

9 Decomposition Algorithms for DDM

Olivier Pironneau¹ and Stéphane Del Pino² and Jacques-Louis Lions³

Control and DDM

We present here decomposition algorithms similar to Schwarz' for the numerical solution of the elliptic and parabolic problems in complex domain. They are well suited to domains described by Constructive Solid Geometry (CSG), set operations on simple shapes, a data structure often used in image synthesis and Virtual Reality[BC94].

We introduce briefly also the decomposition of evolution problems into subproblems on overlapping and nonoverlapping subdomains obtained with Lagrange multipliers without referring to an optimization problem so as to avoid two point boundary value problems.

This paper summarizes several earlier ones, parts of a long term project aimed at solving PDEs with the data structures of VR [LP99b][LP99c] [LP99a] [LP98b] [LP98a] [GLP99] [HLP99] [BLP01]. It is being implemented into freefem3d, a user-friendly, language driven PDE solver. Freefem3d takes VRML[HW96] data and POV-Ray input (<http://www.povray.org>); it uses the fictitious domain method with finite element discretization and it is well suited to DDM[BW86] at the algorithmic level.

Consider the problem of adjusting v so that $y(v)$ be nearest to $y_0 \in L^2(\mathcal{O})$ and subject to

$$y \in V, \quad a(y, \hat{y}) = s(v, \hat{y}) \quad \forall \hat{y} \in V. \quad (1)$$

where

$$s(v, \hat{y}) = \int_{\Sigma} v \hat{y} d\Gamma, \quad v \in L^2(\Sigma) \quad (2)$$

Naturally it can be solved by minimizing

$$J(v) = \frac{\alpha}{2} s(v, v) + \frac{1}{2} \int_{\mathcal{O}} (y - y_0)^2 dx \quad (3)$$

and the method is feasible for any non empty $\mathcal{O} \subset \Omega$.

When $\Omega = \cup \Omega_i$ and $\Omega_i \cap \Omega_j \neq \emptyset$ (see Figure 2 for the notations) we can combine optimal control and domain decomposition.

Consider the solutions of

$$a_i(y_i, \hat{y}_i) = s_i(v_i, \hat{y}_i) + \sum_j \int_{\Gamma_{ij}} \lambda_i \hat{y}_i d\Gamma \quad (4)$$

¹UP6: pironneau@ann.jussieu.fr

²UP6: Université Paris VI

³Collège de France

Let

$$\text{supp } \tau_i \subset \mathcal{O} \cap \Omega_i, \sum \tau_i = 1 \quad c_i(f_i) = \int_{\mathcal{O}} \tau_i f_i^2 dx. \quad (5)$$

$$K(v, \lambda) = \frac{\alpha}{2} \sum_i s_i(v_i) + \frac{1}{2} \sum_i c_i(y_i - y_{oi}) + \frac{1}{2} \sum_{i,j} \|\lambda_i\|_{L^2(\Gamma_{ij})}^2, \quad (6)$$

where $v = \{v_i\} \in \Pi L^2(\Sigma \cap \bar{\Omega}_i)$. We solve

$$\inf_{v, \lambda} K(v, \lambda), \quad \text{subject to } r_{ji}y_i = r_{ij}y_j \quad (7)$$

The key point is to observe that this problem decouples when solved by a gradient method so that the overhead is small when control is added to DDM.

Note that the method works also when the problem is only to solve a PDE like

$$y \in V, \quad a(y, \hat{y}) = (f, \hat{y}) \quad \forall \hat{y} \in V. \quad (8)$$

Then the (virtual) control is an artefact to convert the problem into an optimization problem which decouples on each sub-domain.

Some numerical results are shown on Figure 1 for a Laplace equation (see [LP99c] for more details). It shows that the cost and convergence is comparable to Schwarz' (see [Lio78]).

The same is true of time dependent partial Differential equations (see [LP99a]).

Virtual Control

The previous exercise leads us to investigate "virtual controls" in a more general framework. To solve

$$a_{\Omega}(u, \hat{u}) = (f, \hat{u}) \quad \forall \hat{u} \in V = H_0^1(\Omega). \quad (9)$$

we introduce the *virtual controls* $\lambda_1, \lambda_2 \in L^2(\mathcal{O})$ with $\lambda_1 + \lambda_2 = 0$ (see figure 2). Let a_i be the same operator as a but with integrals on Ω_i . Then solve by a conjugate gradient algorithm for example:

$$\min_{\lambda_i} \mathcal{J}(\lambda) = \frac{\varepsilon}{2} \int_{\mathcal{O}} (\lambda_1^2 + \lambda_2^2) dx + \frac{1}{2} \sum_i \int_{\Gamma_{ij}} u_i^2 \quad (10)$$

subject to

$$a_{\Omega_1}(u_1, \hat{u}_1) = (f_1, \hat{u}_1) + (\lambda_1, \hat{u}_1)_{\mathcal{O}} \quad \forall \hat{u}_1 \in V_1, \quad (11)$$

$$a_{\Omega_2}(u_2, \hat{u}_2) = (f_2, \hat{u}_2) + (\lambda_2, \hat{u}_2)_{\mathcal{O}} \quad \forall \hat{u}_2 \in V_2 \quad (12)$$

with $V_i = H^1(\Omega_i) \cap V$. Notice that the solution to (9) is the sum of functions each having its support in Ω_i (see figure 3): $y = u_1 + u_2$. At the solution $\partial u_1 / \partial n = u_1 = 0$ on $\Omega_2 \cap \partial \Omega_1$

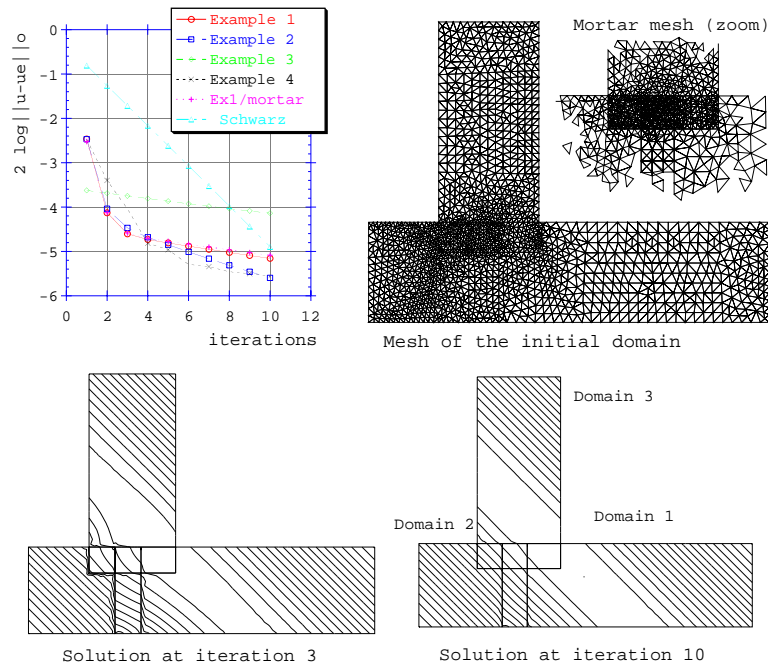


Figure 1: A Laplace equation whose solution is $y = x_1 + x_2$ is solved on a domain decomposed into 3 subdomains by virtual controls on their common boundaries. Several formulations are compared with the Schwarz algorithm. Each domain has its own mesh, so the method is non-conforming

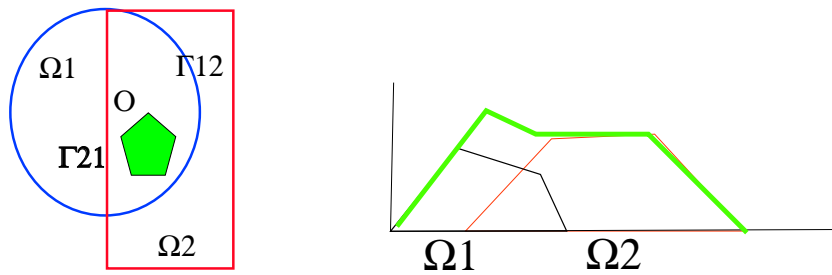


Figure 2: **Left** The virtual control set \mathcal{O} is in $\Omega_1 \cap \Omega_2$. **Right** Decomposition of a function (thick line) into two functions with support each in $\Omega_i, i = 1, 2$

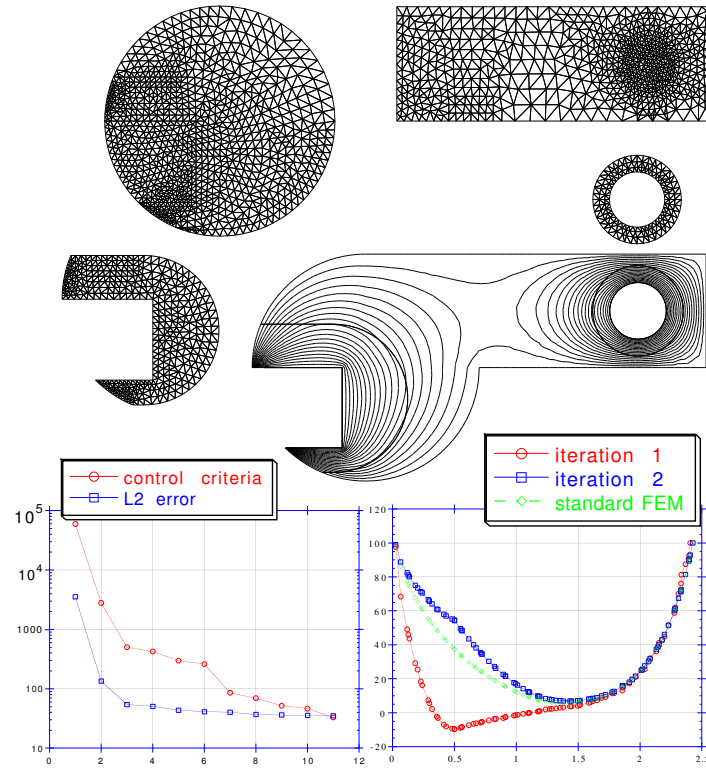


Figure 3: Computation of a Laplace equation on a four piece domain by the method of virtual control and comparison with Schwarz' algorithm

Decomposition of the Space

While introducing the concept of virtual control there was another important idea when we wrote that the solution is the sum of functions with support in each sub-domain; in fact a decomposition of the variational space can be used:

$$H_0^1(\Omega) = H_0^1(\Omega_1) + H_0^1(\Omega_2) \quad (13)$$

Consider again a simple elliptic problem like

$$u|_\Gamma = 0 \quad -\Delta u = f \text{ in } \Omega = \Omega_1 \cup \Omega_2 \quad \Omega_1 \cap \Omega_2 \neq \emptyset \quad (14)$$

$$u = u_1 + u_2, u_i^n|_{\Omega_i} \in H_0^1(\Omega_i).$$

Optimal control is not the only tool to apply the decomposition of the space; the fixed point algorithm for instance works too. Let u_i^n be defined recursively by

$$\begin{aligned} \beta(u_1^{n+1} - u_1^n) - \Delta(u_1^{n+1} + u_2^n) &= f \text{ in } \Omega_1 \\ \beta(u_2^{n+1} - u_2^n) - \Delta(u_1^n + u_2^{n+1}) &= f \text{ in } \Omega_2 \end{aligned} \quad (15)$$

Such an iterative scheme is a sort of regularized Schwarz algorithm (it is Schwarz when $\beta = 0$); converges is shown in [HLP99] for the continuous case and in [BLP01] for the discrete case, the numerical difficulty being the evaluation of mixed integrals like

$$\int_{\Omega_1 \cap \Omega_2} \nabla u_1 \cdot \nabla \hat{u}_2 \quad (16)$$

Convergence

More precisely to evaluate

$$a_h(u_1 + u_2, w_1 + w_2) = a(u_1, w_1) + a(u_2, w_2) + a_h(u_1, w_2) + a_h(u_2, w_1) \quad (17)$$

we use the following quadrature with quadrature points $\{\xi_{jk}^i\}_{j=1..J}$, $i = 1, 2$ in triangle T_k :

$$a_h(u, v) = \frac{1}{2} \sum_k |T_k^1| (\nabla u(\xi_{jk}^1) \cdot \nabla v(\xi_{jk}^1)) + \frac{1}{2} \text{idem on } T_k^2. \quad (18)$$

Proposition(F. Brezzi) *When the quadrature points are the vertices of both triangulations (18) is an admissible quadrature for a and a coercive bilinear forms Furthermore the fixed point algorithm converges when discretized with P^1 elements and the error is optimal.*

Chimera

Chimera as introduced by Steger[SB87] is a Schwarz algorithm with $\Omega = \mathcal{C} \setminus \mathcal{O} = \Omega_1 \cup \Omega_2$. For instance, potential flow around an airfoil involves solving Laplace's equation in a domain outside the airfoil[Pir87].

$$-\Delta \psi_i^{n+1} = f \quad \psi_1^{n+1}|_{\Gamma_{12}} = \psi_2^n \quad \psi_2^{n+1}|_{\Gamma_{21}} = \psi_1^n \quad (19)$$

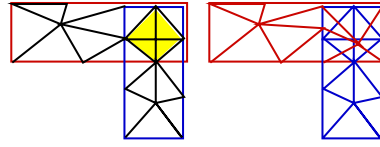


Figure 4: *With non matching grid one must compute integrals of products of functions piecewise linear on each grid.*

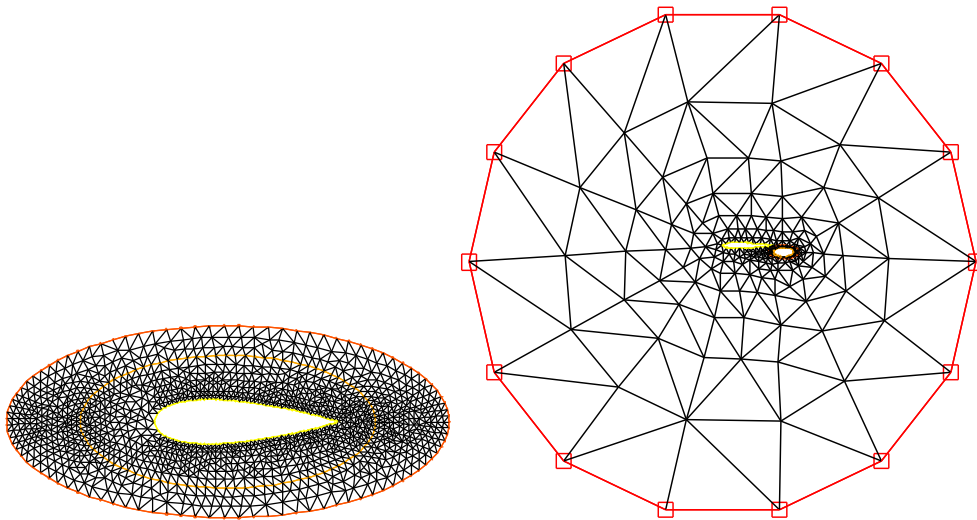


Figure 5: *Meshes and domain decomposition to compute the stream function around a two-pieces airfoil, namely the solution of $\Delta\psi = 0$ with Dirichlet data by the Chimera method. A finer mesh is built around the smaller airfoil (on the left) and a coarse mesh for the rest of the domain, with an elliptic hole in place of the small airfoil (the scale for both domains is not the same on this picture). The whole domain is the union of the fine and coarse domains.*

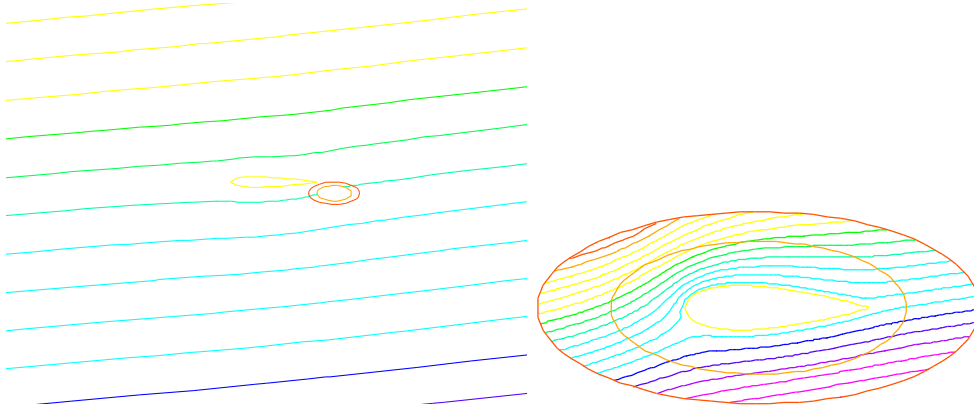


Figure 6: *Stream function around a two-piece airfoil (namely solution of $\Delta\psi = 0$ with Dirichlet data) by the Chimera method (i.e. Schwarz algorithm). The convergence is obtained after 4 iterations.*

Here too we can use a decomposition of the variational space and therefore prove convergence of the Chimera method for arbitrary meshes. In our numerical test the domain is the region outside two airfoils and within a circle which approximates infinity. The finite element method of order one on triangles has been used. The domain is divided in two: a domain near the airfoil which is triangulated with small triangles and the rest of the domain which uses bigger triangles. Here the domain has two airfoils, a large one and a small one. The decomposition must be such that the physical domain is the union of both domain, and the domains must overlap. Then Schwarz algorithm is used with translation and quadratures at the vertices as explained above. Four iterations are sufficient for convergence to machine accuracy (figures 5 and 6.)

Decomposition of Operators

Our last idea is in the family of operator splitting methods. Consider

$$\begin{aligned} a_i(u, \hat{u}) &= \sum \int_{\Omega_i} \rho_i a_{kl} \frac{\partial u}{\partial x_l} \frac{\partial \hat{u}}{\partial x_k}, \\ c_i(u, \hat{u}) &= \int_{\Omega_i} \sigma_i u \hat{u} dx, \\ \rho_1 + \rho_2 &= 1, \quad \sigma_1 + \sigma_2 = 1 \text{ in } \Omega \end{aligned}$$

($\rho_i \geq 0$, $\sigma_i \geq 0$ are extended by 0 outside Ω_i).

Let $\epsilon_1, \epsilon_2, \eta_1, \eta_2$ be positive and small; we now introduce the system

$$\begin{aligned} c_i\left(\frac{\partial u_i}{\partial t}, \hat{u}_i\right) + a_i(u_i, \hat{u}_i) + (\hat{u}_i, \lambda_i - \lambda_j)_{H_{i,j}} &= c_i(f, \hat{u}_i) \\ \epsilon_i\left(\frac{\partial \lambda_i}{\partial t}, \hat{\lambda}_i\right)_{H_{i,j}} + \eta_i b_i(\lambda_i, \hat{\lambda}_i) - (\hat{\lambda}_i, u_i - u_j)_{H_{i,j}} &= 0 \end{aligned}$$

This method works with/without overlapping.

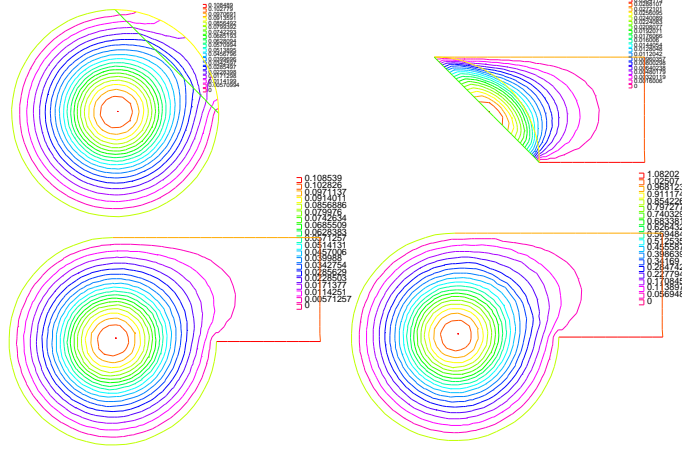


Figure 7: *Decomposition of operator: a Laplace equation solved on a composite domain with overlapping.*

Numerical Example

We consider the heat equation

$$\partial_t u - \Delta u = 1_D \text{ in } \Omega \times (0, T) \quad (20)$$

with zero initial and boundary conditions.

The domain $\Omega = \Omega_1 \cup \Omega_2$ is made of the unit circle centered at the origin and a rectangle $(0, 2) \times (0, 1)$.

The source term is in a disk centered at the origin and of radius 0.4.

The algorithm is

$$\begin{aligned} \frac{1}{\delta t}(u_i^{m+1} - u_i^m) - \Delta u_i^{m+1} &= 1_D + (-1)^i 1_C \lambda, \text{ in } \Omega_i \\ \frac{1}{\delta t}(\lambda^{m+1} - \lambda^m) - \Delta \lambda^{m+1} &= \frac{1}{\epsilon}(u_1 - u_2) \text{ in } C \end{aligned} \quad (21)$$

with Dirichlet conditions on $\partial\Omega$ and Neumann conditions on ∂C , where $C = \Omega_1 \cap \Omega_2$.

The parameters of the computations are $\delta t = 0.015$, $\epsilon = 0.01$ and the results are shown on figure 7. For problems with discontinuous coefficients the method without overlapping is more attractive. We consider the convection-diffusion equation

$$\partial_t u + \mathbf{v} \cdot \nabla u - \nabla \cdot (\nu \nabla u) = 0 \text{ in } \Omega \times (0, T) \quad (22)$$

with initial and boundary conditions.

The problem is in $\Omega = (0, 1) \times (0, 2)$. It is an academic example of the dissipation of a pollutant from an enclosure C into a medium Ω_1 (rock) with low diffusion but cracked (boundary Σ). Furthermore below in Ω_1 in another medium Ω_2 (sand) with large diffusion constant ν_2 , the pollutant is also convected (water in sand) at velocity \mathbf{v} . The velocity derives from a potential ϕ solution of

$$-\nabla \cdot (\mu \nabla \phi) = 0 \text{ in } \Omega_2 \quad \phi|_{x=0} = 0 \quad \phi|_{x=1} = 1 \quad (23)$$

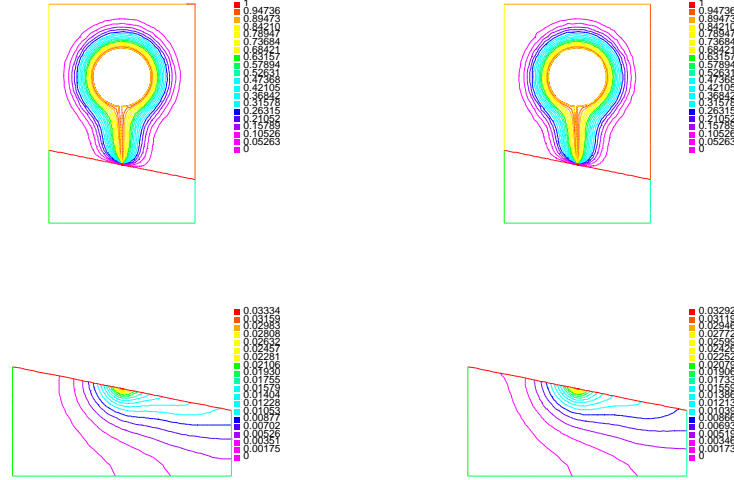


Figure 8: **Top:** reconstructed solution at the two instants $t = 0.3$ and $t = 0.6$. **Bottom:** solution in each subdomain at $t = 0.6$

and $\mathbf{v} = -\mu \nabla \phi$.

Equation (22) is discretized in time by an implicit Euler scheme and in space by the finite element method of degree one on triangles. The convection term is treated by the Galerkin-Characteristic method. Equation (23) is also discretized by the same finite element method.

We have chosen the following Domain Decomposition Method:

$$\frac{1}{\delta t_i} (u_i^{m+1} - u_i^m \circ X^m, \hat{u}_i) + (\nu \nabla u_i^{m+1}, \nabla \hat{u}_i) + b_i(u_i - u_j, \hat{u}_i) = 0 \quad (24)$$

$$\forall \hat{u}_i \in V_i \quad i, j = 1, 2, j \neq i \quad (25)$$

where $u^m \circ X^m(x) \approx u^m(x - \mathbf{v}^m(x)\delta t)$, V_i is the finite element space on Ω_i and

$$b_i(u, v) = \int_S (\alpha_i uv + \beta_i \frac{\partial u}{\partial s} \frac{\partial v}{\partial s}) \quad S = \bar{\Omega}_1 \cap \bar{\Omega}_2 \quad (26)$$

The parameters chosen are:

$$\mu = 10, \nu_1 = 0.001, \nu_2 = 0.05, \delta t_1 = 0.005, \delta t_2 = 0.02, T = 0.15 \quad (27)$$

$$\alpha_1 = 0, \beta_1 = 0.01, \alpha_2 = 100, \beta_2 = 0 \quad (28)$$

The mesh of Ω_1 is 1.5 times finer than the mesh of Ω_2 . The method is not unconditionally stable; we have tried several values for the operator b and not all of them work; but the fact that the coefficients of the PDE are constant in each sub-domain and the inherent parallelism are the two major advantages. The results are on figure 8.

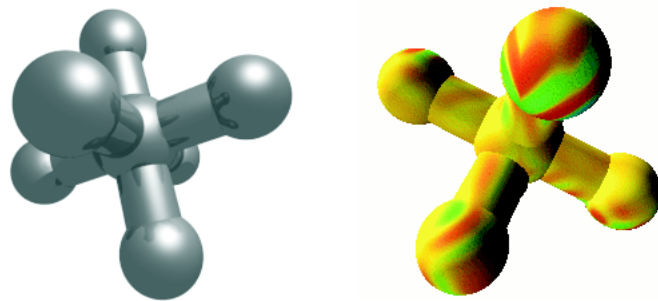


Figure 9: **Left** A scene displayed by *POV-Ray*. The objects are never intersected, it is the graphic rendering that takes care of the problem. **Right** The trace of the real part of the scattered acoustic field on the surface of the geometry [PHPT00]

Computation in Virtual Realities

Parallel computing with non-conforming meshes is easy to implement once a fast and robust interpolator is available to compute functions on all the meshes. Such is the case of *freefem+* [BHOP99], a public domain software written by one of the authors. The previous numerical results of this chapter were obtained with it.

Freefem3d

DDM is potentially useful to speed-up computation of virtual scenes created by Constructive Solid Geometry. However all of them are in 3d, so we are currently developing a 3d version of *freefem*. It has a language which is interpreted using *bison*, it reads geometries created with *POV-Ray* (the file `cross.pov` below for instance) and it uses the Fictitious Domain Embedding Method (FDEM). Results are displayed with IBM's Data Explorer (cf <http://www.dx.com>) or even *Pov-Ray* (Suzuki's path in <http://www.public.usit.net/rsuzuki/e/povray/iso/index.html>)

```
vector a = (0,0,0);
vector b = (1,1,1);
vector n = (100,100,100);
structmesh Mesh(n,a,b);
scene S("cross.pov",Mesh);
array mu(Mesh) = 1;
w = 5;
solve(u) {
u * w^2 + div(mu*grad(u)) = 0;
dnu(u) - I*w*u = I*w *(Nx-1) on<1,0,0>
};
plot(u);
```

Operator Overloading in C++ makes it easy to program the vector case by using templates over the scalar case (*generic programming*). Thus the following is possible with any number

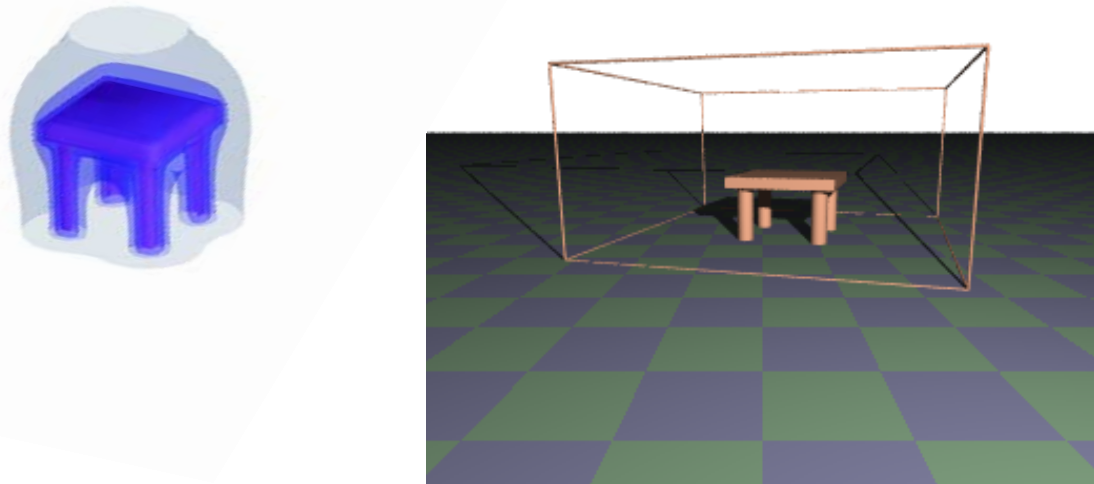


Figure 10: Displayed are 4 iso-temperature surfaces (0.95, 0.5, 0.25, 0.05) for a transient solution of the heat equation at time 0.1, around a table-shaped object at temperature 1 with Neumann conditions on the boundary of the computational domain, (shown on the right with POV-Ray) and initial temperature zero; the program in `freefem3d` language is given above.

of unknowns (2 here):

```
solve(u,v){
  pde(u) - laplace(u) = f1;
  on(a) dnu(u) + v = g;
  pde(v) u - laplace(v) = f2;
  on(a) u + 10*v = h };
```

However it will be quite a challenge to find a general preconditioner for iterative solutions of the linear systems.

We conclude with a last example with the heat equation

```
double i=0; double dt=0.1; do{
  solve(u) { u - div (dt * grad(u)) = u;
            u=1 on <1,0,0>;
            dnu(u) = 0 on Mesh;
          };
  i=i+1;
  dxplot("u.dat",u,Mesh);
}while(i<=5);
```

The results are shown on figure 10

References

- [BC94]Gordon Burden and Philippe Coiffe. *Virtual Reality Technology*. Wiley, Chichester, 1994.
- [BHOP99]Dominique Bernardi, Frederic Hecht, Kohji Otsuka, and Olivier Pironneau. freefem+, a finite element software to handle several meshes. <http://www.freefem.org>, 1999.
- [BLP01]Franco Brezzi, Jacques-Louis Lions, and Olivier Pironneau. Analysis of a chimera method. *C.R.A.S. I*, 332(7):655–664, 2001.
- [BW86]Petter E. Björstad and Olof B. Widlund. Iterative methods for the solution of elliptic problems on regions partitioned into substructures. *SIAM J. Numer. Anal.*, 23(6):1093–1120, 1986.
- [GLP99]R. Glowinski, J.L. Lions, and O. Pironneau. Decomposition of energy spaces and applications. C.R.A.S., Paris, 1999.
- [HLP99]F. Hecht, J.L. Lions, and O. Pironneau. Domain decomposition algorithm for computed aided design. In A. Sequeira et al, editor, *Applied Nonlinear Analysis*, pages 185–198. Kluwer Academic-Plenum Publishers, New York, 1999.
- [HW96]John Hartman and John Wernecke. *The VRML 2.0 Handbook*. Addison Wesley, New-York, 1996.
- [Lio78]Pierre Louis Lions. Interprétation stochastique de la méthode alternée de Schwarz. *C. R. Acad. Sci. Paris*, 268:325–328, 1978.
- [LP98a]J.L. Lions and O. Pironneau. Algorithmes parallèles pour la solution des problèmes aux limites. *C.R.A.S., Paris*, 327:947–952, 1998.
- [LP98b]J.L. Lions and O. Pironneau. Sur le contrôle parallèle des systèmes distribués. *C.R.A.S., Paris*, 327:993–998, 1998.
- [LP99a]J.L. Lions and O. Pironneau. Contrôle virtuel, répliques et décomposition d’opérateurs. *C.R.A.S.*, 1999.
- [LP99b]J.L. Lions and O. Pironneau. Domain decomposition method for CAD. *C.R.A.S., Paris*, 328:73–80, 1999.
- [LP99c]J.L. Lions and O. Pironneau. Domain decomposition methods for cad. *C.R. Acad. Sci. Paris*, 328:73–80, 1999.
- [PHPT00]Stephan Del Pino, Errki Heikkola, Olivier Pironneau, and Jari Toivanen. A finite element method for virtual reality data. *C.R.A.S. I*, 330(12):1107–1115, 2000.
- [Pir87]Olivier Pironneau. *Finite Element Methods for Fluids*. Wiley, Chichester, 1987.
- [SB87]J. Steger and J. Benek. On the use of composite grid schemes in computational aerodynamics. *Comp. Meth. Appl. Mech. Eng.*, 64:301–320, 1987.