA 23529-0162 and ICASE, NASA Langley Res. Ctr., Hampton, VA 23681-2199, Email: kaushik@cs.odu.edu. Supported in part by NASA under contract NAGI-1692 and by a GAANN fellowship from the U. S. Department of Education.

2 Computer Science Department, Old Dominion University, Norfolk, VA 23529-0162 and ICASE, NASA Langley Res. Ctr., Hampton, VA 23681-2199, Email: keyes@icase.edu. Supported in part by the National Science Foundation under grant ECS-9527169, by NASA under contracts NAS1-19480 and NAS1-97046, and by Argonne National Laboratory under contract 982232402.

3 Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL 60439-4844, Email: bsmith@mcs.anl.gov. Supported by U.S. Department of Energy, under Contract W-31-109-Eng-38.

Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

state problems with local CFL numbers approaching infinity. Whether time-accurate or steady, we employ Newton's method on each (pseudo-)timestep. The coupling of analysis with design in aerodynamic practice is another motivation for implicitness. Design processes that make use of sensitivity derivatives and the Hessian matrix require operations with the Jacobian matrix of the state constraints (i.e., of the governing PDE system); if the Jacobian is available for design, it may be employed with advantage in a nonlinearly implicit analysis, as well.

Implicit methods tend to contain global operations, which makes them challenging to parallelize. Nevertheless, the increasing resolution requirements of PDE analyses require access to the large memories provided by parallelism.

Parallel ΨNKS Solvers and Software

Our framework for an implicit PDE solution algorithm, with pseudo-timestepping to advance towards an assumed steady state, has the form: $(\frac{1}{\Delta t^{\ell}})\mathbf{u}^{\ell}+\mathbf{f}(\mathbf{u}^{\ell}) = (\frac{1}{\Delta t^{\ell}})\mathbf{u}^{\ell-1}$, where $\Delta t^{\ell} \rightarrow \infty$ as $\ell \rightarrow \infty$, where $\mathbf{u}$ represents the fully coupled vector of unknowns, and the steady-state solution satisfies $\mathbf{f}(\mathbf{u}) = 0$.

Each member of the sequence of nonlinear problems, $\ell = 1, 2, \ldots$, is solved with an inexact Newton method. The resulting Jacobian systems for the Newton corrections are solved with a Krylov method, relying directly only on matrix-free operations. The Krylov method needs to be preconditioned for acceptable inner iteration convergence rates, and the preconditioning can be the "make-or-break" feature of an implicit code. A good preconditioner saves time and space by permitting fewer iterations in the Krylov loop and smaller storage for the Krylov subspace. An additive Schwarz preconditioner [Cai89] accomplishes this in a concurrent, localized manner, with an approximate solve in each subdomain of a partitioning of the global PDE domain. Applying any preconditioner in an additive Schwarz manner tends to increase flop rates over the same preconditioner applied globally, since the smaller subdomain blocks maintain better cache residency, even apart from concurrency considerations. Combining a Schwarz preconditioner with a Krylov iteration method inside an inexact Newton method leads to a synergistic parallelizable nonlinear boundary value problem solver with a classical name: Newton-Krylov-Schwarz (NKS) [CKV97, GMTK98]. Combined with pseudo-timestepping, we write ΨNKS.

The basic philosophy of any efficient distributed computation is "owner computes", together with message merging and overlapping communication with computation where possible with split transactions. To minimize communication, each processor "ghosts" its stencil dependences on its neighbors' data. Grid functions are mapped from a global (user) ordering into contiguous local orderings (which, in unstructured cases are designed to maximize spatial locality for cache line reuse). Scatter/gather operations are created between local sequential vectors and global distributed vectors, based on connectivity patterns determined at runtime. Global NKS operations are thus translated into local tasks and communication tasks.

We employ the PETSc package [BGMS98], which features distributed data structures — index sets, vectors, and matrices — as fundamental objects. Iterative linear and nonlinear solvers, implemented in as data structure-neutral a manner as possible, are combinable modularly, recursively, and extensibly through a uniform application programmer interface. Portability is achieved through MPI, but message-passing detail is not required in user code. We use MeTiS [KK99] to partition the unstructured grid.

Incompressible and Compressible Flows

Our discretization routines are adapted from FUN3D, a tetrahedral unstructured grid code developed by W. K. Anderson and co-workers at NASA Langley for compressible [AB94] and incompressible [ARB95] Euler and Navier-Stokes equations. It is used in aeronautical and automotive external flow applications for analysis and (recently) design optimization. Unknown fields are defined at the vertices of tetrahedra, with a typical edge-connectivity to other vertices of approximately 15, when a control volume discretization is carried out on the polyhedral control volumes that are dual to the tetrahedra. The discrete $\mathbf{f}(\mathbf{u})$ is constructed in a conservative manner by looping over the edges of the tetrahedral grid, evaluating fluxes across the dual cell face pierced by the edge, and allocating the identical flux with opposite sign to the two control volumes on either side.

The incompressible version of the code employs four unknowns (pressure and three momenta) at each vertex and Chorin's artificial compressibility technique [Cho67]. The compressible version uses five unknowns (density, momenta, and internal energy) at each vertex. Roe's flux-difference splitting is used to discretize the convective terms. This requires a local eigendecomposition of the flux Jacobian to determine the characteristic directions and speeds of the waves passing through each control volume face element — an operation that is significantly more complex, computationally, for the compressible case than for the incompressible. Either first- or second-order fluxes can be computed, a feature that we exploit in different ways in the pseudo-transient NKS technique. We never employ higher than first-order fluxes in the preconditioner, since the preconditioner matrix is incompletely factored without pivoting, which is stabilized by the artificial viscosity of the upwinded first-order discretization. We always employ second-order fluxes in the residual evaluation as steady state is approached, and thereby obtain close to second-order discretization accuracy in the converged solution. We switch from first-order to second-order fluxes in the residual evaluation as a "continuation" device in compressible flow problems, in which rapid Newton convergence is difficult to achieve from a second-order discretization alone.

Besides the discretization-order switch, we employ the traditional continuation device of false timestepping when pursuing steady states, as in this paper. (False timestepping may also be employed as a subiteration, if necessary, in transient problems.) The timestep is advanced towards infinity by a power-law variation of the switched evolution/relaxation (SER) heuristic of Van Leer & Mulder [ML95]. To be specific, within each of the first-order and second-order phases of computation, we adjust the timestep according to

$$N_{CFL}^{\ell} = N_{CFL}^{0} \left(\frac{||f(u^0)||}{||f(u^{\ell-1})||} \right)^p ,$$

where p is normally unity, but damped down to 0.75 for robustness in cases in which shocks are expected to appear.

The overall solution process for nonlinear steady states has been found to be competitive with FAS multigrid in execution time when compared in specific two-dimensional external Euler flow contexts on vector computers [Key95]. We have not yet compared ΨNKS to FAS multigrid in the parallel three-dimensional context, but we know that, at a minimum, nonlinear grid sequencing is required for pseudo-transient NKS to scale acceptably as the mesh is refined.

Comparisons

In this paper, we present three sets of comparisons: (1) the parallel scalability of a transonic compressible flow problem on three different parallel machines (Cray T3E, SGI Origin, IBM SP), (2) the parallel scalability of four different flow problems on an Origin, and (3) the cache efficiency of two different flow problems on five common sequential processors. The first set of experiments exposes relative performance advantages of important machines. The second set exposes the relative difficulties of solving different regimes of flow. We consider incompressible, compressible subsonic, transonic, and mildly supersonic flows. The grid and geometry are held fixed. (This is somewhat unrealistic, since no special care is given to adaptively resolve shocks when they appear, but we are primarily interested in the algebraic aspects of the problem, not the discretization aspects.) The third set of experiments illustrates the sensitivity of per-processor floating-point performance to cache organization, and the organization of the "busy" data structures in the flow code.

Incompressible (Mach zero) flow is relevant to low velocity portions of flight envelope (and to automotive speeds), where the incompressible formulation offers considerable savings in storage and execution time over the compressible formulation while capturing the physics accurately for Mach numbers up to approximately 0.3. Our transonic (Mach 0.839) flow is a classical $\alpha = 3.06°$ "lambda shock" test case. This is relevant to airliner cruise conditions. Runs at Mach 0.3 verify the adequacy of the incompressible model and help interpolate trends. Runs at Mach 1.2 explore changing nonlinear and linear character of the discretized system.

Scalability Across Platforms

Cross-platform performance comparisons of a medium-size wing problem, closed with a symmetry plane inboard, are given in Table 1. The 16-processor run has approximately 22,369 vertices per processor; the 80-processor run has approximately 4,473. Decreasing volume-to-surface ratios in the subdomains and increasing depth of the global reduction spanning tree of the processors lead to gradually decaying efficiency. The convergence rate, in terms of pseudo-time steps to achieve a relative reduction of steady-state residual norm of 10^{-12}, is not much affected by increased partitioning. Exactly one Newton iteration is performed on each pseudo-time step, and the Krylov space restart size is 30, with a maximum of one restart. The slight differences in the numbers of timesteps arise from slightly different floating point arithmetic and/or noncommutative summation of global inner products, which lead to slightly different trajectories to the same steady state. The Origin is the fastest

Table 1 Transonic flow over M6 wing; fixed-size grid of 357,900 vertices.

No.	Cray T3E			IBM SP			SGI Origin		
Procs.	Steps	Time	Eff.	Steps	Time	Eff.	Steps	Time	Eff.
16	55	2406s	—	55	1920s	—	55	1616s	—
32	57	1331s	.90	57	1100s	.87	56	862s	.94
48	57	912s	.88	57	771s	.83	56	618s	.87
64	57	700s	.86	56	587s	.82	57	493s	.82
80	57	577s	.83	59	548s	.70	57	420s	.77

per processor (achieving the highest percentage of peak sequentially). The T3E has the best scalability, due to its torus network, which is fast compared to sequential processor performance. The full problem fits on smaller numbers of processors on the Origin, but "false" superunitary parallel scalability results due to cache-thrashing when too many vertices are assigned to a processor; 5K to 20K vertices per processor is reasonable for this code.

Scalability Across Flow Regimes

Trans-Mach convergence comparisons of the same problem are given in Table 2. Here efficiencies are normalized by the number of timesteps, to factor convergence degradation out of the performance picture and measure implementation factors alone (though convergence degradation with increasing granularity is modest). The number of steps increases dramatically with the nonlinearity of the flow, as Mach rises; however, the linear work per step decreases on average. Reasons for this include: more steps spent in the cheaper, first-order discretization phase of the continuation process, smaller CFL in early steps, and the increased hyperbolicity of the flow. The compressible Jacobian is far more complex to evaluate, but it also concentrates locality, achieving much higher computational rates than the corresponding incompressible Jacobian.

Memory Hierarchy Aspects

As observed in [KKS98] for the same unstructured flow code, data structure storage patterns for primary and auxiliary fields should adapt to hierarchical memory through: (1) interlacing, (2) blocking of degrees of freedom (DOFs) that are defined at the same point in point-block operations, and reordering (3) of edges for reuse of vertex data. Blocking allows efficient use of registers by reducing integer overhead and permitting hardwired unrolling of dense inner loops. Interlacing allows efficient reuse of cached operands, since components at the same point interact more intensely with each other than do the same fields at other points. Similarly, edge-reordering for vertex reuse reflects the fact that nearby points interact more intensely than distant points.

Table 3 illustrates these three effects on five processors with different cache (and

Table 2 Flow over M6 wing on SGI Origin; fixed-size grid of 357,900 vertices (1,431,600 DOFs incompressible, 1,789,500 DOFs compressible).

No. Procs.	Steps	Time per Step	Per-Step Speedup	Impl. Eff.	FcnEval Mflop/s	JacEval Mflop/s
Incompressible (4×4 blocks)						
16	19	41.6s	—	—	2,630	359
32	19	20.3s	2.05	1.02	5,366	736
48	21	14.1s	2.95	0.98	7,938	1,080
64	21	11.2s	3.71	0.93	10,545	1,398
80	21	10.1s	4.13	0.83	11,661	1,592
Subsonic (Mach 0.30) (5×5 blocks)						
16	17	55.4s	—	—	2,002	2,698
32	19	29.8s	1.86	0.93	3,921	5,214
48	19	20.5s	2.71	0.90	5,879	7,770
64	20	14.3s	3.88	0.97	8,180	10,743
80	20	12.7s	4.36	0.87	9,452	12,485
Transonic (Mach 0.84) (5×5 blocks)						
16	55	29.4s	—	—	2,009	2,736
32	56	15.4s	1.91	0.95	4,145	5,437
48	56	11.0s	2.66	0.89	5,942	7,961
64	57	8.7s	3.39	0.85	8,103	10,531
80	57	7.4s	3.99	0.80	9,856	12,774
Supersonic (Mach 1.20) (5×5 blocks)						
16	80	19.2s	—	—	2,025	2,679
32	81	10.6s	1.81	0.90	3,906	5,275
48	81	7.1s	2.72	0.91	6,140	7,961
64	82	5.8s	3.31	0.83	7,957	10,398
80	80	4.6s	4.20	0.84	9,940	12,889

processor) parameters. The original ordering is the native FUN3D ordering, which is based on vector register-oriented multicoloring. The combination of the three effects can enhance overall execution time by a factor of 2.5 on the Pentium to as much as 7.5 on the Power2. We are currently studying hardware counter profiles of similar runs to build more detailed causal explanations.

Conclusions and Future Directions

Unstructured implicit CFD solvers are amenable to scalable implementation, but careful tuning is needed to obtain the best product of per-processor efficiency and parallel efficiency. We [KKS98] and others have already solved problems of millions of vertices on hundreds of processors at rates in the tens of gigaflop/s, and we believe such performance is extensible, with further effort, to the teraflop/s regime.

In the future, we hope to enhance per-processor performance through improved spatial and temporal locality. We also hope to enhance parallel efficiency through algorithms that synchronize less frequently, and through multiobjective partitioning, which equidistributes communication work as well as computational work.

Acknowledgements

The authors thank W. Kyle Anderson of the NASA Langley Research Center for providing FUN3D. Satish Balay, Bill Gropp, and Lois McInnes of Argonne National Laboratory co-developed (with Smith) the PETSc software employed in this paper. Computer time was supplied by DOE (through Argonne and NERSC). Sandra Bittner graciously assisted in providing dedicated Origin access.

REFERENCES

[AB94] Anderson W. K. and Bonhaus D. L. (1994) An implicit upwind algorithm for computing turbulent flows on unstructured grids. *Computers and Fluids* 23: 1–21.

[ARB95] Anderson W. K., Rausch R. D., and Bonhaus D. L. (1995) Implicit/multigrid algorithms for incompressible turbulent flows on unstructured grids. AIAA 95-1740.

[BGMS98] Balay S., Gropp W. D., McInnes L. C., and Smith B. F. (1998) The Portable, Extensible Toolkit for Scientific Computing, version 2.0.22. `http://www.mcs.anl.gov/petsc`.

[Cai89] Cai X.-C. (1989) Some domain decomposition algorithms for nonselfadjoint elliptic and parabolic partial differential equations. Courant Institute TR 461.

[Cho67] Chorin A. (1967) A numerical method for solving incompressible viscous flow problems. *J. Comp. Phys.* 2: 12–26.

[CKV97] Cai X.-C., Keyes D. E., and Venkatakrishnan V. (1997) Newton-Krylov-Schwarz: An implicit solver for CFD. In *Proceedings of the Eighth International Conference on Domain Decomposition Methods*, pages 387–400. Wiley.

[GMTK98] Gropp W. D., McInnes L. C., Tidriri M. D., and Keyes D. E. (1998) Parallel implicit PDE computations: Algorithms and software. In *Proceedings of Parallel CFD'97*, pages 333–344. Elsevier.

[Key95] Keyes D. E. (1995) Aerodynamic applications of Newton-Krylov-Schwarz solvers. In *Proceedings of the 14th International Conference on Numerical Methods in Fluid Dynamics*, pages 1–20. Springer.

[KK99] Karypis G. and Kumar V. (1999) A fast and high quality schema for partitioning irregular graphs. *SIAM Journal of Scientific Computing* 20(1): 359–392.

[KKS98] Kaushik D. K., Keyes D. E., and Smith B. F. (1998) On the interaction of architecture and algorithm in the domain-based parallelization of an unstructured grid incompressible flow code. In *Proceedings of the Tenth International Conference on Domain Decomposition Methods*, pages 311–319. AMS.

[ML95] Mulder W. and Leer B. V. (1995) Experiments with implicit upwind methods for the Euler equations. *J. Comp. Phys.* 59: 232–246.

Table 3 Flow over M6 wing; fixed-size grid of 22,677 vertices (90,708 DOFs incompressible; 113,385 DOFs compressible). Activation of a layout enhancement is indicated by a "×" in the corresponding column. Improvement ratios are averages over the entire code; different subroutines benefit to different degrees. The F77 and C compilers are the vendor's in each case, except for the Pentium, where versions 5.0 of Visual Fortran and C++ are used.

Enhancements			Results			
Field Interlacing	Structural Blocking	Edge Reordering	Incompressible Time/Step	Incompressible Ratio	Compressible Time/Step	Compressible Ratio
Alpha 21164, 450MHz, cache: 8KB data / 8KB instr/ 96KB L2						
			153.2s	—	216.8s	—
×			67.8s	2.26	93.8s	2.31
×	×		56.0s	2.74	72.1s	3.00
		×	55.7s	2.75	91.0s	2.38
×		×	38.9s	3.94	58.3s	3.72
×	×	×	29.2s	5.24	40.8s	5.31
IBM P2SC ("thin"), 120MHz, cache: 128KB data / 32 KB instr						
			165.7s	—	237.6s	—
×			62.1s	2.67	85.8s	2.77
×	×		50.0s	3.31	65.7s	3.62
		×	43.3s	3.82	67.5s	3.52
×		×	33.5s	4.95	50.8s	4.68
×	×	×	22.1s	7.51	32.2s	7.37
MIPS R10000, 250MHz, cache: 32KB data / 32KB instr / 4MB L2						
			83.6s	—	140.0s	—
×			36.1s	2.31	57.5s	2.44
×	×		29.0s	2.88	43.1s	3.25
		×	29.2s	2.86	59.1s	2.37
×		×	23.4s	3.57	35.7s	3.92
×	×	×	16.9s	4.96	24.5s	5.71
Intel Pentium II (NT), 400MHz, cache: 16KB data / 16KB instr / 512KB L2						
			70.3s	—	108.5s	—
×			44.1s	1.59	70.1s	1.55
×	×		37.4s	1.88	57.3s	1.89
		×	43.8s	1.61	72.4s	1.50
×		×	34.0s	2.07	54.5s	1.99
×	×	×	27.6s	2.55	43.2s	2.51
Sun UltraSPARC II, 300MHz, cache: 2MB external						
			120.5s	—	185.0s	—
×			61.6s	1.96	86.3s	2.14
×	×		50.8s	2.37	70.9s	2.61
		×	51.0s	2.36	103.1s	1.79
×		×	37.8s	3.19	55.7s	3.32
×	×	×	28.5s	4.22	42.1s	4.39

57

Block preconditioners for nonsymmetric saddle point problems

Piotr Krzyżanowski[1]

Introduction

We discuss a class of methods for preconditioning nonsymmetric, indefinite saddle point problems arising from mixed finite element discretization of partial differential equations, in particular, the linearized Navier–Stokes equation. The corresponding matrix has block structure

$$\begin{pmatrix} A & B^T \\ B & -C \end{pmatrix},$$

where the matrix block A is positive definite, yet not necessarily symmetric, and C is nonnegative. We present new mathematical results concerning block diagonal and block triangular preconditioners based on *symmetric, positive definite* blocks. In both cases, the convergence of iterative method is independent of the mesh parameter h. Our analysis is also valid for inexact preconditioning blocks. Detailed proofs of theorems discussed here may be found in [Krz97].

Block preconditioners for saddle point problems were discussed by many authors before, see for example [D'y87], [BP88], [BP90], [BP97], [RW92], [SW94], [Elm96b], [ES96], [ESW97], [Elm96a], [Kla98b], [Kla98a], [KS97]. The approach we propose gives an application programmer a great opportunity to reuse, in an efficient way, existing very powerful methods (or software) like the domain decomposition or the multigrid method for symmetric positive definite problems. We believe that in certain cases this may be a more robust approach than to use custom domain decomposition preconditioners developed especially for the whole

1 Warsaw University, Institute of Applied Mathematics, Banacha 2, 02–097 Warszawa, Poland
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

saddle point problem.

General assumptions

The generic constant "Const", which appears later in this paper is independent of the parameter h.

Let $\bar{V}, \bar{W}$ be real Hilbert spaces with scalar products denoted by $((\cdot,\cdot))$ and $(\cdot,\cdot)$, respectively. The norms in these spaces, induced by the inner products, will be denoted by $||\cdot||$ and $|\cdot|$. We consider a family of finite dimensional subspaces indexed by parameter $h \in (0,1)$, $V_h \subset \bar{V}$, and $W_h \subset \bar{W}$.

Let us introduce three continuous bilinear forms: $a : \bar{V} \times \bar{V} \to R$, $b : \bar{V} \times \bar{W} \to R$, $c : \bar{W} \times \bar{W} \to R$, and assume that $a(\cdot,\cdot)$ is $\bar{V}$–elliptic, i.e.

$$\exists \alpha > 0 \quad \forall v \in \bar{V} \qquad a(v,\, v) \geq \alpha ||v||^2 \tag{1}$$

and that $c(\cdot,\,\cdot)$ satisfies

$$\exists \gamma \geq 0 \quad \forall p \in \bar{W} \qquad c(p,\, p) \geq \gamma |p|^2 \tag{2}$$

(we allow $\gamma = 0$). Notice that $a(\cdot,\cdot)$ does not need to be symmetric. Throughout this paper we also assume that V_h and W_h satisfy the uniform LBB condition, see [GR86],

$$\exists \beta > 0 \quad \forall h \in (0,1) \quad \forall p \in W_h \qquad \beta |p| \leq \sup_{v \in V_h, v \neq 0} \frac{b(v,\, p)}{||v||}. \tag{3}$$

In what follows we consider preconditioners for a family of finite dimensional problems (we drop the subscript h for simplicity of notation)

Problem 1 *Find* $(u,p) \in V \times W$ *such that*

$$M \begin{pmatrix} u \\ p \end{pmatrix} \equiv \begin{pmatrix} A & B^* \\ B & -C \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} F \\ G \end{pmatrix}. \tag{4}$$

The operators in (4) are:

$$\begin{aligned} A &: V \to V, \quad ((Au,v)) = a(u,\, v) \quad \forall u,v \in V, \\ B &: V \to W, \quad (Bu,p) = b(u,\, p) \quad \forall u \in V, p \in W, \\ C &: W \to W, \quad (Cp,q) = c(p,\, q) \quad \forall p,q \in W, \end{aligned}$$

while the right hand side $F \in V, G \in W$ is defined through $((F,v)) \equiv \langle\langle f,v \rangle\rangle$ and $(G,w) \equiv \langle g,w \rangle$, where f,g are given continuous functionals on $\bar{V}$, $\bar{W}$, and $\langle\langle \cdot,\cdot \rangle\rangle$, $\langle \cdot,\cdot \rangle$ denote the duality pairing in $\bar{V}$, $\bar{W}$, respectively. B^* denotes the formal adjoint operator to B, i.e. $(Bu,p) = ((u,B^*p))$ for all $u \in V$, $p \in W$.

We introduce two more operators, $A_0 : V \to V$ and $J_0 : W \to W$. We assume that they are self-adjoint, their inverses are easy to apply, and there exist positive constants a_0, a_1, b_0, b_1, which are independent of h, such that

$$a_0((u,u)) \leq ((A_0 u,u)) \leq a_1((u,u)) \qquad \forall u \in V, \tag{5}$$

$$b_0(p,p) \leq (J_0 p,p) \leq b_1(p,p) \qquad \forall p \in W. \tag{6}$$

The product space $V \times W$ is equipped with a natural scalar product $\langle \cdot, \cdot \rangle$,

$$\left\langle \begin{pmatrix} u \\ p \end{pmatrix}, \begin{pmatrix} v \\ q \end{pmatrix} \right\rangle = ((u,v)) + (p,q),$$

however, we are going to analyse the preconditioned problem using custom inner product $[\cdot, \cdot]$, which depends on the preconditioner being used:

$$\left[\begin{pmatrix} u \\ p \end{pmatrix}, \begin{pmatrix} v \\ q \end{pmatrix} \right] = \left\langle \begin{pmatrix} A_0 & 0 \\ 0 & J_0 \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix}, \begin{pmatrix} v \\ q \end{pmatrix} \right\rangle .$$

Lemma 1 *Under the above assumptions,*

$$\begin{array}{llll} ||A||_{V\to V}, & ||B||_{V\to W}, & ||C||_{W\to W}, & \\ ||A^*||_{V\to V}, & ||B^*||_{W\to V}, & ||C^*||_{W\to W}, & \\ ||A_0||_{V\to V}, & ||J_0||_{W\to W}, & ||A_0^{-1}||_{V\to V}, & ||J_0^{-1}||_{W\to W}, \\ ||M||_{V\times W\to V\times W}, & ||M^*||_{V\times W\to V\times W} \leq Const. & & \end{array}$$

Preconditioners leading to symmetric, positive definite problems

In this section we are going to extend the results obtained previously for symmetric saddle point problems e.g. in [D'y87] and [BP90]. This approach is of normal equations type, which influences the convergence speed of the method; though, it has certain integesting advantages, too. The transformed system allows for using the conjugate gradients (PCG) method, which is less memory consuming than, e.g. GMRES; the transformed system is well conditioned, so the number of iterations required to damp the error by a given factor is independent of h. Moreover, no scaling is needed for A_0, in contrast to results for preconditioner for the GMRES in the next section and the analysis remains valid for A only V^0-elliptic, see Remark 2.

Block diagonal preconditioner

Using a block diagonal preconditioner

$$M_D = \begin{pmatrix} A_0 & 0 \\ 0 & J_0 \end{pmatrix},$$

we transform system (4) into the following one:

Problem 2 *Find* $(u,p) \in V \times W$ *such that*

$$M_D^{-1} M^* M_D^{-1} M \begin{pmatrix} u \\ p \end{pmatrix} = M_D^{-1} M^* M_D^{-1} \begin{pmatrix} F \\ G \end{pmatrix}.$$

Clearly, the operator $P = M_D^{-1} M^* M_D^{-1} M$ is selfadjoint with respect to the inner product $[\cdot, \cdot]$. The following theorem guarantees P is well conditioned with respect to h:

Theorem 1 *There exist constants* $0 < m_0 \leq m_1$*, independent of* h*, such that*

$$m_0 \left[\begin{pmatrix} u \\ p \end{pmatrix}, \begin{pmatrix} u \\ p \end{pmatrix} \right] \leq \left[P \begin{pmatrix} u \\ p \end{pmatrix}, \begin{pmatrix} u \\ p \end{pmatrix} \right] \leq m_1 \left[\begin{pmatrix} u \\ p \end{pmatrix}, \begin{pmatrix} u \\ p \end{pmatrix} \right]. \qquad (7)$$

Proof. Observe that

$$\left[\mathcal{P}\begin{pmatrix} u \\ p \end{pmatrix}, \begin{pmatrix} u \\ p \end{pmatrix}\right] = ((A_0^{-1}(Au + B^*p), Au + B^*p)) + (J_0^{-1}(Bu - Cp), Bu - Cp). \qquad (8)$$

Defining $\tilde{f} = Au + B^*p$ and $\tilde{g} = Bu - Cp$, we obviously have that (u, p) satisfies

$$\begin{aligned} Au + B^*p &= \tilde{f} \\ Bu - Cp &= \tilde{g}, \end{aligned}$$

so from (5), (6) and the stability result for Problem 1, see [GR86], we obtain

$$\left[\mathcal{P}\begin{pmatrix} u \\ p \end{pmatrix}, \begin{pmatrix} u \\ p \end{pmatrix}\right] \geq \text{Const}\left(||u||^2 + |p|^2\right).$$

This yields the lower bound. The upper bound follows from (8) and Lemma 1. *QED*

Remark 1 *From practical point of view it is important to observe that when using the PCG method for solving Problem 2, one has to solve* only one *system with* M_0 *per inner product* $[Px, y]$.

Block triangular preconditioner

It is also possible to construct a symmetric preconditioner for the operator M, based on a block lower triangular matrix

$$M_T = \begin{pmatrix} A_0 & 0 \\ B & J_0 \end{pmatrix}.$$

We transform system (4) into equivalent one,

$$P\begin{pmatrix} u \\ p \end{pmatrix} \equiv M^* L_0^* K_0^{-1} L_0 M \begin{pmatrix} u \\ p \end{pmatrix} = M^* L_0^* K_0^{-1} L_0 \begin{pmatrix} F \\ G \end{pmatrix},$$

where

$$L_0^* K_0^{-1} L_0 \equiv \begin{pmatrix} I & A_0^{-1}B^* \\ 0 & -I \end{pmatrix}\begin{pmatrix} A_0^{-1} & 0 \\ 0 & J_0^{-1} \end{pmatrix}\begin{pmatrix} I & 0 \\ BA_0^{-1} & -I \end{pmatrix}.$$

Observe that $K_0^{-1}L_0$ is nothing else but M_T^{-1} and that P is symmetric with respect to the scalar product $\langle \cdot, \cdot \rangle$.

Theorem 2 *There exist constants* $0 < c_0 \leq c_1$ *independent of parameter* h, *such that*

$$c_0 \left\langle \begin{pmatrix} u \\ p \end{pmatrix}, \begin{pmatrix} u \\ p \end{pmatrix} \right\rangle \leq \left\langle P\begin{pmatrix} u \\ p \end{pmatrix}, \begin{pmatrix} u \\ p \end{pmatrix} \right\rangle \leq c_1 \left\langle \begin{pmatrix} u \\ p \end{pmatrix}, \begin{pmatrix} u \\ p \end{pmatrix} \right\rangle$$

for all $(u, p) \in V \times W$.

In contrast to the diagonal preconditioner, the triangular symmetrized preconditioner requires three applications of A_0^{-1} per inner product.

Remark 2 *From theoretical point of view it is worth noticing that Theorems 1 and 2 hold if (1) is replaced by a weaker condition,*

$$a(u, u) \geq \alpha ||u||^2 \qquad \forall v \in \ker B.$$

Preconditioner for the GMRES method

It is well known that after symmetrizing the system, its condition number increases. This fact is also indicated by numerical experiments below. Instead, we may use efficiently preconditioned GMRES method for our problem. If the preconditioner works so good that we obtain satisfactory approximation after few iterations, then the GMRES memory consumption is less painful, and we can benefit from its better convergence properties.

Let us consider block triangular preconditioner,

$$M_T = \begin{pmatrix} A_0 & 0 \\ B & -J_0 \end{pmatrix},$$

which was previously discussed e.g. in [ES96], [ESW97] and [Kla98a], [KS97].

Theorem 3 *If A_0 is scaled so that the symmetric part*[2] *A_{symm} of A together with $A^*A_0^{-1}A$ satisfy*

$$\Big(\big((A_{symm} - \frac{1}{2}A^*A_0^{-1}A)u, u\big)\Big) \geq \delta \cdot ((A_0 u, u)) \qquad \forall u \in V, \tag{9}$$

for some positive constant δ independent of h, then the convergence rate of the GMRES for $M_T^{-1}M$ is independent of h.

Proof.

Observe that

$$\left[M_T^{-1}M\begin{pmatrix} u \\ p \end{pmatrix}, \begin{pmatrix} u \\ p \end{pmatrix}\right] \geq (((A_{symm} - \frac{1}{2}A^*A_0^{-1}A)u,\, u)) + \frac{1}{2}|||B^*p|||^2_{A_0^{-1}},$$

with $|||v|||^2_{A_0^{-1}} \equiv ((A_0^{-1}v,\, v))$. Obviously, $|||B^*p|||^2_{A_0^{-1}} \geq \mathrm{Const}(J_0 p,\, p)$, by (5), (6) and the LBB condition (3), see [BP88].

Therefore, if only A_0 satisfies (9) with δ independent of h, then we can conclude that

$$\left[M_T^{-1}M\begin{pmatrix} u \\ p \end{pmatrix}, \begin{pmatrix} u \\ p \end{pmatrix}\right] \geq \mathrm{Const}\left[\begin{pmatrix} u \\ p \end{pmatrix}, \begin{pmatrix} u \\ p \end{pmatrix}\right].$$

On the other hand, from Lemma 1 we have

$$\left[M_T^{-1}M\begin{pmatrix} u \\ p \end{pmatrix}, M_T^{-1}M\begin{pmatrix} u \\ p \end{pmatrix}\right] \leq \mathrm{Const}\left[\begin{pmatrix} u \\ p \end{pmatrix}, \begin{pmatrix} u \\ p \end{pmatrix}\right].$$

In view of the last two inequalities, the theorem follows from [EES83]. *QED*

Remark 3 *Let us notice that the scaling requirement is similar to that of Bramble and Pasciak [BP88] for the triangular preconditioner in the symmetric case.*

A good scaling factor for A_0^{-1} may be obtained, by finding certain extreme eigenvalues of two generalized eigenproblems. Also observe that for $A = A^* = A_0$ ("exact" preconditioning block), (9) holds with $\delta = 1/2$ without any scaling.

---

2 By definition, $A_{symm} = \frac{A+A^*}{2}$.

Numerical experiments

Our test problem was to solve the homogeneous Dirichlet boundary value problem for the Oseen equations in $\Omega = (-1,1)\times(-1,1)$

$$\begin{aligned} -\Delta u + (k\cdot\nabla)u + \nabla p &= f,\\ \operatorname{div} u &= 0, \end{aligned}$$

discretized with Q_2–Q_1 Taylor–Hood rectangular finite elements. The mesh was uniform in both directions, with nx inner pressure nodes along the x-axis and also y-axis. The function $k(\cdot,\cdot)$ was defined as a simple vortex,

$$k(x,y) = \begin{pmatrix} 2y(1-x^2) \\ -2x(1-y^2) \end{pmatrix}.$$

For f, we took a random vector with elements uniformly distributed between $(-1,1)$. All algorithms were implemented using Krylov iterative solvers and additive Schwarz preconditioners from the PETSc library [SGM97]. We examined three different solver/preconditioner configurations: GMRES(30) method with block diagonal preconditioner (GMRES/diag); GMRES(30) method with block triangular preconditioner (GMRES/triang); the symmetrized conjugate gradient method with block diagonal preconditioner (CG/symm). The stopping criterion was to reduce the initial residual by a factor of 10^6.

We report on two choices of preconditioning blocks. The first was to test the "best" (yet still symmetric, positive definite) preconditioners, i.e. A_0 = Laplacian, J_0 = mass matrix. The second choice was to use inexact preconditioners for A_0, J_0, namely the additive Schwarz method (with no coarse space), with standard black-box decomposition into 6 rectangular subdomains with small and fixed overlap (2 nodes) provided by PETSc. The results are presented in Figure 1 and Table 1.

The experiments confirm theoretical results, showing that the number of iterations reflects the quality of the preconditioning blocks being used. When using the symmetric part of relevant operators as the preconditioning blocks, the number of iterations is virtually independent of the mesh size.

Table 1 Flops count relative to GMRES/triang for different solver/preconditioner strategies. Iteration count in brackets.

	nx	DOF	GMRES/diag	GMRES/triang	CG/symm
"best"	20	4182	1.51 (39)	1.00 (21)	2.14 (46)
ASM	20	4182	1.18 (128)	1.00 (76)	1.62 (127)

For additive Schwarz preconditioning blocks, according to [SBG96], the number of iterations with fixed small overlap and no coarse grid, should increase as the square root of nx, and this is approximately what we actually see. Theorems 1 and 3 ensure that the convergence rate is fully independent of nx, if two-level additive Schwarz method was used. The superiority of GMRES/triang was not so apparent when additive Schwarz method preconditioners were used: the number of iterations required by symmetrized CG was competitive to that of GMRES/triang.

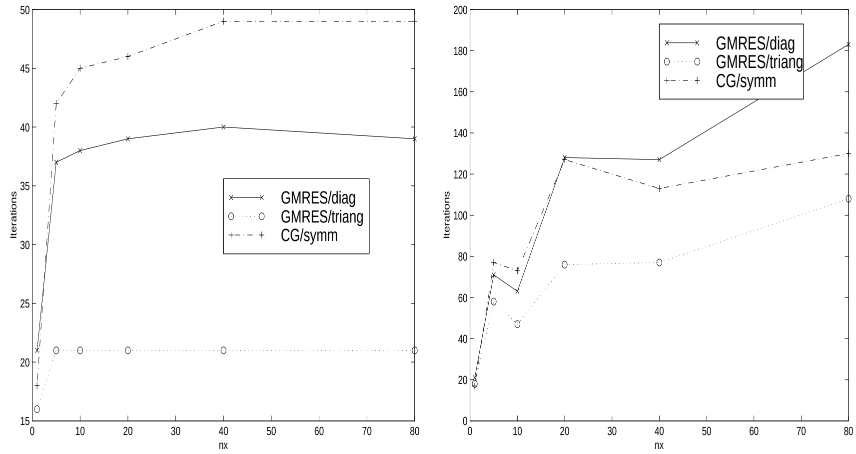


Figure 1 The growth of iterations as a function of problem size for different solvers and preconditioning strategies. **Left**: A_0 = Laplacian, J_0 = Mass matrix. **Right**: additive Schwarz preconditioners (small overlap, no coarse grid).

Acknowledgements

I would like to thank Prof. Maksymilian Dryja and Prof. Krzysztof Moszyński for their valuable remarks and comments. This research was partially sponsored by Polish Scientific Research Committee grant for PhD students, KBN 8T11F01209.

REFERENCES

[BP88] Bramble J. and Pasciak J. (1988) A preconditioning technique for indefinite problems resulting from mixed approximation of elliptic problems. *Math. Comp.* 50(181): 1–17.

[BP90] Bramble J. and Pasciak J. (1990) A domain decomposition technique for Stokes problems. *App. Num. Math.* 6(4): 251–261.

[BP97] Bramble J. and Pasciak J. (1997) Iterative techniques for time dependent Stokes problems. *Comput. Math. Appl.* 33(1-2): 13–30. Approximation theory and applications.

[D'y87] D'yakonov E. (1987) On iterative methods with saddle operators. *Soviet Math. Dokl.* 35(1): 166–170.

[EES83] Eisenstat S., Elman H., and Schultz M. (1983) Variational iterative methods for nonsymmetric systems of linear equations. *SIAM J. Numer. Anal.* 20(2): 345–357.

[Elm96a] Elman H. (November 1996) Preconditioning for the steady-state Navier–Stokes equations with low viscosity. Technical Report CS-TR-3712, University of Maryland.

[Elm96b] Elman H. (September 1996) Multigrid and Krylov subspace methods for the discrete Stokes equations. In Melson N., Manteuffel T., McCormick S., and Douglas C. (eds) *Seventh Copper Mountain Conference on Multigrid*

Methods. NASA Conference Publication 3339, Hampton, Virginia.

[ES96] Elman H. and Silvester D. (1996) Fast nonsymmetric iterations and preconditioning for Navier–Stokes equations. *SIAM J. Sci. Comput.* 17(1): 33–46.

[ESW97] Elman H. C., Silvester D. J., and Wathen A. J. (1997) Iterative methods for problems in computational fluid dynamics. In *Iterative methods in scientific computing (Hong Kong, 1995)*, pages 271–327. Springer, Singapore.

[GR86] Girault V. and Raviart P. (1986) *Finite Element Method for Navier–Stokes Equations. Theory and Algorithms*. Springer-Verlag, Berlin, Heidelberg, New York.

[Kla98a] Klawonn A. (1998) Block-triangular preconditioners for saddle point problems with a penalty term. *SIAM J. Sci. Comput.* 19(1): 172–184 (electronic). Special issue on iterative methods (Copper Mountain, CO, 1996).

[Kla98b] Klawonn A. (1998) An optimal preconditioner for a class of saddle point problems with a penalty term. *SIAM J. Sci. Comput.* 19(2): 540–552 (electronic).

[Krz97] Krzyżanowski P. (1997) A note on preconditioning nonsymmetric saddle point problems. Technical Report RW 97-09, Institute of Applied Mathematics and Mechanics, Warsaw University.

[KS97] Klawonn A. and Starke G. (1997) Block triangular preconditioners for nonsymmetric saddle point problems: Field-of-values analysis. Technical Report 04/97-N, Westfälishe Wilhelms-Universität Münster, Germany.

[RW92] Rusten T. and Winther R. (1992) A preconditioned iterative method for saddle point problems. *SIAM J. Matr. Anal. Appl.* 13: 887–904.

[SBG96] Smith B., Bjorstad P., and Gropp W. (1996) *Domain Decomposition. Parallel Multilevel Methods for Elliptic Partial Differential Equations*. Cambridge University Press, Cambridge.

[SGM97] Smith B., Gropp W., and McInnes L. (1997) PETSc 2.0 users manual. Technical Report ANL-95/11, Argonne National Laboratory. Also available via ftp://www.mcs.anl/pub/petsc/manual.ps.

[SW94] Silvester D. and Wathen A. (1994) Fast iterative solution of stabilized Stokes systems, Part II: Using general block preconditioners. *SIAM J. Numer. Anal.* 31(5): 1352–1367.

58

Ordering techniques for convection dominated problems on unstructured three-dimensional grids

S. LE BORNE[1]

INTRODUCTION

For convection dominated problems in two spatial dimensions, multigrid methods using a simple Gauß-Seidel method as a smoother can lead to excellent results if the unknowns are ordered appropriately. However, the difficulties with constructing robust multigrid solvers in three dimensions are much greater: Many ordering strategies that have been proposed for two-dimensional problems make explicit use of the planarity of the underlying graphs and are not directly applicable to three-dimensional problems.

In this paper we introduce the concept of interior surfaces in tetrahedral grids which we use to construct an ordering for the unknowns. The proposed ordering algorithm produces a decomposition of the unknowns into disjoint blocks so that block iterative methods may be applied.

We study problems where the matrix A arises from the finite element discretisation of the convection-diffusion equation

$$-\Delta u + (b \cdot \nabla)\, u \quad = \quad f \qquad \text{in } \Omega \subset \mathbb{R}^n. \tag{1}$$

Here n denotes the dimension (2 or 3) and suitable boundary conditions are assumed. Let $\mathcal{T}$ be the underlying triangulation of the domain Ω. We mention that in the numerical solution of

1 Praktische Mathematik, University Kiel, Hermann-Rodewald-Str. 3, 24098 Kiel. email: sg@numerik.uni-kiel.de
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

systems of equations, such as the Navier-Stokes equations, solving a set of scalar convection-diffusion equations can be required as a subproblem (see e.g., [BWY90], [BS97] or [HGMM98] for smoothers for the Stokes problem that are based on a block ILU decomposition and hence depend on the ordering of the unknowns).

Here we will use the graph of a matrix to develop and illustrate ordering techniques and thus recall the definition of the graph of a matrix: The matrix graph $G = G(A)$ consists of the vertex set $V = V(G)$ and the edges $E = E(G) = \{(i,j) \in V \times V : \ a_{ij} \neq 0\}$. For continuous piecewise linear basis functions with the standard nodal basis, the vertices and edges of the triangulation correspond, respectively, to the vertices and edges of the matrix graph.

Typically, the size $|a_{ij}|$ of the matrix entries varies considerably over the grid due to the convective term. For the proposed ordering strategies we neglect the small matrix entries and define the graph of dominant entries as the graph with the reduced edge set $E_0 = \{(i,j) \in E : |a_{ij}| \text{ 'is large'}\}$. Typically, at most one of the entries $|a_{ij}|$ and $|a_{ji}|$ is large, thus making the reduced graph a directed graph. Henceforth, all graphs are assumed to be directed graphs. There are several reasonable classifications for an entry to be large, one of which defines $|a_{ij}|$ as 'large' if $|a_{ij}| > \kappa\ |a_{ii}|$ with a parameter $\kappa \in (0,1)$.

The task is to find a permutation P such that PAP^T is easier to solve. The ideal case would have PAP^T an upper (or lower) triangular matrix for which a simple backward (or forward) substitution process can be employed. With respect to the matrix graph, this situation corresponds to an acyclic graph.

However, many applications involve cyclic flows that may result in cyclic graphs, cf. [GDN95]. In this case, several approaches have been suggested for two-dimensional problems where the graph is planar. These ordering algorithms provide a numbering strategy that produces a block structure which can be used for block iterative methods. Numerical results for the two-dimensional convection-diffusion equation as well as for the Stokes equations with a convective term have been provided in [GP97], [HGMM98] and [HP97]. Additionally, ordering techniques to improve multigrid convergence have also been developed and applied by other authors, e.g., in [RR96], [Tur97].

In three spatial dimensions we also have a structure consisting of vertices and edges but additionally, we can involve the faces of the tetrahedra. A collection of neighbouring faces describes an interior surface in the tetrahedron. In turn, an ordering of these surfaces in addition to an ordering within each surface defines an ordering for the vertices in the tetrahedral grid.

Following, we provide the details needed to accomplish the task of constructing interior surfaces. In section 58 we briefly review (prerequisites for) ordering strategies that have been proposed for planar graphs and point out the difficulties in generalising the concepts to graphs that are not planar. Section 58 deals with ordering concepts for the three-dimensional case.

THE TWO-DIMENSIONAL CASE

In preparation for the three-dimensional case we now review some of the ideas that have been previously considered for two dimensions. The motivation will not change: find a permutation P such that PAP^T has a structure that is better suited for applying linear equation algorithms.

Given a planar graph $G(V,E)$, the edges associated with a vertex can be ordered periodically with respect to their defining angles. A useful property of graphs that arise from the discretisation of the convection-diffusion equation is the *one-flow-direction condition*: for all vertices $p \in V$ there is a (unique) sequence $\{e_1, \cdots, e_m\}$ of outward edges and a sequence $\{e_{m+1}, \cdots, e_k\}$ of inward edges enumerated in the counter-clockwise direction.

For the two-dimensional case, several ordering techniques such as *ordering with concentric cycles*, *feedback vertex set ordering* or *parallel ordering* have been developed for planar graphs that fulfil the one-flow-direction condition.

For further details on these ordering algorithms as well as numerical results for the convection-diffusion equation and the Stokes equations with a convective term, the interested reader is referred to [GP97], [Hac97], [HGMM98], [HP97].

However, for the three-dimensional case it unfortunately remains unclear how to order the edges adjacent to a vertex or how to characterise minimal or concentric cycles. For this reason the concepts that apply to orderings for two-dimensional graphs cannot be generalised to three-dimensional graphs in a straightforward way.

In the following section new concepts are proposed for the three-dimensional case. The goal is to identify subgraphs that possess properties of planar graphs so that ordering strategies for planar graphs can be applied.

THE THREE-DIMENSIONAL CASE

In the three-dimensional case, looking for graph cycles may not be the best concept: As seen in section 58, minimal and concentric cycles produce favourable orderings. In three dimensions, it is not clear how to characterise cycles as minimal or concentric. Alternatively, we propose to decompose the graph into a set of disjoint planar subgraphs that might contain cycles, and then apply ordering techniques for planar graphs on each subgraph separately.

In order to determine suitable subgraphs, we must first introduce the concept of interior surfaces. For example, if the underlying triangulation consists of tetrahedra, an interior surface can be described through a combination of neighbouring faces (of the tetrahedra). If such a surface is constructed carefully its vertices and edges define a planar graph in which we can apply the ordering algorithms of section 58. The cylinder depicted in Figure 1 illustrates this concept. Interior to the cylinder we have detailed the cyclic flow, however, the tetrahedral triangularisation has been left out. Slicing through the cylinder in a horizontal direction (i.e. parallel to the top and bottom but orthogonal to the cylindrical walls,) produces circular surfaces in which the flow can be seen to be concentric cycles. On these surfaces, the ordering algorithms for planar graphs are applicable.

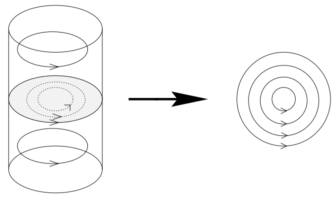


Figure 1 Cyclic flow in a cylinder

Before an algorithm can be developed that will construct such interior surfaces we first must establish terminology for the description of edges, faces and surfaces.

Surfaces and surface graphs

In the three-dimensional case we are dealing with tetrahedral grids, $\mathcal{T}(V, E, F)$, consisting of a set of vertices, $V(\mathcal{T})$, and a set of edges, $E(\mathcal{T})$ and, additionally, we have a set of faces, $F(\mathcal{T})$. We call two faces *neighbours* if they have a common edge (note that a common vertex is not sufficient).

Definition 1 *A subset $\mathcal{S}$ of $\mathcal{T}$ is called a* surface *if it possesses the following three properties:*

(i) *The* degree *of every edge in $\mathcal{S}$, i.e., the number of faces in $\mathcal{S}$ sharing this edge, is at most two.*
(ii) *$\mathcal{S}$ is* connected*, i.e., for any $f, g \in F(\mathcal{S})$ there is a path $f = f_0, f_1, \cdots, f_k = g$ of faces $f_i \in F(\mathcal{S})$ where f_i and f_{i+1} are neighbours for all $i = 1, \cdots, k-1$.*
(iii) *For any $f, g \in F(\mathcal{S})$ with a common vertex p there is a path $f = f_0, \cdots, f_k = g$ of faces $f_i \in F(\mathcal{S})$ each having p as a vertex where f_i and f_{i+1} are neighbours for all $i = 1, \cdots, k-1$.*

Figure 2 shows an example for a surface and three examples that violate one or more of the required properties.

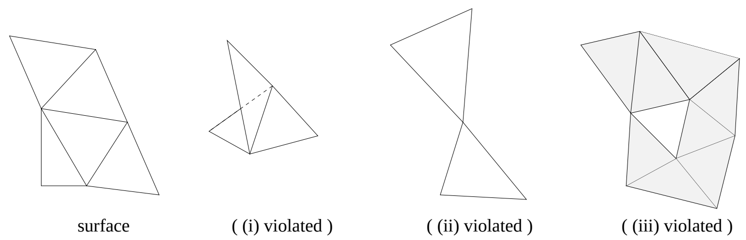


Figure 2 A surface and three violated situations

The *surface graph* $G(\mathcal{S}) = (V(\mathcal{S}), E(\mathcal{S}))$ is the graph consisting of vertices and edges on the surface. In the following we only consider orientable surfaces.

As in the two-dimensional case, for all vertices $v \in V(\mathcal{S})$ of an orientable surface $\mathcal{S}$, we can order the edges adjacent to v counter-clockwise with respect to the orientation. We note that this would not be the case without property *(iii)* above. Hence, we can determine whether a one-flow-direction condition holds for the vertices of the surface graph $G(\mathcal{S})$ which establishes an important prerequisite for applying the ordering techniques for planar graphs.

Flow surface

As illustrated in the introductory example in Figure 1, the goal is to construct surfaces that lie parallel to the flow while retaining the information about the flow, in particular concerning cyclic dependencies, is contained in the surface. To obtain such surfaces one begins with an arbitrary face and then iteratively attaches faces in the flow direction across the edges. In order to formalise this procedure we will define a function $\varphi : F(\mathcal{T}) \times \tilde{E}(\mathcal{T}) \to F(\mathcal{T})$ $(\tilde{E}(\mathcal{T}) \subseteq E(\mathcal{T}))$ that assign a face to a given face f across one of its edges e in the following way:

We assume that the convection direction b of the partial differential equation (1) is available. For an edge e with endpoints $v, w \in V(\mathcal{T})$, let $m = \frac{1}{2}(v + w)$ denote the edge midpoint. Define $\tilde{E}(\mathcal{T})$ to be the set of edges with degree greater than one that have a convection direction $b(m)$ with $\|b(m)\|_2 > \sigma > 0$ in the edge midpoint for a parameter $\sigma > 0$. Let $\tilde{E}_1(\mathcal{T}) \subseteq \tilde{E}(\mathcal{T})$ be the subset of edges for which the convection in the edge midpoint is not parallel to the edge itself, and let $\tilde{E}_2(\mathcal{T}) = \tilde{E}(\mathcal{T}) \setminus \tilde{E}_1(\mathcal{T})$. Then functions φ_f, $\varphi_b : E_1(\mathcal{T}) \to F(\mathcal{T})$ (the index 'f' refers to *forward*, 'b' to *backward*) can be defined as follows: Let $\gamma \in (0, \pi/2)$ be a given angle. For an edge $e \in \tilde{E}_1(\mathcal{T})$, let $\varphi_f(e)$ be the face adjacent to e that has the smallest angle to the plane spanned by the convection direction in the edge midpoint and the edge e, provided that the angle between $b(m)$ and this face is smaller than γ. Analogously, we define $\varphi_b(e)$ as the face adjacent to e that has the smallest angle to the plane spanned by the *negative* convection direction in the edge midpoint and the edge e, again provided that the angle between $-b(m)$ and this face is smaller than γ.

Note that in these definitions the orientation of the convection direction is of importance when making the decision whether an angle is α or $\pi - \alpha$. For example, in Figure 3, the angle between $b(m)$ and the face f is 0, whereas the angle between $b(m)$ and the face g is π. Hence we have $\varphi_f(e) = f$ and $\varphi_b(e) = g$. The angle γ was introduced to ensure that the face $\varphi_f(e)$ ($\varphi_b(e)$) is nearly parallel (up to an error smaller than γ) to the flow (negative flow).

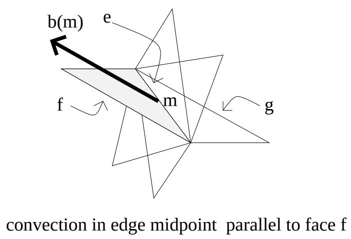


Figure 3 Illustration for the mappings φ_f and φ_b.

For an edge $e \in \tilde{E}_2(\mathcal{T})$ of a face f, the convection direction is by definition of $\tilde{E}_2(\mathcal{T})$ nearly parallel to the edge so that we cannot define faces in the flow direction as above. Instead, we define $\varphi_p : F(\mathcal{T}) \times \tilde{E}_2(\mathcal{T}) \to F(\mathcal{T})$ (index 'p' refers to *parallel*) where $\varphi_p(f, e)$ is defined to be the face with edge e that is closest to being *parallel* to f, i.e., that has an angle to f as close to π as possible, provided that the angle between this face and f is at least $\pi - \gamma$. Using the example given in figure 3, only changing the convection direction $b(m)$ to be parallel to e so that $e \in \tilde{E}_2(\mathcal{T})$, we have $\varphi_p(g, e) = f$.

We define the sets

$$\begin{aligned} \tilde{F}_f(\mathcal{T}) &= \varphi_f(\tilde{E}_1(\mathcal{T})), \\ \tilde{F}_b(\mathcal{T}) &= \varphi_b(\tilde{E}_1(\mathcal{T})), \\ \tilde{F}_p(\mathcal{T}) &= \varphi_p(\tilde{F}_f(\mathcal{T}) \times \tilde{E}_2(\mathcal{T})) \cup \varphi_p(\tilde{F}_b(\mathcal{T}) \times \tilde{E}_2(\mathcal{T})), \\ \tilde{F}(\mathcal{T}) &= \tilde{F}_f(\mathcal{T}) \cup \tilde{F}_b(\mathcal{T}) \cup \tilde{F}_p(\mathcal{T}). \end{aligned}$$

The set $\tilde{F}(\mathcal{T})$ can be considered as the set of faces lying in the flow. The collection of faces in $\tilde{F}(\mathcal{T})$, however, does not typically represent a surface since an edge might have a degree higher than two. We next define a flow surface:

Definition 2 *Let $\alpha \in (0, \pi)$ be a given angle. An oriented surface $\mathcal{S}$ consisting of a set of faces $F(\mathcal{S}) \subseteq \tilde{F}(\mathcal{T})$ is called a* flow surface of minimal angle α *if, for every edge $e \in E(\mathcal{S})$ with adjacent faces f and g, the angle between these faces is greater than α, i.e., $\angle(f, g) \geq \alpha$.*

Surface algorithm

Presented is an algorithm that constructs flow surfaces. The intuitive description of the algorithm is as follows: Start with an arbitrary edge of $\tilde{E}_1(\mathcal{T})$ and determine the face $\varphi_f(e)$. Next append faces across the edges using the functions φ_f, φ_b or φ_p. The functions φ_f, φ_b or φ_p are combined into a single function by defining

$$\varphi \quad : \quad F(\mathcal{T}) \times \tilde{E}(\mathcal{T}) \to F(\mathcal{T}),$$

$$\varphi(f,e) \quad = \quad \begin{cases} \varphi_p(f,e) & : \quad e \in \tilde{E}_2(\mathcal{T}) \\ \varphi_f(e) & : \quad e \in \tilde{E}_1(\mathcal{T}) \text{ and } \angle(\varphi_f(e), f) > \angle(\varphi_b(e), f) \\ \varphi_b(e) & : \quad \mathit{otherwise}. \end{cases}$$

The following Pascal-like procedure is the main ingredient to an algorithm to construct a flow surface $\mathcal{S}$.

```
procedure surface( e, f );
begin
    g := φ(f, e);
    if g ∉ S and S ∪ {g} is a flow surface then
    begin
        S := S ∪ {g};
        surface( ẽ, g ) for all edges ẽ ≠ e of g
end end;
```

Now an algorithm can be introduced that constructs disjoint flow surfaces in $\mathcal{T}$:

```
procedure decompose( T );
begin
    while F̃(T) not empty
    begin
        for an arbitrary f ∈ F̃(T) initialise S := {f};
        surface( ẽ, f ) for all edges ẽ of f;
        delete all faces that have at least one vertex in common with S
end end;
```

Remark 1 *The constructed surfaces depend on the starting edge. For the example involving the cylinder depicted in Figure 1 we could either generate vertical or horizontal surfaces.*

An example

We applied the surface algorithm to a cyclic flow in the unit cube. The flow direction and the resulting flow surfaces for a regular grid as well as an irregular grid are depicted in Figure 4. In the case of the regular grid, a single vertical surface and several horizontal surfaces have been constructed. On each of these surfaces we can now apply ordering techniques that have been developed for the two-dimensional case.

Conclusions and future work

We have provided a general framework for constructing (disjoint) surfaces in a tetrahedral grid. To this end, we have constructed surfaces that retain important information about

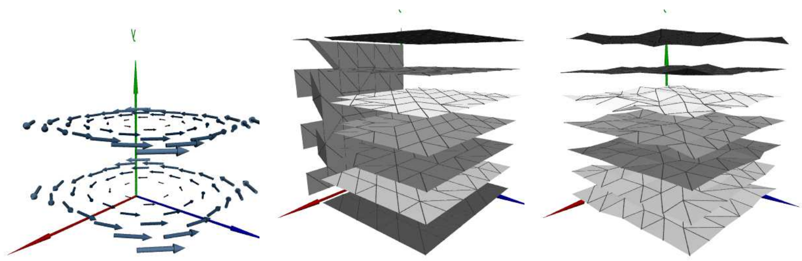


Figure 4 Cyclic flow and resulting flow surfaces on a regular and irregular grid.

the flow direction. Therefore existing ordering techniques that have proved successful in the two-dimensional case may be applied to the three-dimensional case.

Even though the investigation of iterative schemes for solving the linear system of equations was not the goal of this paper, any analysis of the proposed reordering scheme (in terms of improved convergence rates) requires the application of the algorithm to flow problems. This work is presently being performed and the results will be forthcoming. Additional future activities include the choice of parameters in the algorithm, e.g. the parameter κ in the definition of strong edges as well as the parameters α and γ in the definition of the function φ and of flow surfaces. By extending the theory that has been applied to the two-dimensional case to the three-dimensional case, the proposed ordering scheme serves to efficiently decompose the set of unknowns of a convection dominated problem into disjoint blocks for which well-known ordering algorithms are available. Once the disjoint blocks are found, the well-studied block iterative methods may then be applied.

Acknowledgements. The author wishes to thank Thomas Probst and Ronald Kriemann (both University of Kiel) for helpful discussions as well as for implementing the proposed algorithms.

REFERENCES

[BS97] Braess D. and Sarazin R. (1997) An efficient smoother for the Stokes problem. *Applied Numerical Mathematics* 23: 3–19.

[BWY90] Bank R. E., Welfert B. D., and Yserentant H. (1990) A class of iterative methods for solving saddle point problems. *Numerische Mathematik* 56: 645–666.

[GDN95] Griebel M., Dornseifer T., and Neunhoeffer T. (1995) *Numerische Simulation in der Strömungsmechanik - Eine praxisorientierte Einführung.* Vieweg.

[GP97] Gutsch S. and Probst T. (1997) Cyclic and feedback vertex set ordering for the 2d convection-diffusion equation. Technical Report 97-22, Christian-Albrechts-Universität Kiel.

[Hac97] Hackbusch W. (1997) On the feedback vertex set problem for a planar graph. *Computing* 58: 129–155.

[HGMM98] Hackbusch W., Gutsch S., Maitre J.-F., and Musy F. (1998) The appropriate numbering for the multigrid solution of convection dominated problems. In *Notes on Numerical Fluid Mechanics*, volume 66, pages 75–88.

Vieweg.

[HP97] Hackbusch W. and Probst T. (1997) Downwind Gauß-Seidel Smoothing for Convection Dominated Problems. *Numerical Linear Algebra with Applications* 4: 85–102.

[RR96] Rentz-Reichert H. (1996) *Robuste Mehrgitterverfahren zur Lösung der inkompressiblen Navier-Stokes Gleichung: Ein Vergleich.* PhD thesis, Institut für Computeranwendungen der Universität Stuttgart.

[Tur97] Turek S. (1997) On ordering strategies in a multigrid algorithm. In *Notes on Numerical Fluid Mechanics*, volume 41. Vieweg. Proc. 8th GAMM-Seminar, Kiel.

59

Domain decomposition and parallel processing in microwave applicator design.

D.H. Malan[1], A.C. Metaxas[2]

Introduction

In this paper, spatial domain decomposition with a direct solver is presented. It is used with the frequency domain finite element method to compute the electric field in microwave heating applicators. Parallel implementation of the scheme is discussed, and the analysis of a loaded microwave heating cavity is presented. It is demonstrated how this method can be used to very rapidly determine the change in reflection coefficient as a function of localised changes in geometry.

The finite element method is a powerful technique for the electromagnetic analysis of microwave heating cavities, and a number of authors have reported on its use, notably Dibben and Metaxas [DM94] and Chassecourte et al. [CLM93].

It was found by Dibben and Metaxas [DM94] that for multimode microwave cavities loaded with a food-like material, the matrices produced by the frequency domain finite element method become extremely ill-conditioned. If an iterative method is to be used with the highly ill-conditioned system that is produced by the frequency domain formulation, very sophisticated and complicated preconditiong would be required. An alternative approach is to use a direct solver. Although direct solvers require a great deal more storage than iterative schemes, with the continual decrease in the cost of computer memory, they are becoming an attractive alternative, even for three dimensional structures of medium size.

The direct solution spatial domain decomposition technique described here, has been

1 Electricity Utilisation Group, Cambridge University Engineering Department, UK,
CB2 1PZ, *tel: +44 1223 332867 fax: +44 1223 332662*
2 Address same as above.
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

extensively used in structural analysis in past decades [Prz63], but comparatively few applications have thus far been reported in electromagnetics [WRH92, SC95].

The domain decomposition method

The problem domain is subdivided into arbitrarily shaped subregions and a mesh is created for each subregion, with the meshes of adjacent subregions conforming on the shared boundaries. Using elemental matrices found from applying the FEM in the normal way [Jin93]pp. 244–250, a finite element matrix and vector is built up for each subregion independently. Each such system is then reordered so that the unknowns lying on the boundary $\{x_{BB}\}$ of the subregion are numbered last, and the unknowns lying in the interior $\{x_{II}\}$ are numbered first:

$$\begin{bmatrix} A_{II} & A_{IB} \\ A_{BI} & A_{BB} \end{bmatrix} \begin{Bmatrix} x_{II} \\ x_{BB} \end{Bmatrix} = \begin{Bmatrix} f_{II} \\ f_{BB} \end{Bmatrix} \tag{1}$$

The interior unknowns can now be written in terms of the boundary unknowns, by using the first row of the above equation:

$$\{x_{II}\} = [A_{II}]^{-1}\{f_{II}\} - [A_{II}]^{-1}[A_{IB}]\{x_{BB}\} \tag{2}$$

Substituting this into the second row of equation (1) allows the internal unknowns of the subregion to be eliminated, resulting in the reduced system in terms of only the boundary unknowns:

$$[A_{SC}]\{x_{BB}\} = \{f_{SC}\} \tag{3}$$

where

$$\begin{aligned} [A_{SC}] &= [A_{BB}] - [A_{BI}][A_{II}]^{-1}[A_{IB}] \\ \{f_{SC}\} &= \{f_{BB}\} - [A_{BI}][A_{II}]^{-1}\{f_{II}\} \end{aligned} \tag{4}$$

$[A_{SC}]$ is the Schur complement matrix of the system [SBG96]pp. 105–106. In the same way, the internal unknowns are eliminated for every subregion. The reduced systems of the subregions can then be combined into a reduced system for the whole region. After this is solved, the internal unknowns for each subregion are recovered using equation (2).

It is possible to execute this process in several steps. The internal unknowns for each subregion are eliminated, after which adjacent subregions are grouped together, and their reduced linear systems combined. The internal unknowns of these larger subregions, corresponding to unknowns on shared boundaries, are then eliminated, and the resulting larger subregions combined with adjacent ones into even larger subregions, until the whole domain is encompassed. In this way a pyramidal or hierarchical structure is created which is traversed upwards in order to eliminate unknowns, and then downwards in order to find their values.

Implementation

A domain decomposition FE program was created in C++ and Fortran 90 using the above concept, to solve the vector wave equation [Met96]pp. 41–43:

$$\nabla \times \frac{1}{\mu}\nabla \times \mathbf{E} + \sigma_e \frac{\partial \mathbf{E}}{\partial t} + \epsilon \frac{\partial^2 \mathbf{E}}{\partial t^2} = -\frac{\partial \mathbf{J_s}}{\partial t} \tag{5}$$

Using this program, solutions were computed in closed cavities fed by a waveguide, with the boundary condition on the metal walls being zero tangential electric field, and excitation by an imposed field distribution across the width of the waveguide.

The code was written specifically with parallel execution in mind, and to this end, the elimination of the internal unknowns for each subregion runs as a separate program. Programs communicate using the PVM (Parallel Virtual Machine) library of message passing routines, allowing the code to be used on non-homogenous clusters of workstations.

Because the subregions are combined in a hierarchical manner, the parallelism decreases as one moves upwards in the pyramid. To achieve load balancing, a *pool of tasks* paradigm was implemented, with a single controller processor continually assigning new subregions to the processors as they become available. In this way, reasonable parallel efficiency is achieved, as long as the domain is partitioned into many more subregions than there are processors.

A feature of the direct solution scheme that was deliberately exploited, is that it if the model geometry is changed, it is necessary to recompute the Schur complement matrices for only those subregions affected by the change. This means that if a geometry change is fairly localised, the bulk of the computation does not have to be repeated, allowing a new solution to be very rapidly computed.

Although the intention is eventually to use an unstructured tetrahedral mesh, initially a regular mesh of brick edge-elements [Jin93]pp. 244–250 was used to facilitate geometric mesh decomposition.

Results

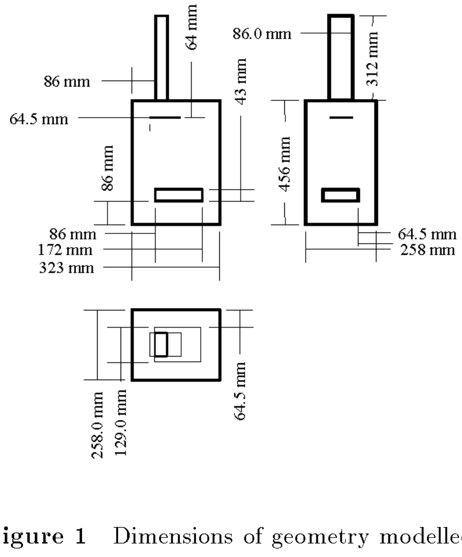


Figure 1 Dimensions of geometry modelled.

For a test problem, a multimode rectangular microwave heating cavity fed by a waveguide was chosen, as shown in Figure 1.

A block of lossy material with permittivity 2-1j was placed some way above the bottom of the cavity. From the computed electric field in the waveguide, it was possible to determine the reflection coefficient of the structure. It was decided to attempt to reduce this by suspending a thin metal plate below the waveguide aperture, and varying its length in order to improve the matching. Meshing the problem resulted in 78,000 elements, and 138,000 free unknowns.

Using spatial domain decomposition, the geometry was divided into twenty-one subregions at the lowest level, with six levels in the hierarchy. To do the initial elimination of unknowns and field solution for the entire structure took more than two hours, using 16 processors on a Hitachi SR2201 parallel machine.

After this, the length of the thin plate was varied from 7.16mm to 200.6 mm in 28 steps, requiring 28 new field solutions to be computed. However, since the changing plate was confined to only one subregion, a field calculation for each new plate size could be very rapidly computed - in around 45 minutes on a *single* processor. Moreover, the calculations for each new plate size could be done completely in parallel, so that is possible to compute the field patterns, and hence reflection coefficients, for all twenty eight plate sizes simultaneously using twenty eight processors, in 45 minutes.

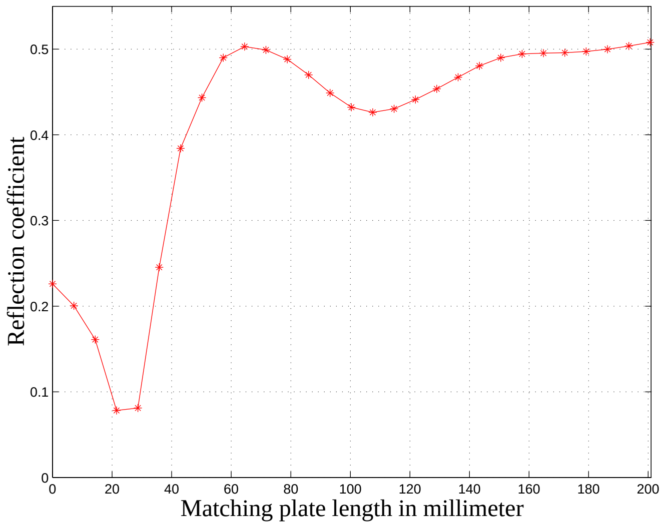


Figure 2 Reflection coefficient against matching plate length.

The results are displayed in Figure 2. From this graph it can be seen that by introducing a small piece of plate 21.5 mm long, the matching of the cavity can be significantly improved - from 0.22 to 0.08.

The field patterns for three points on the graph are displayed in Figure 3. The first has no plate, while in the second the field concentration around the short little plate can be clearly seen. In the third, with a slightly longer plate, the large standing wave ratio in the waveguide, indicating poor matching, can be observed.

Conclusions

Domain Decomposition allows the Finite Element Method to be used in the frequency domain with a direct solver to analyse resonant structures several wavelengths in size. Once an initial

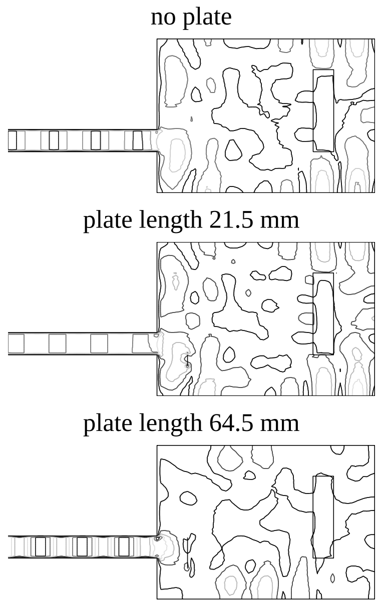


Figure 3 Electric field pattern with three different plate lengths.

solution for the complete structure has been computed, localised changes in the geometry can be isolated in a subregion, and a field solution for the entire structure recomputed very rapidly. This might allow FE-based optimisation of electromagnetic cavities to be done in the future.

Significant parallelism is introduced into the solution process which can be exploited given suitable computer resources.

REFERENCES

[CLM93] Chassecourte P., Lamaudiere J., and Maestrali B. (1993) *Electromagnetic Field Modelling of a Loaded Microwave Cavity.* Presented at 28th IMPI Microwave Power Symposium, Montreal, Canada, July 11 – 14.

[DM94] Dibben D. and Metaxas A. (1994) Finite-Element Time-Domain Analysis of Multimode Applicators Using Edge Elements. *International Journal of Microwave Power and Electromagnetic Energy* 29(4): 242–251.

[Jin93] Jin J. (1993) *The Finite Element Method in Electromagnetics.* John Wiley & Sons, Inc., New York NY.

[Met96] Metaxas A. (1996) *Foundations of Electroheat.* John Wiley & Sons, Inc., Chichester, UK.

[Prz63] Przemieniecki J. S. (1963) Matrix structural analysis of substructures. *Am. Inst. Aero. Astro. Journal* 1: 138–147.

[SBG96] Smith B., Bjørstad P., and Gropp W. (1996) *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations.* Cambridge University Press, Cambridge.

[SC95] Spring C. and Cangellaris A. (1995) Parallel implementation of domain decomposition methods for the electromagnetic analysis of guided wave systems. *Journal of Electromagnetic Waves and Applications* 9(1/2): 175–192.

[WRH92] Weeber K., Ratnajeevan S., and Hoole H. (March 1992) The subregion method in magnetic field analysis and design optimisation. *IEEE Transactions on Magnetics* 28(2): 1561–1564.

60

On the Choice of Krylov Methods and Preconditioners for a Domain Decomposed Iterative Solution of the Exterior Helmholtz Problem

Antonini Puppin Macedo [1]

Introduction

From a finite element viewpoint, the solution of the Helmholtz problem has stability problems related to its operator, in particular the loss of ellipticity for high wave numbers. Regardless of the stability properties of the finite element method used in its discretization, the Helmholtz problem requires always a minimal number of elements per wave length to correctly represent the physical phenomena and to avoid an under-resolved numerical solution. The engineering literature often refers to 10 elements per wave length $kh = \pi/5$ as providing an acceptable resolution. Therefore realistic scattering problems intrinsically require fine meshes leading to significantly large-scale systems of equations that must be solved to high precision in engineering applications.

It is also important to note that the finite element discretization of the Helmholtz operator results in systems which combine stiffness and mass terms in the form,

$$[\mathbf{K} - k^2\mathbf{M}]\mathbf{u} = \mathbf{f} \tag{1}$$

that is symmetric and *indefinite* for high wave numbers. Also, for the exterior Helmholtz problem, the Sommerfeld operator adds a contribution of a general form $ik\mathbf{M}_S$ to the system matrices resulting in a system often called complex, symmetric and indefinite. We note that $\mathbf{M}_S$ is a mass matrix restricted to the dof's on the external boundary.

1 Department of Aerospace Engineering Sciences, and Center for Aerospace Structures, University of Colorado at Boulder, Boulder, CO 80309-0429, U. S. A.
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

Domain decomposition - DD - methods for such problem in general lead to systems with the same characteristics, which poses several challenges for their development. In this context, the recent DD literature is very rich in relevant contributions to the solution of the exterior Helmholtz problem [Des90, BD96, CCEW98, dLBFM+98, FML98a, FML+98b]. Among these works the so-called FETI-H method [FML98a, FML+98b] is one of the most efficient for large scale systems. This method is a dual Schur complement method that comes from a modified Lagrangian operator using Lagrange multipliers to enforce continuity between subdomains, resulting in an interface problem that is solved by means of iterative Krylov methods. These systems, as a result of the use of dual Schur complements, are very ill-conditioned degrading convergence. To solve this complex, symmetric, indefinite and ill-conditioned systems we intend to investigate the application of well-known algorithms for the Helmholtz operator, namely: Conjugate Residuals - CR -, Generalized Conjugate Residuals - GCR -, Bi-conjugate Gradients - BCG - and Generalized Minimal Residual - GMRES.

This paper is organized as follows. In section 2, we present the system of interface equations resulting from the FETI-H method. In section 3, we discuss the iterative solution of this system and appropriate preconditioners. We end the paper with a series of numerical experiments.

The regularized FETI method for complex problems

The Euler equations associated with the FETI-H modified Lagrangian can be written as (see [FML98a, FML+98b] for more details):

$$(\widetilde{\mathbf{K}}^s + ik\mathbf{M}_I^s)\mathbf{u}^s = (\mathbf{K}^s - k^2\mathbf{M}^s + ik\mathbf{M}_I^s)\mathbf{u}^s = \mathbf{f}^s - \mathbf{B}^{s^T}\lambda \tag{2}$$

$$\sum_{s=1}^{s=N_s} \mathbf{B}^s\mathbf{u}^s = 0 \tag{3}$$

where

$$\mathbf{M}_I^s = \begin{bmatrix} 0 & 0 \\ 0 & \sum_{\Omega^s\cap\Omega^q\neq\{\emptyset\}} \epsilon^{s,q}\mathbf{M}_{bb}^{s,q} \end{bmatrix} \tag{4}$$

and it follows that $\mathbf{M}_I^s$ has a constant sign within a subdomain and opposite sign in its neighbors, that is, if $\forall q/\Omega^s\cap\Omega^q \neq \{\emptyset\}$ $\epsilon^{s,q}=1$, so $\forall s/\Omega^q\cap\Omega^s \neq \{\emptyset\}$ $\epsilon^{q,s}=-1$ or both $\epsilon^{q,s}$ and $\epsilon^{s,q}$ are set to zero. Note that in [FML98a] it is shown that a modified Lagrangian approach ensures that the resulting subdomain problem matrices $\widetilde{\mathbf{K}}^s + ik\mathbf{M}_I^s$ are always non singular and an algorithm is proposed to ensure that the signing condition imposed by $\epsilon^{s,q}$ is always satisfied and at least one edge of each subdomain has a non-zero ϵ.

From (3), we find that the interface problem associated with the regularized subdomain equations (2) is given by

$$\mathbf{F}_I\lambda = \mathbf{d} \tag{5}$$

where

$$\mathbf{F}_I = \sum_{s=1}^{s=N_s} \mathbf{B}^s(\mathbf{K}^s - k^2\mathbf{M}^s + ik\mathbf{M}_I^s)^{-1}\mathbf{B}^{s^T}$$

$$\mathbf{d} = \sum_{s=1}^{s=N_s} \mathbf{B}^s(\widetilde{\mathbf{K}}^s + ik\mathbf{M}_I^s)^{-1}\mathbf{f}^s$$

where $\mathbf{B}$ is a signed boolean matrix that extracts the boundary components from a subdomain vector, $\mathbf{M}_I$ is the "interface mass" emanating from the modified Lagrangian of the FETI-H method (see [FML98a]).

The performance of iterative solvers for this system is then crucial in FETI type methods and the efficiency of the Krylov methods used will play a significant rule. Recalling the discussion in the introduction, this system presents challenging difficulties that has required the development of preconditioning techniques that make the solution by Krylov methods feasible. In the next section, we discuss the appropriate Krylov methods for this problem as well as the development of preconditioning techniques at both coarse and local levels for the FETI-H method.

Iterative solution of the interface problem

For the iterative solution of complex systems emanating from the Helmholtz discretization, we intend to focus on basically two approaches. First, methods that take advantage of the symmetry of the problem are considered, namely the Conjugate Residuals - CR - and the Bi-Conjugate Gradients - BCG - algorithms. Because of this, the inner product used for these solvers is redefined from what is expected for complex systems and the requirement of a full-reorthogonalization for the search directions is circumvented. Second, methods traditionally well regarded for the solution of indefinite systems are considered, but in this case they require full reorthogonalization of the search directions, as in the Generalized Conjugate Residuals - GCR - and the Generalized Minimal Residuals - GMRES - algorithms. [2]

The BCG algorithm, derived from the Lanczos recursion, is a Conjugate Gradient like method and will be implemented in its transpose-free version [Fre92]. [3]

The CR algorithm, on the other hand, is derived from GMRES in the particular case of a symmetric system matrix $\mathbf{A}$. In this case, the residual vectors should be $\mathbf{A}$-orthogonal, i.e., they are conjugate (hence the name of the algorithm). In addition, the search directions $\mathbf{p}$, are also $\mathbf{A}$ orthogonal. The resulting algorithm has the same structure as the standard Conjugate Gradients algorithm, but satisfies these conditions and requires only one more vector update, i.e., $2n$ more operations, than CG [Saa95].

A very important point related to the application of these algorithms is that the definition of the inner-product must be redefined as:

$$(\mathbf{x}, \mathbf{y}) = \mathbf{y}^T \mathbf{x}, \quad \mathbf{x}, \mathbf{y} \in \mathcal{C}^n \tag{6}$$

for *symmetric* complex matrices, and not as

$$(\mathbf{x}, \mathbf{y}) = \mathbf{y}^H \mathbf{x}, \quad \mathbf{x}, \mathbf{y} \in \mathcal{C}^n \tag{7}$$

as normaly used for complex matrices. See [Cra69, Fre92, Mac99] for a deeper discussion on this.

The other algorithms we intend to apply to this problem, GCR and GMRES, require a full reorthogonalization of the search directions in their standard form. For the Helmholtz problem, within the current framework, the number of iterations may become very large and the storage and computational requirements may become impractical. The literature is rich in restarted and truncated versions, but they are well known to stagnate when the matrix

2 Because of the page limit for this contribution, we refer the reader to [Saa95, Fre92, Mac99] for a detailed description of these algorithms.

3 In [ZW94], we find that the iterations of another well regarded solver for Helmholtz problems, the Quasi-Minimal Residuals - QMR - algorithm, can be obtained from those of BCG as a particular case of residual smoothing.

is not positive definite. In our numerical experiments, large scale problems have a strong tendency to stagnate for all algorithms, even when full reorthogonalization is used. The use of preconditioning techniques is an obvious option to keep the number of iterations small making it feasible to use any method.

For the FETI-H method the development of a two-level coarse projection approach (see [FML98a, FML+98b] for details) represent the a most crucial point in the high performance of the method, combined with the modified Lagrangian formulation that *per se* provides scalability with respect to the element size as well as guarantees that the local subdomain problems are solvable. In addition, we propose efficient local preconditioning techniques that will improve convergence even more. We will compare the performance of the different algorithms. To make this paper self-contained, we briefly present the coarse projection technique in the next section and we then discuss the ideas of the local preconditioner.

A coarsening strategy: the improved FETI-H algorithm

The idea of the coarse projection is, in fact, the main characteristic of the FETI-H method. It basically accelerates convergence by modifying the Krylov iterations by enforcing the following *optional admissible constraint* on the residuals $\mathbf{r}^k$ in each iteration:

$$\mathbf{Q}^T\mathbf{r}^k \;=\; 0\,. \tag{8}$$

Here $\mathbf{Q}$ is a coarse subspace containing a set of plane waves (see [FML98a, FML+98b] for details). This leads to the following projector:

$$\mathbf{P} \;=\; \mathbf{I} - \mathbf{Q}(\mathbf{Q}^T\mathbf{F}_I\mathbf{Q})^{-1}\mathbf{Q}^T\mathbf{F}_I \tag{9}$$

which transforms the FETI-H interface problem into:

$$\mathbf{P}^T\mathbf{F}_I\mathbf{P}\lambda \;=\; \mathbf{F}_I\mathbf{P}\lambda \;=\; \mathbf{P}^T\mathbf{d}\;; \tag{10}$$

$\mathbf{P}$ may be interpreted as a right preconditioner.

A suitable local preconditioning technique for the FETI-H algorithm

For FETI methods in general, it is also possible to derive another right preconditioner similar to the coarse projector $\mathbf{P}$, but instead of coming from a coarse space of optional admissible constraint, we approximate the interface problem using a local, subdomain based approximation.

The development of preconditioners for the traditional FETI method for elasticity problems can be presented in terms of mechanical arguments. In each iteration the FETI solver computes a "force" vector $\mathbf{p}^k$ and then the corresponding jump of the displacement across the interfaces. A simple way to accelerate convergence is then to invert this process, i.e., to impose the jump of the displacement field across the interface and compute the corresponding interface forces. These forces are then added back to $\mathbf{p}^k$ and the convergence is accelerated [FR94].

The implementation of such a preconditioner can be interpreted as follows. The forces on the interface are calculated by solving the following system (to simplify the notations we replace $\check{\mathbf{K}}$ by $\mathbf{K}$):

$$\begin{bmatrix} \mathbf{K}_{ii} & \mathbf{K}_{ib} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \widetilde{\mathbf{w}}_i^k \\ \{\widetilde{\mathbf{w}}_b^k\} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{w}_b \end{bmatrix} \tag{11}$$

The solution is

$$\widetilde{\mathbf{w}}_i^k = -\mathbf{K}_{ii}^{-1}\mathbf{K}_{ib}\mathbf{w}_b \tag{12}$$

The subscripts i and b mean the internal and boundary d.o.f. and the brackets refers to known values. Now the corresponding interface force vector is:

$$\begin{bmatrix} \mathbf{K}_{ii} & \mathbf{K}_{ib} \\ \mathbf{K}_{ib}{}^T & \mathbf{K}_{bb} \end{bmatrix} \begin{bmatrix} \widetilde{\mathbf{w}}_i^k \\ \{\widetilde{\mathbf{w}}_b^k\} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{K}_{bb}^{(s)} - \mathbf{K}_{ib}^{(s)\,T}\mathbf{K}_{ii}^{(s)\,-1}\mathbf{K}_{ib}^{(s)} \{\widetilde{\mathbf{w}}_b^k\} \end{bmatrix} \tag{13}$$

For a more efficient implementation of this preconditioner, we assemble the primal subdomain operator instead of approaching the inverse of the operator after assembling. This leads to the Dirichlet preconditioner:

$$\widetilde{\mathbf{F}_{\mathbf{I}}^{-1}}^{\mathbf{D}} = \sum_{s=1}^{s=N_s} \mathbf{B}^{(s)} \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{K}_{bb}^{(s)} - \mathbf{K}_{ib}^{(s)\,T}\mathbf{K}_{ii}^{(s)\,-1}\mathbf{K}_{ib}^{(s)} \end{bmatrix} \mathbf{B}^{(s)\,T} \tag{14}$$

The computation of the Dirichlet preconditioner needs an additional factorization of $\mathbf{K}_{ii}$, and storage. In each iteration, we also need one more forward/backward substitution. The following preconditioner,

$$\widetilde{\mathbf{F}_{\mathbf{I}}^{-1}}^{\mathbf{L}} = \sum_{s=1}^{s=N_s} \mathbf{B}^{(s)} \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{K}_{bb}^{(s)} \end{bmatrix} \mathbf{B}^{(s)\,T} \tag{15}$$

does not need any addtional storage or factorization. The idea of this *lumped* preconditioner is to lump all of the system matrix to the interface. From the mechanical point of view, for elasticity problems, it tries to find a set of lumped interface forces to reproduce the displacement jumps at the subdomains interface and only the interface nodes are allowed to displace.

For the Helmholtz problem, we intend to filter the vanishing modes by using the preconditioner, i.e., the modes that have a spatial frequency on the interface that is greater than the wavenumber of the global problem, i.e, to filter the eigenmodes associated with the eigenvalues that are close to zero. It has the drawback as it also changes the behavior of the other modes, in particular those of low frequency [dLBFM+98]. This process in the FETI-H method is reminiscent of the *lumped* preconditioner for elasticity problems, as our physical interpretation points out.

These considerations are logical and straightforward, but they do not consider the physical aspects of the Helmholtz problem, in particular the Sommerfeld condition. Furthermore, in the FETI-H method, we use a modified Lagrangian formulation that results in local interface matrices similar to the Sommerfeld condition contribution. The simplest idea to account for these facts is to incorporate into $\mathbf{K}_{bb}$ the contribution of the local interface matrices emanating from the modified Lagrangian. Then the lumped preconditioner is redefined as:

$$\widetilde{\mathbf{F}_{\mathbf{I}}^{-1}}^{\mathbf{L}} = \sum_{s=1}^{s=N_s} \mathbf{B}^{(s)} \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{K}_{bb}^{(s)} \pm ik\mathbf{M}_{bb}^{(s)} \end{bmatrix} \mathbf{B}^{(s)\,T} \tag{16}$$

REMARK: Note that the contribution $\mathbf{M}_{bb}^{(s)}$ incorporates into all of the interface of each subdomain is restricted by the "interface signing" compatibility constraint in [FML98a, FML+98b]. This may be interpreted not as a *Dirichlet* preconditioner, but as a *Dirichlet-Sommerfeld* preconditioner (at least locally). Although the numerical experiments in the next section show that this preconditioner improves convergence, it is still not clear if other definitions of this local *Dirichlet-Sommerfeld* preconditioner may provide a better option.

Numerical experiments

Table 1 The guided wave problem in 3D – $h = 1/36$, $ka = 10$, total size of the problem – 46656 dofs

N_θ	N_S	CR	CR*	BCG	BCG*	GCR	GCR*	GMRES	GMRES*
0	3×1	19	2	19	–	19	2	19	17
1	3×1	17	–	17	–	17	12	17	17
2	3×1	34	–	34	–	32	24	32	31
3	3×1	23	–	21	–	19	15	19	19
0	4×1	30	3	30	–	25	3	25	23
1	4×1	23	–	24	–	22	16	22	23
2	4×1	55	–	57	–	39	32	43	39
3	4×1	27	–	30	–	23	20	24	24
0	6×1	41	10	40	–	34	5	34	32
1	6×1	30	–	31	–	27	22	28	29
2	6×1	87	–	99	–	60	48	60	58
3	6×1	37	–	37	–	33	27	33	32
0	$2 \times 2 \times 2$	90	–	87	–	62	51	61	66
1	$2 \times 2 \times 2$	74	–	78	–	55	45	55	61
2	$2 \times 2 \times 2$	61	–	70	–	51	46	51	59
3	$2 \times 2 \times 2$	68	–	83	–	50	44	50	58
0	$3 \times 3 \times 3$	88	–	83	–	76	63	76	107
1	$3 \times 3 \times 3$	75	–	80	–	61	53	60	102
2	$3 \times 3 \times 3$	101	–	110	–	78	70	78	121
3	$3 \times 3 \times 3$	68	–	69	–	57	48	57	100
0	$4 \times 4 \times 4$	260	–	263	–	162	155	162	188
1	$4 \times 4 \times 4$	103	–	107	–	75	65	75	92
2	$4 \times 4 \times 4$	79	–	83	–	64	54	64	82
3	$4 \times 4 \times 4$	79	–	88	–	61	54	61	79
0	$6 \times 6 \times 6$	–	–	–	–	381	366	–	–
1	$6 \times 6 \times 6$	91	–	100	–	64	57	64	87
2	$6 \times 6 \times 6$	93	–	101	–	65	55	65	84
3	$6 \times 6 \times 6$	96	–	110	–	62	54	62	83

(*) indicates the algorithms with the local preconditioner and (–) stagnation

In this section we consider a 3D guided wave problem in a unit cube domain discretized using $36 \times 36 \times 36$ linear brick elements. Two opposite faces are subject to an unit Dirichlet b.c. and a Sommerfeld condition, respectively. All the other faces are subject to a homogeneous Neumann b.c.. The wave number for this problem is fixed to $ka = 10$ which gives almost 20 elements per wave length. The domain is decomposed into "slices" ($N \times 1$ subdomains) or cubes, i.e. $N \times N \times N$ subdomains. The results for this numerical experiment are given in Table 1 where we can assess the efficiency of the preconditioner and different algorithms. In this table, each column, corresponding to a solver, gives the number of iterations for the following convergence criteria: $||\mathbf{f} - [\mathbf{K} - k^2\mathbf{M}]\bar{\mathbf{u}}|| < 1e-6$, where $\bar{\mathbf{u}}$ is the solution resulting from the FETI-H iterative process. The same criteria was used for the results in Table 2.

Table 2 2D Submarine-shaped scatterer – SGI Origin 2000, 10 cpu's

GCR alg.	No Preconditioner	Preconditioner
Iterations	69	40
Total CPU Time	584.70	487.28
Preconditioning CPU Time	0.0	0.75

Comparing the different algorithms , the GCR algorithm has the best convergence properties and robustness, and, by using the preconditioner, we get good improvement in iteration count, for any mesh partition, as well as when combined with the coarse space. In contrast, the other algorithms do not get the same improvement as GCR and all of them stagnate when the preconditioner is combined with the coarse space or when a general mesh partition is used. For all these experiments, the residuals of the GCR algorithm are much smoother than those of all other algorithms, except for GMRES, which are similarly smooth. Also, it is important to note that the improvement of the preconditioner is worthwhile as it costs only about 0.15 % of the total solver cpu time, for a speed-up of the order of 15 %.

In table 2, we show the significant improvement observed for the FETI-H method by using the preconditioned GCR algorithm for an irregular mesh partition and decomposition in 2D. This is the case presented in [FML98a] and it concerns a submarine-shaped 2D scatterer with 1,077,432 nodes and 1,075,264 elements partitioned in 128 irregular subdomains.

Acknowledgements

The author acknowledges the guidance of his advisor, Prof. Charbel Farhat and the computational help and fruitful comments from Dr. Radek Tezaur. Also, the comments of the referee are greatly appreciated. Partial financial support was provided by a CAPES Fellowship, Process No. 1233/95-2.

REFERENCES

[BD96] Benamou J. and Desprès B. (1996) A domain decomposition method for the Helmholtz equation and related optimal control problems. Rapport de recherche INRIA No. 2791.

[CCEW98] Cai X.-C., Casarin Jr M., Elliott Jr F., and Widlund O. (1998) Two-level algebraic multigrid for the Helmholtz problem. *Contemporary Mathematics* 218: 390–398.

[Cra69] Craven B. (1969) Complex symmetric matrices. *J. Austral. Math. Soc.* 10: 341–354.

[Des90] Desprès B. (1990) Décomposition de domaine et problème de Helmholtz. *Compte Rendu de l'Académie des Sciences* No 311 (Série I): 313–316.

[dLBFM+98] de La Bourdonnaye A., Farhat C., Macedo A., Magoulès F., and Roux F. X. (1998) A non-overlapping domain decomposition method for the exterior Helmholtz problem. *Contemporary Mathematics* 218: 42–66.

[FML98a] Farhat C., Macedo A., and Lesoinne M. (1998) A two-level domain decomposition method for the iterative solution of high frequency exterior

Helmholtz problems. *Num. Math.* Submitted.
[FML+98b] Farhat C., Macedo A., Lesoinne M., Magoulès F., Roux F.-X., and de La Bourdonnaye A. (1998) Two-level domain decomposition methods with Lagrange multipliers for the fast iterative solution of acoustic scattering problems. Univ. of Colorado, Report CU-CAS-98-18.
[FR94] Farhat C. and Roux F.-X. (1994) Implicit parallel processing in structural mechanics. *Computational Mechanics Advances* 2: 1–124.
[Fre92] Freund R. (1992) Conjugate gradient-type methods for linear systems with complex symmetric coefficient matrices. *SIAM J. Sci. Stat. Comput.* 13: 425–448.
[Mac99] Macedo A. (1999) On the choice of Krylov methods and preconditioners for the FETI-H method for exterior Helmholtz scattering problems. Univ. of Colorado, CU-CAS Report. In preparation.
[Saa95] Saad Y. (1995) *Iterative methods for sparse linear systems.* PWS Publishing Company, Boston.
[ZW94] Zhou A. and Walker H. (1994) Residual smoothing techniques for iterative methods. *SIAM J. Sci. Comput.* 15: 297–312.

61

Overlapping Methods with Perfectly Matched Layers for the Solution of the Helmholtz Equation

ANDREA TOSELLI [1]

INTRODUCTION

In recent years, a considerable effort has been devoted to the the study of preconditioners for scalar and vector Helmholtz equations; see, e.g., [Des91, Gha96, CCEW97, MSRKA97]. In this paper, we build a class of overlapping Schwarz preconditioners for a finite element approximation of a scalar Helmholtz problem with a first–order Sommerfeld condition, in two dimensions. Perfectly Matched Layers and two kinds of boundary conditions are employed to build the local problems.

Let $\Omega \subset R^2$ be a bounded connected polygon. We consider the following Helmholtz problem for the complex–valued function u:

$$P(u) := -\Delta u - k^2 u = f, \quad \text{in } \Omega \tag{1}$$

$$\frac{\partial u}{\partial n} + ik\, u = 0, \quad \text{on } \partial\Omega, \tag{2}$$

where the frequency k is positive and the source f has support contained in Ω.

Equations 1 and 2 can be derived from the full 3D Maxwell's equations for time-harmonic fields and first–order Silver–Muller boundary conditions, when considering waves with the electric and magnetic fields, respectively, parallel and perpendicular to the xy–plane (TM waves). Then, 1 and 2 are the equations for the z-component of the magnetic field.

---

1 Courant Institute of Mathematical Sciences, 251 Mercer Street, New York, N.Y. 10012.
E-mail: `toselli@cims.nyu.edu` URL: http://www.math.nyu.edu/~toselli. This work was supported in part by the National Science Foundation under Grants NSF-CCR-9732208 and NSF-ECS-9527169, and in part by the U.S. Department of Energy under Contract DE-FG02-92ER25127.
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

We then approximate Equations 1 and 2, by introducing a triangulation $\mathcal{T}_h$ of Ω, made of quadrilaterals of maximum diameter h, and considering the standard finite element (FE) space $V_h \subset H^1(\Omega)$, consisting of continuous piecewise bilinear functions. Triangular linear FE spaces can also be employed; see [QV94]. The well–posedness, stability, and accuracy of the corresponding linear system are studied in [IB95]. In particular, we recall that for the stability of the linear problem, the condition $kh < 1$ must be satisfied; see [IB95]. As is well–known, a restriction on kh requires that there are enough discretization points per wavelength. The number of points per wavelength is defined as

$$ppw = 2\pi/kh.$$

SCHWARZ METHODS

In the following, we suppose that the domain Ω is a rectangle and, for simplicity, that the triangulation $\mathcal{T}_h$ is uniform. We want to build a preconditioner for Equations 1 and 2, and consider an overlapping decomposition, built in the following way:

Starting from a decomposition of the rectangle Ω into M nonoverlapping rectangles $\{\Omega_i\}$, we extend each rectangle, and obtain a family of overlapping subdomains $\{\Omega_i'\}$, such that their boundaries do not cut through the elements. Let $\Omega_{i,d}'$ be the extended part of Ω_i and let d be the the overlap,

$$d := \delta h,$$

defined as the thickness of $\Omega_{i,d}'$. Each subdomain can then be decomposed as

$$\overline{\Omega}_i' = \overline{\Omega}_i \cup \overline{\Omega}_{i,d}'.$$

For the generic subproblem on the subdomain Ω_i', we solve a Helmholtz problem with the Perfectly Matched Layers studied in [ZC96]. The Perfectly Matched Layer in Ω_i' is exactly $\Omega_{i,d}'$ and the corresponding operator is defined as

$$P_d(u) := \begin{cases} -\Delta u - k^2 u, & \text{in } \Omega_i \\ -\text{div } (\Lambda \, \mathbf{grad}\, u) - k^2 a_z u, & \text{in } \Omega_{i,d}'. \end{cases} \tag{3}$$

Here $\Lambda = \text{diag}\,\{a_x(x,y), a_y(x,y)\}$ and $a_z(x,y)$ are suitable complex functions, that ensure that incident waves on the boundary between Ω_i and $\Omega_{i,d}'$ are not reflected and that the energy of a wave traveling inside $\Omega_{i,d}'$ is partially absorbed. In the layer that is perpendicular to the x axis, for instance, they are given by

$$a_x^{-1} = a_y = a_z = 1 - i\alpha \left(\frac{y}{d}\right)^m, \tag{4}$$

where α is an absorption coefficient. We refer to [ZC96] for the exact expression of the coefficients a_x, a_y and a_z in the general case, and to [CM97] for a discussion of practical issues of Perfectly Matched Layers.

We obtain different preconditioners, by choosing different boundary conditions for the local problems. We consider Dirichlet conditions (*Algorithm 1L*) and Sommerfeld conditions (*Algorithm 2L*). We define the following linear iterations. We start with an initial vector u^0. A full iteration step is performed through M fractional steps, where $u^{n+\frac{j}{M}}$ is the solution of the following problem on the subdomain Ω_j', $j = 1, \cdots, M$:

- **Algorithm 1L** (Dirichlet + Layers)

$$\begin{cases} P_d\left(u_j^{n+\frac{j}{M}} - u^{n+\frac{j-1}{M}}\right) = f - P\left(u^{n+\frac{j-1}{M}}\right), & \text{in } \Omega_j', \\ u_j^{n+\frac{j}{M}} = u_{out}^{n+\frac{j-1}{M}}, & \text{on } \partial\Omega_j'. \end{cases} \tag{5}$$

- **Algorithm 2L** (Sommerfeld + Layers)

$$\begin{cases} P_d\left(u_j^{n+\frac{j}{M}} - u^{n+\frac{j-1}{M}}\right) = f - P\left(u^{n+\frac{j-1}{M}}\right), & \text{in } \Omega'_j, \\ \frac{\partial u_j^{n+\frac{j}{M}}}{\partial n_{int}} - ik\, u_j^{n+\frac{j}{M}} = -\frac{\partial u_{out}^{n+\frac{j-1}{M}}}{\partial n_{out}} - ik\, u_{out}^{n+\frac{j-1}{M}}, & \text{on } \partial\Omega'_j. \end{cases} \tag{6}$$

Here n_{int} and n_{out} are the outward and inward normal vectors to $\partial\Omega'_j$, respectively. The function $u_{out}^{n+\frac{j-1}{M}}$ is the iterate at step $n + \frac{j-1}{M}$, defined in $\Omega \setminus \Omega'_j$.

In the definition of the fractional steps 5 and 6, we have chosen to solve the local problems in sequence and obtained multiplicative algorithms, but additive algorithms can also be considered. These basic iterations can be employed to build preconditioners to be combined with a Krylov–type accelerator. A coarse solver can also be added in a standard way, by using the FE discretization of Equations 1 and 2 on a coarse mesh $\mathcal{T}_H$, $H > h$. See [SBG96] for a general discussion of these issues. We also refer to [QV99], for other examples of iteration schemes, similar to Algorithms 1L and 2L, where the continuity of suitable traces across the boundaries of the subdomains is enforced. We note that the corresponding algorithms with no absorption ($\alpha = 0$) have already been studied: see Algorithms 1 and 2 in [CCEW97]. We also remark that Perfectly Matched Layers are only employed for the local solvers.

NUMERICAL RESULTS

In this section, we compare the performance of the two algorithms introduced in the previous section, when varying the overlap, the number of subregions and the diameter of the coarse mesh. We will only present results for multiplicative preconditioners, since we have observed that they give far better performances than the corresponding additive ones, for this particular problem.

We have taken $\Omega = (0,1)^2$ in all our experiments. In the following, n and nc denote the number of discretization nodes in each direction, for the fine and the coarse mesh, respectively, and $nsub \times nsub$ is the number of subdomains. We also define the *wavelap* as the fraction of a wavelength that is covered by the overlap:

$$wlp = \frac{\delta}{ppw};$$

see [CCEW97].

In our numerical results, we have employed GMRES acceleration and right preconditioning, with restart equal to 40, a maximum number of iterations equal to 70, and a reduction of the relative residual of the preconditioned system, by a factor of 10^{-6}. For the absorption in the Perfectly Matched Layers, we have chosen $m = 2$, in 4.

For our two methods, an optimal range of values of the absorption coefficient α can be found, which is fairly insensitive to the frequency, the number of points per wavelength, the number of subregions and the diameter of the coarse triangulation. The supporting results are not shown here. We also note that our methods is pretty robust with respect to variations of the absorption coefficient, if $\alpha > 1$ for Algorithm 1L, and $\alpha > 0.25$ for Algorithm 2L. In our experiments, we have chosen $\alpha = 2.0$ for Algorithm 1L and $\alpha = 0.75$ for Algorithm 2L.

The first set of tables shows the dependence on ppw (or n, equivalently), the overlap and the number of subregions, for a fixed value of the frequency and no coarse space; see Table 1 for $ppw = 10.1$ and Table 2 for $ppw = 13.5$. The tables show results for Algorithm 1L with $\alpha = 2.0$, Algorithm 2L with $\alpha = 0.75$ and Algorithm 2L with $\alpha = 0$.

Table 1 Number of GMRES iterations, versus δ (wavelap) and $nsub$; $n = 121$, $ppw = 10.1$, $k = 75$, $nc = 0$; first rows for Algorithm 1L with $\alpha = 2.0$, second rows for Algorithm 2L with $\alpha = 0.75$ and third rows for Algorithm 2L with $\alpha = 0$.

δ wavelap	1 0.10	2 0.20	3 0.30	4 0.39	5 0.49	6 0.59
nsub = 4	>70	37	29	27	26	25
	21	20	18	18	17	17
	24	19	19	20	18	21
nsub = 5	>70	44	32	29	28	26
	25	22	21	20	19	18
	28	22	23	23	23	26
nsub = 8	>70	61	43	36	33	31
	35	31	28	27	26	23
	41	34	38	48	44	>70

As expected, without a coarse space and for a fixed δ, the number of iterations increases with the number of subregions. By comparing the results for Algorithm 2L, one can see that the increase is larger if no absorption is present. We also remark that Sommerfeld boundary conditions for the local problems ensure faster convergence; see also [CCEW97]. In this case, results for $\alpha > 0$ are somewhat better than those for $\alpha = 0$ for a few subregions, and considerably better for many subregions. In particular, some absorption ensures a steady decrease of the number of iterations when the overlap is increased.

By comparing Tables 1 and 2, one can see that, for a fixed number of subregions, a constant value of the wavelap gives comparable numbers of iterations. This shows the importance of this parameter in the analysis of overlapping methods for Helmholtz problems. This has already been pointed out in [CCEW97, MSRKA97] for other Schwarz algorithms.

Tables 3 and 4 show the results when a coarse space is added. For a fixed value of ppw and $nsub$, they show the number of iterations when varying the wavelap and the size of the coarse space, for different values of the frequency. Results are given for Algorithm 1L with $\alpha = 2.0$, Algorithm 2L with $\alpha = 0.75$ and Algorithm 2L with $\alpha = 0$.

We observe an initial deterioration of the performances when a very coarse space is added, but note a considerable improvement, when the number of coarse points per wavelength ($cppw$) is sufficiently large (greater than or equal to 4).

As for Tables 1 and 2, we remark that for Algorithm 2L, some absorption ensures better performances. We also remark that, for a fixed value of the wavelap and the number of coarse points per wavelength, the number of iterations increases with the frequency.

CONCLUDING REMARKS

We do not show any results for Algorithm 1L with $\alpha = 0$. This case was considered in [CCEW97] and it generally performs very poorly. From the numerical results, we can deduce that, in general, adding Perfectly Matched Layers to the local problems, improves the

Table 2 Number of GMRES iterations, versus δ (wavelap) and $nsub$; $n = 161$, $ppw = 13.5$, $k = 75$, $nc = 0$; first rows for Algorithm 1L with $\alpha = 2.0$, second rows for Algorithm 2L with $\alpha = 0.75$ and third rows for Algorithm 2L with $\alpha = 0$.

δ wavelap	1 0.07	2 0.15	3 0.22	4 0.30	5 0.37	6 0.44
nsub = 4	>70	48	33	28	28	27
	26	21	20	19	19	19
	56	21	20	21	22	22
nsub = 5	>70	64	40	32	30	28
	29	24	22	22	21	20
	>70	25	24	25	27	27
nsub = 8	>70	>70	53	43	39	36
	42	33	32	31	30	29
	>70	38	37	42	65	65

performance of Schwarz methods for Helmholtz equations.

The key parameters of the algorithms are the wavelap for the one–level algorithms and the wavelap, the number of coarse points per wavelengths, and the frequency, for the two–level algorithms.

The methods developed in this paper can be easily generalized to the full three–dimensional Maxwell's equations, using the theory of PMLs developed in [ZC96] for the three–dimensional case and results are forthcoming.

ACKNOWLEDGMENTS

The author is grateful to Olof Widlund for his help and enlightening discussions of my work and to Jean David Benamou for suggesting the problem.

REFERENCES

[CCEW97] Cai X.-C., Casarin M. A., Elliott Jr F. W., and Widlund O. B. (1997) Overlapping Schwarz methods for solving Helmholtz's equation. In Cai X.-C., Farhat C., and Mandel J. (eds) *Tenth International Symposium on Domain Decomposition Methods for Partial Differential Equations*. AMS. TR 753, Computer Science Department, Courant Institute, New York, NY.

[CM97] Collino F. and Monk P. (1997) Optimizing the perfectly matched layer. *Comput. Meth. in Appl. Mech. and Eng.* To appear.

[Des91] Després B. (October 1991) *Méthodes de Décomposition de Domaine pour les Problémes de Propagation d'Ondes en Régime Harmonique*. PhD thesis, Paris IX Dauphine.

[Gha96] Ghanemi S. (January 1996) *Méthode de Décomposition de Domaine avec Conditions de Transmissions Non Locales pour des Problémes de Propagation*

Table 3 Number of GMRES iterations, versus δ (wavelap) and nc; $ppw = 20$, $nsub = 8$, $n = 121$, $k = 38$; first rows for Algorithm 1L with $\alpha = 2.0$, second rows for Algorithm 2L with $\alpha = 0.75$ and third rows for Algorithm 2L with $\alpha = 0$.

δ wavelap	1 0.05	2 0.10	3 0.15	4 0.20	5 0.25	6 0.30
nc = 0	>70	>70	>70	58	51	49
cppw = 0.0	68	55	43	39	41	47
	>70	>70	52	47	52	>70
nc = 16	>70	>70	61	44	42	40
cppw = 2.6	>70	63	44	46	52	68
	>70	>70	51	55	>70	>70
nc = 24	30	31	28	27	25	24
cppw = 4.0	51	28	24	22	22	22
	>70	>70	27	24	25	53
nc = 32	25	25	24	23	23	21
cppw = 5.3	43	21	18	19	19	21
	>70	43	18	18	27	33

d'Ondes. PhD thesis, Paris IX Dauphine.

[IB95] Ihlenburg F. and Babuška I. (1995) Finite element solution of the Helmholtz equation with high wave number, Part I: The h-version of the FEM. *Computers Math. Applic.* 30(9): 9–37.

[MSRKA97] McInnes L. C., Susan-Resiga R. F., Keyes D. E., and Atassi H. M. (1997) Additive Schwarz methods with nonreflecting boundary conditions for the parallel computation of Helmholtz problems. In Cai X.-C., Farhat C., and Mandel J. (eds) *Tenth International Symposium on Domain Decomposition Methods for Partial Differential Equations*. AMS.

[QV94] Quarteroni A. and Valli A. (1994) *Numerical approximation of partial differential equations*. Springer-Verlag, Berlin.

[QV99] Quarteroni A. and Valli A. (1999) *Domain decomposition methods for partial differential equations*. Oxford University Press, Oxford.

[SBG96] Smith B. F., Bjørstad P. E., and Gropp W. D. (1996) *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*. Cambridge University Press.

[ZC96] Zhao L. and Cangellaris A. C. (1996) GT-PML: generalized theory of perfectly matched layers and its application to the reflectionless truncation of finite–difference time–domain grids. *IEEE Trans. Microwave Theory Tech.* 44: 2555–2563.

Table 4 Number of GMRES iterations, versus δ (wavelap) and nc; $ppw = 20$, $nsub = 8$, $n = 161$, $k = 50.6$; first rows for Algorithm 1L with $\alpha = 2.0$, second rows for Algorithm 2L with $\alpha = 0.75$ and third rows for Algorithm 2L with $\alpha = 0$.

δ wavelap	1 0.05	2 0.10	3 0.15	4 0.20	5 0.25	6 0.30
nc = 0 cppw = 0.0	>70 69 >70	>70 46 >70	>70 39 47	58 37 43	48 35 47	45 37 >70
nc = 16 cppw = 2.6	>70 >70 -	>70 >70 -	>70 >70 -	>70 >70 -	>70 >70 -	>70 >70 -
nc = 24 cppw = 4.0	>70 59 >70	63 47 -	52 37 49	41 48 60	40 52 >70	44 40 >70
nc = 32 cppw = 5.3	69 46 >70	36 30 >70	30 36 43	33 30 31	33 25 28	31 24 >70

62

Parallel multi-grid for turbulent reacting flow simulations

A.Twerda[1], R.L.Verweij[2], T.W.J. Peeters[2] & A.F. Bakker[2]

Introduction

This paper reports on the implementation of a multi-grid acceleration technique in a computer code FURNACE, which simulates turbulent combustion in large-scale furnaces. The need for accurate predictions of these furnaces is still increasing, due to strict government regulations. Simulations of large-scale industrial furnaces require fine meshes and accurate models, which makes these simulations very CPU and memory demanding. Convergence becomes troublesome, especially in combination with domain decomposition. Full multi-grid with full approximation scheme is implemented to accelerate the convergence [Wes92, PSS92].

The implementation of the multi-grid is validated by a well-known CFD problem: a laminar lid-driven cavity flow. Next, results for a furnace simulation will be presented. These numerical examples show that multi-grid improves the convergence behaviour and can be applied successfully in conjunction with domain decomposition.

Mathematical model

The conservation equations of mass, momentum, energy and species are applied to describe the turbulent reacting flow. Favre averaging is applied to obtain density averaged equations. The Favre-averaged incompressible (variable density) Navier-Stokes equations are solved for *conservation of momentum and mass*. The standard high-Reynolds number $k-\varepsilon$ turbulence model is used. Wall-functions are applied to bridge the low-Reynolds number region near

1 Department of Applied Physics, Delft University of Technology, Lorentzweg 1, 2628 CJ Delft, The Netherlands, E-mail : A.Twerda@tn.tudelft.nl
2 Address same as above.
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

the walls [Rod84]. For the *conservation of energy* the transport equation for the enthalpy is solved. The *conservation of species* is modelled with a conserved scalar approach. Here, the concentrations of all species are directly coupled to the mean mixture fraction, which varies from 1 in the fuel-inlet to 0 in air-inlet. Finally, to close the set equations, an equation of state is needed. For this the ideal gas law is used. Because of the strong non-linearity of the thermo-chemical quantities on the mean mixture fraction also the mean mixture fraction variance is needed, and a β- probability density function is used for computing the mean values of these quantities. The chemistry is modelled with an constraint equilibrium model. [Pos88] and [Boe97] give more details about applied models and methods for furnace simulations.

Numerical model

The set of equations described in the previous section is discretized using the *Finite Volume Method* (FVM). The computational domain is divided in a finite number of control volumes (CVs). A cell-centered colocated Cartesian grid arrangement is applied [FP95]. As the equations are solved in the integral form, one must discretize the fluxes through the cell faces. The diffusive fluxes are approximated with the central difference scheme. For laminar flows, the convective fluxes are approximated with the central difference scheme. For turbulent (reacting) flow, a first order upwind scheme is used. For pressure velocity coupling the SIMPLE scheme is applied [Pat80, FP95]. The porosity method is used to match the Cartesian grid to the geometry of the furnace [Pos88].

Domain Decomposition (DD) with minimal overlap is used as parallelisation technique. This technique has proven to be very efficient for creating a parallel algorithm [VTP98]. A grid-embedding technique with static load balancing is applied to divide the global domain into sub-domains (blocks) [CPC91], assigning one processor to every sub-domain. Message passing library MPI is used for communication [GL96].

Multi-grid method

One of the drawbacks of DD is, as implicit solvers are used to solve the discretized set of equations, that the convergence deteriorates as the number of domains increases [PSS92]. Also if the number of CVs increases the convergence becomes more difficult. To minimize both effects, a multi-grid algorithm is implemented. The multi-grid method is applied over the set of equations, which means that *all* equations are solved per prolongation- and restriction-step, rather than applying multi-grid for each equation separately. The Full Approximation Scheme (FAS) is implemented with tri-linear interpolation for restriction and prolongation operators. The V-cycle is applied to determine the sequence in which the grids are visited. It is efficient to start with a good initial guess. A solution on a coarser grid is used and interpolated onto the finer grid. This leads to the Full Multi-Grid method (FMG) [Wes92].

As mentioned in [FP95], the value of some turbulent quantities, like the eddy viscosity, can differ several orders of magnitude over the domain. Computing them directly on the coarse grid could destabilize the multi-grid algorithm. Therefore, these variables are only computed on the finest grid-level and interpolated on the coarser grids. During a multi-grid cycle they are kept constant. Near the solid walls, the variable ε is calculated from wall-functions. This gave rise to large corrections on the coarser grid which destabilised the multi-grid iteration. Therefore, ε is not corrected near a solid wall, but of the fine grid value is interpolated on the coarse grid.

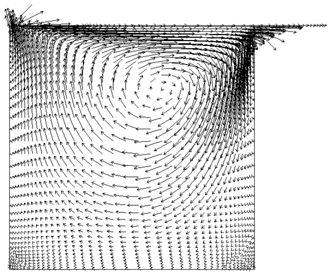


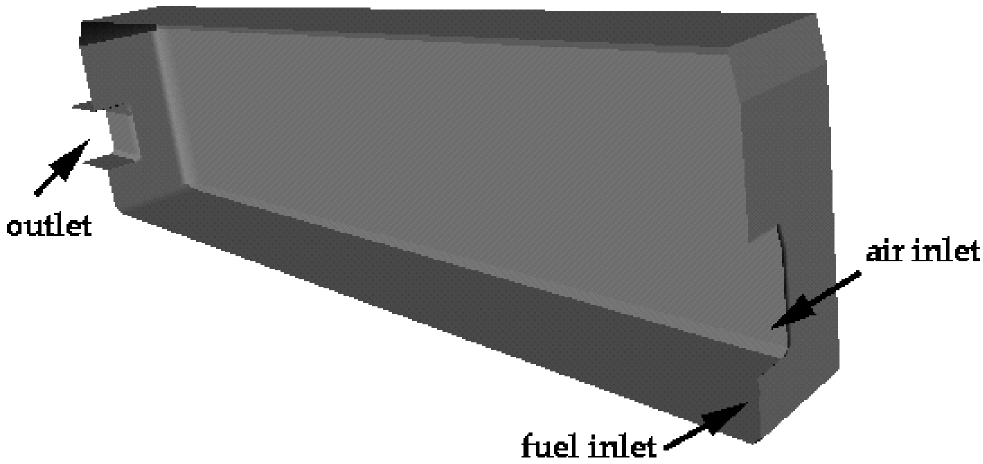
Figure 1 Velocity profile in the $y = 0.5$m plane of the laminar lid-driven cavity with $Re = 100$.

Results

Description of the test cases

The FURNACE code has been validated on a well known CFD test problem: a 3D laminar lid-driven cavity [TF89]. The Reynolds number is based on the uniform top-lid velocity $U_{lid} = 1$m/s, the size of the cavity $L = 1$m and the kinematic viscosity $\nu = 0.01\text{s/m}^2$. Thus, $Re = \frac{U_{lid}L}{\nu} = 100$. Figure (1) shows the characteristic velocity profile of the cavity in the $x-z$ plane for $y = 0.5$m. The number of CVs applied are 4^3 to 64^3 with respectively 2 to 6 levels in the multi-grid calculations. Thus, the coarsest grid-level consisted of 2^3 number of CVs. The residual, $r_h = \frac{\|A_h\phi_h - b_h\|}{\|b_h\|}$ and the relative change of each of the velocity components in a monitor point are used as convergence criteria. The residual and the relative change must be smaller than 10^{-5} for convergence. The first, is a measure for the approximation of the discretized equations, the latter is a measure for the coupling of the equations with respect to each other.

Next, results of a full furnace simulation are presented. The furnace computed is the IFRF glass-melting furnace at IJmuiden (0.440m $\times$ 4.09m $\times$ 0.955m). The inlet consist of two parts, the pre-heated air-inlet $T = 1400$K, $v_{air} = 10$m/s, and a small gas-inlet $T = 370$K, $v_{gas} = 125$m/s. [Boe97] gives more detailed results of these simulations. The geometry of the furnace is plotted in figure (2 left). Because symmetry boundary conditions are applied in the negative x-direction, only half of the furnace is computed. The coarse grid contains ($16\times24\times20$) CVs, which is shown in figure (2 right). The number of grid points for the coarsest grid could not be reduced due to the small radius of the fuel inlet (0.06m). Furthermore, the relative thin flame front would not be correctly captured by a coarser grid. In figure (3) a contour plot of the mean mixture fraction in the symmetry plane of the furnace is shown. On the left, the fuel- and air-inlet, are clear visible. In the outlet, on the right, the flow has almost perfectly mixed.



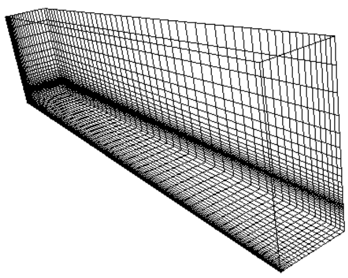


Figure 2 **Left**: Geometry of the IFRF Furnace. **Right**: Coarse grid used for the simulations.

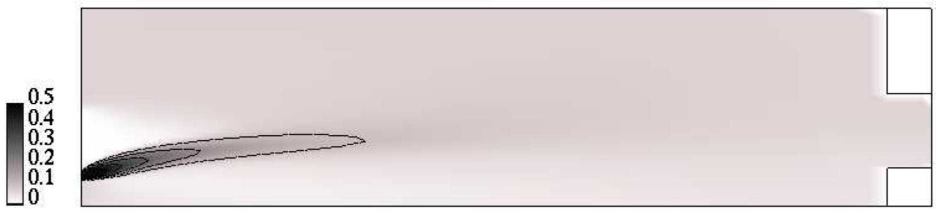


Figure 3 Contour plot of mean mixture fraction in the symmetry plane, of the IFRF furnace. Contour lines denote values of 0.1 to 0.5.

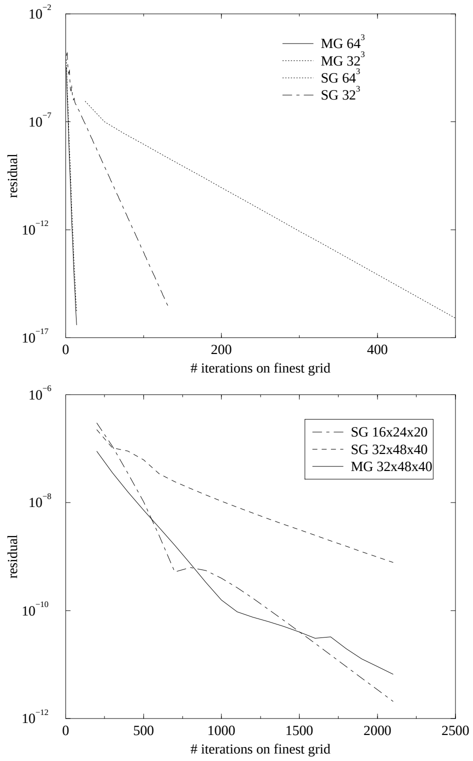


Figure 4 Convergence history of single domain multi-grid and single grid. **Left**: Residual of the velocity components for laminar flow. **Right**: Residual of the mean mixture fraction for a furnace simulation.

Influence of multi-grid on the number of iterations

The residual is plotted in figure (4) versus number of iterations performed on the finest grid-level for both the laminar test case and the furnace. For the cavity flow (left) the residual of the velocity components is plotted. The multi-grid (MG) algorithm converges much faster than the single grid (SG) algorithm. Moreover, the convergence of the MG-algorithm is independent of the number of CVs used, which is the main reason for using the algorithm. The main part of the computation time is spent in calculations on the finest grid, so the acceleration in execution time is of the same order as shown in figure (4), a factor of 25 for the 64^3 grid.

In figure (4 right) the convergence behaviour for the mean mixture fraction equation is plotted. The MG converges better than the SG solver for the finer grid. Also the convergence rate of the finer grid does not differ much from the single coarse grid computation. The acceleration of the MG-algorithm is now approximate a factor of 3, which is less than for the laminar case, but the set of equations for these simulations are more non-linear, which makes the MG-algorithm less efficient.

Influence of number of sub-domains on the multi-grid

The impact of the number of sub-domains is investigated for both cases, using 1 to 16 processors on a CRAY T3E. As one processor is used for every sub-domain, the number of processors can be read where the number of blocks or sub-domains are plotted or mentioned.

For the laminar case, the time per MG-cycle (left) and the number of MG-cycles needed for convergence (right) are plotted in figure (5) for grid sizes 4^3 to 64^3. The latter was only computed with 8 and 16 blocks due to memory restrictions. Because a grid on 1 block consists of at least 1 grid-point, the coarsest grid used is now 4^3. Thus, the number of grid-levels used for the MG-algorithm is 1 for 4^3 grid to 5 for the 64^3 grid. It takes approximately the same time to perform one MG-cycle on a 32^3 grid using one block and on 64^3 grid using 8 blocks. This means that the algorithm is suitable for large grid-sizes because, if the number of blocks increases with problem size, the CPU time per MG-cycle time will remain approximate the same. The number of iterations slowly increases with the number of blocks because implicit solvers are used.

For the 32^3 grid the number of grid-levels used in the MG-algorithm has been varied. The number of iterations needed for convergence (left) and the total execution times (right) are plotted in figure (6). There is an improvement in execution time of a factor 24, from approximate $2900s$ on 1 block without MG, to $120s$ with 8 blocks using 3 levels of multi-grid. A factor of 4 originates from the parallelisation and another factor 6 is gained by the MG-algorithm. Even for the case where no multi-grid is applied, the deterioration of the convergence when more blocks are used is small. This can be explained by the fact that the pressure correction equation in the SIMPLE scheme is solved several times (typically 4). This improves the convergence, especially as DD is applied, as has been shown by [Ver99]. There is a trade-off between MG-levels used in the MG-algorithm and the number of blocks, because the iterations on the coarser grid-levels do not scale properly with large number of blocks, as already shown in figure (5 left).

Finally, the furnace is computed with several sub-domains. The converge history is plotted using 8 and 16 blocks, with two levels of MG being applied. There is a slight influence of the DD on the convergence rate. The single grid computation, with the same number of iterations performed on the fine grid, is especially inferior to the MG computations in the beginning of the iterations, but also in the latter stage the convergence rate is less. Also with more blocks used, the factor gained by using MG is a factor of 3. The time per MG/SG-cycle is shown in table (1). The MG does not increase the time per cycle considerable. The time spend in on

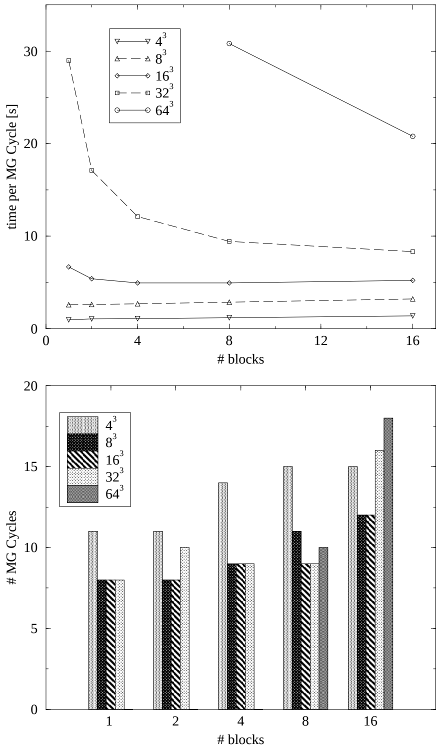


Figure 5 Time per MG-cycle (**left**) and number of iterations (**right**) on different grid sizes using 1 to 16 blocks for the lid-driven cavity flow.

Table 1 Time per SG/MG-cycle for several blocks for the furnace simulation.

Case	Time per cycle [s]
SG 8 blocks	12.97
MG 8 blocks	13.88
MG 16 blocks	8.48

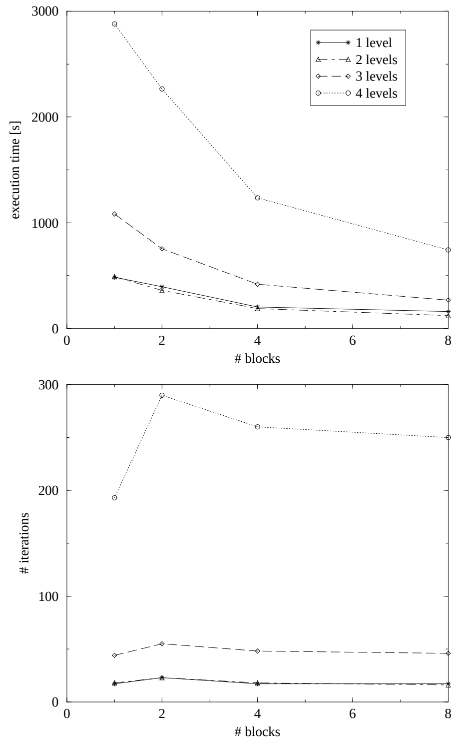


Figure 6 Execution time (**left**) and number of iterations (**right**) for different number of grid levels applied versus the number of blocks

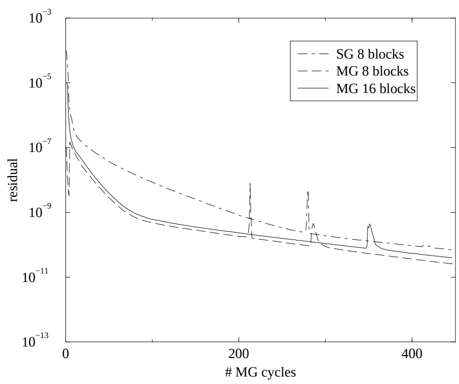


Figure 7 Convergence history of the mean mixture fraction equation for multiple blocks for the furnace simulation.

the coarse grid can be neglected compared with the time spend on the fine grid. The speed-up using 16 in stead of 8 blocks is 1.64 which is moderate and could be matter of concern when using larger number of blocks.

Conclusions

The multi-grid algorithm as described in this paper, has been implemented in the FURNACE code. It has proven to be a good acceleration technique for laminar flow simulations. For turbulent reacting flow simulations an acceleration of a factor 3 for only two grid-levels has been measured.

The multi-grid algorithm also improves the convergence behaviour when domain decomposition is applied. The algorithm is scalable, meaning that if the number of blocks is scaled with the number of control volumes, the time per MG-cycle will stay constant. It can be efficient to use less MG-levels on more blocks, because the iterations on the coarser levels are not scalable to higher number of blocks. The parallel speed-up is moderate, and must be improved if a larger number of blocks is needed.

Acknowledgments

The authors would like to thank the HPαC centre for the computing facilities.

REFERENCES

[Boe97] Boerstoel G. (1997) *Modelling of gas-fired furnaces.* PhD thesis, Delft University of Technology.

[CPC91] Coelho P., Pereira J., and Carvalho M. (1991) Calculation of laminar recirculating flows using a local non-staggered grid refinement system. *Int. J. Num. Meth. Fl.* 12: 535–557.

[FP95] Ferziger J. and Perić M. (1995) *Computational Methods for Fluid Dynamics.* Springer-Verlag, New York.

[GL96] Gropp W. D. and Lusk E. (1996) *User's Guide for* `mpich`*, a Portable Implementation of MPI.* Mathematics and Computer Science Division, Argonne National Laboratory. ANL-96/6.

[Pat80] Patankar S. (1980) *Numerical heat transfer and fluid flow.* McGraw-Hill, London.

[Pos88] Post L. (1988) *Modelling of flow and combustion in a glass melting furnace.* PhD thesis, Delft University of Technology.

[PSS92] Perić M., Schäfer M., and Schreck E. (1992) Computation of fluid flow with a parallel multigrid solver. *Parallel Computational Fluid Dynamics* pages 297–312.

[Rod84] Rodi W. (1984) *Turbulence models and their application in hydraulics–a state of the art review.* IAHR, Karlsruhe.

[TF89] Thompson M. and Ferziger J. (1989) An adaptive multigrid technique for the incompressible navier stokes equations. *Journ. Comp. Phys.* 82: 94–121.

[Ver99] Verweij R. (1999) *Parallel computing for furnace simulations using domain decomposition.* PhD thesis, Delft University of Technology.

[VTP98] Verweij R., Twerda A., and Peeters T. (1998) Parallel computing for reacting flows using adaptive grid refinement. In *Tenth International Conference on Domain Decomposition Methods, Boulder, Colorado, USA.*

[Wes92] Wesseling P. (1992) *An introduction to multigrid methods.* John Wiley & Sons, Chichester.

63

Parallel Domain Decomposition for Reaction-Diffusion Problems

N. VOLFOVSKY [1], M. BERCOVIER [2]

INTRODUCTION

A parallel domain decomposition technique has been applied to construct a solution of 3d non linear reaction-diffusion problems arising in the mathematical modeling of calcium dynamic in neurons. The present method defines the domain decomposition iterations not at each time step of computation but at automatically determined common times for updating the individual subdomains boundary conditions. This method uses over-lapping not necessarily "mesh consistent" geometries. Thus such an approach provides both different space and different time resolution on each subdomain.

The first extension of the classical Schwarz alternative algorithm to a class of parabolic equations appeared in the paper of [Lio88, Lio89]. The basic idea is to divide the domain into several overlapping subdomains and then to solve the parabolic problem in each subdomain alternatively with boundary information from the neighboring subdomains.

There are basically two approaches for which a domain decomposition method can be used to solve a nonlinear problem. The first approach is to locally linearize the nonlinear equation and then to solve the resulting linearized problems at each nonlinear iteration by a domain decomposition method. The second approach is to use domain decomposition, such as the Schwarz alternating method, directly on the nonlinear problems. In this case, a number of smaller nonlinear problems need to be solved in each domain decomposition iteration. In this paper we focus on the last approach. Inside each subdomain we used the standard FEM package, FIDAP 7. 5 [Eng96] which was modified for PVM [Gei94] parallelization [Ah96] and was supplemented by the options of time management introduced in [Ber97]. Convergence

1 Institute of Computer Science, The Hebrew University of Jerusalem, Jerusalem 91904, Israel
2 Address same as above.
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

analysis of the algorithm is based on the convergence theorem [Ran93] and was presented in [Ber97]. The computations were performed on the SP2 machine of High Power Computing Center of Israel.

The parallel implementation of the algorithms greatly reduces the computational time. However the problems of load balancing and synchronization in solving complex nonlinear problems on separate domains become dominant through the inter-processor communication. The present domain decomposition method concentrates on the time step management.

Numerical tests of the algorithm have been undertaken. It took several days to calculate the general problem described here without parallel domain decomposition on an Indy SGI workstation. Parallel computations on a SP2 machine gave us the results in hours. The algorithm presented here is fully parallel relative to the number of domains, that is the total solution time is the same for 4 domains on 4 processors as it is for 32 domains on 32 processors.

THE MODEL PROBLEM

Consider a bounded domain, Ω of R^3 with the boundary $\partial\Omega$. Then, the full problem to be solved is the following : Find u_i (i =1,..n), real valued functions, defined on Ω and satisfying :

$$\begin{cases} \partial_t u_l - \nu_l \bigtriangleup u_l = \partial_t f_l \quad \text{in} \quad \Omega \times (0,T] \\ \partial_t f_l = \sum_{m=1}^{n} (-\lambda_m u_l u_m + \mu_m u_{lm}) \quad \text{in} \quad \Omega \times (0,T] \\ \partial_n u_l = g_l \quad \text{on} \quad \partial\Omega \times (0,T] \\ u_l|_{t=0} = u_{l0} \quad \text{in} \quad \Omega \end{cases} \tag{1}$$

where u_l is the concentration of the l-th component of the system, ν_l is the corresponding diffusion coefficient, where $\partial_t f_l$ denotes its reaction with the rest of the system, and where λ_{lm} and μ_{lm} are the rate constants and u_{lm} is the binding complex of components l and m .

GENERAL ALGORITHM

We consider the domain decomposition of the parent domain Ω into M subdomains ($\{\Omega_i\}_{i=1}^{M}$) in such a way that each pair of adjacent subdomains sharing a common boarder will have a non-empty interior intersection. Thus,

$$\begin{aligned} &\Omega = \bigcup_{i=1}^{M} \Omega_i \\ &\partial\Omega_i \cap \partial\Omega_j \neq \emptyset \quad \Rightarrow \quad \Omega_i \cap \Omega_j \neq \emptyset \quad \text{for all} \quad i \neq j \end{aligned} \tag{2}$$

where $\Omega_i \subset R^3$ is the i-th open subdomain with the boundary $\partial\Omega_i$. The steps followed to define the domain decomposition scheme are as follows

Step 1: Initialization and definitions

- set dt_i the time step associated with the various reactions in the i th subdomain. Note that this " internal " time step can vary from subdomain to subdomain.
- set T - the global time over all subdomains and let $dT_n = T_{n+1} - T_n$ be the time step for updating the information between the subdomains.
- set $u_i(0) = u_0$ in Ω_i
- let k be the current number of the domain decomposition iterations
- let j be the current number of of the internal iterations on each subdomain.

Step 2: Domain-decomposition iteration

- for k,n solve M subproblems:
  Find u_i^k, a real valued functions, defined on Ω_i and satisfying :

$$\begin{cases} \partial_t u_i^k - \nu_i \bigtriangleup u_i^k = \partial_t f_i & \text{in} \quad \Omega \times (T_n, T_{n+1}] \\ \partial_n u_i^k = g_i \quad \text{on} & (\partial\Omega_i \cap \partial\Omega) \times (T_n, T_{n+1}] \\ \partial_n u_i^k = u_*^k \quad \text{on} & (\partial\Omega_i \backslash \partial\Omega) \times (T_n, T_{n+1}], \end{cases} \tag{3}$$

  where

$$u_*^k = \begin{cases} u_{j_1}^{k-1} & \text{on} \quad \partial\Omega_i \cap \Omega_j \cap [\cup_{j \neq i, J_1} \Omega_{j_1}^c] \quad \text{for all} \quad j_1 \neq i \\ u_{j_1 j_2}^{k-1} & \text{on} \quad \partial\Omega_i \cap \Omega_{j_1} \cap \Omega_{j_2} \cap [\cup_{j \neq i, j_1, j_2} \Omega_j^c] \quad \text{for all} \\ \vdots & \qquad\qquad j_1 \neq j_2 \neq i \\ u_{j_1 \cdots j_{M-1}}^{k-1} & \text{on} \quad \partial\Omega_i \cap \Omega_{j_1} \cap \Omega_{j_2} \cap \ldots \cap \Omega_{j_{M-1}}, \end{cases} \tag{4}$$

  and $u_{j_1 \cdots j_m}^{k-1}$ $(m \leq M-1)$ is selected to satisfy $Min(u_{j_1}^{k-1}, \ldots u_{j_m}^{k-1}) \leq u_{j_1 \cdots j_m}^{k-1} \leq Max(u_{j_1}, \ldots u_{j_m}^{k-1})$.
- a segregated algorithm [Eng96] is used to solve these problems; internal iterations for the current local time step of the implicit Euler integration scheme are stopped when the following convergence criterion is satisfied

$$\frac{\| u_i^{k(j+1)} - u_i^{k(j)} \|}{\| u_i^{k(j)} \|} \leq \epsilon_i \tag{5}$$

  where $u_i^{k(j)}$ is the solution of subprocess i at domain decomposition iteration k at iteration j of the segregated algorithm, and ϵ_i is the tolerance for subdomain computations.
- *management of time steps*
  if the convergence (5) is not attained then
  (i) the solution restarts with the smallest dt_i obtained on the current local step, these dt_i being estimated directly in FIDAP.
  (ii) if (i) is not sufficient for convergence, the solution restarts from the last global time step T_n with a decreased dt_i;
  (iii) if (ii) does not converge, the solution restart s from the same global time step T_n, but with a decreased global time step $dT_n \leq T - T_n$.

Step 3: Global convergence

- the domain decomposition iterations are stopped when the following convergence criterion is met in all subdomains

$$\frac{\| u_i^k - u_i^{k-1} \|}{\| u_i^k \|} \leq \epsilon_T \tag{6}$$

  where ϵ_T is the selected tolerance
- *management of time step*
  if criterion (6) is not satisfied then
  (i) the solution restarts from the last global time step T_n but with the last evaluated internal boundaries conditions (u_*^k).
  (ii) if the number of iteration exceeds the limit, the solution restarts from the last global time step T_n with a decreased global time step dT_n.

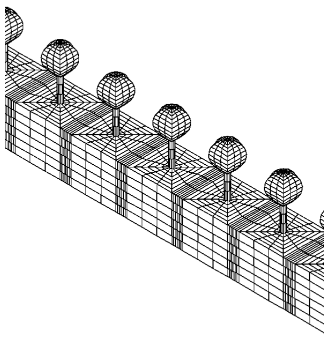


Figure 1 Fragment of the dendrite with eight spines

MODEL OF Ca^{2+} DYNAMICS IN THE DENDRITIC SPINES

The reaction-diffusion problem of Ca^{2+} dynamic in the dendritic spines includes the following processes associated with Ca^{2+} distribution [Gam87, Hol90] : 1) diffusion; 2) Ca^{2+} buffering, 3) Ca^{2+} influx and 4) Ca^{2+} extrusion.

1) The diffusion and 2) Ca^{2+} buffering is modeled by

$$\begin{array}{c} \frac{\partial C}{\partial t} = D \triangle C - k_{+} C \cdot B + k_{-}(BC) \\ \frac{\partial B}{\partial t} = -\frac{\partial (BC)}{\partial t} = -k_{+} C \cdot B + k_{-}(BC) \end{array} \tag{7}$$

where D is the Ca^{2+} diffusion coefficient, C is the Ca^{2+} concentration, B is the free buffer and $BC$is the Ca^{2+} bounded buffer. k_{+} and k_{-} denote the binding and dissociation rate constants, respectively (for parameter values see Table 1).

Boundary conditions of the problem are determined by the following equations; a description of the physical context and real values of the parameters are given in the Appendix.

3) Ca^{2+}influx

$$J_{in} \mid_{\Gamma_{in}} = \alpha_1 \cdot [1 - \alpha_2 ln(\frac{\alpha_3}{C(t)})] \tag{8}$$

4) Ca^{2+} extrusion

$$J_p \mid_{\Gamma} = \frac{\beta_1 \cdot C}{C + \beta_2} \tag{9}$$

where Γ is the spine surface, Γ_{in} is the surface of head of spine.

NUMERICAL TESTS

The numerical example represents a three dimensional model of second-messenger dynamics in the dendrite (Figure 1) . The physical domain is a long wide cylinder with any number of

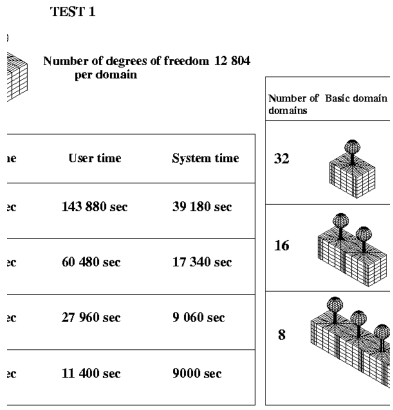


Figure 2 TEST1: A reaction-diffusion problem was solved for a domain with 4, 8, 16, and 32 spines. The DD algorithm was applied for partitions on 4, 8, 16, and 32 domains. TEST2: A reaction-diffusion problem was solved for a domain with 32 spines. The DD algorithm was applied for partitions on 32, 16, and 8 domains

spines. Each spine is the union of a thin cylinder and a ball.
To implement the proposed Domain Decomposition (DD) algorithm, the initial dendritic domain was divided in basic ones, describing a spine head and a fragment of the dendrite.

The numerical tests concerned the numerical efficiency of the present algorithm as well as an original 3d model of Ca^{2+} dynamics in the dendrite. To evaluate the parallel properties of the DD algorithm two different tests were performed and the following computational parameters were discussed: real time of computations, user time - total time of the computation and the system time, which is the communications and I/O time.

First, problems of different sizes were considered. Fragments of dendrite with 4, 8, 16, and 32 spines were divided in 4, 8, 16, and 32 unitary subdomains respectively. Results of the first series of the tests are presented in Figure 2. This indicates that, as expected, an increase in the number of unitary domains (Test1) leads to a proportional increase in user and system time of the simulation , but the "real time" i.e. the actual clock time a user has to wait for an answer is growing significantly less than the computer resources usage.

In the second series of computations a domain with 32 spines (Figure 2) was studied. Three cases of partition were selected on 32, 16, and 8 subdomains, each with one, 2 and 4 spines correspondingly. For all simulations , convergence was defined by requiring that $\epsilon_T = 10^{-3}$.

The second test is significantly different than the first one. Real time of simulation declined proportionally to the increase in the number of subdomains. User time increased with the largest number of subdomains. However, relative changes are small. In these tests, unlike the first ones, the system time decreased with the number of subdomains, which suggests that this parameter is insensitive to the size of the problem on each CPU.

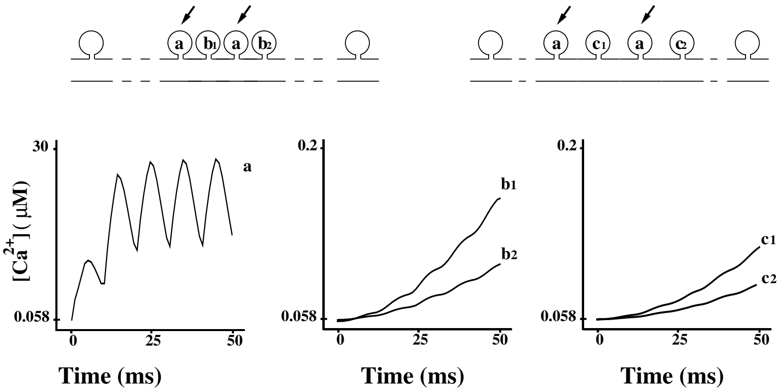


Figure 3 Interaction of the spines with different activation protocol; line **a.** denotes distribution of calcium in the stimulated spine, lines $\mathbf{b}_1$, $\mathbf{b}_2$ denote distribution of calcium in the instimulated spines (distance between spines 0.65μm) and lines $\mathbf{c}_1$, $\mathbf{c}_2$ -the same but with a distance of 0.95μm between the spines

A specific characteristic of the DD algorithm presented here is that it accepts different time steps on separate subdomains, such as in the problem of chemical interactions of dendritic spines with different activation protocols. The results of simulations are shown on Figure 3.

The dynamics of calcium (Ca^{2+}) was monitored in a spine which was not stimulated at all and only two neighbors were stimulated (an index 1 denotes a spine whose two neighbors are stimulated and an index 2 when only one neighbor is stimulated.) Five depolarizing pulses of 5msec were given at 100 Hz. The accumulation of Ca^{2+} in nonstimulated spines shows interspine interaction, which is one of experimentally unsolved problems of Ca^{2+} dynamics in neurons. As expected this interaction is more profound at the high density of 0.65 μm interval between spines ($\mathbf{b}_1$ and $\mathbf{b}_2$). At the lower density of 0.95 μ m ($\mathbf{c}_1$ and $\mathbf{c}_2$) interval the interaction is less profound, but clearly apparent.

The main conclusions of this interdisciplinary research are : (1) the present DD algorithm is fully parallel relative to the number of domains and (2) parallel domain decomposition methods provide an efficient procedure for solution of reaction-diffusion problems arising in the modeling of complex biological systems.

APPENDIX

Boundary conditions and description of parameters.

Boundary conditions are determined by the influx equation on the spine head boundary and extrusion equation on the all boundary of the spine.

The Ca^{2+} *flux* through voltage dependent channels situated in the membrane of the spine head and dendrite is described by

$$J_{in} \mid_{\Gamma_{in}} = g_{max} \cdot [V - \frac{RT}{ZF} ln(\frac{C_{out}}{C(t)})] \tag{10}$$

where Γ_{in} is the segment of the membrane on the spines head and on the dendrite which include voltage-dependent channels, V is the membrane potential, C_{out} is the extracellular concentration of Ca^{2+}, g_{max} is the maximal conductance, R is the gas constant, T is the absolute temperature (in degrees Kelvin), Z is the valence of Ca^{2+}, F is the Faraday constant, and $C(t)$ is the concentration of intracellular Ca^{2+} beneath the head membrane at time t.

The Ca^{2+} *extrusion* which is conducted by pumps located throughout the entire spine membrane is given by

$$J_p \mid_\Gamma = \frac{x_p \cdot C}{C + k_p} \tag{11}$$

where Γ is the spine surface.

Table 1 Parameters of the model

Parameter	Description	Value
D	Diffusion coefficient	$4 \cdot 10^{-6}\ cm^2 s^{-1}$
C_0	Initial concentration of Ca^{2+}	$0.06\ \mu M$
B_{cl0}	Initial concentration of calcineurin	$10\ \mu M$
B_{cn0}	Initial concentration of calmodulin	$25\ \mu M$
k_{cl+}	on rate constant of calcineurin	$50\ \mu M^{-1} s^{-1}$
k_{cl-}	off rate constant of calcineurin	$25\ s^{-1}$
k_{cn+}	on rate constant of calmodulin	$50\ \mu M^{-1} s^{-1}$
k_{cn-}	off rate constant of calmodulin	$500\ s^{-1}$
x_p	maximal rate of extrusion	$15\ \mu M s^{-1}$
k_p	half saturation constant	$0.9\ \mu M$

REFERENCES

[Ah96] Aharon S., Bercovier M., and Parnas H. (1996) Parallel computation enables precise description of Ca^{2+} distribution in nerve terminals. *Bull. Math. Biol.* 58: 1075–1097.

[Ber97] Bercovier, M., Volfovsky, N., and Parnas, H. (1997) Parallel domain decomposition for reaction-diffusion problems: theory and experiments. In Bristeau, M-O., Etgen, G., Fitzgibbon, W., Lions, P.L., and Wheeler, M.F (ed) *Computational Science for the 21st century.* INRIA, University of Houston and GAMNI/SMAI.

[Eng96] Engelman, M.S. (1996) *FIDAP Theoretical Manual, version 7.5.* FDI, Evanston, IL.

[Gam87] Gamble, E. & Koch, C. (1987) The dynamics of free calcium in dendritic spines in response to repetitive synaptic input. *Science* 236: 1311–1315.

[Gei94] Geist, A., Beguelin, A., Dongarra, J., Jiang, W., Manchek, R., and Sunderam, V. (1994) *PVM 3 user's guide and reference manual.* Oak Ridge National Laboratory, Oak Ridge, Tennessee 37831.

[Hol90] Holmes W.R. & Levy W.B. (1990) Insights into long-term potentiation from computational models of nmda receptors-mediated calcium influx and intracellular calcium concentration changes. *J Neurophysiol* 63: 1148–1168.

[Lio88] Lions, P.L. (1988) On the Schwarz alternating method. i. In Glowinski, R., Golub, G.H., Meurant, G.A., and Périaux J. (ed) *1st Int. Symp. on Dom. Decomp. Meth. for PDE.* SIAM, Philadelphia.

[Lio89] Lions, P.L. (1989) On the Schwarz alternating method. ii: Stochastic interpretation and order properties. In Chan, T., Glowinski, R., Périaux, J., and Widlund, O. (ed) *Domain Decomposition Methods*. SIAM, Philadelphia.
[Ran93] Rannacher, R. (1993) Domain decomposition in the nonstationary streamline diffusion finite element method. In Neitaanmäki P. (ed) *Proc FEM 50 Fifty Years Finite Element Method,*.

64

FETI domain decomposition algorithms for sensitivity analysis in contact shape optimization

Vít Vondrák, Zdeněk Dostál [1], John Rasmussen [2]

Introduction

In this paper, we show that the computational cost of the contact shape optimization may be essentially reduced by the application of a domain decomposition method to the solution of the state variational inequality that describes the equilibrium of a system of elastic bodies. In particular, we describe an algorithm for the minimization of a compliance of one body in a coercive system of bodies during their mutual contacts. After discretization by the finite element method, the algorithm uses a feasible directions method for minimization of the cost functional.

To evaluate gradients of the cost function that are necessary for implementation of the feasible direction method, we describe two different methods for sensitivity analysis. The first one, the so-called overall finite difference method, is based on a simple approximation of partial derivatives of the cost function by the finite differences. It turns out that the decomposition of the stiffness matrices of bodies that have prescribed shape is carried out only once, so that the proposed method of solution of the discretized variational inequality can partly exploit the specific structure of the shape optimization problem. However, the stiffness matrix of the body whose shape is to be designed must be decomposed for each design variable in each design step. This leads naturally to application of the semianalytic method [OL94] that, though algebraically more complicated, works in each design step with

1 Department of Applied Mathematics, Technical University in Ostrava, 17. listopadu, CZ-708 33 Ostrava-Poruba,Czech Republic
2 Institute of Mechanical Eng., Aalborg University, Pontoppidanstraede 101,DK-9220 Aalborg, Denmark
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

only one stiffness matrix, so that only one decomposition of one block of the block diagonal stiffness matrix is necessary to carry out one design step, regardless of the number of design variables. The state variational inequality is then solved by an efficient algorithm [ZAS98]

The algorithm has been implemented into the system ODESSY [OL94, RLO93] developed at the Institute of Mechanical Engineering in Aalborg and tested on several model problems. Results of numerical experiments indicate that there are problems for which the algorithm presented is effective.

Discretized contact shape optimization problem

We shall start our exposition from the discretized contact problem. Suppose that $\mathbf{K}$ is the stiffness matrix of order n resulting from the finite element discretization of a system of elastic bodies $\Omega_1, \ldots, \Omega_p$ with enhanced bilateral boundary conditions. With a suitable numbering of nodes, we can achieve that $\mathbf{K} = \mathrm{diag}\,(\mathbf{K}_1, \ldots, \mathbf{K}_p)$, where each $\mathbf{K}_i$ denotes a band matrix which may be identified with the stiffness matrix of the body Ω_i. We assume that $\mathbf{K}$ is positive definite. Let us denote by $\mathbf{f}$ the vector of nodal forces.

Let us now assume that the shape of the first region Ω_1 depends on a vector of design variables α, so that the energy functional will have the form

$$J(\mathbf{u}, \alpha) = \frac{1}{2}\mathbf{u}^T\mathbf{K}(\alpha)\mathbf{u} - \mathbf{f}^T(\alpha)\mathbf{u} \tag{1}$$

where the stiffness matrix $\mathbf{K}(\alpha)$ and possibly the vector of nodal forces $\mathbf{f}(\alpha)$ depend on α. The matrix $\mathbf{N}$ and the vector $\mathbf{c}$ that describe the linearized incremental condition of non-interpenetration also depend on α, so that the solution $\mathbf{u}(\alpha)$ of the contact problem with the region $\Omega_1 = \Omega_1(\alpha)$ satisfies

$$\mathbf{u}(\alpha) = \arg\min J(\mathbf{u}, \alpha) : u \in C(\alpha), \tag{2}$$

where

$$C(\alpha) = \{\mathbf{u} : \mathbf{N}(\alpha)\mathbf{u} \leq \mathbf{c}(\alpha)\}.$$

More details about formulation and discretization of contact problems may be found in Kikuchi and Oden [KO88] or Hlaváček at al [HHNL88].

We shall consider the contact shape optimization problem to find

$$\min\{\mathcal{J}(\alpha) : \alpha \in D_{adm}\} \tag{3}$$

where $\mathcal{J}(\alpha)$ is the cost functional that derives optimality criterion for design of body $\Omega_1(\alpha)$. The set of admissible design variables D_{adm} defines all feasible designs. For example, if the cost functional is defined by $\mathcal{J}(\alpha) \equiv J(\mathbf{u}, \alpha)$, then the minimal compliance problem is obtained. Set of admissible design parameters could be given by

$$D_{adm} = \{l \leq \alpha \leq r : \mathrm{vol}(\Omega(\alpha)) \leq \mathrm{vol}(\Omega(0))\} \tag{4}$$

It has been proved that the minimal compliance problem has at least one solution and that the functional $J(\mathbf{u}, \alpha)$ considered as a function of α has derivatives under natural assumption [HN96].

Duality-based sensitivity analysis

The goal of the sensitivity analysis is to find the influence of design change to the solution of state problem and to the value of the cost function. It means, that we are looking for the directional derivative of solution of the state problem

$$\mathbf{u}'(\alpha,\beta) = \lim_{t\to 0+} \frac{\mathbf{u}(\alpha+t\beta)-\mathbf{u}(\alpha)}{t} \tag{5}$$

where β denotes direction of this directional derivative which is substituted during computation by vectors $\Delta\alpha = (0,\ldots,0,\Delta\alpha_i,0,\ldots,0)^T$ for $i=1,\ldots,k$, where k is the number of design variables that control the design of bodies.

The simplest method for computation of this derivative is to use the overall forward finite difference approximation $\Delta\mathbf{u}/\Delta\alpha_i$ to the design sensitivity $\partial\mathbf{u}/\partial\alpha_i$ that is given by

$$\frac{\partial\mathbf{u}(\alpha)}{\partial\alpha_i} = \frac{\Delta_i\mathbf{u}(\alpha)}{\Delta\alpha_i} = \frac{\mathbf{u}(\alpha_1,\ldots,\alpha_i+\Delta\alpha_i,\ldots,\alpha_k)-\mathbf{u}(\alpha_1,\ldots,\alpha_k)}{\Delta\alpha_i} \tag{6}$$

It follows that the overall finite difference method for evaluation of the gradient of $\mathbf{u}$ as a function of the design variables α requires $k+1$ solutions of (2). An unpleasant complication is that the Hessian of the quadratic form (1) is different for each auxiliary problem so that we have to carry out $k+1$ times the decomposition of the block K_d that corresponds to the body whose shape is to be computed.

This drawback may be removed by extending the analytic or semianalytic method of sensitivity analysis for problems with state equality [OL94] in contact problems [HN96]. In the rest of this section, the semianalytic aproach will be described.

The Lagrange function of the problem (2) has the form

$$\mathcal{L}(\mathbf{u},\mathbf{x},\alpha) = \frac{1}{2}\mathbf{u}^T\mathbf{K}(\alpha)\mathbf{u} - \mathbf{f}^T(\alpha)\mathbf{u} + \mathbf{x}^T\left(\mathbf{N}(\alpha)\mathbf{u}-\mathbf{c}(\alpha)\right) \tag{7}$$

where $\mathbf{u}$ and $\mathbf{x}$ also depend on the vector of design variables α. For the problem (2) we can prescribe Karush-Kuhn-Tucker conditions in following terms

$$\begin{aligned}
&\mathbf{K}(\alpha)\mathbf{u} = \mathbf{f}(\alpha) - \mathbf{N}^T(\alpha)\mathbf{x}\\
&\mathbf{N}(\alpha)\mathbf{u} - \mathbf{c} \le \mathbf{o}\\
&\lambda \ge \mathbf{o}
\end{aligned} \tag{8}$$

Let the set $I = \{i : \mathbf{n}_{i*}(\alpha)\mathbf{u} = \mathbf{c}_i(\alpha)\}$ denote set of indices of nodal variables in contact, let $\mathbf{n}_{i*}(\alpha)$ denote the i^{th} row of matrix $\mathbf{N}(\alpha)$ from the problem (2) and let vector $\mathbf{u}$ denote solution of the state problem (2). Further, for analysis of all contact cases we divide the set I to the two sets

$$\begin{aligned}
I_s &= \{i : i\in I \wedge \mathbf{x}_i > 0\}\\
I_w &= \{i : i\in I \wedge \mathbf{x}_i = 0\}
\end{aligned} \tag{9}$$

where I_s is the set of indices of nodal variables in, so called, strong contact, I_w is the set of indices in weak contact and $\mathbf{x}$ is the solution of the dual formulation of the state problem (2). After formal differentiation of conditions (8) and after some simplification we obtain the new problem

$$\min_{\mathbf{z}\in G(\alpha,\beta)} \mathcal{H}(\alpha,\beta) \tag{10}$$

where

$$\begin{aligned}
\mathcal{H}(\alpha,\beta) &= \frac{1}{2}\mathbf{z}^T\mathbf{K}(\alpha)\mathbf{z} - \mathbf{z}^T\left(\mathbf{f}'(\alpha,\beta) - \mathbf{K}'(\alpha,\beta)\mathbf{u} - \mathbf{N}'^T(\alpha,\beta)\mathbf{x}\right)\\
G(\alpha,\beta) &= \big\{\mathbf{z} : \mathbf{n}_{j*}(\alpha)\mathbf{z} \le \mathbf{f}'(\alpha,\beta) - \mathbf{n}'_{j*}(\alpha,\beta)\mathbf{u} \quad \text{for} \quad j\in I_w,\\
&\qquad \mathbf{n}_{j*}(\alpha)\mathbf{z} = \mathbf{f}'(\alpha,\beta) - \mathbf{n}'_{j*}(\alpha,\beta)\mathbf{u} \quad \text{for} \quad j\in I_s\big\}
\end{aligned} \tag{11}$$

Symbols $\mathbf{K}'(\alpha,\beta)$, $\mathbf{f}'(\alpha,\beta)$ and $\mathbf{N}'(\alpha,\beta)$ represent directional derivatives in direction β that have the same definition as $\mathbf{u}'(\alpha,\beta)$. At this place it is important to notice that these derivatives can be simply evaluated. It has been proved [HN96] that the solution of this problem is the directional derivative $\mathbf{u}'(\alpha,\beta)$ of solution of problem (2).

Let us make some notifications for simplifying the problem (10)

$$\begin{aligned}
&\tilde{\mathbf{f}}(\alpha,\beta) = \mathbf{f}'(\alpha,\beta) - \mathbf{K}'(\alpha,\beta)\mathbf{u} - \mathbf{N}'^T(\alpha,\beta)\mathbf{x} \\
&\mathbf{N}_w(\alpha) = (\mathbf{n}_{j*}(\alpha))_{j\in I_w}, \qquad \mathbf{c}_w(\alpha,\beta) = \left(\mathbf{f}'(\alpha,\beta) - \mathbf{n}'_{j*}(\alpha,\beta)\mathbf{u}\right)_{j\in I_w} \\
&\mathbf{N}_s(\alpha) = (\mathbf{n}_{j*}(\alpha))_{j\in I_s}, \qquad \mathbf{c}_s(\alpha,\beta) = \left(\mathbf{f}'(\alpha,\beta) - \mathbf{n}'_{j*}(\alpha,\beta)\mathbf{u}\right)_{j\in I_s}
\end{aligned}$$

where $\mathbf{N}_w(\alpha)$,$\mathbf{N}_s(\alpha)$ are matrices that decompose the original matrix $\mathbf{N}(\alpha)$ of contact conditions from problem (2) and $\mathbf{c}_w(\alpha,\beta)$,$\mathbf{c}_s(\alpha,\beta)$ are vectors of dimensions corresponding to number of rows of matrices $\mathbf{N}_w(\alpha)$,$\mathbf{N}_s(\alpha)$. Then, we can rewrite problem (10) in the form

$$\min_{\mathbf{z}\in\tilde{G}(\alpha,\beta)} \tilde{\mathcal{H}}(\alpha,\beta) \tag{12}$$

where

$$\begin{aligned}
\tilde{\mathcal{H}}(\alpha,\beta) &= \frac{1}{2}\mathbf{z}^T\mathbf{K}(\alpha)\mathbf{z} - \tilde{\mathbf{f}}^T(\alpha,\beta)\mathbf{z} \\
\tilde{G}(\alpha,\beta) &= \{\mathbf{z} : \mathbf{N}_w(\alpha)\mathbf{z} \le \mathbf{c}_w(\alpha,\beta), \mathbf{N}_s(\alpha)\mathbf{z} = \mathbf{c}_s(\alpha,\beta)\}
\end{aligned}$$

It is easy to see that the last problem is again a quadratic programming problem with linear constraints in the form of equalities and inequalities. Using the theory of duality, we can convert our problem to the following problem

$$\Phi(\lambda) \to \min \quad \text{subject to} \quad \lambda \ge \mathbf{o}, \mathbf{M}\lambda = \mathbf{d} \tag{13}$$

where

$$\Phi(\lambda) = \frac{1}{2}\lambda^T\mathbf{N}_w(\alpha)\mathbf{K}^{-1}(\alpha)\mathbf{N}_w^T(\alpha)\lambda - \lambda^T\left(\mathbf{N}_w(\alpha)\mathbf{K}^{-1}(\alpha)\tilde{\mathbf{f}}^T(\alpha,\beta) - \mathbf{c}_w(\alpha,\beta)\right)$$

and

$$\mathbf{M} = \mathbf{N}_s(\alpha)\mathbf{K}^{-1}(\alpha)\mathbf{N}_w^T(\alpha), \mathbf{d} = \mathbf{N}_s(\alpha)\mathbf{K}^{-1}(\alpha)\tilde{\mathbf{f}}^T(\alpha,\beta) - \mathbf{c}_s(\alpha,\beta)$$

Finally, the derivative $\mathbf{u}'(\alpha,\beta)$ can be obtained from equation

$$\mathbf{u}'(\alpha,\beta) = \mathbf{K}^{-1}(\alpha)\left(\tilde{\mathbf{f}}(\alpha,\beta) - \mathbf{N}_w^T(\alpha)\lambda\right).$$

The dual problem (13) with simple inequality constraint and linear equality constraint is efficiently solvable by the algorithm using the augmented Lagrangians with adaptive precision control described by Dostál, Friedlander and Santos in [ZAS96]. Thus the semi-analytic method for sensitivity analysis requires solution of k quadratic programming problems (13) with the same matrix

$$\mathbf{K}^{-1}(\alpha) = \operatorname{diag}\left({\mathbf{K}_1}^{-1}(\alpha), ..., {\mathbf{K}_p}^{-1}(\alpha)\right).$$

Using this method we exploit not only the advantages of dual formulation of quadratic programming problems, but we can use the decomposition of matrix $\mathbf{K}(\alpha)$ from the solution of the state problem to the sequence of problems in the semi-analytic sensitivity analysis. Thus the semi-analytic approach requires only one decomposition of the stiffness matrix which compares favorably with k+1 decompositions of the overall finite difference approach.

The directional derivatives obtained by the sensitivity analysis can then be exploited for shape optimization. We use sequential linearization in our experiments. More discussion about the outer minimization procedure may be found in Kirsch [Kir94] or Fancello and Feijóo [FF94].

Numerical experiments

We have tested our algorithm on the solution of a simple model problem. The problem was to find the shape of the lower part of the upper body of the system of elastic bodies in Figure 1 so that the compliance of the system is minimal while the volume of the modified upper body does not exceed the volume of the body in the original design. The system has been discretized by the finite element method, so that the discretized system had 1600 nodal variables with possibly 41 nodes in contact. The latter number is the number of dual variables. The Poisson ratio of both bodies was 0.3, the Young modulus of the upper and the lower body was 210000MPa and 100000MPa, respectively. The distributed force with density -1000MPa was acting on the upper surface of the upper body. The bodies were fixed on the right, while zero normal displacements were prescribed on the left and on the bottom of the lower body. It was also required that the bodies do not penetrate in the reference configuration. The design is controlled by vertical movement of six points that are uniformly distributed on the lower boundary of the upper body.

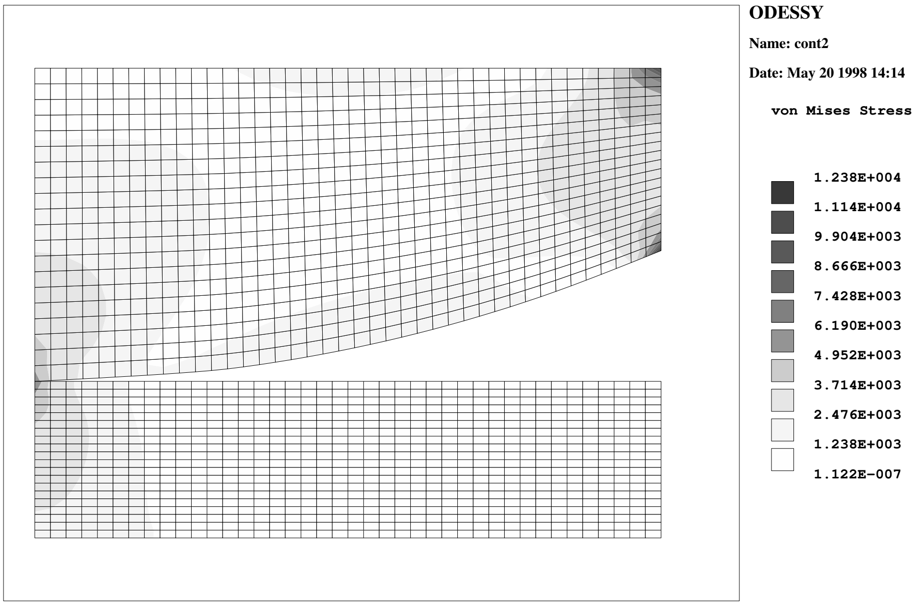


Figure 1 Model problem

The algorithm has been included in the system ODESSY with the overall finite difference sensitivity analysis and sequential linear programming. The performance of the algorithm is given in Table 1 and the final design is depicted in Figure 2. The distribution of the contact pressure is nearly uniform as expected. We explain small variations of the contact stresses by the imposed condition of non-interpenetration in the reference configuration. The computations were carried out on one processor of an IBM SP/2.

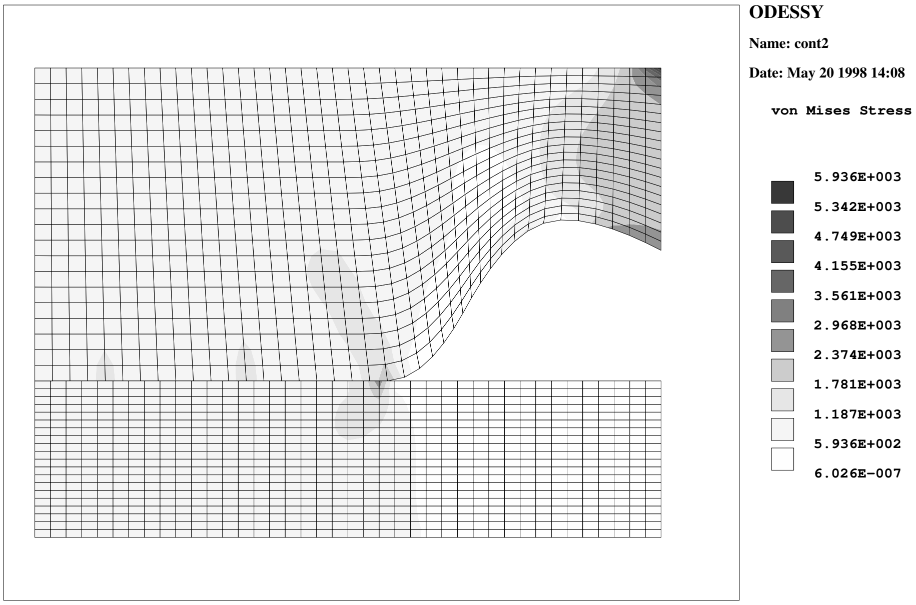


Figure 2 Optimized design

Table 1 Performance in cg iterations and seconds

	Analysis		Sensitivity anal. Average DVs		Total		Value of cost fun.	Change of design
design	iter	time	iter	time	iter	time	$\times 10^4$	%
1	13	3	14	4	96	26	1.238	—
2	11	4	11	4	77	30	1.111	4.34
3	13	4	14	5	96	34	0.988	6.21
4	13	5	15	5	101	33	0.823	8.69
5	15	4	17	5	114	32	0.618	8.31
6	16	5	15	5	106	32	0.465	6.03
7	18	6	20	5	135	37	0.367	6.34
8	14	4	14	4	96	27	0.286	5.25
9	16	4	17	4	117	28	0.281	3.47
10	17	5	16	4	114	29	0.278	0.03
11	13	4	15	4	102	28	0.277	0.00
Total	159	48	166	48	1154	336		

Comments and conclusions

A new duality-based method of solution of strictly convex quadratic programming problems has been applied to the minimization of the compliance of a system of elastic bodies. Theoretical results [ZAS98] guarantee the convergence and the robustness of the method. The method has been applied to minimization of compliance of a system of elastic bodies and the efficiency of the method has been confirmed by results of a numerical experiment. The method may be extended to the solution of problems with friction [DV97b, DV97a] and to the solution of semicoercive problems [ZAS96, ZAS98]. The implementation of the algorithm to the solution of these problems is in progress together with implementation of the semianalytic method of sensitivity analysis and a more sophisticated outer minimization procedure. The salient feature of the algorithms presented is essentially the reduction in the cost of decomposition in preparing domain decomposition based solutions for the state variational inequality that is enabled by the special structure of the problem considered. The algorithm may be more efficient in a parallel environment.

Acknowledgements

This research has been supported by grants GAČR 201/97/0421 and 101/98/0535.

REFERENCES

[DV97a] Dostál Z. and Vondrák V. (1997) Duality based domain decomposition for contact problems with coulomb friction. In 10^{th} *Int. conf. on domain decomposition methods*, page 21. Univ. of Colorado in Boulder, Boulder. Contributed presentation, abstract in Book of Abstracts.

[DV97b] Dostál Z. and Vondrák V. (1997) Duality based solution of cotact problems with coulomb friction. *Arch. Mech.* 3(49): 453–460.

[FF94] Fancello E. A. and Feijóo R. (1994) Shape optimization in frictionless contact problems. *Int. J. Num. Meth. Engng.* (37): 2311–2335.

[HHNL88] Hlaváček I., Haslinger J., Nečas J., and Lovíšek J. (1988) *Solution of Variational Inequalities in Mechanics.* Springer-Verlag, Berlin.

[HN96] Haslinger J. and Neittanmaki P. (1996) *Finite element approximation for optimal shape, material and topology design.* John Wiley & Sons, second edition.

[Kir94] Kirsch U. (1994) Efficient sensitivity analysis for structural optimization. *Comp. Meth. in Appl. Mech. and Engng.* (117): 143–156.

[KO88] Kikuchi N. and Oden J.-T. (1988) *Contact Problems in Elasticity.* SIAM, Philadelphia.

[OL94] Olhoff N. and Lund E. (1994) Finite element based engineering sensitivity analysis and optimization. In Olhoff N. (ed) *Concurrent engineering tools for dynamic analysis and optimization*, volume 2, pages 1–45.

[RLO93] Rasmussen J., Lund E., and Olhoff N. (1993) Parametric modelling and analysis for optimum design. In Herskowicz J. (ed) *Structural optimization 93 - the world congress on optimal design of structural systems*, volume 2, pages 407–414. Rio de Janeiro.

[ZAS96] Z.Dostál, A.Friedlander, and S.A.Santos (1996) Augmented lagrangians with adaptive precision control for quadratic programming with simple bounds and equality constraints. Res. report RP 74/96, IMECC-UNICAMP, Campinas.

[ZAS98] Z.Dostál, A.Friedlander, and S.A.Santos (1998) Solution of contact problems of elasticity by feti domain decomposition. *Contemporary Math.*

(218): 82–93.