INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING
Int. J. Numer. Meth. Engng 2005; **62**:1873–1894
Published online 9 February 2005 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/nme.1258

An interface strip preconditioner for domain decomposition methods: application to hydrology

Rodrigo R. Paz*, † and Mario A. Storti

Centro Internacional de Métodos Computacionales en Ingeniería (CIMEC) CONICET-INTEC-U.N.L., Güemes 3450, (3000) Santa Fe, Argentina

SUMMARY

In this paper, the efficiency of a parallelizable preconditioner for domain decomposition methods in the context of the solution of non-symmetric linear equations arising from discretization of the Saint-Venant equations, is investigated. The proposed interface strip preconditioner (IS) is based on solving a problem in a narrow strip around the interface. It requires much less memory and computing time than classical Neumann–Neumann preconditioner, and handles correctly the flux splitting among sub-domains that share the interface. The performance of this preconditioner is assessed with an analytical study of Schur complement matrix eigenvalues and numerical experiments conducted in a parallel computational environment (consisting of a Beowulf cluster of 20 nodes). Copyright © 2005 John Wiley & Sons, Ltd.

KEY WORDS: domain decomposition; preconditioner; parallel computing; Saint-Venant equations

1. INTRODUCTION

The large spread in length scales present in hydrological problems (like river, estuaries, lakes, open channels, levees or dam breaks, etc.) requires a high degree of refinement in the finite element mesh and, then, requires very large computational resources. Also, in a 2D coupled surface–subsurface flow problem, a typical multi-aquifer model, the number of unknowns per surface node is, at least, equal to the number of aquifers and aquitards. Due to this fact, it is expected to have a very high demand of CPU computation time, calling for parallel processing

*Correspondence to: R. R. Paz, Centro Internacional de Métodos Computacionales en Ingeniería (CIMEC). Güemes 3450, (3000) Santa Fe, Argentina.
†E-mail: rodrigop@intec.unl.edu.ar

Contract/grant sponsor: Consejo Nacional de Investigaciones Científicas y Técnicas; contract/grant number: PIP 198/*Germen-CFD*
Contract/grant sponsor: Agencia Nacional de Promoción Científica y Tecnológica; contract/grant number: ANPCyT-PID-99/74 *FLAGS*, ANPCyT-FONCyT-PICT-6973 *PROA*
Contract/grant sponsor: Universidad Nacional del Litoral; contract/grant number: CAI+D-UNL-PIP-02552-2000

Received 12 April 2004
Revised 20 September 2004
Accepted 3 October 2004

techniques. Linear systems obtained from discretization of PDE's by means of finite difference or finite element methods are normally solved in parallel by iterative methods [1, 2] because they require much less communication compared to direct solvers.

The Schur complement domain decomposition method leads to a reduced system better suited for iterative solution than the global system, since its condition number is lower ($\propto 1/h$ vs $\propto 1/h^2$ for the global system, h being the mesh size) and the computational cost per iteration is not so high once the sub-domain matrices have been factorized.

Iterative substructuring methods rely on a non-overlapping partition into sub-domains (substructures). The efficiency of these methods can be further improved by using preconditioners [3]. Once the degrees of freedom inside the substructures have been eliminated by block Gaussian elimination (or other algorithm), a preconditioner for the resulting Schur complement system is built with matrix blocks relative to a decomposition of interface finite element functions into subspaces related to geometrical objects (vertices, edges, faces, single substructures) or simply by the coefficients of sub-domain matrices near the interface. Iterative methods like conjugate gradient and GMRES are then employed. Early works, such as References [4, 5], have influenced most of the later work in the field. They proposed two spaces for the coarse problem. One of their coarse spaces is given in terms of the averages of the nodal values over the entire substructure boundaries $\partial\Omega_i$. The other space is defined by extending the *wire basket* (we recall that the wire basket is the union of the boundaries of the faces which separate the substructures) values as a 2D discrete harmonic function onto the faces, and then as discrete harmonic function into the interiors of the sub-domains.

For auto-adjoint positive semidefinite problems, Neumann–Neumann preconditioner is the most classical one. From a mathematical point of view, the preconditioner is defined by approximating the inverse of the global Schur complement matrix by the weighted sum of local Schur complement matrices. From a physical point of view, Neumann–Neumann preconditioner is based on splitting the flux applied to the interface in the preconditioning step and solving local Neumann problems in each sub-domain. This strategy is good only for symmetric operators.

The preconditioner proposed here is based on solving a problem in a '*strip*' of nodes around the interface (Figure 1). When the width of the strip is narrow, the computational cost and memory requirements are low and the iteration count is relatively high, when the strip is wide, the converse is verified.

This preconditioner performs better for non-symmetric operators and does not have *rigid body* modes for internal *floating sub-domains*, as is the case for the Neumann–Neumann preconditioner. Recall that for operators that involve only derivatives of the unknowns (as Laplace equation, steady elasticity, steady advection–diffusion, for instance) a portion of the boundary should have Dirichlet or mixed boundary conditions. Otherwise, the problem is ill-posed and the matrix is singular. When using the Neumann–Neumann preconditioner, sub-domains inherit the boundary condition of the original problem in the external boundary, whereas Neumann boundary conditions are imposed at the internal sub-domain interfaces. Sub-domains that have a non-empty intersection with a portion of the Dirichlet part of the external boundary do not have rigid modes. Sub-domains whose boundary has empty intersection with the external Dirichlet or mixed portion of the boundary, would have Neumann condition imposed on their whole boundary and would have rigid modes for the kind of operators described above.

In contrast with the wire-basket algorithms, the IS preconditioner is purely algebraic, i.e. it can be assembled from a subset of the matrix coefficients. There are no requirements on the

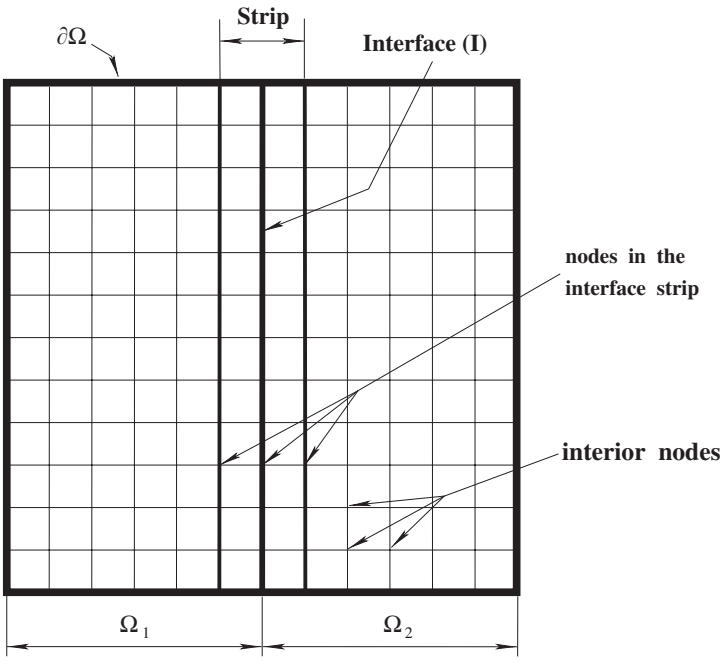

Figure 1. Domain decomposition.

topology of the mesh, and even it could be applied to sparse matrices coming from other kind of problems, not necessarily from PDE discretizations.

2. THE OPEN CHANNEL FLOW MODEL [6, 7]

The equations for the 2D Saint-Venant open channel flow are the well-known mass and momentum conservation equations integrated in the vertical direction. If we write these equations in the conservation matrix form (Einstein summation convention is assumed), we have

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathscr{F}_i(\mathbf{U})}{\partial x_i} = \mathbf{G}_i(\mathbf{U}), \quad i = 1, 2, \quad \text{on } \Omega_{\text{st}} \times [0, t] \tag{1}$$

where Ω_{st} is the stream domain, $\mathbf{U} = (h, hu, hv)^{\text{T}}$ is the state vector and the advective flux functions in (1) are

$$\begin{aligned} \mathscr{F}_1(\mathbf{U}) &= \left(hu, hu^2 + g\frac{h^2}{2}, huv \right)^{\text{T}} \\ \mathscr{F}_2(\mathbf{U}) &= \left(hv, huv, hv^2 + g\frac{h^2}{2} \right)^{\text{T}} \end{aligned} \tag{2}$$

where h is the height of the water in the channel with respect to the channel bottom, $\bar{u} = (u, v)^{\text{T}}$ is the velocity vector and g is the acceleration due to gravity. G_{s} represents the gain (or loss)

of the river, the source term is

$$\mathbf{G}(\mathbf{U}) = (G_s,\, gh(S_{0x} - S_{fx}),\;\; gh(S_{0y} - S_{fy}))^{\mathrm{T}} \tag{3}$$

where S_0 is the bottom slope and S_f is the slope friction.

$$\begin{aligned} S_{fx} &= \frac{1}{C_h^2 h}\, u|\bar{u}|, \quad S_{fy} = \frac{1}{C_h^2 h} v|\bar{u}| \quad \text{Chézy model} \\ S_{fx} &= \frac{n^2}{h^{4/3}} u|\bar{u}|, \quad S_{fy} = \frac{n^2}{h^{4/3}}\, v|\bar{u}| \quad \text{Manning model} \end{aligned} \tag{4}$$

where C_h and n (the Manning roughness) are model constants. In the case of great lakes, wide rivers and estuaries we should take into account the effect of Coriolis force (see Reference [7]).

Initial and boundary conditions [8]: To obtain a well-posed problem we add some initial conditions

$$\mathbf{U}|_{t=0} = \tilde{\mathbf{U}}^0 \quad \text{on } \Omega_{st} \tag{5}$$

and boundary conditions on $\partial\Omega_{st}$ (the stream boundary). We recall that the type of a flow in a stream or in an open channel depends on the value of the Froude number $F_r = |\bar{u}|/c$ (where $c = \sqrt{gh}$ is the wave celerity). A flow is said *fluvial* if $|\bar{u}| < c$ and *torrential* if $|\bar{u}| > c$.

Fluvial boundary:

- Inflow boundary: $\bar{u}$ specified and h is extrapolated from interior points, for instance.
- Outflow boundary: h specified and velocity field extrapolated from interior points, for instance.

Torrential boundary:

- Inflow boundary: $\bar{u}$ and the depth h are specified.
- Outflow boundary: all variables are extrapolated from interior points.

Solid Wall Condition: We prescribe the simple slip condition over $\partial\Omega_{slip} (\subset \partial\Omega_{st})$

$$\bar{u} \cdot n = 0 \tag{6}$$

Upon using the SUPG stabilized finite element discretization with linear triangles and/or bilinear rectangular elements, and the trapezoidal rule for time integration we obtain the system to be solved at each time step,

$$\mathbf{R} = \mathbf{K}(\mathbf{U})[\theta \mathbf{U}^{\tau+1} + (1-\theta)\mathbf{U}^{\tau}] + \mathbf{B}(\mathbf{U}) \frac{\mathbf{U}^{\tau+1} - \mathbf{U}^{\tau}}{\Delta t} - \mathbf{Q}^{\tau+1} = 0 \tag{7}$$

where θ is the time-weighting factor satisfying $0 \leqslant \theta \leqslant 1$, Δt is the time increment and τ denotes the number of time steps. $\mathbf{K}$ and $\mathbf{B}$ are the stiffness and mass matrix, respectively ($\mathbf{K}$ and $\mathbf{B}$ depend on $\mathbf{U}$), $\mathbf{Q}$ is the source vector and $\mathbf{R}$ is the residual vector.

3. SCHUR COMPLEMENT DOMAIN DECOMPOSITION METHOD

We consider solving in each time step a linearized form of system (7) (i.e. $\mathbf{A}u = f$) resulting from finite element discretization as described in the previous section. Let Ω denote the computational domain of the hydrological problem, and $\{\Omega^i\}_{i=1}^{i=n}$ its decomposition into n non-overlapping sub-domains. We shall re-order u and f as $u = (u_\mathrm{L}, u_\mathrm{I})^\mathrm{T}$ and $f = (f_\mathrm{L}, f_\mathrm{I})^\mathrm{T}$, numbering the global nodes such that the coefficient matrices of hydraulic height and velocities assume block-ordered structure

$$\mathbf{A} = \begin{bmatrix} A_{\mathrm{LL}} & A_{\mathrm{LI}} \\ A_{\mathrm{IL}} & A_{\mathrm{II}} \end{bmatrix} \tag{8}$$

where $A_{\mathrm{LL}} = \mathrm{diag}[A_{11}, A_{22}, \ldots, A_{nn}]$ is a block-diagonal with each block A_{ii}, $i = 1, 2, \ldots, n$ being the matrix corresponding to the unknowns belonging to the *interior* vertices of sub-domain Ω_i. A_{LI} and A_{IL} represents connections between sub-domains to interfaces.

A_{II} corresponds to the discretization of the differential operator restricted to the interfaces and represents the coupling between local interface points.

The numerical solution of $\mathbf{A}u = f$ is equivalent to solving

$$\mathbf{S}u_\mathrm{I} = g \quad \text{on interfaces } \Gamma \tag{9}$$

$$\mathbf{A}_{\mathrm{LL}}u_\mathrm{L} = f_\mathrm{L} - \mathbf{A}_{\mathrm{LI}}u_\mathrm{I} \quad \text{in } \Omega_i \tag{10}$$

where

$$\mathbf{S} = \mathbf{A}_{\mathrm{II}} - \sum_{i=1}^{n} \mathbf{A}_{\mathrm{IL}}\mathbf{A}_{\mathrm{LL}}^{-1}\mathbf{A}_{\mathrm{LI}} \tag{11}$$

and

$$g = f_\mathrm{I} - \sum_{i=1}^{n} \mathbf{A}_{\mathrm{IL}}\mathbf{A}_{\mathrm{LL}}^{-1} f_\mathrm{L} \tag{12}$$

where $\mathbf{S}$ is the well-known Schur complement matrix.

The Schur domain decomposition method starts by first determining u_I on the interfaces between sub-domains by solving (9). Upon obtaining u_I, the sub-domain problems (10) decouple and may be solved in parallel. The main computational cost for the iterative solution of (9) depends on the number of iteration, i.e. the condition number, to achieve convergence to a given accuracy criterion.

4. PRECONDITIONERS FOR DOMAIN DECOMPOSITION METHODS

It is clear that knowing the eigenvalue spectrum of the Schur complement matrix is one of the most important issues in order to develop suitable preconditioners. We start with the advection–diffusion equation, which is a simple model equation that captures many of the characteristics of advective systems like the Saint-Venant one (1). In this section the null advection case (Poisson equation) is considered, while the full advection–diffusion case is studied in Section 6. This simplification allows us to compute the eigenvalues in closed form and to assess the efficiency of several preconditioners.

The Poisson problem in a unit square is given by

$$\Delta\phi = g \quad \text{in } \Omega = \{0<x, y<1\} \tag{13}$$

and the boundary conditions

$$\phi = \bar{\phi} \quad \text{at } \Gamma = \{x, y = 0, 1\} \tag{14}$$

where ϕ is the unknown, $g(x, y)$ is a given source term and Γ is the boundary.

Consider now the partition of Ω in N_s non-overlapping sub-domains $\Omega_1, \Omega_2, \ldots, \Omega_{N_s}$, such that $\Omega = \Omega_1 \cup \Omega_2 \cup \cdots \cup \Omega_{N_s}$. For the sake of simplicity, we assume that the sub-domains are rectangles of unit height and width L_j. In practice this is not the best partition, but it is used (see Reference [9]) to compute the eigenvalues of the interface problem in closed form. Let $\Gamma_{\text{int}} = \Gamma_1 \cup \Gamma_2 \cup \cdots \cup \Gamma_{N_s-1}$ be the interior interfaces among adjacent sub-domains. Given a guess ψ_j for the trace of ϕ in the interior sub-domains $\phi|_{\Gamma_j}$, we can solve each interior problem independently as

$$\begin{aligned} &\Delta\phi = g \quad \text{in } \Omega_j \\ &\phi = \begin{cases} \psi_{j-1} & \text{at } \Gamma_{j-1} \\ \psi_j & \text{at } \Gamma_j \\ \bar{\phi} & \text{at } \Gamma_{\text{up},\,j} + \Gamma_{\text{down},\,j} \end{cases} \end{aligned} \tag{15}$$

where $\psi_0 = \bar{\phi}|_{x=0}$ and $\psi_{N_s} = \bar{\phi}|_{x=1}$ are given.

4.1. The Steklov operator

Not all combinations of trace values $\{\psi_j\}$ give the solution of the original problem (13). Indeed, the solution to (13) is obtained when the trace values are chosen in such a way that the flux balance condition at the internal interfaces is satisfied,

$$f_j = \left.\frac{\partial\phi}{\partial x}\right|_{\Gamma_j}^{-} - \left.\frac{\partial\phi}{\partial x}\right|_{\Gamma_j}^{+} = 0 \tag{16}$$

where the $\pm$ superscripts stand for the derivative taken from the left and right sides of the interface. We can think of the correspondence between the ensemble of interface values $\boldsymbol{\psi} = \{\psi_1, \ldots, \psi_{N_s-1}\}$ and the ensemble of flux imbalances $\mathbf{f} = \{f_1, \ldots, f_{N_s-1}\}$ as an interface operator $\mathcal{S}$ such that

$$\mathcal{S}\boldsymbol{\psi} = \mathbf{f} - \mathbf{f}_0 \tag{17}$$

where all inhomogeneities coming from the source term and Dirichlet boundary conditions are concentrated in the constant term $\mathbf{f}_0$, and the homogeneous operator $\mathcal{S}$ is equivalent to solving the equation set (15) with source term $g = 0$ and homogeneous Dirichlet boundary conditions $\bar{\phi} = 0$ at the external boundary Γ.

Here, $\mathcal{S}$ is the *Steklov operator*. In a more general setting, it relates the unknown values and fluxes at boundaries when the internal domain is in equilibrium. In the case of internal boundaries, it can be generalized by replacing the fluxes by the flux imbalances. The Schur complement matrix is a discrete version of the Steklov operator. In Reference [9] the eigenvalues

of Steklov operator are computed in a closed form for this simplified case. Hence, good estimates for the corresponding Schur complement matrix eigenvalues are obtained.

4.2. Eigenvalues of Steklov operator

We assume that only two sub-domains are present, one of them at the left of width L_1 and the other at the right of width L_2, so that $L = L_1 + L_2 = 1$ is the side length.

We solve first the Laplace problem in each sub-domain with homogeneous Dirichlet boundary condition at the external boundary and ψ at the interface,

$$\Delta\phi = 0 \quad \text{in } \Omega_{1,2}$$
$$\phi = \begin{cases} 0 & \text{at } \Gamma \\ \psi & \text{at } \Gamma_1 \end{cases} \tag{18}$$

The flux imbalance resulting from the solution $\phi_n(x, y)$ of (18) on each sub-domain is (see Reference [9])

$$f_n = \left.\frac{\partial\phi_n}{\partial x}\right|_{x=L_1^-} - \left.\frac{\partial\phi_n}{\partial x}\right|_{x=L_1^+}$$
$$= k_n[\coth(k_n L_1) + \coth(k_n L_2)]\sin(k_n y) \tag{19}$$

where the wave number k_n and the wavelength λ_n are defined as

$$k_n = 2\pi/\lambda_n, \quad \lambda_n = 2L/n, \quad n = 1, \ldots, \infty \tag{20}$$

A given interface value function ψ is an eigenfunction of the Steklov operator if the corresponding flux imbalance $f = \mathcal{S}\psi$ is proportional to ψ, i.e. $\mathcal{S}\psi = \omega\psi$, ω being the corresponding eigenvalue. The eigenfunctions of the Steklov operator are

$$\psi_n(y) = \sin(k_n y) \tag{21}$$

with eigenvalues

$$\omega_n = \mathrm{eig}(\mathcal{S})_n = \mathrm{eig}(\mathcal{S}^-)_n + \mathrm{eig}(\mathcal{S}^+)_n$$
$$= k_n[\coth(k_n L_1) + \coth(k_n L_2)] \tag{22}$$

where $\mathcal{S}^{\mp}$ are the Steklov operators of the left and right sub-domains,

$$\mathcal{S}^{\mp}\psi = \pm \left.\frac{\partial\phi}{\partial x}\right|_{L_1^{\mp}} \tag{23}$$

and their eigenvalues are

$$\mathrm{eig}(\mathcal{S}^{\mp})_n = k_n \coth(k_n L_{1,2}) \tag{24}$$

For large n, the hyperbolic cotangents in (24) both tend to unity. This shows that the eigenvalues of the Steklov operator grow proportionally to n for large n, and then its condition

number is infinity. However, when considering the discrete case the wave number k_n is limited by the largest frequency that can be represented by the mesh, which is $k_{\max} = \pi/h$ where h is the mesh spacing. The maximum eigenvalue is

$$\omega_{\max} = 2k_{\max} = \frac{2\pi}{h} \tag{25}$$

which grows proportionally to $1/h$. As the lowest eigenvalue is independent of h, this means that the condition number of the Schur complement matrix grows as $1/h$. Note that the condition number of the discrete Laplace operator typically grows as $1/h^2$. Of course, this reduction in the condition number is not directly translated to total computation time, since we have to take account of factorization of sub-domain matrices and forward and backward substitutions involved in each iteration to solve internal problems. However, the overall balance is positive and reduction in the condition number, beside being inherently parallel, turns out to be one of the main strengths of domain decomposition methods.

The eigenvalue magnitude is related to eigenfunction frequency along the inter-sub-domain interface, and the penetration of the eigenfunctions toward sub-domains interiors decays strongly for higher modes.

5. PRECONDITIONERS FOR THE SCHUR COMPLEMENT MATRIX

In order to further improve the efficiency of iterative methods, a preconditioner has to be added so that the condition number of the Schur complement matrix is lowered. The most known preconditioners for structural (symmetric and positive semidefinite) problems are Neumann–Neumann and its variants [10, 11] for Schur complements methods, and Dirichlet for finite element tearing and interconnecting (FETI) methods and its variants [12–15]. It can be proved that they reduce the condition number of the preconditioned operator to $O(1)$ (i.e. independent of h) in some special cases.

5.1. The Neumann–Neumann preconditioner

Consider the Neumann–Neumann preconditioner

$$\mathscr{P}_{\mathrm{NN}} v = f \tag{26}$$

where

$$v(y) = \tfrac{1}{2}\left[v_1(L_1, y) + v_2(L_1, y)\right] \tag{27}$$

and v_i, $i = 1, 2$, are defined through the following problems:

$$\begin{aligned} \Delta v_i &= 0 \quad \text{in } \Omega_i \\ v_i &= 0 \quad \text{at } \Gamma_0 + \Gamma_{\mathrm{up},i} + \Gamma_{\mathrm{down},i} \\ (-1)^{i-1}\frac{\partial v_i}{\partial x} &= \frac{1}{2} f \quad \text{at } \Gamma_1 \end{aligned} \tag{28}$$

The preconditioner consists in assuming that the flux imbalance f is applied on the interface. Since the operator is symmetric and the domain properties are homogeneous, this '*load*' is equally split among the two sub-domains. Then, we have a problem in each sub-domain with the same boundary conditions in the exterior boundaries, and a non-homogeneous Neumann boundary condition at the inter-sub-domain interface.

Again, we will show that the eigenfunctions of the Neumann–Neumann preconditioner are (21). Effectively, we can propose for v_1 the form

$$v_1 = C \sinh(k_n x) \sin(k_n y) \tag{29}$$

where C is determined from the boundary condition at the interface in (28) and results in

$$C = \frac{1}{2k_n \cosh(k_n L_1)} \tag{30}$$

and similarly for v_2, so that

$$\begin{aligned} v_1(x, y) &= \frac{1}{2k_n} \frac{\sinh(k_n x)}{\cosh(k_n L_1)} \sin(k_n y) \\ v_2(x, y) &= \frac{1}{2k_n} \frac{\sinh(k_n (L - x))}{\cosh(k_n L_2)} \sin(k_n y) \end{aligned} \tag{31}$$

Then, the value of $v = \mathscr{P}_{\mathrm{NN}}^{-1} f$ can be obtained from (27)

$$v(y) = \mathscr{P}_{\mathrm{NN}}^{-1} f = \frac{1}{4k_n} [\tanh(k_n L_1) + \tanh(k_n L_2)] \sin(k_n y) \tag{32}$$

so that the eigenvalues of $\mathscr{P}_{\mathrm{NN}}$ are

$$\mathrm{eig}(\mathscr{P}_{\mathrm{NN}})_n = 4k_n [\tanh(k_n L_1) + \tanh(k_n L_2)]^{-1} \tag{33}$$

As its definition suggests, it can be verified that

$$\mathrm{eig}(\mathscr{P}_{\mathrm{NN}})_n = 4\left[\mathrm{eig}(\mathscr{S}^-)_n^{-1} + \mathrm{eig}(\mathscr{S}^+)_n^{-1} \right]^{-1} \tag{34}$$

As the Neumann–Neumann preconditioner (26) and the Steklov operator (17) diagonalize in the same basis (21) (i.e. they '*commute*'), the eigenvalues of the preconditioned operator are simply the quotients of respective eigenvalues, i.e.

$$\mathrm{eig}(\mathscr{P}_{\mathrm{NN}}^{-1} \mathscr{S})_n = \tfrac{1}{4} [\tanh(k_n L_1) + \tanh(k_n L_2)] [\coth(k_n L_1) + \coth(k_n L_2)] \tag{35}$$

We see that all $\tanh(k_n L_j)$ and $\coth(k_n L_j)$ factors tend to unity for $n \to \infty$, then we have

$$\mathrm{eig}(\mathscr{P}_{\mathrm{NN}}^{-1} \mathscr{S})_n \to 1 \quad \text{for } n \to \infty \tag{36}$$

so that this means that the preconditioned operator $\mathscr{P}_{\mathrm{NN}}^{-1} \mathscr{S}$ has a condition number $O(1)$, i.e. it does not degrade with mesh refinement. This is optimal, and is a well-known feature of the Neumann–Neumann preconditioner. In fact, for a symmetric decomposition of the domain $\left(\text{i.e. } L_1 = L_2 = \frac{1}{2}\right)$, we have

$$\mathrm{eig}(\mathscr{P}_{\mathrm{NN}}^{-1} \mathscr{S})_n = \tfrac{1}{4}\, 2 \tanh(k_n/2) 2 \coth(k_n/2) = 1 \tag{37}$$

so that the preconditioner is equal to the operator and convergence is achieved in one iteration.

Note that comparing (22) and (34) we can see that the preconditioning is good as long as

$$\mathrm{eig}(\mathscr{S}^-)_n \approx \mathrm{eig}(\mathscr{S}^+)_n \tag{38}$$

This is true for symmetric operators and symmetric domain partitions (i.e. $L_1 \approx L_2$). Even for $L_1 \neq L_2$, if the operator is symmetric, then (38) is valid for large eigenvalues. However, this fails for non-symmetric operators as in the advection–diffusion case, and also for irregular interfaces.

Another aspect of the Neumann–Neumann preconditioner is the occurrence of indefinite internal Neumann problems, which leads to the need of solving a coarse problem [10, 11] in order to solve the '*rigid body modes*' for internal floating sub-domains. The coarse problem couples the sub-domains and hence ensures scalability when the number of sub-domains increases. However, this adds to the computational cost of the preconditioner.

5.2. The interface strip (IS) preconditioner

A key point about the Steklov operator is that its high-frequency eigenfunctions decay very strongly far from the interface, so that a preconditioning that represents correctly the high-frequency modes can be constructed if we solve a problem on a narrow strip around the interface. In fact, the nth eigenfunction with wave number k_n given by (21) decays far from the interface as $\exp(-k_n|s|)$ where s is the distance to the interface (the hyperbolic sine factors appearing in (19)). Then, this high-frequency modes will be correctly represented if we solve a problem on a strip of width b around the interface, provided that the interface width is very large with respect to the mode wave length λ_n.

The IS preconditioner is defined as

$$\mathscr{P}_{\mathrm{IS}} v = f \tag{39}$$

where

$$f = \left.\frac{\partial w}{\partial x}\right|_{x=L_1^-} - \left.\frac{\partial w}{\partial x}\right|_{x=L_1^+} \tag{40}$$

and

$$\begin{aligned} \Delta w &= 0 \quad \text{in } |x - L_1| < b \\ w &= 0 \quad \text{at } |x - L_1| = b \text{ and } y = 0, 1 \\ w &= v \quad \text{at } x = L_1 \end{aligned} \tag{41}$$

Please note that for high frequencies (i.e. $k_n b$ large) the eigenfunctions of the Steklov operator are negligible at the border of the strip, so that the boundary condition at $|x - L_1| = b$ is justified. The eigenfunctions for this preconditioner are again given by (21) and the eigenvalues can be taken from (22), replacing $L_{1,2}$ by b, i.e.

$$\mathrm{eig}(\mathscr{P}_{\mathrm{IS}})_n = 2\,\mathrm{eig}(\mathscr{S}_b)_n = 2k_n \coth(k_n b) \tag{42}$$

where $\mathscr{S}_b$ is the Steklov operator corresponding to a strip of width b.

For the preconditioned Steklov operator, we have

$$\mathrm{eig}(\mathscr{P}_{\mathrm{IS}}^{-1}\mathscr{S})_n = \tfrac{1}{2}\tanh(k_n b)[\coth(k_n L_1) + \coth(k_n L_2)] \tag{43}$$

We note that $\mathrm{eig}(\mathscr{P}_{\mathrm{IS}}^{-1}\mathscr{S})_n \to 1$ for $n \to \infty$, so that the preconditioner is optimal, independently of b. Also, for b large enough we recover the original problem so that the preconditioner is exact (convergence is achieved in one iteration). However, in this case the use of this preconditioner is impractical, since it implies solving the whole problem. Note that in order to solve the problem for v, we need information from both sides of the interface, while the Neumann–Neumann preconditioner solves the problem without communication of information between sub-domains. This is a disadvantage in terms of efficiency, since we have to waste communication time in sending the matrix coefficients in the strip from one side to the other or otherwise compute them in both processors. However, we will see that efficient preconditioning can be achieved with few node layers and negligible communication. Moreover, we can solve the preconditioner problem by iteration, so that no migration of coefficients is needed.

6. THE SCALAR ADVECTIVE–DIFFUSIVE CASE

Consider now the advective–diffusive case,

$$\kappa\Delta\phi - u\phi_{,x} = g \quad \text{in } \Omega \tag{44}$$

where κ is the thermal conductivity of the medium and u the advection velocity. The problem can be treated in a similar way, and the Steklov operators are defined as

$$\mathscr{S}^{\mp}\psi = \pm\phi_{,x}|_{L_1^{\mp}} \tag{45}$$

where

$$\begin{gathered}\kappa\Delta\phi - u\phi_{,x} = 0 \quad \text{in } \Omega_{1,2} \\ \phi = \begin{cases} 0 & \text{at } \Gamma \\ \psi & \text{at } \Gamma_1 \end{cases}\end{gathered} \tag{46}$$

The eigenfunctions are still given by (21). Looking for solutions of the form $v \propto \exp(\mu x)\sin(k_n y)$ we find that the eigenvalues are

$$\begin{aligned}\mathrm{eig}(\mathscr{S}^{-})_n &= \frac{u}{2\kappa} + \delta_n \coth(\delta_n L_1) \\ \mathrm{eig}(\mathscr{S}^{+})_n &= -\frac{u}{2\kappa} + \delta_n \coth(\delta_n L_2)\end{aligned} \tag{47}$$

For low-frequency modes, advective effects are more pronounced and the first eigenfunction is notably biased to the right. In contrast, for high-frequency modes the diffusive term prevails and the eigenfunction is more symmetric about the interface, and (as in the pure diffusive case) concentrated around it (see Reference [9]). Note that now the eigenvalues for the right and left part of the Steklov operator may be very different due to the asymmetry

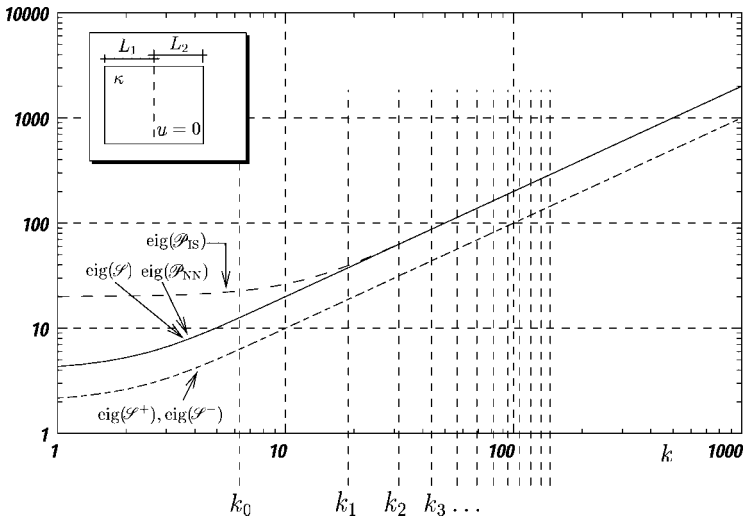

Figure 2. Eigenvalues of Steklov operators and preconditioners for the Laplace operator ($Pe = 0$) and symmetric partitions ($L_1 = L_2 = L/2$, $b = 0.1L$).

introduced by the advective term. This difference in splitting is more important for the lowest mode.

In Figures 2–5 we see the eigenvalues as a function of the wave number k_n. Note that for a given side length L only a certain sequence of wave numbers, given by (21) should be considered. However, it is perhaps easier to consider the continuous dependence of the different eigenvalues upon the wave number k.

For a symmetric operator and a symmetric partition (see Figure 2), the symmetric flux splitting is exact and the Neumann–Neumann preconditioner is optimal. The largest discrepancies between the IS preconditioner and the Steklov operator occur at low frequencies and yield a condition number less than two.

If the partition is non-symmetric (see Figure 3) then the Neumann–Neumann preconditioner is no longer exact, because $\mathscr{S}^+ \neq \mathscr{S}^-$. However, its condition number is very low whereas the IS preconditioner condition number is still under two.

For a relatively important advection term, given by a global Pèclet number of $uL/2\kappa = 5$ (see Figure 4), the asymmetry in the flux splitting is much more evident, mainly for small wave numbers, and this results in a large discrepancy between the Neumann–Neumann preconditioner and the Steklov operator. On the other hand, the IS preconditioner is still very close to the Steklov operator.

The difference between the Neumann–Neumann preconditioner and the Steklov operator increases for larger Pe (see Figure 5).

This behaviour can be directly verified by computing the condition number of Schur complement matrix and preconditioned Schur complement matrix for the different preconditioners (see Tables I and II). We can see that both the Neumann–Neumann and IS preconditioners give a similar preconditioned condition number regardless of mesh refinement (it almost doesn't change from a mesh of 50×50 to a mesh of 100×100), whereas the Schur complement matrix

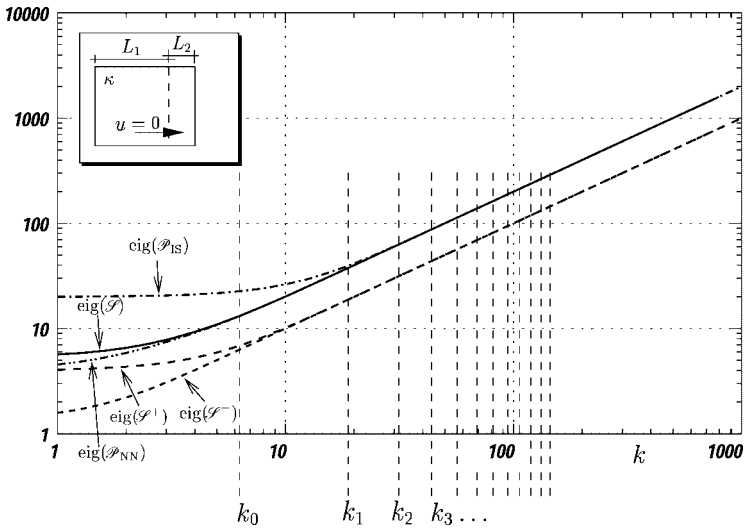

Figure 3. Eigenvalues of Steklov operators and preconditioners for the Laplace operator ($Pe = 0$) and non-symmetric partitions ($L_1 = 0.75L,\ L_2 = 0.25L,\ b = 0.1L$).

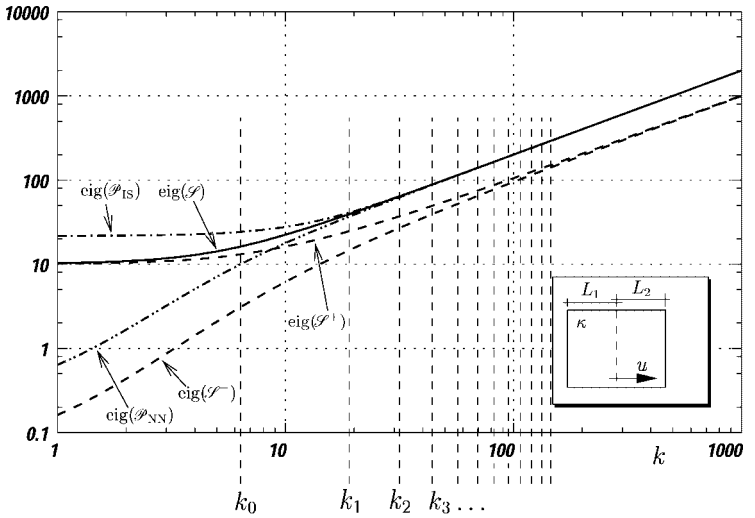

Figure 4. Eigenvalues of Steklov operators and preconditioners for the advection–diffusion operator ($Pe = 5$) and symmetric partitions ($L_1 = L_2 = L/2,\ b = 0.1L$).

exhibits a condition number roughly proportional to $1/h$. However, the Neumann–Neumann preconditioner exhibits a large condition number for high Pèclet numbers whereas the IS preconditioner performs better for advection dominated problems.

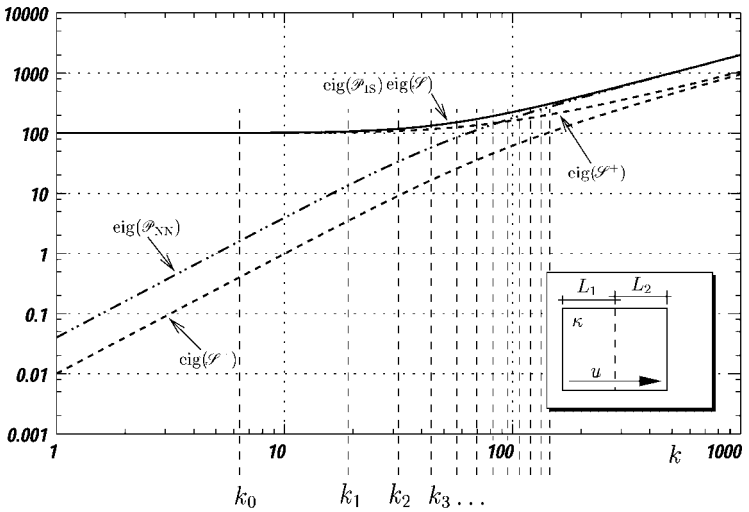

Figure 5. Eigenvalues of Steklov operators and preconditioners for the advection–diffusion operator ($Pe = 50$) and symmetric partitions ($L_1 = L_2 = L/2,\ b = 0.1L$).

Table I. Condition number for the Steklov operator and several preconditioners mesh: 50×50 elements, strip: five layers of nodes.

Pe	$\mathrm{cond}(\mathscr{S})$	$\mathrm{cond}(\mathscr{P}_{\mathrm{NN}}^{-1}\mathscr{S})$	$\mathrm{cond}(\mathscr{P}_{\mathrm{IS}}^{-1}\mathscr{S})$
0	41.00	1.00	4.92
0.5	40.86	1.02	4.88
5	23.81	3.44	2.92
25	5.62	64.20	1.08

Table II. Condition number for the Steklov operator and several precond. (mesh: 100×100 elements, strip: 10 layers of nodes).

u	$\mathrm{cond}(\mathscr{S})$	$\mathrm{cond}(\mathscr{P}_{\mathrm{NN}}^{-1}\mathscr{S})$	$\mathrm{cond}(\mathscr{P}_{\mathrm{IS}}^{-1}\mathscr{S})$
0	88.50	1.00	4.92
0.5	81.80	1.02	4.88
5	47.63	3.44	2.92
25	11.23	64.20	1.08

7. SOLUTION OF THE STRIP PROBLEM

Some hints are given for an efficient implementation of the IS preconditioner in a parallel environment.

Consider a sub-domain interface with a strip of two element layers ($n_{\mathrm{lay}} = 2$), as shown in Figure 6. The preconditioning consists in, given a vector f_{I} defined on the nodes at the

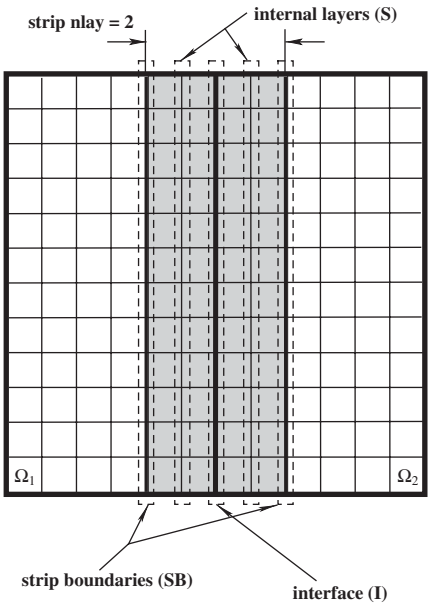

Figure 6. Strip interface problem.

interface (I in the figure) to compute an approximate solution v_{I} given by

$$\begin{bmatrix} A_{\mathrm{II}} & A_{\mathrm{IS}} & A_{\mathrm{I,SB}} \\ A_{\mathrm{SI}} & A_{\mathrm{SS}} & A_{\mathrm{S,SB}} \\ A_{\mathrm{SB,I}} & A_{\mathrm{SB,S}} & A_{\mathrm{SB,SB}} \end{bmatrix} \begin{bmatrix} v_{\mathrm{I}} \\ v_{\mathrm{S}} \\ v_{\mathrm{SB}} \end{bmatrix} = \begin{bmatrix} f_{\mathrm{I}} \\ 0 \\ 0 \end{bmatrix} \tag{48}$$

with '*Dirichlet boundary conditions*' at the strip boundary $v_{\mathrm{SB}} = 0$, so that it reduces to

$$\begin{bmatrix} A_{\mathrm{II}} & A_{\mathrm{IS}} \\ A_{\mathrm{SI}} & A_{\mathrm{SS}} \end{bmatrix} \begin{bmatrix} v_{\mathrm{I}} \\ v_{\mathrm{S}} \end{bmatrix} = \begin{bmatrix} f_{\mathrm{I}} \\ 0 \end{bmatrix} \tag{49}$$

Once this equation is solved, v_{I} is the value of the proposed preconditioner applied to f_{I}, i.e.

$$v_{\mathrm{I}} = \mathscr{P}_{\mathrm{IS}}^{-1} f_{\mathrm{I}} \tag{50}$$

A direct solution of this interface problem is not easily parallelizable. This approach would involve transferring all the interface matrix to a single processor and solving the problem there. So that, the possibility is to partition the strip problem among processors, much in the same way as the global problem is, and solving the strip problem by an iterative method. The idea of an iterative method is also suggested by the fact that the preconditioning matrix (i.e. the matrix obtained by assembling on the strip domain with Dirichlet boundary conditions at the strip boundary) is highly diagonal dominant for narrow strips. Care must be taken to avoid nesting a *non-stationary* method like CG or GMRES inside another outer non-stationary method [16]. We recall that in a *stationary* method the solution x at the iteration k depends,

only, on the solution at the previous step (i.e. $x_k = f(x_{k-1})$), then we can find the guess x_k after k successive applications of the same operator to the initial value x_0. The problem here is that a non-stationary method executed a finite number of times *is not* a linear operator, unless the inner iterative method is iterated enough and then approaches the inverse of the preconditioner. In this respect, relaxed Richardson iteration is suitable.

For the Richardson interface problem, a fixed predetermined number m of Richardson iterations are performed. If m is too low, then the preconditioner has no effect, and if it is too large the efficiency of the preconditioner tends to saturate, while the cost is roughly proportional to m, so in general there is an optimal value for m. We have found that adjusting m so that Richardson iteration converges one order of magnitude (relative to the initial residual) is fine for most problems. Note that the number of iterations may depend on the intrinsic conditioning of the interface problem and also on the strip width. For small strip widths ($n_{\text{lay}} < 5$) m was chosen in the range $5 \leqslant m \leqslant 10$.

A subsequent possibility is preconditioning the IS preconditioner problem itself with block Jacobi. In general, in parallel implementation, each processor may have several sub-domains. In this way, the memory and time computation requirements (i.e. the cost of factorize smaller matrices is reduced) are reduced. If the number of dof's in the interfaces grows toward the number of total dof's the method results in a fully iterative method.

Even if the preconditioner has been described through figures in terms of finite element structured meshes, the implementation is purely algebraic (in contrast to previous approaches, like, notably, the wire-basket one) based on the graph connectivity of the matrix. The preconditioner has been implemented in a FEM production code [17] and tested in large scale problems with unstructured tetrahedral meshes with up to one million tetras.

8. SOME NUMERICAL EXAMPLES IN PARALLEL ENVIRONMENT

In this section, we present numerical results for diffusive and advective problems and some discussions about these results. The tests were carried out on a *Beowulf* cluster of PC's. The cluster at CIMEC laboratory has 20 (uniprocessor) nodes; where 10 nodes are Pentium IV—2.4 GHz, 1 GB RAM (DDR, 333 MHz), 7 nodes Pentium IV—1.7 GHz, 512 MB RAM (RIMM, 400/800 MHz) and 2 nodes Pentium IV—1.7 GHz, 256 MB RAM (RIMM, 400/800 MHz). Usually, the first node works as server. The nodes are connected through a switch Fast Ethernet (100 Mbit/s, latency $= O(100)$ μs).

The iteration counts of the IS and Neumann–Neumann preconditioners are shown, for a sequential environment, in Reference [9]. In this paper, the performance of the proposed preconditioner is studied in a parallel environment. For this purpose, we consider two different problems. The domain Ω in both cases is the unit square discretized on an structured mesh of 500×500 nodes, and decomposed into four rectangular sub-domains. We compare the residual norm versus iteration count by using no preconditioner, Neumann–Neumann preconditioner, block Jacobi preconditioner, global Jacobi preconditioner and the IS preconditioner (with several strip widths at the interfaces). Global Jacobi is a diagonal scaling preconditioning algorithm. Block Jacobi preconditioner is a block-diagonal preconditioner and is obtained by (approximately) inverting the local diagonal blocks on each processor (see Reference [1] for a detailed description of these preconditioners).

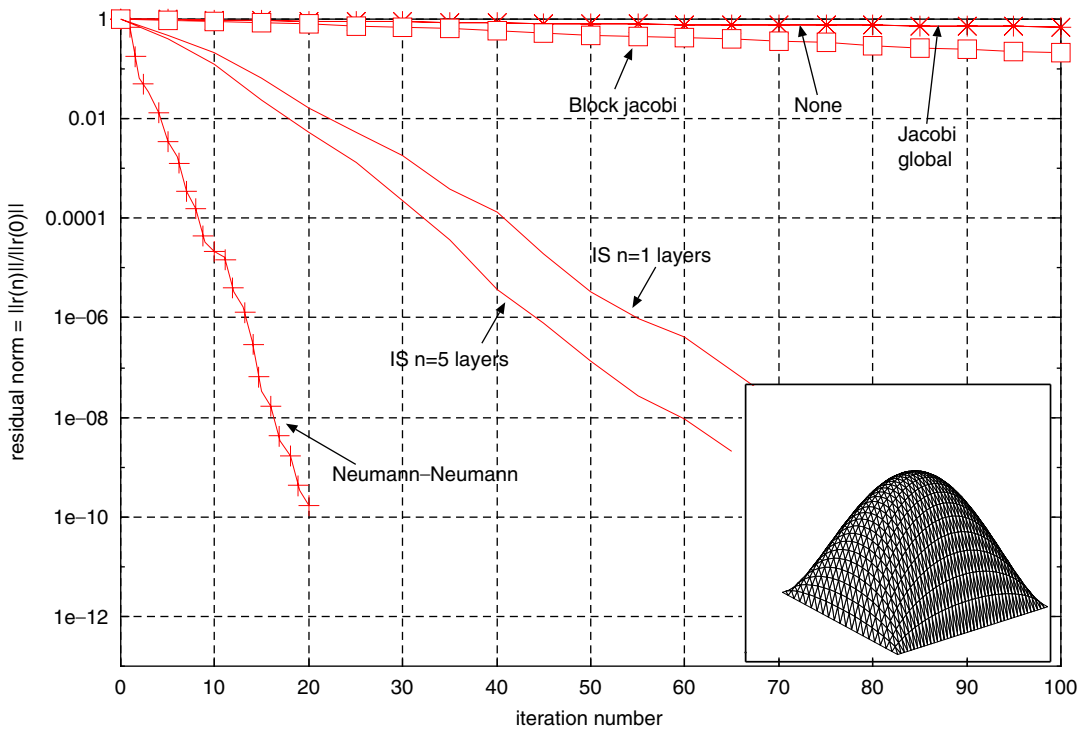

Figure 7. Solution of Poisson's problem (*mesh* 500×500).

The first example is the Poisson's problem $\Delta\phi = g$, where $g = 1$ and $\phi = 0$ on all the boundary Γ. The iteration counts and the problem solution (obtained in a coarse mesh for visualization purposes) are plotted in Figure 7. As it can be seen, the Neumann–Neumann preconditioner has a very low iteration count, as it is expected for a symmetric operator. The IS preconditioner has a larger iteration count for thin strip widths, but it decreases as the strip is thickened. Regarding memory use, the required core memory for thin strip is much less than for the Neumann–Neumann preconditioner. The strip width acts in fact as a parameter that balances the required amount of memory and the preconditioner efficiency. We split the system solution in two stages, the factorization stage (for the local problems) and the GMRES iteration stage (including the Richardson iteration for the IS preconditioner), in order to compute the time consumed to achieve a given tolerance in the residual vector (see Table III). CPU times for the iteration stage and memory requirements are not given in Table III for Jacobi preconditioning and not preconditioning at all because these methods failed to converge.

The second example is an advective–diffusive problem at a global Péclet number of $Pe = 25$, $g = \delta\left(\frac{1}{4}, \frac{3}{4}\right) + \delta\left(\frac{3}{4}, \frac{1}{4}\right)$, and $\phi(-0.5, y) = 0$, where δ is the Dirac's delta function. Therefore, the problem is strongly advective. We compare the iteration counts in two different meshes and two different decompositions. The mesh of 500×500 nodes is decomposed into four rectangular domains, one per processor, and the mesh of 1000×1000 is partitioned into seven sub-domains. The iteration count and the problem solution (interpolated in a coarse mesh for visualization purposes) are plotted in Figures 8 and 9. In this example, the advective term

Table III. CPU time and memory requirements per proc. for Poisson problem (*mesh* 500×500 elements).

Preconditioner	None	Jacobi glob.	Block Jacobi	N–N	IS($n_{\text{lay}} = 1$)	IS($n_{\text{lay}} = 5$)
Factorization (s)	—	—	1.9	4.7	2.3	2.3
GMRES stage (s)	*	*	*	1.51	5.4	4.9
Tolerance	1.e-10	1.e-10	1.e-10	1.e-10	1.e-10	1.e-10
Memory/proc. (Mb)	*	*	*	70	62	62.5

Note: * in table means iteration failed to converge to a specified tolerance in a maximum of 200 its.

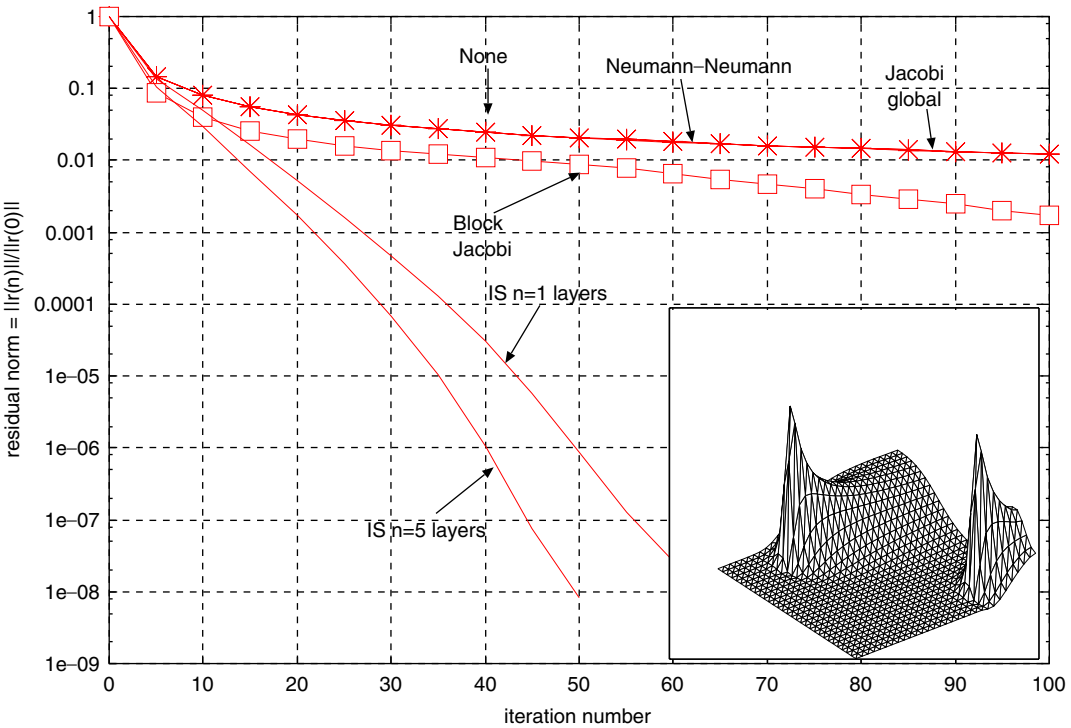

Figure 8. Solution of advective–diffusive problem (*mesh* 500×500).

introduces a strong asymmetry. CPU times and memory requirements are not given in Table IV for N–N preconditioner because this method failed to converge. However, only to give an idea, the required memory for N–N preconditioner (coarse mesh) for 50/60 iterations (IS was converged at this point) is 73 Mb/proc (megabytes per processor), whereas for 200 iterations (the maximum allowed) the consumed memory was 120 Mb/proc. For the refined mesh, the memory used in 70/80 iterations is 210 Mb/proc and for the 200 iterations (the maximum allowed) was 320 Mb/proc. Clearly, the Neumann–Neumann preconditioner is outperformed by IS preconditioner in iteration count (and consequently in computing time) and memory demands, even for thin strips. The CPU time and memory used (per processor) are shown in Table IV.

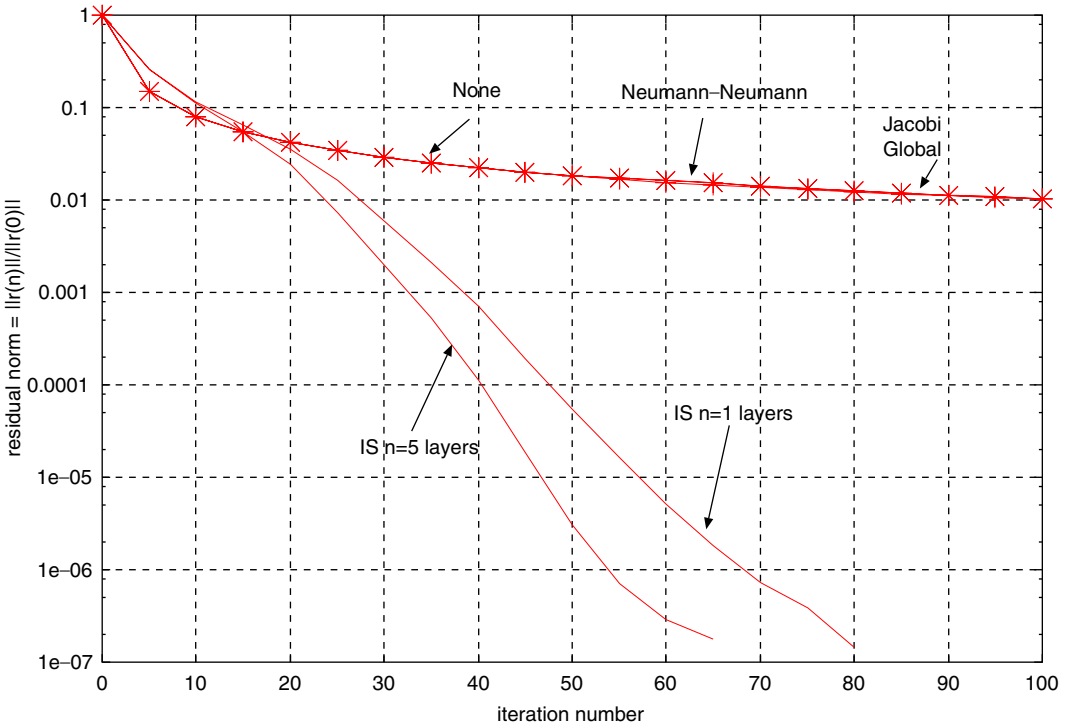

Figure 9. Iteration counts for advective–diffusive problem (*mesh* 1000×1000).

Table IV. CPU time and memory requirements per proc. for advective–diffusive problem (*mesh* 1000×1000 elements).

Preconditioner	None	Jacobi glob.	N–N	IS($n_{\text{lay}} = 1$)	IS($n_{\text{lay}} = 5$)
Factorizations (s)	—	—	4.0	8.0	7.8
GMRES stage (s)	*	*	*	13.0	12.0
Tolerance	0.25e-06	0.25e-06	0.25e-06	0.25e-06	0.25e-06
Memory/proc. (Mb)	*	*	*	140	142

Note: * in table means iteration failed to converge to a specified tolerance in a maximum of 200 its.

9. SAINT-VENANT NUMERICAL EXAMPLES

The example is a 2D Saint-Venant subcritical flow over an impermeable unit square channel with a parabolic bump in the bottom and a sinusoidal wave-train perturbation in x-velocity at the inflow boundary. The parabolic variation of the bottom has the form $\eta(x, y) = \min\{h_1, h_2 + (h_1 - h_2)(r/R)^2\}$, where r is the distance to the centre of the bump, located at $(0, 0)$, $h_1 = 1$, $h_2 = 0.5$ and $R = 0.3$. The period of the plane inciding wave is $T = 0.1$ s. Hence, roughly, five wave-lengths enter in the diameter of the bump. The initial global Froude and Courant numbers (based in longitudinal velocity u) are $Fr = u/\sqrt{gh} = 0.3$ and $C = u(\Delta t/\Delta x) = 15$. Null flux is considered in $y = \pm 0.5$ and fluvial boundary conditions at the inflow/outflow sections. For

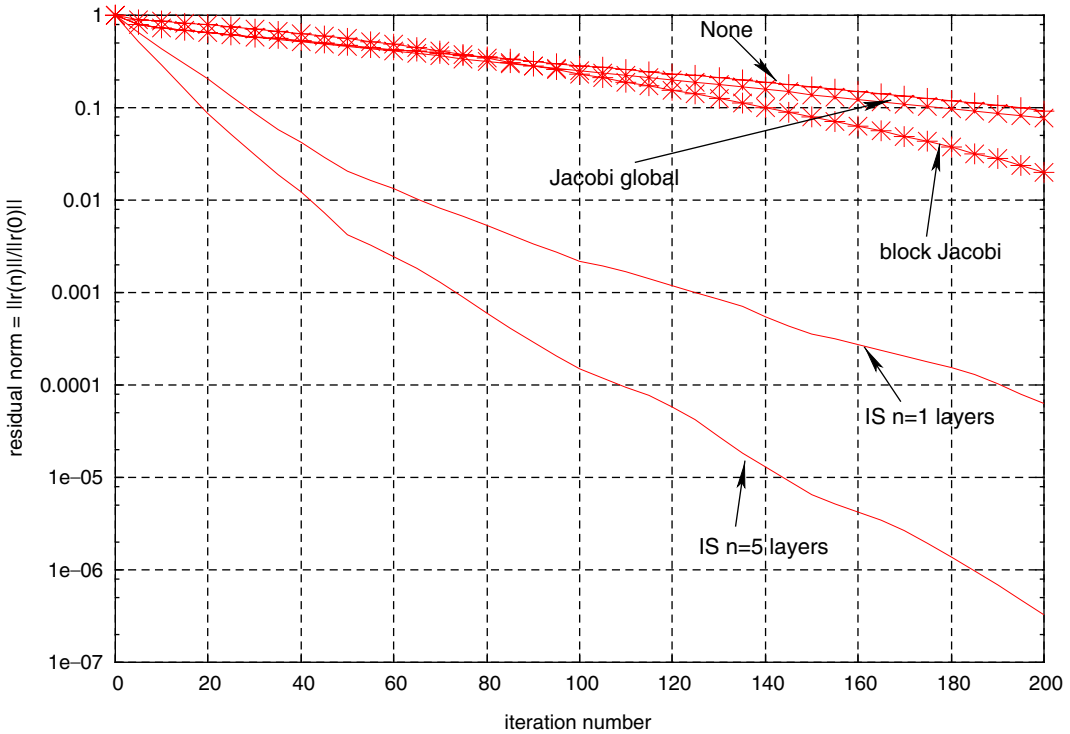

Figure 10. Iteration counts for Saint-Venant system of equations (*mesh* 500×500).

the computations we use the Chézy model with friction coefficient $C_h = 110\,\mathrm{m}^{1/2}/\mathrm{s}$. The mesh of 10^5 linear triangles was partitioned with METIS into five sub-domains (one per processor).

The iteration counts for the linear system corresponding to a typical Newton iteration at a given time step is plotted in Figure 10. Figure 11 shows the elevation for the steady periodical state. In this example, the system of conservation laws (1) introduces a strong asymmetry. As in the linear advection–diffusion problem, the IS preconditioner improves the iteration counts and memory demands. Although each iteration is more expensive for the IS preconditioner, the consumed time to reach a given tolerance is smaller. The CPU consumed time, tolerances and consumed memory are shown in Table V.

10. CONCLUSIONS

We have presented the parallel version of a new preconditioner for Schur complement domain decomposition methods and the convergence improvement for hydrological problems. This preconditioner is based on solving a problem posed in a narrow strip around the inter-subdomain interfaces. Some analytical results have been derived to present its mathematical basis. Numerical experiments of several physical problems have been carried out to show its convergence properties and the computation time.

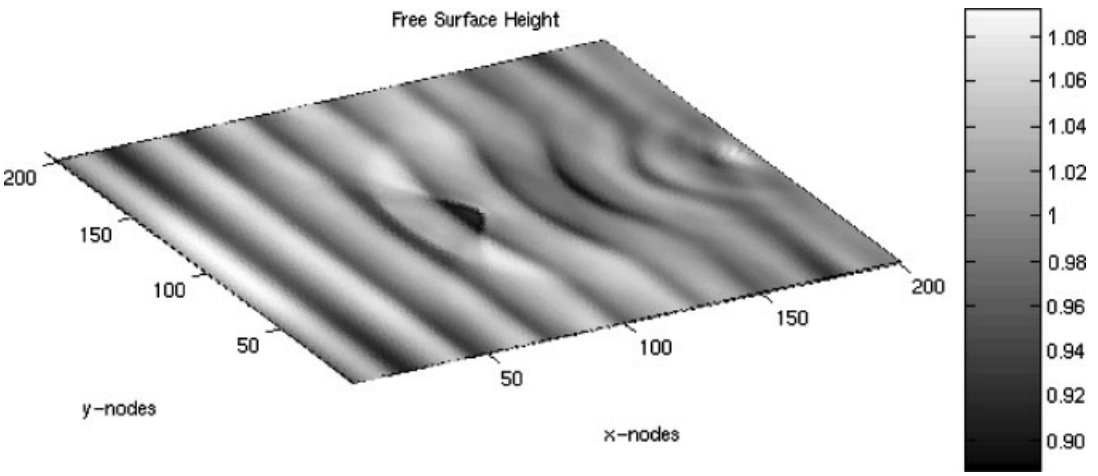

Figure 11. Solution of Saint-Venant system of equations (*mesh* 500×500).

Table V. CPU time and memory requirements for Saint-Venant equations (*mesh* 500×500 elements).

Preconditioner	None	Jacobi glob.	Block Jacobi	IS($n_{\text{lay}} = 1$)	IS($n_{\text{lay}} = 5$)
Factorization (s)	—	—	8.1	9.0	9.2
GMRES stage (s)	*	*	522	68	43
Tolerance	1.e-05	1.e-05	1.e-05	1.e-05	1.e-05
Memory/proc. (Mb)	*	*	605	548	550

Note: * in table means iteration failed to converge to a specified tolerance in a maximum of 400 iterations.

The IS preconditioner is easy to construct as it does not require any special calculation (it can be assembled with a subset of sub-domain matrix coefficients). It is much less memory-consuming than classical optimal preconditioners such as Neumann–Neumann in primal methods (or Dirichlet in FETI methods). Moreover, it permits to decide how much memory to assign for preconditioning purposes.

The IS preconditioner is well suited for hydrological problems where advective terms are present in governing equations, while it is capable to handle reasonably well diffusion–dominated regions.

ACKNOWLEDGEMENTS

This work has received financial support from *Consejo Nacional de Investigaciones Científicas y Técnicas* (CONICET, Argentina), *Agencia Nacional de Promoción Científica y Tecnológica* (ANPCyT) and *Universidad Nacional del Litoral* (UNL) through grants CONICET PIP 198/*Germen-CFD*, ANPCyT-PID-99/74 *FLAGS*, ANPCyT-FONCyT-PICT-6973 *PROA* and CAI+D-UNL-PIP-02552-2000. We made extensive use of freely distributed software such as *GNU/Linux* OS, MPI, PETSc, Metis, Octave, the CGAL geometrical library, OpenDX and many others.

REFERENCES

1. Saad Y. *Iterative Methods for Sparse Linear Systems*. PWS Publishing Co.: Massachusetts, MA, 2000. http://www-users.cs.umn.edu/~saad/books.html.
2. Meurant G. *Computer Solution of Large Linear Systems.* Studies in Mathematics and Its Applications, vol. 28. North-Holland: Amsterdam, 1999.
3. Le Tallec P, Vidrascu M. Solving large scale structural problems on parallel computers using domain decomposition techniques. *Parallel Solution Methods in Computational Mechanics*, Papadrakakis M (ed.). Chapter 2. Wiley: New York, 1997; 49–85.
4. Bramble JH, Pasciak JE, Schatz AH. The construction of preconditioners for elliptic problems by substructuring, I. *Mathematics of Computation* 1986; **47**(175):103–134.
5. Bramble JH, Pasciak JE, Schatz AH. The construction of preconditioners for elliptic problems by substructuring, IV. *Mathematics of Computation* 1989; **53**(187):1–24.
6. Whitham GB. *Linear and Nonlinear Waves*. Pure and Applied Mathematics, A Wiley-Interscience Series of Texts, Monographs, and Tracts, 1974.
7. Paz RR, Storti MA, Idelsohn SR, Rodríguez LB, Vionnet C. Parallel finite element model for coupled surface and subsurface flow in hydrology: province of Santa Fe Basin, absorbent boundary condition. *XIII Argentine Congress on Computational Mechanics—ENIEF2003*, 2003.
8. Hirsch C. *Numerical Computation of Internal and External Flows—vol. II.* Wiley Series in Numerical Methods in Engineering, 1990.
9. Storti MA, Dalcin L, Paz RR, Yommi A, Sonzogni V, Nigro N. An interface strip preconditioner for domain decomposition methods. *Journal of Computational Methods in Science and Engineering* 2003, to appear.
10. Mandel J. Balancing domain decomposition. *Communications in Applied Numerical Methods* 1993; **9**:233–241.
11. Cros JM. A preconditioner for the Schur complement domain decomposition method. *14th International Conference on Domain Decomposition Methods*, 2002.
12. Farhat C, Roux FX. A method of finite element tearing and interconnecting and its parallel solution algorithm. *International Journal for Numerical Methods in Engineering* 1991; **32**:1205–1227.
13. Farhat C, Mandel J, Roux FX. Optimal convergence properties of the FETI domain decomposition method. *Computer Methods in Applied Mechanics and Engineering* 1994; **115**:365–385.
14. Farhat C, Mandel J. The two-level FETI method for static and dynamic plate problems. *Computer Methods in Applied Mechanics and Engineering* 1998; **155**:129–152.
15. Farhat C, Lesoinne M, Le Tallec P, Pierson K, Rixen D. FETI-DP: a dual-primal unified FETI method-part I: a faster alternative to the two-level FETI method. *International Journal for Numerical Methods in Engineering* 2001; **50**:1523–1544.
16. Kelley CT. *Iterative Methods for Linear and Nonlinear Equations*. Frontiers in Applied Mathematics, vol. 16. SIAM: Philadelphia, PA, 1995.
17. Storti MA, Nigro N, Paz RR. PETSc-FEM: a general purpose, parallel, multi-physics FEM program. http://www.cimec.org.ar/petscfem.

