Contemporary Mathematics
Volume 157, 1994

Factorization of the Convection-Diffusion Operator and a (Possibly) Non Overlaping Schwarz Method

F. NATAF and F. ROGIER

ABSTRACT. We propose an iterative algorithm which consists, at each step, in the solving of the convection-diffusion equation on each subdomain. Each subproblem is subject to boundary conditions issued from the approximate factorization of the convection-diffusion operator. The method is all the more efficient as the viscosity is small. Numerical tests are shown.

1. Exact Factorization

We propose a domain decomposition method (DDM) to solve the convection-diffusion equation. For an alternating Schwarz method, it is well-knowm that the convergence rate depends crucially on the transmission conditions. In this paper, we use extensively the Wiener-Hopf factorization to design the efficient transmission conditions. In order to simplify the paper, we consider the following model problem set on the domain $]0, L[\times]0, h[$:

$$\begin{gathered} \mathcal{L}(u) = \frac{u}{\Delta t} + a(x,y)\frac{\partial u}{\partial x} + b(x,y)\frac{\partial u}{\partial y} - \nu\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) = f + \frac{u(t-\Delta t)}{\Delta t} \\ u = g_d \text{ on } x = 0,\ y = 0 \text{ and } y = h \\ \frac{\partial u}{\partial n} = g_n \text{ on } x = L \end{gathered} \tag{1}$$

arising from an implicit discretization in time of the convection-diffusion equation. This problem has to be solved at each time step. To design the DDM, we consider temporarily the same problem set on the vertical strip $]0, L[\times\mathbf{R}$. We write a factorization of the operator $\mathcal{L}$. By factorization, we mean the possibility to write $\mathcal{L}$ as the product of two operators which are of order one in the x-direction. One corresponds to a well-posed parabolic problem in the direction of positive x and the other one in the opposite direction. To find the explicit form of the factorization, we consider the constant coefficients operator. We take the Fourier transform in y of (1) (the dual variable is denoted by k):

$$\hat{\mathcal{L}} = \frac{1}{\Delta t} + a\frac{\partial}{\partial x} - i\,b\,k - \nu\frac{\partial^2}{\partial x^2} + \nu k^2$$

1991 Mathematics Subject Classification. Primary 65N55, 47A68, 76R05.
The final version of this paper will be submitted for publication elsewhere

0271-4132/94 $1.00 + $.25 per page

which can be written as $\hat{\mathcal{L}} = -\nu(\frac{\partial}{\partial x} - \lambda^+)(\frac{\partial}{\partial x} - \lambda^-)$, where

$$\lambda^{\pm} = \frac{a \pm \sqrt{a^2 + \frac{4\nu}{\Delta t} - 4i\nu bk + 4k^2\nu^2}}{2\nu} \tag{2}$$

Let Λ^+ and Λ^- be the operators of symbols λ^+ and λ^-, we have

$$\mathcal{L} = -\nu(\frac{\partial}{\partial x} - \Lambda^+)(\frac{\partial}{\partial x} - \Lambda^-) \tag{3}$$

The term λ^+ is positive and the term λ^- is negative whatever the sign of a is. Thus, the operator $\partial_x - \Lambda^+$ (resp. $\partial_x - \Lambda^-$) corresponds to a well-posed parabolic problem in the direction of negative (resp. positive) x. The expressions of λ^+ and λ^- are not polynomial in k. The operators Λ^+ and Λ^- are thus non local.

We consider a domain $]0, L[\times\mathbf{R}$ covered by N vertical strips $[l_i, L_i] \times \mathbf{R}$ $(i = 1, \ldots, N)$with possible overlap of size δ. This factorization suggests the following algorithm:

Let u_i^n be an estimate of the solution in domain i, u_i^{n+1} is defined as:

$$\begin{aligned} &\mathcal{L}(u_i^{n+1}) = f \text{ in domain } i \\ \text{at } x = l_i,\ (i \neq 1)\ &(\frac{\partial}{\partial x} - \Lambda^+)(u_i^{n+1}) = (\frac{\partial}{\partial x} - \Lambda^+)(u_{i-1}^n) \\ \text{at } x = L_i,\ (i \neq N)\ &(\frac{\partial}{\partial x} - \Lambda^-)(u_i^{n+1}) = (\frac{\partial}{\partial x} - \Lambda^-)(u_{i+1}^n) \end{aligned} \tag{4}$$

at $x = 0$ and $x = L$, we use the boundary conditions of the global problem. At each step, N independent subproblems have to be solved. This domain decomposition method is valuable for parallel computation.

We have here an interesting result. In the case where the operators $\partial_x - \Lambda^+$ and $\partial_x - \Lambda^-$ are used as boundary operators at $x = 0$ and $x = L$ respectively, it can be proved that with or without overlap there is convergence in N steps. This is optimal since $\mathcal{L}$ is elliptic in x. Nevertheless, the algorithm (4) can not be used for practical purposes for two reasons. The first one is the non locality of the boundary conditions in each subdomain. The second one, is that for variable coefficients a and b, the explicit form of the factorization is not known. This is why we shall go into approximate factorizations involving local operators.

2. Approximate Factorization

In order to obtain approximate factorizations involving local operators, we have to take Taylor approximations with respect to k of λ^+ and λ^-. We are restricted to at most third order approximations since for higher order approximations, the sign of the approximations of λ^+ (resp. λ^-) is negative (resp. positive) for high wavenumbers. This would yield boundary conditions which lead to ill-posed problems. We do not consider either the approximation of order three since it yields a boundary condition of order three in y. Thus, we simply propose as approximate factorizations (even in the case where a and b are variable)

$$\mathcal{L} \simeq -\nu(\partial_x - \frac{a + \sqrt{a^2 + \frac{4\nu}{\Delta t}}}{2\nu})(\partial_x - \frac{a - \sqrt{a^2 + \frac{4\nu}{\Delta t}}}{2\nu}) \tag{5}$$

$$(6)\qquad \simeq -\nu(\partial_x - \frac{a+\sqrt{a^2+\frac{4\nu}{\Delta t}}}{2\nu} - \frac{b}{\sqrt{a^2+\frac{4\nu}{\Delta t}}}\partial_y)$$

$$(\partial_x - \frac{a-\sqrt{a^2+\frac{4\nu}{\Delta t}}}{2\nu} + \frac{b}{\sqrt{a^2+\frac{4\nu}{\Delta t}}}\partial_y)$$

$$(7)\qquad \simeq -\nu(\partial_x - \frac{a+\sqrt{a^2+\frac{4\nu}{\Delta t}}}{2\nu} - \frac{b}{\sqrt{a^2+\frac{4\nu}{\Delta t}}}\partial_y + \frac{\nu\left(1+\frac{b^2}{a^2+\frac{4\nu}{\Delta t}}\right)}{\sqrt{a^2+\frac{4\nu}{\Delta t}}}\partial_{yy})$$

$$(\partial_x - \frac{a-\sqrt{a^2+\frac{4\nu}{\Delta t}}}{2\nu} + \frac{b}{\sqrt{a^2+\frac{4\nu}{\Delta t}}}\partial_y - \frac{\nu\left(1+\frac{b^2}{a^2+\frac{4\nu}{\Delta t}}\right)}{\sqrt{a^2+\frac{4\nu}{\Delta t}}}\partial_{yy})$$

The algorithm (4) leads obviously to new algorithms depending on the approximate factorization which is used. They read as follows:

$$(8)\qquad \begin{aligned} &\mathcal{L}(u_i^{n+1}) = f \text{ in domain } i \\ &\text{at } x = l_i,\ (i \neq 1)\ \ (\frac{\partial}{\partial x} - \Lambda_j^+)(u_i^{n+1}) = (\frac{\partial}{\partial x} - \Lambda_j^+)(u_{i-1}^n) \\ &\text{at } x = L_i,\ (i \neq N)\ \ (\frac{\partial}{\partial x} - \Lambda_j^-)(u_i^{n+1}) = (\frac{\partial}{\partial x} - \Lambda_j^-)(u_{i+1}^n) \end{aligned}$$

where $j = 0$, 1 or 2 and $\Lambda_j^{\pm}$ are obvious notations for the operators arising from the approximate factorizations. At $x = 0$ and $x = L$, we use the boundary conditions of the global problem. Of course, contrarily to the algorithm (4) based on the exact factorization, the convergence of (8) in a finite number of steps is no more satisfied.

At this point, we should make clear that since the operators (5), (6) and (7) are local, the Fourier transform is not necessary to solve (8). Thus, the proposed algorithm may be written for variable coefficients and a general geometry. In the general case, we still do not have convergence results and we present here convergence results in the constant coefficients case.

3. Convergence Results

When a and b are constant coefficients, we carry out a Fourier analysis of the convergence of the method for $N = 2$. The convergence rate $\rho_j(k)$ (j denotes the order of the approximate factorization used in (8)) in the Fourier space between every two steps is

$$(9)\qquad \rho_j(k) = \left(\frac{\lambda^-(k) - \lambda_j^-(k)}{\lambda^+(k) - \lambda_j^-(k)}\right)^2 e^{(-\frac{a}{\nu}+2\lambda^-(k))\delta}$$

It can be proved that for a.e. k and any δ, $|\rho_j(k)| < 1$. For small wavenumbers, ρ_j tends to zero. For large wavenumbers, ρ_j tends to 1 if $\delta = 0$, and to zero as soon as $\delta \neq 0$ (see fig. 3). This shows the importance of the overlap.

4. Numerical Results

In order to illustrate the validity of the method, a 2D test problem has been performed on the problem (1). All the numerical tests have been implemented on a IPSC2 with 32 processors. The problem that we have tested consists in finding the temperature of a fluid moving at the velocity $\vec{v} = (a, b)$. The computational domain is the square $[0, 1] \times [0, 1]$ (see figure 1) , the viscosity ν is equal to .001.

The picture (fig. 2) shows the isovalues of the temperature at different step time of the algorithm for a velocity $\vec{v} = (1, 0)$ and for an initial condition equal to 1 in the middle of the domain. As the picture shows, the boundary layer at $y = 0$ is taken into account with accuracy along the time. The curvature of the isovalues is due to the effect of the initial condition. We also made computations with a variable velocity a.

The figure 4 has been obtained in the case of a steady-state computation ($\Delta t = \infty$) by changing the order of the approximation. The transmission conditions of order 2 yield an improved convergence since they can take into account the boundary layer at $y = 0$.

From one iteration to the other, only the right hand side of the boundary conditions are modified. That is why we compute the LU factorization of the convection-diffusion problem in each subdomain before beginning the iterative procedure. This is also a parallel task. Then, at each step, this LU factorization is used to solve the subproblems. The last drawing (Table 1) gives the computational time and the convergence rate of the algorithm as a function of the number of processors . The global meshsize is 64×64 and is kept fixed. It can be seen that the computational time is divided by two when the number of processors is doubled. But for $nb = 32$ the computational time is the same as for $nb = 16$. This fact is due to the communication cost between the processors which is then large compared to the computational time for each processor since we have only 4 points along x in each subdomain. The factor three between $nb = 1$ and $nb = 2$ case is due to the fact that with one processor there is no domain decomposition method (DDM) and that a direct method (the LU factorization) is indeed used. While, with two processors, the DDM is used and the LU factorization which is made at the beginning of the iterative algorithm in each subdomain is much less costly that the LU factorization of the global problem with one processor.

CMAP, CNRS, ECOLE POLYTECHNIQUE, PALAISEAU, 91128, FRANCE
Current address: CMAP, Ecole Polytechnique, 91128 Palaiseau Cedex, France
E-mail address: nataf@cmapx.polytechnique.fr

DIVISION CALCUL PARALLÈLE, ONERA, CHÂTILLON, 92322, FRANCE
Current address: Division Calcul parallèle, ONERA, 29 Av. de la division Leclerc 92322 Châtillon, France
E-mail address: rogier@sun01.onera.fr

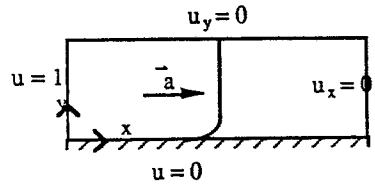
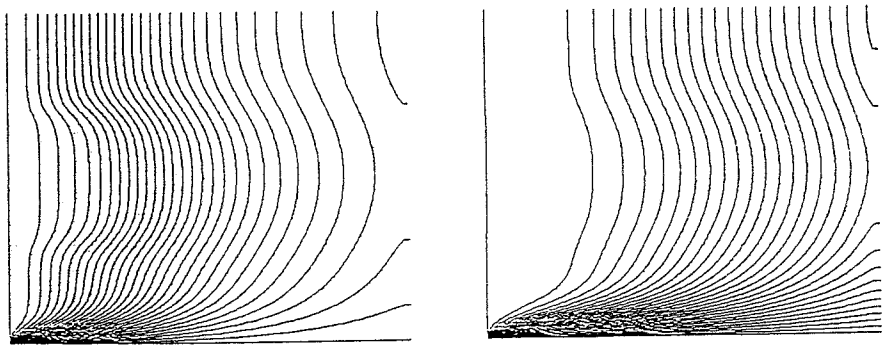


FIGURE 1 - Computational domain



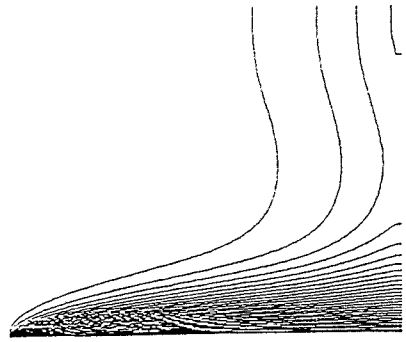


FIGURE 2 - Isovalues of the temperature at time = .2, .4 and .8

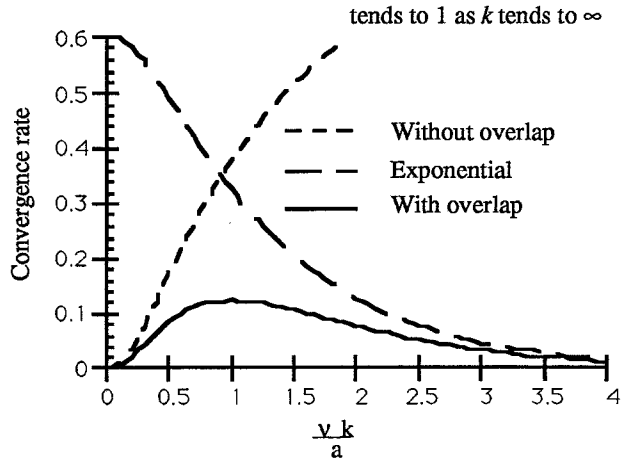


FIGURE 3 - Convergence rate vs. wavenumber k

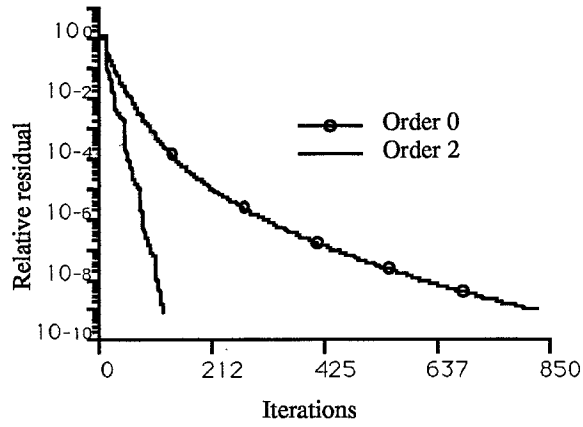


FIGURE 4 - Relative residual vs. nb of iterations

TABLE 1 - Computational time vs. number of processors

Nb of processors :	1	2	4	8	16	32
Computational time:	170,0	51,0	26,7	11	6,0	6,1