THE SWISS-CARPET DOMAIN DECOMPOSITION PRECONDITIONER

ALFIO QUARTERONI[1], MARZIO SALA[2] and ALBERTO VALLI[3]

Abstract. In this paper we consider domain decomposition preconditioners based on a vertex-oriented decomposition of the computational domain. In element-oriented (EO) decompositions, each element of the grid belongs to a different domain, while in vertex-oriented (VO) decompositions each vertex belongs to a different subdomain. Based on VO decompositions, we present several preconditioners for the solution of the original (unreduced) system, as well as for that of the Schur complement system. Theoretical properties are investigated for a finite element approximation of a scalar problem. Numerical results and comparison with state-of-art preconditioners are also reported. The numerical results here presented show the effectiveness of the proposed preconditioners and their good parallel properties.

1 Introduction

We consider the iterative solution by Krylov methods of sparse linear systems

$$A\mathbf{u} = \mathbf{f}, \tag{1}$$

where $A \in \mathbb{R}^{n\times n}$ is a non-singular matrix (typically, derived from the discretisation of a boundary value problem for partial differential equations), $\mathbf{u} \in \mathbb{R}^n$ the solution vector, and $\mathbf{f} \in \mathbb{R}^n$ the right-hand side.

The convergence rate depends on the spectral properties of A, mainly on its condition number $\kappa(A)$. The latter is defined as $|||A|||\cdot|||A^{-1}|||$ (for a suitable matrix norm $|||\cdot|||$), or simply as $\lambda_{\max}(A)/\lambda_{min}(A)$ if the matrix is symmetric and positive definite. For ill-conditioned systems one typically resorts to solve the (left) preconditioned system

$$P^{-1}A\mathbf{u} = P^{-1}\mathbf{f}, \tag{2}$$

where P is a non-singular matrix, or else, if P is a symmetric and positive definite matrix,

$$P^{-1/2}AP^{-1/2}\mathbf{w} = P^{-1/2}\mathbf{f}, \tag{3}$$

with $\mathbf{w} = P^{1/2}\mathbf{u}$. The preconditioner P should approximate A^{-1} as closely as possible, so that $\kappa(P^{-1}A) \ll \kappa(A)$, while still being reasonably cheap to invert. Ideally, P should be *optimal*—*i.e.* $\kappa(P^{-1}A)$ ought not depend on the mesh size, and, when considering parallel preconditioners, a further requirement for P is that it is *scalable*—*i.e.* $\kappa(P^{-1}A)$ should not depend on the number of processors used in the computations.

[1]CMCS-IACS-SB, EPFL, CH-1015 Lausanne, Switzerland, and MOX-Politecnico di Milano, I-20133 Milano, Italy
[2]CMCS-IACS-SB, EPFL, CH-1015 Lausanne, Switzerland
[3]Dipartimento di Matematica, Università di Trento, I-38050 Povo (Trento)

In this paper, we address parallel preconditioners issuing from domain decomposition (DD) [19, 25].

DD is nowadays a very active area of research in the field of numerical approximation of boundary value problems for PDEs. The key idea of (homogeneous) domain decomposition techniques is to split the original computational domain, say Ω, into smaller components $\Omega_i, i = 1, \ldots, M$ (which may or may not overlap), called subdomains, and then to solve the boundary value problem on each subdomain. Clearly, additional interface conditions have to be imposed on $\partial\Omega_i \backslash \partial\Omega$. Usually, the original global problem is reformulated as an iterative procedure on the subdomains.

In overlapping DD preconditioners, local Dirichlet-type problems are then solved on each Ω_i. The communication between the solutions on the different subdomains is here guaranteed by the overlapping regions. This procedures, often referred to as one-level Schwarz method, is non-scalable. In fact, the information exchange among the subdomains is only through the overlapping regions. A good scalability may be recovered by the addition of a coarse correction, whose role is to spread out information among far away subdomains.

For problems arising from PDEs, a typical choice to define the coarse correction is to solve the original PDE problem on a coarse grid. However, the construction of a coarse grid and of the corresponding restriction and extension operators may be difficult or computationally expensive on general unstructured grids and arbitrary geometries. In addition, for a general, possibly non-convex domain, it is difficult to ensure that the boundary conditions are correctly represented on the coarse level.

To overcome these difficulties, the VO decomposition easily allows the definition of a coarse operator based on aggregation procedures. This procedure has been previously used for the shallow water equations and 2D potential flows [17], for 3D potential flow computations [6], for groundwater flows [9], for multiphase flows [11], for discontinuous Galerkin approximation of advection-diffusion problems [12], and for the 3D compressible Euler equations on unstructured tetrahedral grids [23, 24], to mention some.

On the other side, non-overlapping DD preconditioners are usually reformulated in terms of the so-called Schur complement (SC) system. The idea is to partition the set of nodes assigned to each subdomain into the set of boundary and internal nodes. This latter set can be "condensed". Thus, instead of solving a linear system with matrix A, one can solve a linear system with the SC matrix S. If A is symmetric and positive definite matrix, the spectral condition number $\kappa(S)$ of S is lower than $\kappa(A)$ [2].

The first step in every DD approach is the decomposition of the computational domain Ω into the subdomains. Let $\Omega \subset \mathbb{R}^d$, $d = 2, 3$, be a polygonal domain. Let $\mathcal{T}_h$ be a triangulation made of polygons or polyhedra, and, for the sake of simplicity, assume that the boundary of Ω coincides with that of the triangulation. We may face the two following cases. In the first approach, $\mathcal{T}_h$ is partitioned into M non-overlapping parts, namely $\mathcal{T}_{h,1}, \mathcal{T}_{h,2}, \ldots, \mathcal{T}_{h,M}$. For each $i = 1, \ldots, M$ we associate to $\mathcal{T}_{h,i}$ a subdomain Ω_i, formed by the interior of the union of the elements of $\mathcal{T}_{h,i}$, while Ω_Γ is composed by a finite number of $d - 1$ manifolds; see Figure 1, left, for an example in the case $M = 2$. This situation represents the common case where Ω_Γ is the union of the part of the boundary of Ω_i that is internal to Ω. This type of decomposition is called *element-oriented* (EO) decomposition, because each element of $\mathcal{T}_h$ belongs exclusively to one of the M subdomains Ω_i.

An alternative approach consists of partitioning Ω into $M + 1$ non-overlapping

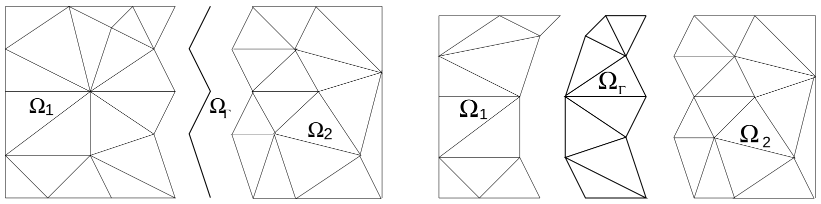

Figure 1: Example of element-oriented (left) and vertex-oriented (right) decomposition.

parts $\mathcal{T}_{h,1}$, $\mathcal{T}_{h,2}, \dots, \mathcal{T}_{h,M}$, and $\mathcal{T}_{h,\Gamma}$. By construction, the portion of space Ω_Γ of which $\mathcal{T}_{h,\Gamma}$ is a triangulation, is now formed by the union of a finite number of "strips" laying between the Ω_i. This is called *vertex-oriented* (VO) decomposition, because each vertex belongs exclusively to one of the M subdomains Ω_i. This time Ω_Γ is a $d-$dimensional domain as well (see Figure 1, right).

Note that by construction, in an EO decomposition all the vertices on Ω_Γ are shared by two or more subdomains Ω_i, while in a VO decomposition they belong to exactly two subdomains: Ω_Γ and one of the Ω_i. EO decompositions lead to well-known SC based schemes, while VO decompositions allow a variational reinterpretation and give rise to new schemes.

From the point of view of parallel computing, a distinguishing feature of VO decompositions is that the transition region Ω_Γ can be replicated among the processors which hold the Ω_i. This provides a way for data communication. In particular, if the discrete operator associated to the matrix A has a compact stencil, matrix-vector products based on VO decompositions may be easily computed in parallel.

VO techniques are also adopted by many parallel linear algebra packages. Indeed, with a VO approach one may derive the local operators directly from the assembled global matrix A, while adopting an EO decomposition would require to work at the level of the (problem dependent) assembly process. On the other side, some quasi-optimal results have been proved for EO SC matrices.

In this paper we mainly focus on a new Dirichlet-Neumann domain decomposition preconditioner that we call *swiss-carpet* preconditioner, based on a VO decomposition like that presented in Figure 2, right. The Dirichlet step is performed in parallel in the domains Ω_i, while the Neumann step is performed in the domain Ω_Γ. When considering the stiffness matrix deriving by the finite element approximation of the Laplace operator in two dimensions, the condition number of the preconditioned system turns out to be $O(H/h)$, H being a bound for the diameter of Ω_i and h the mesh size. We point out that this estimate is less favourable than others, like that obtained for the balancing Nemann-Neumann method or the FETI method; however, the algorithm is *one-level* (no coarse space), and the preconditioner is *cheap*, as only one Neumann solver in the domain Ω_Γ (that contains very few mesh nodes) is needed. This problem is global and, at some extent, plays the role of the coarse correction.

The paper is organised as follows. In Section 2 we introduce the algebraic formalism for EO and VO decompositions. Linear system (1) is reformulated for both the EO and the VO approach, pointing out the main differences. The focus is mainly on Schur complement systems. The EO approach has been extensively investigated in the literature (see for instance [19, 14, 3, 25]); hence, in Section 3 we present several preconditioners for the VO decomposition. The solution of both the unreduced system and the SC system is considered. For the latter, a $O(H/h)$-theoretical estimate of the condition number is obtained. Numerical results are presented in Section 4, while in Section 5 we draw some conclusions.

2 Element-oriented and vertex-oriented decomposition

The difference between EO and VO decompositions can be easily appreciated in the case of two subdomains. Let us introduce some notation first. The nodes on $\overline{\Omega_\Gamma}$ will be called *border* nodes, and in particular those on $\overline{\Omega_i} \cap \overline{\Omega_\Gamma}$ are the border nodes of the domain Ω_i. A node of $\overline{\Omega_i}$ which is not a border node is said to be *internal* to Ω_i, $i = 1, \dots, M$.

For vectors, we will consistently use the subscripts I and B to indicate internal and border, respectively, while the superscript (i) will denote the domain we are referring to. Thus, $\mathbf{u}_I^{(i)}$ will indicate the vector of unknowns associated to nodes internal to Ω_i, while $\mathbf{u}_B^{(i)}$ is associated to the border nodes. The dimension of these vectors will be denoted $n_I^{(i)}$ and $n_B^{(i)}$, respectively. For ease of notation, we will often refrain from distinguishing between a domain Ω and its triangulation $\mathcal{T}_h$, when no ambiguity is possible. Unless otherwise specified, for matrices we will use the subscript i to indicate the subdomain to which we are referring to.

Let us consider the EO approach first. By rewriting equation (1) so that the values associated to nodes internal to Ω_1 are ordered first, followed by the ones internal to Ω_2, and finally those on Ω_Γ, one obtains

$$\begin{pmatrix} A_{II}^{(1)} & 0 & A_{IB}^{(1)} \\ 0 & A_{II}^{(2)} & A_{IB}^{(2)} \\ A_{BI}^{(1)} & A_{BI}^{(2)} & A_{BB}^{(1)} + A_{BB}^{(2)} \end{pmatrix} \begin{pmatrix} \mathbf{u}_I^{(1)} \\ \mathbf{u}_I^{(2)} \\ \mathbf{u}_B \end{pmatrix} = \begin{pmatrix} \mathbf{f}_I^{(1)} \\ \mathbf{f}_I^{(2)} \\ \mathbf{f}_B \end{pmatrix}. \tag{4}$$

If A is symmetric, then $A_{BI}^{(i)} = {A_{IB}^{(i)}}^T$. $A_{BB}^{(i)}$ represents the mutual influence of the border nodes of Ω_i. For example, in the case of a finite element discretisation, one has

$$[A_{BB}^{(i)}]_{k,j} = a(\varphi_j|_{\Omega_i}, \varphi_k|_{\Omega_i}),$$

where φ_k, φ_j are the finite element shape functions associated to border nodes k and j, respectively, restricted to the subdomain Ω_i. The two null blocks arise because of the local support of the shape functions.

Eliminating the internal nodes from system (4) gives

$$S_{EO}\mathbf{u}_B = \mathbf{g} \tag{5}$$

where $S_{EO} = S_1 + S_2$ and $\mathbf{g} = \mathbf{g}^{(1)} + \mathbf{g}^{(2)}$, with

$$\begin{aligned} S_i &= A_{BB}^{(i)} - A_{BI}^{(i)} {A_{II}^{(i)}}^{-1} A_{IB}^{(i)} && (6) \\ \mathbf{g}^{(i)} &= \mathbf{f}_B^{(i)} - A_{BI}^{(i)} {A_{II}^{(i)}}^{-1} \mathbf{f}_I^{(i)}. && (7) \end{aligned}$$

The generalisation to an arbitrary number of subdomains, see Figure 2, left, is straightforward. We introduce the restriction matrix $\bar{R}_i \in \mathbb{R}^{n_{B,i} \times n_B}$ which from a

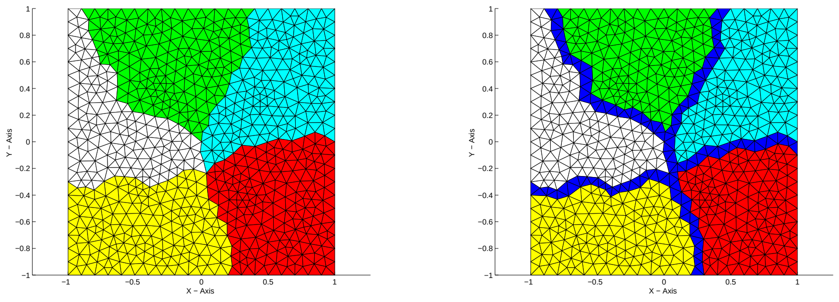

Figure 2: On the left, element-oriented decomposition: each element is assigned to a different subdomain $\Omega_i, i = 1, \dots, M$. On the right, vertex-oriented decomposition: each vertex is assigned to a different subdomain. The blue domain is called Ω_Γ.

global vector $\mathbf{u}_B \in \mathbb{R}^{n_B}$ extracts the entries corresponding to the border nodes of Ω_i, i.e. $\mathbf{u}_{B,i} = \bar{R}_i \mathbf{u}_B$. Its transpose $\bar{R}_i^T$ is the prolongation matrix relative to the subdomain Ω_i which extends by zero a vector of border values. Then,

$$S_{EO} = \left(\sum_{i=1}^{M} \bar{R}_i^T S_i \bar{R}_i \right), \quad \mathbf{g} = \mathbf{f}_B - \sum_{i=1}^{M} \bar{R}_i^T A_{BI}^{(i)} {A_{II}^{(i)}}^{-1} \mathbf{f}_I^{(i)}.$$

For the VO approach, in the case of two subdomains, we have

$$A\mathbf{u} = \begin{pmatrix} A_{II}^{(1)} & A_{IB}^{(1)} & 0 & 0 \\ A_{BI}^{(1)} & A_{BB}^{(1)} + A_\Gamma^{(1,1)} & 0 & A_\Gamma^{(1,2)} \\ 0 & 0 & A_{II}^{(2)} & A_{IB}^{(2)} \\ 0 & A_\Gamma^{(2,1)} & A_{BI}^{(2)} & A_{BB}^{(2)} + A_\Gamma^{(2,2)} \end{pmatrix} \begin{pmatrix} \mathbf{u}_I^{(1)} \\ \mathbf{u}_B^{(1)} \\ \mathbf{u}_I^{(2)} \\ \mathbf{u}_B^{(2)} \end{pmatrix} = \begin{pmatrix} \mathbf{f}_I^{(1)} \\ \mathbf{f}_B^{(1)} \\ \mathbf{f}_I^{(2)} \\ \mathbf{f}_B^{(2)} \end{pmatrix}. \quad (8)$$

The border nodes are split in two subsets, one belonging to $\overline{\Omega_1}$, the other to $\overline{\Omega_2}$. A_Γ represents the mutual influence of border nodes, considering only the contribution from the subdomain Ω_Γ. For instance, in the case of a finite element discretisation, one has

$$[A_\Gamma^{(l,m)}]_{i,j} = a(\varphi_j|_{\Omega_\Gamma}, \varphi_i|_{\Omega_\Gamma})$$

for $i \in \overline{\Omega_l} \cap \overline{\Omega_\Gamma}$ and $j \in \overline{\Omega_m} \cap \overline{\Omega_\Gamma}$.

The elimination of the vector associated to the internal nodes leads to the system

$$S_{VO}\mathbf{u}_B = \begin{pmatrix} S_1 + A_\Gamma^{(1,1)} & A_\Gamma^{(1,2)} \\ A_\Gamma^{(2,1)} & S_2 + A_\Gamma^{(2,2)} \end{pmatrix} \begin{pmatrix} \mathbf{u}_B^{(1)} \\ \mathbf{u}_B^{(2)} \end{pmatrix} = \begin{pmatrix} \mathbf{g}^{(1)} \\ \mathbf{g}^{(2)} \end{pmatrix}, \quad (9)$$

with S_i and $\mathbf{g}^{(i)}$ defined by (6) and (7). Its generalisation to the case of M subdomains

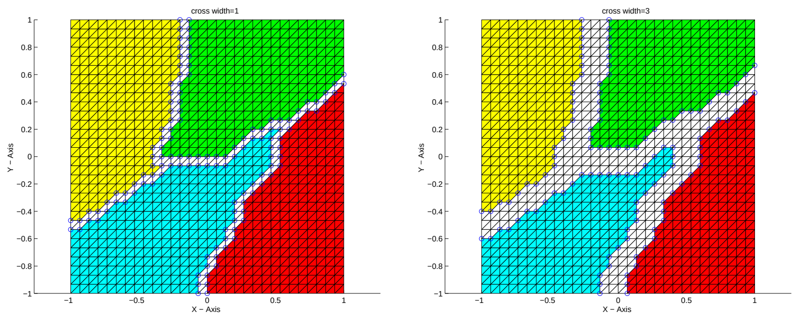

Figure 3: Two VO unstructured decompositions, with $M = 4$ and cross width $\delta = 1$ (left) and $\delta = 3$ (right).

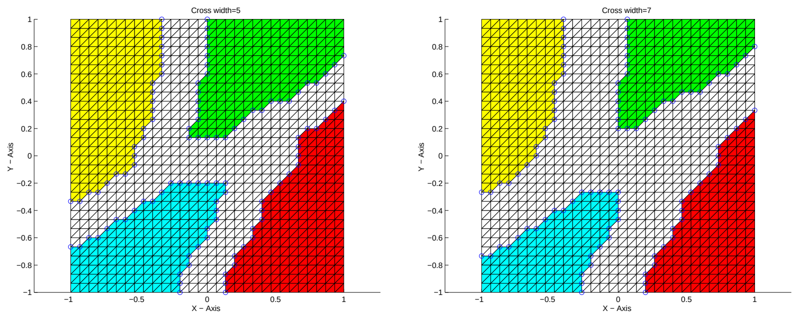

Figure 4: Two VO unstructured decompositions, with $M = 4$ and cross width $\delta = 5$ (left) and $\delta = 7$ (right).

gives

$$\begin{pmatrix} S_1 + A_\Gamma^{(1,1)} & A_\Gamma^{(1,2)} & \dots & A_\Gamma^{(1,M)} \\ A_\Gamma^{(2,1)} & S_2 + A_\Gamma^{(2,2)} & \dots & A_\Gamma^{(2,M)} \\ \vdots & \vdots & \ddots & \vdots \\ A_\Gamma^{(M,1)} & A_\Gamma^{(M,2)} & \dots & S_M + A_\Gamma^{(M,M)} \end{pmatrix} \begin{pmatrix} \mathbf{u}_B^{(1)} \\ \mathbf{u}_B^{(2)} \\ \vdots \\ \mathbf{u}_B^{(M)} \end{pmatrix} = \begin{pmatrix} \mathbf{g}^{(1)} \\ \mathbf{g}^{(2)} \\ \vdots \\ \mathbf{g}^{(M)} \end{pmatrix}, \tag{10}$$

Some of the blocks $A_\Gamma^{(i,j)}$ may be in fact empty. This happens when no element in the triangulation contains nodes belonging to both $\overline{\Omega_i}$ and $\overline{\Omega_j}$.

For a decomposition with many subdomains, let us indicate with Δ_{ij} the minimum number of edges that one has to cross when moving from an arbitrary node belonging to Ω_i to a node of Ω_j, $i, j = 1, \dots, M, i \neq j$; then

$$\delta = \min_{i,j} \Delta_{ij}$$

represents the width of Ω_Γ. In the VO decomposition described so far, we have $\delta = 1$. However, this value can be easily increased using the following algorithm.

Algorithm 1.

```
1  For $k = 1, \dots, N$ Do
2      For every element $j$ of $\mathcal{T}_{h,i}$, $i = 1, \dots, M$ Do:
3          If $k$ is a border node, assign the element $j$ to $\mathcal{T}_{h,\Gamma}$
4      End For
5      Update the set of internal and border nodes
6  End For
```

Example of VO decomposition for $M = 4$ and $\delta = 1, 3, 5, 7$ are given in figures 3 and 4.

3 VO DD Preconditioners

3.1 Solution of the unreduced system

The VO decomposition can be used to define an overlapping (Schwarz) preconditioner. Let us introduce the matrices

$$A_i = \begin{pmatrix} A_{II}^{(i)} & A_{IB}^{(i)} \\ A_{BI}^{(i)} & A_{BB}^{(i)} + A_{\Gamma}^{(i,i)} \end{pmatrix}, \; i = 1, \dots, M.$$

Note that $A_i = R_i A R_i^T$, where R_i is the rectangular (restriction) matrix which, when applied to a vector corresponding to $\overline{\Omega}$, returns its components in $\overline{\Omega_i}$.

A two-level Schwarz preconditioner $P_{S,C,add}$ satisfies

$$P_{S,C,add}^{-1} = \sum_{i=0}^{M} R_i^T A_i^{-1} R_i. \tag{11}$$

A_0 is the so-called *coarse operator*, whose role is to spread out information among the far-away subdomains. The construction of A_0 involves the discretisation of the PDE problem on a coarser grid, whose number of elements n_0 is much lower than that of the original fine grid.

This procedure may be difficult or computationally expensive on general unstructured grids and arbitrary geometries. In addition, for a general, possibly non-convex domain, it is difficult to ensure that the boundary conditions are correctly represented on the coarse level.

In VO decompositions, however, the definition of a coarse operator can be based on aggregation procedures. Precisely, on each subdomain Ω_i we build $n_{0,i}$ *aggregates* $\vartheta_{i,s}, i = 1, \dots, M, s = 1, \dots, n_{0,i}$, each of them representing a set of contiguous vertices, see Figure 5 for an example in 2D. Then on every Ω_i, each aggregate is associated to a vector $\boldsymbol{\eta}_{s,i}$, $s = 1, \dots, n_{0,i}$, whose elements are built following the rule

$$\boldsymbol{\eta}_{s,i}(k) = \begin{cases} 1 & \text{if } k \text{ belongs to } \vartheta_{s,i} \\ 0 & \text{otherwise.} \end{cases}$$

At this stage, the construction of R_0, called smoothed aggregation (SA) procedure, is made in 2 steps: coarsening and grid transfer construction. The former corresponds to build a non-smoothed restriction operator $\tilde{R}_0$ so that each fine grid node is included in just one aggregate. This decomposition can be obtained using a graph partitioning algorithm (vertex-oriented decomposition of the grid). The value $n_0 = \sum_{i=1}^{M} n_{0,i}$ represents the dimension of the coarse space, since each aggregate will be given a

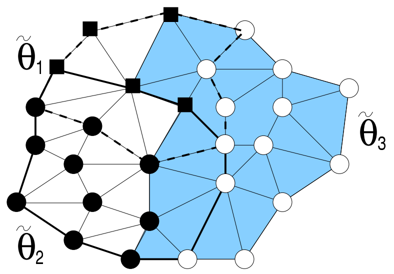

Figure 5: Examples of aggregates for a 2D domain. The nodes marked with '■', '○', and '•' belong to $\tilde{\vartheta}_1$, $\tilde{\vartheta}_2$, and $\tilde{\vartheta}_3$, respectively.

coarse grid basis function. We will indicate with $\tilde{\vartheta}_i$ the set of nodes that form the non-smoothed aggregate i. The entries of $\tilde{R}_0$ are thus as follows:

$$\tilde{R}_0(i,j) = \begin{cases} 1 & \text{if } j \in \tilde{\vartheta}_i \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

$\tilde{R}_0$ can be viewed as a simple grid transfer operator corresponding to piecewise constant interpolation. A 2D example is reported in figure 5. The second step (that may actually be avoided) consists of applying a prolongation smoother $\tilde{S}_0$ to produce the final prolongation operator $R_0^T = \tilde{R}_0^T \tilde{S}_0$.

For a more exhaustive presentation on this subject, the reader is referred to [21].

3.2 Preconditioners for the SC system

Equation (10) may be reinterpreted as a two-subdomain formulation, the two subdomains being given by Ω_Γ and $\Omega_\star = \cup_{i=1}^M \Omega_i$. Then

$$S_{VO} = S_\star + A_\Gamma, \tag{13}$$

with $S_\star = \text{blockdiag}\{S_i, i = 1, \dots, M\}$.

In the two-subdomain formulation $\overline{\Omega} = \overline{\Omega_\star} \cup \overline{\Omega_\Gamma}$, $S_\star$ is the Schur complement obtained by eliminating the internal nodes of $\Omega_\star$, while the Schur complement of domain Ω_Γ, usually indicated by S_Γ, coincides with A_Γ, since no internal nodes have been included in Ω_Γ.

Note that $S_\star$ has a (dense) diagonal block form, while A_Γ is sparse and, in general, features many zero-blocks.

Three preconditioners may be derived for the matrix (13):

$$P_\star = S_\star \; ; \quad P_\Gamma = A_\Gamma \; ; \quad P_{NN} = (S_\star^{-1} + A_\Gamma^{-1})^{-1}. \tag{14}$$

$P_\star$ and P_Γ are formally equivalent to Dirichlet-Neumann (DN) preconditioners [19, Section 3.2]. The inversion of $P_\star$ involves M Neumann solves (one in every Ω_i), whereas only one Neumann solve in Ω_Γ is required to invert P_Γ. The latter solve is somehow unusual since all the nodes of Ω_Γ stand on the border of the domain Ω_Γ. P_{NN} is the 2-subdomain Neumann-Neumann (NN) preconditioner.

Remark 1. *The DN and NN preconditioners are optimal for the two-subdomain case. However, the analysis requires the subdomains to have a measure independent of h. This is not verified by preconditioners $P_\star$, P_Γ, and P_{NN}, since the width of Ω_Γ is h. Clearly, if the subdivision is modified so that Ω_Γ has a measure independent of h, then $P_\star$, P_Γ, and P_{NN} become optimal preconditioners.*

The P_Γ preconditioner behaves better (both theoretically and numerically) than the $P_\star$ preconditioner (see the theoretical results in Proposition 1, that we have been able to prove only for the P_Γ preconditioner; see moreover the numerical results in Table 5). On the other hand, though the condition number of $P_{NN}^{-1}S_{VO}$ is in general slightly lower than that of $P_\Gamma^{-1}S_{VO}$ (see Table 5), the computational complexity of P_{NN} is higher than that of P_Γ, as it requires $M+1$ solves. Moreover, in the "floating" subdomains Ω_i (those internal to Ω), the solution of a pure Neumann problem can demand for compatibility conditions on the data. For this reason, in the sequel we will focus almost exclusively on the preconditioner P_Γ.

Remark 2. *We take the liberty of calling P_Γ the swiss-carpet preconditioner. In fact, this is a little abuse that we grant ourselves, and that is motivated by the following analogy. A typical VO decomposition can be obtained starting from the Swiss flag (a white cross on a red background) and applying to it a recursive procedure (see Figure 6) somehow similar to the one used for constructing the Sierpiński carpet (see, e.g., [16], p. 144). Indeed, the real procedure consists of moving from step n to step $n+1$ by eliminating from each square a central cross of thickness equal to the finite element mesh-size h.*

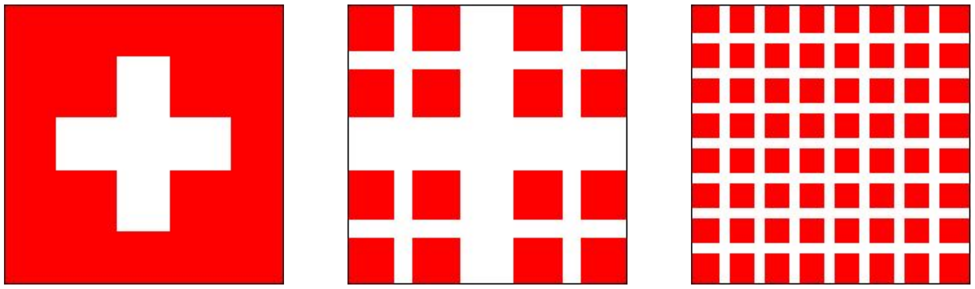

Figure 6: Derivation of the Swiss-carpet preconditioner (warning: this analogy is not rigorous): starting from the Swiss flag (on the left), we apply a recursive procedure, to obtain the desired decomposition (on the right).

We now derive a convergence estimate for the swiss-carpet preconditioner. For the sake of simplicity, the estimate of the condition number of the preconditioned Schur complement $P_\Gamma^{-1}S$ will be presented only in a case of simple geometry and the Laplace operator with homogeneous boundary conditions. However, using similar arguments, the interested reader could easily extend the result to many more general situations (for instance, the result is true for any second order elliptic operator (without zero order terms) and being associated to a symmetric and coercive bilinear form, or for the two-dimensional elasticity system; however, other boundary conditions and domains are allowed as well).

Let us consider the square $\Omega = (0,1)\times(0,1)$, and a family of uniform triangulations $\mathcal{T}_h$ of it. We assume that Ω is subdivided into the union of the disjoint squares Ω_j,

$j = 1, \dots, M$, with side-length H such that $\sqrt{M}(H+h) = 1+h$, and of the domain Ω_Γ, which is connected and consists of the union of $2(\sqrt{M}-1)$ non-disjoint strips of length 1 and width h (see Figure 6). Let us also set

$$\Omega_\star = \cup_{j=1}^{M} \Omega_j \quad , \quad \Omega_\Gamma = \Omega \setminus \overline{\Omega_\star} \quad , \quad \Gamma = \overline{\Omega_\star} \cap \overline{\Omega_\Gamma} \ . \tag{15}$$

We denote by η_h the trace on Γ of the piecewise *linear* functions v_h defined in Ω. Then, we indicate by $E_{h,\star}\eta_h$ the discrete harmonic extension of η_h in $\Omega_\star$, and by $E_{h,\Gamma}\eta_h$ the discrete harmonic extension of η_h in Ω_Γ. Since in Ω_Γ there are no internal nodes (the width of Ω_Γ is equal to h), the function $E_{h,\Gamma}\eta_h$ is simply the interpolant of η_h in $\overline{\Omega_\Gamma}$.

The action of both the Schur complement S and P_Γ can be expressed through the local bilinear forms in the following way (here $\boldsymbol{\eta}$ and $\boldsymbol{\mu}$ denote the vector of the nodal values of η_h and μ_h, respectively) (see, e.g., [19, pag. 22]):

$$S\boldsymbol{\eta} \cdot \boldsymbol{\mu} = \textstyle\int_{\Omega_\star} \nabla E_{h,\star}\eta_h \cdot \nabla E_{h,\star}\mu_h + \int_{\Omega_\Gamma} \nabla E_{h,\Gamma}\eta_h \cdot \nabla E_{h,\Gamma}\mu_h$$

$$P_\Gamma\boldsymbol{\eta} \cdot \boldsymbol{\mu} = \textstyle\int_{\Omega_\Gamma} \nabla E_{h,\Gamma}\eta_h \cdot \nabla E_{h,\Gamma}\mu_h \ .$$

We can prove the following proposition.

Proposition 1. *There is a positive constant C_1 independent of h and H such that*

$$P_\Gamma\boldsymbol{\eta}\cdot\boldsymbol{\eta} \le S\boldsymbol{\eta}\cdot\boldsymbol{\eta} \le C_1 \frac{H}{h} P_\Gamma\boldsymbol{\eta}\cdot\boldsymbol{\eta} \quad \forall \boldsymbol{\eta} \in \mathbb{R}^{n_B}.$$

Therefore, the condition number of $P_\Gamma^{-1}S$ satisfies $\kappa(P_\Gamma^{-1}S) \le C_1 H/h$.

Proof. The first inequality is obvious. To prove the second one, we need to bound $\int_{\Omega_\star} |\nabla E_{h,\star}\eta_h|^2$, and this can be done by adding the contributions to the integral from each square Ω_j.

By means of a translation, instead of Ω_j we can consider the square Q_H centred in $(0,0)$ and with sides of length H. Then by the change of variable $(\widehat{x},\widehat{y}) = T(x,y) = (x/H, y/H)$, we map Q_H over the unit square $\widehat{Q}$. The triangulation over Q_H is transformed into a triangulation onto $\widehat{Q}$; the correspondent mesh size is given by h/H. Introducing, for each function g defined in Q_H, the function $\widehat{g} = g \circ T^{-1}$ defined in $\widehat{Q}$, and denoting by $\widehat{E}_{h/H}\widehat{\eta_h}$ the discrete harmonic extension of $\widehat{\eta_h}$ with respect to the triangulation in $\widehat{Q}$, it is easily verified that $\widehat{E}_{h/H}\widehat{\eta_h} = \widehat{E_{h,\star}\eta_h}$ and that

$$\int_{Q_H} |\nabla E_{h,\star}\eta_h|^2 = \int_{\widehat{Q}} |\widehat{\nabla}\widehat{E}_{h/H}\widehat{\eta_h}|^2. \tag{16}$$

The uniform extension theorem for finite elements (see, e.g., [27], Sect. 4.2.2) gives that

$$\int_{\widehat{Q}} |\widehat{\nabla}\widehat{E}_{h/H}\widehat{\eta_h}|^2 \le C_0 |\widehat{\eta}_h|^2_{1/2,\partial\widehat{Q}} \ , \tag{17}$$

where the constant C_0 does not depend on h and H and $|\cdot|_{1/2,\partial\widehat{Q}}$ denotes the seminorm of $H^{1/2}(\partial\widehat{Q})$.

Let us now observe that the set Ω_Γ is the union of $F_j \cap \Omega$, where the "frames" F_j are the sets of width h around Ω_j(note that these sets are not disjointed, but any possible intersection concerns at most four of them). Moreover, by means of a translation, we can think that each set F_j is the frame $F_{H,h}$ of the square Q_H.

Let us consider now the set $\widehat{F}_{1,1/2}$, that is, the frame of width 1/2 around $\widehat{Q}$. We want to construct a function in $\widehat{F}_{1,1/2}$ such that its value on the internal boundary of $\widehat{F}_{1,1/2}$ (that is, on $\partial\widehat{Q}$) is equal to $\widehat{\eta}_h$. The natural idea is to start from $E_{h,\Gamma}\eta_h$, that is, from the interpolant of η_h in $\overline{\Omega_\Gamma}$. For simplicity, we will write $\psi_h = E_{h,\Gamma}\eta_h$, and we recall that $\psi_{h|\Gamma} = \eta_h$.

Now we have to pass from the domain F_j (or, equivalently, from $F_{H,h}$) onto $\widehat{F}_{1,1/2}$. First of all, note that the transformation T maps $F_{H,h}$ onto the set $\widehat{F}_{1,h/H}$. Moreover, we define

$$t(\xi) := \begin{cases} \xi & \text{if } |\xi| \le 1/2 \\ 1/2[1 + h^{-1}H(|\xi| - 1/2)]\xi/|\xi| & \text{if } 1/2 \le |\xi| \le 1/2 + h/H \end{cases} ,$$

then we introduce the transformation

$$K(\widehat{x}, \widehat{y}) := (t(\widehat{x}), t(\widehat{y})) ,$$

that maps $\widehat{F}_{1,h/H}$ onto $\widehat{F}_{1,1/2}$, leaving the internal boundary $\partial\widehat{Q}$ unchanged. We are now in a position to set

$$\psi^\sharp : \widehat{F}_{1,1/2} \to \mathbb{R} \ , \ \psi^\sharp := \psi_{h|F_{H,h}} \circ T^{-1} \circ K^{-1} ,$$

and we easily verify that $\psi^\sharp_{|\partial\widehat{Q}} = \widehat{\eta}_h$.

We can thus use the trace theorem and obtain

$$|\widehat{\eta}_h|^2_{1/2,\partial\widehat{Q}} \le C_1 ||\nabla\psi^\sharp||^2_{0,\widehat{F}_{1,1/2}} \tag{18}$$

(see [27, Theorem 4.1]). To estimate the right hand side, taking into consideration the explicit expression of the transformations T and K, we can prove

$$||\nabla\psi^\sharp||^2_{0,\widehat{F}_{1,1/2}} \le C_2 h^{-1} H \int_{F_{H,h}} |\nabla\psi_h|^2 . \tag{19}$$

By collecting (16), (17), (18) and (19) we have thus obtained

$$\begin{aligned} \textstyle\int_{\Omega_j} |\nabla E_{h,\star}\eta_h|^2 &\le C_3 h^{-1} H \textstyle\int_{F_j} |\nabla E_{h,\Gamma}\eta_h|^2 \\ &= C_3 h^{-1} H \textstyle\int_{F_j\cap\Omega} |\nabla E_{h,\Gamma}\eta_h|^2 \end{aligned} \tag{20}$$

(here, for the squares Ω_j that have at least one side on $\partial\Omega$, the function η_h has to be considered extended by 0 on the boundary of Ω and on the boundary of F_j external to Ω).

Bearing in mind the geometry of Ω_Γ we easily obtain the estimate

$$4 \int_{\Omega_\Gamma} |\nabla E_{h,\Gamma}\eta_h|^2 \ge \sum_j \int_{F_j\cap\Omega} |\nabla E_{h,\Gamma}\eta_h|^2 .$$

The thesis now follows by summing up the contribution from all the domains Ω_j and the frames F_j. □

Remark 3. *As we already noted, the above result also holds for the elliptic operator*

$$Lu := -\sum_i D_i[\rho(x)D_i u] \ ,$$

where $\rho(x)$ is a positive and bounded function in Ω.

A more precise conclusion can be reached if the coefficient $\rho(x)$ satisfies $\rho(x) = \rho_j > 0$ in $\Omega_j \cup (F_j^{1/2} \cap \Omega)$, $j = 1, \ldots, M$, where $F_j^{1/2}$ is the frame of width $h/2$ around Ω_j: in this case the constant C_1 in Proposition 1 does not depend on the jumps of the coefficients ρ_j.

In fact, in each domain Ω_j the extension operator with respect to the operator L coincides with the harmonic extension, therefore (17) holds. Moreover, proceeding as before we obtain (20) (with $F_j^{1/2}$ instead of F_j) and therefore

$$\begin{aligned} S_\star \boldsymbol{\eta} \cdot \boldsymbol{\eta} \ &= \textstyle\sum_{j=1}^M \rho_j \int_{\Omega_j} |\nabla E_{h,\star}\eta_h|^2 \\ &\leq C_3 h^{-1} H \textstyle\sum_{j=1}^M \rho_j \int_{F_j^{1/2}\cap\Omega} |\nabla E_{h,\Gamma}\eta_h|^2 \\ &= C_3 h^{-1} H P_\Gamma \boldsymbol{\eta} \cdot \boldsymbol{\eta} \ , \end{aligned}$$

where the constant C_3 does not depend on the jumps of the coefficients ρ_j.

4 Numerical Experiments

We consider a finite element approximation of the problem

$$\begin{cases} -\Delta u &= f \quad \text{in } \Omega \\ \quad u &= g \quad \text{on } \partial\Omega, \end{cases} \tag{21}$$

(where the data f and g correspond to $u(x,y) = \sin x \, \sin y$ in 2D and to $u(x,y,z) = \sin x \, \sin y \, \sin z$ in 3D), with the aim of numerically validating the VO DD preconditioners presented in Section 3, and comparing them with popular DD preconditioners.

For 2D problems, $\Omega = (0,1) \times (0,1)$, and the mesh is built by dividing Ω into $(1/h)^2$ squares of uniform size, which are then subdivided into two triangles. For 3D problems, $\Omega = (0,1) \times (0,1) \times (0,1)$, and the mesh is divided into $(1/h)^3$ hexahedra, subdivided into 5 tetrahedra.

As regards the domain decomposition, we consider overlapping squares (for 2D problems) or hexahedra (for 3D problems) Ω_i, of equal area H^2 or volume H^3, plus the subdomain Ω_Γ in the case of a VO decomposition. If not otherwise specified, the width of Ω_Γ is of one element. We use piecewise *linear* finite elements, and we solve the linear systems (1), (5), and (10) by a preconditioned conjugate gradient (PCG) method [8, 20].

The tables report the estimated condition number (according to the algorithm given in [7], pp. 128–130) of the preconditioned system.

We start by considering the solution of system (1), using a one-level Schwarz preconditioner on a VO decomposition of the grid. In this case, Ω_Γ represents the union of the overlapping regions among the subdomains Ω_i. In Table 1 we report the estimated condition number (ECN) of the preconditioned system. Table 2 reports the ECN for two-level Schwarz preconditioners, using respectively a coarse grid, and the aggregation procedure with non-smoothed aggregation. One can appreciate that both methods share the same asymptotic behaviour with respect to H and h (see for instance [13]). The easiness in the construction of the aggregation coarse space has a drawback in that, for given H and h, the condition number results larger than using a classical coarse space. For further results on this subject, see [22].

Then, we focus on the solution of system (10). As it can be expected, for regular decompositions as that presented in Proposition 1 the Schur matrix S_{VO} is bad conditioned: from Table 3 one sees that $\kappa(S_{VO})$ is slightly more than linear with h^{-1} in 2D, and slightly more than quadratic in 3D: therefore a preconditioner for this matrix is mandatory. The preconditioner P_Γ proposed in Section 3.2 is studied in the same Table 3. The results of Table 3 confirm the theoretical estimates of Proposition 1, and suggest that a similar estimate may hold for 3D domains as well.

The other preconditioners proposed in Section 3.2 are studied in Table 5. For a given h, we have first constructed a VO decomposition into 4 subdomains, as described in Section 2 (that is, $\delta = 1$). Then, Algorithm 1 was used to increase the width. In the table, $P_{NN}^{-1} = S_\star^{-1} + A_\Gamma^{-1}$. All the tested preconditioners share the same asymptotic behaviour; however, $P_\star$ results in a consistently larger condition number than P_{NN} and P_Γ. As M additional Neumann problems have to be solved to impose P_{NN}, in the following we will focus only on P_Γ.

In Table 6 the ECN of S_{VO} and $P_\Gamma^{-1}S_{VO}$ is again reported, this time for 2D unstructured decompositions into M subdomains; the subdomain partition has been obtained by using the software METIS (see [10]). The analogous results for the 3D case are reported in Table 7.

In order to compare P_Γ with more standard preconditioners, we have used one of the state-of-the-art preconditioners for non-overlapping decompositions, namely the balancing Neumann-Neumann (BNN).

We recall briefly the Neumann-Neumann preconditioner, which is based on the assembly of the global SC from the local subdomain SC matrices:

$$P_{NN}^{-1} = \sum_{i=1}^{M} \bar{R}_i D_i S_i^{-1} D_i \bar{R}_i. \tag{22}$$

The weighting diagonal matrices D_i allow to preserve good conditioning even when the subdomains have different characteristics, for instance because of the different values of the coefficients of the differential operator.

For a small number of subdomains, the convergence of the Neumann-Neumann algorithm is quite good.

It is possible to prove that $\kappa\left(P_{NN}^{-1}S\right) \leq \frac{C}{H^2}\left(1+\log(H/h)\right)^2$. The term $1/H^2$ cannot be avoided with this formulation since the NN algorithm fails to provide any means of global distribution of information beyond that of local exchange and, as expected, suffers from steep deterioration of convergence as the number of subdomains increases.

As noted in [14], the NN preconditioner becomes impractical when applied to problems with a number of subdomains larger than about 16. Mandel [15] proposed the so-called balancing Neumann-Neumann preconditioner (BNN) by adding to the NN preconditioner a simple coarse correction constructed by using piecewise constant coarse grid space. The corresponding coarse operator involves one unknown per subdomain in the scalar case. It can be proved that the resulting preconditioner satisfies

$$\kappa(P_{BNN}^{-1}A) \leq C\left(1+\log\frac{H}{h}\right)^2. \tag{23}$$

$\kappa(P_S^{-1}A)$	$H = 1/4$	$H = 1/8$	$H = 1/16$	$H = 1/32$
$h = 1/15$	24.29	45.92	-	-
$h = 1/31$	51.66	98.63	194.97	-
$h = 1/63$	106.31	203.80	403.31	804.48
$h = 1/127$	215.58	414.03	819.73	1635.34

Table 1: 2D model problem. Condition number of $P_S^{-1}A$ using the one-level Schwarz preconditioner P_S with minimal overlap ($\delta = h$).

$\kappa(P_{S,C,add}^{-1}A)$	$H = 1/4$	$H = 1/8$	$H = 1/16$	$H = 1/32$
$h = 1/15$	4.17 (12.09)	3.86 (8.53)	-	-
$h = 1/31$	6.77 (25.47)	4.79 (17.23)	4.20 (9.76)	-
$h = 1/63$	12.22 (52.86)	7.19 (34.64)	5.13 (19.06)	4.36 (10.11)
$h = 1/127$	23.08 (107.91)	12.84 (69.61)	7.37 (39.95)	5.35 (20.33)

Table 2: 2D model problem. Condition number of $P_{S,C,add}^{-1}A$, with a standard coarse space (in parentheses: with a coarse space built using non-smoothed aggregation).

For numerical results, we refer the reader to [15, 5] or [19, Section 3.2.2]. See also Table 8.

Comparison between BNN (with an EO decomposition) and P_Γ (with a VO decomposition) are shown in Table 9. The 2D grid has been partitioned, by means of the software METIS, into M subdomains using a VO decomposition. Then, in order to obtain a EO decomposition, each of the elements in Ω_Γ has been assigned to one of the neighbouring subdomains. The SC systems has been solved with a tolerance of 10^{-10}.

In Table 9, `it_Γ` and `it_BNN` indicate the number of iterations to converge using P_Γ and P_{BNN}, respectively. `flops_Γ` and `flops_BNN` refer to the flops required to solve the problem, as indicated by the MATLAB command `flops`. We have not included in the flops count the operations required to explicitly form the SC matrix, and the coarse matrix of BNN. The Neumann problems have been solved using a LU decomposition. The operations needed to compute the LU factors have not been included in the flop count.

Few considerations are in order. BNN always outperforms P_Γ in terms of iterations to converge; however, it is worth to note that the preconditioning phase is much

	$H = 1/2$	$H = 1/4$	$H = 1/8$	$H = 1/16$
$h = 1/15$	41.33 (6.72)	62.75 (4.87)	90.52 (2.00)	- (-)
$h = 1/31$	96.85 (13.54)	155.75 (9.76)	264.73 (5.27)	- (2.00)
$h = 1/63$	209.27 (27.71)	347.06 (19.76)	627.94 (10.83)	- (5.44)
$h = 1/127$	431.87 (54.50)	732.06 (39.48)	1366.15 (21.93)	- (11.09)

Table 3: 2D model problem. Condition number of S_{VO} (in parentheses: that of $P_\Gamma^{-1}S_{VO}$).

	$H = 1/2$	$H = 1/3$	$H = 1/4$
$h = 1/15$	74.45 (5.19)	95.54 (4.11)	- (3.23)
$h = 1/23$	- (7.44)	- (5.54)	- (4.49)
$h = 1/31$	390.13 (9.74)	507.33 (6.87)	596.79 (5.82)
$h = 1/47$	- (14.36)	- (10.61)	- (8.54)
$h = 1/63$	1785.84 (19.01)	2391.22 (13.76)	2892.44 (11.28)

Table 4: 3D model problem. Condition number of S_{VO} (in parentheses: that of $P_\Gamma^{-1} S_{VO}$).

	δ	$\kappa(S_{VO})$	$\kappa(S_\Gamma^{-1} S_{VO})$	$\kappa(S_\star^{-1} S_{VO})$	$\kappa(P_{NN}^{-1} S_{VO})$
$h = 1/15$	1	41.33	6.72	16.01	4.92
	3	20.47	2.77	4.67	1.98
	5	12.95	2.08	2.97	1.15
$h = 1/31$	1	96.85	13.54	31.95	8.86
	3	61.59	5.00	9.66	3.22
	5	48.99	3.36	5.66	2.19
$h = 1/63$	1	209.27	27.27	63.86	16.81
	3	149.69	9.66	20.01	5.84
	5	133.05	6.05	11.69	3.72
$h = 1/127$	1	431.87	54.51	127.79	32.78
	3	329.70	18.71	41.28	11.15
	5	309.76	11.52	24.29	6.88

Table 5: 2D model problem. Condition number for the VO SC matrix, with no preconditioning and using the preconditioners proposed in Section 3. The domain is partitioned into 4 subdomains, and the influence of the cross width is reported. (Note that $S_\Gamma = A_\Gamma$ if $\delta = 1$, but not if $\delta \geq 2$.)

	$M = 12$	$M = 20$	$M = 32$
$h = 1/15$	80.54 (11.67)	85.62 (15.69)	90.42 (7.65)
$h = 1/31$	150.22 (47.16)	184.14 (35.76)	219.02 (17.93)
$h = 1/47$	235.24 (20.60)	286.27 (40.57)	330.72 (24.84)
$h = 1/63$	300.47 (45.93)	382.03 (48.63)	464.09 (34.27)

Table 6: 2D model problem. Condition number of S_{VO} (in parentheses: that of $P_\Gamma^{-1} S_{VO}$). Unstructured decomposition into M subdomains (obtained using METIS).

	$M = 12$	$M = 20$	$M = 32$
$h = 1/15$	84.75 (20.50)	92.00 (22.32)	97.78 (20.54)
$h = 1/31$	458.46 (21.44)	500.93 (21.73)	569.71 (21.50)
$h = 1/47$	1177.10 (23.19)	1253.82 (26.23)	1422.79 (21.83)

Table 7: 3D model problem. Condition number of S_{VO} (in parentheses: that of $P_\Gamma^{-1} S_{VO}$). Unstructured decomposition into M subdomains (obtained using METIS).

$\kappa((P_{BNN})^{-1}S_{EO})$	$H=1/2$	$H=1/4$	$H=1/8$	$H=1/16$
$h=1/15$	1.67	1.48	1.27	-
$h=1/31$	2.17	2.03	1.47	1.29
$h=1/63$	2.78	2.76	2.08	1.55
$h=1/127$	3.51	3.67	2.81	2.07

Table 8: 2D model problem. Condition number of $P_{BNN}^{-1}S_{EO}$.

$1/h$	M	$\kappa(S_{EO})$	`it_BNN`	`flops_BNN`	$\kappa(S_{VO})$	`it_Γ`	`flops_Γ`
1/20	4	34.1	9	4.55e+05	80.5	30	4.09e+05
1/20	8	52.7	13	9.20e+05	109.5	33	5.46e+05
1/20	16	91.28	18	1.80e+06	145.8	36	7.99e+05
1/20	32	111.8	16	2.94e+06	190.6	39	1.08e+06
1/40	4	71.9	10	3.65e+06	180.1	38	1.78e+06
1/40	8	128.2	16	5.90e+06	254.1	48	2.99e+06
1/40	16	168.2	19	8.31e+06	343.9	52	3.96e+06
1/40	32	230.1	20	1.33e+07	462.3	53	4.91e+06
1/40	64	343.8	22	2.66e+07	608.7	51	7.72e+06
1/40	128	463.8	21	5.72e+07	770.4	44	1.29e+07
1/60	4	125.1	12	1.65e+07	268.4	45	4.69e+06
1/60	8	187.7	16	1.82e+07	406.3	59	8.04e+06
1/60	16	255.2	20	2.31e+07	522.0	57	8.97e+06
1/60	32	357.8	22	3.37e+07	712.3	59	1.12e+07
1/60	64	539.4	22	5.61e+07	971.4	59	1.53e+07

Table 9: Comparison of P_{BNN} (EO decomposition) and P_Γ (VO decomposition). `it_BNN` and `it_Γ` indicate the iterations to converge of P_{BNN} and P_Γ, respectively, while `flops_BNN` and `flops_Γ` the MATLAB flops required by the solution of the problem using CG.

cheaper for P_Γ than for BNN, for what concerns both CPU-time and memory storage. In fact, to apply the BNN preconditioner we have to solve a Neumann problem on each subdomain. If a direct method is used, we have to compute the LU decomposition of the matrix A_i, while, in the case of an iterative method, we usually have to provide a (local) preconditioner. On the contrary, to apply P_Γ we have to solve just one Neumann problem in Ω_Γ. Therefore, at least for this test case, P_Γ reveals to be computationally cheaper than P_{BNN}.

For the parallel solution of P_Γ we have investigated two approaches:

1. The parallel direct solver for distributed matrices available in MUMPS (see [1]). The bottleneck of this library for our application is that the vector to be preconditioned must be entirely stored on processor 0 (as the resulting preconditioned vector).

2. A nested iterative solver.

Our numerical results for medium-size problems suggest to solve P_Γ iteratively. In this case, several choices are available. We have considered the following:

h	M	`noprec`	`Jacobi`	`PCG`
1/128	2×2	110	93	26
1/128	3×3	120	135	25
1/128	4×4	134	94	25
1/256	2×2	140	90	26
1/256	3×3	209	106	28
1/256	4×4	256	102	30
1/512	2×2	148	114	39
1/512	3×3	374	163	40
1/512	4×4	428	186	46

Table 10: Iterations to converge for the 2D model problem.

h	M	`noprec`	`Jacobi`	`PCG`
1/16	$2 \times 2 \times 2$	26	16	11
1/16	$4 \times 4 \times 4$	33	21	12
1/32	$2 \times 2 \times 2$	51	24	14
1/32	$4 \times 4 \times 4$	57	29	15
1/64	$2 \times 2 \times 2$	65	34	15
1/64	$4 \times 4 \times 4$	70	40	17

Table 11: Iterations to converge for the 3D model problem.

1. few steps of a simple iterative method, like Jacobi's;
2. a nested Krylov accelerator (possibly with preconditioning).

Note that 2. may result in a non-constant preconditioner, leading PCG to diverge (unless the linear system with P_Γ is solved up to machine precision). For this reason, in the numerical experiments using a nested Krylov solver, we have resorted to GMRESR [26], which allows variable preconditioning.

Tables 10 and 11 report the iterations required to solve up to a tolerance of $||\mathbf{r}_k||/||\mathbf{r}_0|| \leq 10^{-5}$ our model problem on $(0,1) \times (0,1)$ and on $(0,1) \times (0,1) \times (0,1)$, respectively. Different strategies for the solution of P_Γ are investigated. In the table, `Jacobi` indicates two steps of the Jacobi method applied to the linear system

$$P_\Gamma \mathbf{z} = \mathbf{r}, \tag{24}$$

with initial guess $\mathbf{z}_0 = \mathbf{0}$ (using PCG as external Krylov solver), and `PCG` indicates that linear system (24) is solved using the PCG method, iterated up to the fulfillment of the criterion

$$\frac{||\mathbf{r} - P_\Gamma \mathbf{z}_i||_2}{||\mathbf{r}||_2} \leq 10^{-2}$$

(using GMRESR as external Krylov solver).

Table 12 reports the CPU-times, obtained on a LINUX cluster, composed by 12 bi-processor Pentium IV.

h	M	`noprec`	`Jacobi`	`PCG`
1/128	2×2	3.29	2.92	1.27
1/128	3×3	1.42	1.65	0.60
1/128	4×4	0.99	0.75	0.45
1/256	2×2	24.09	16.51	6.41
1/256	3×3	12.93	6.75	2.92
1/256	4×4	8.80	3.68	1.78
1/512	2×2	208.75	184.36	75.39
1/512	3×3	150.32	65.69	22.34
1/512	4×4	80.89	35.84	12.71

Table 12: CPU-time for the 2D model problem, obtained on a LINUX cluster, with a tolerance $\|\mathbf{r}_k\|/\|\mathbf{r}_0\| \leq 10^{-5}$.

5 Conclusions

The choice of a VO or EO decomposition largely affects both the numerical algorithm and the data structures used in a parallel code. Often, a VO approach is preferred since the transition region Ω_Γ may be replicated on the processor which holds Ω_i and provides a means of data communication. In this way if the discrete operator associated to the matrix A has a compact stencil (as it is often the case with finite elements or finite volume discretisations) the matrix-vector product may be easily carried out in parallel. The VO technique is also the matter of choice of many parallel linear algebra packages. Indeed, with a VO approach one may derive the local operators directly from the assembled global matrix A, while adopting an EO decomposition would require to work at the level of the (problem dependent) assembly process. On the other hand, some quasi-optimal results have been proved for EO SC matrices.

We point out that the given estimate for the condition number of the proposed Dirichlet-Neumann VO preconditioner is less favourable than others, like BNN, or FETI; however:

- the method is one-level (no coarse space);
- the preconditioner is cheap: one has to solve only one Neumann problem. This problem is global and, at some extent, plays the role of the coarse correction.

For a scalar implementation one can factorize and store P_Γ, while on a parallel computer a strategy to solve the global problem with P_Γ is necessary. Note that the linear system with P_Γ needs not be solved exactly.

To conclude, we have presented a new approach for the iterative solution of the SC system with many subdomains. Instead of defining a coarse space, we have decomposed the original domain in order to have one subdomain which is connected to all the others. This subdomain Ω_Γ is reduced to the minimum width of 1 element, and it allows the definition of global preconditioners. In particular, among the preconditioners here reported, the best choice seems to be a DN preconditioner, where a Neumann problem is solved in Ω_Γ. Consequently, the preconditioning step requires the solution of only one linear system on a subdomain of a special shape. Numerical tests confirm the good properties of such a decomposition, and comparison with the balancing Neumann/Neumann preconditioner shows that, from the point of view of computational cost, the approach advocated in this paper is competitive.

References

[1] P.R. Amestoy, I.S. Duff and J.-Y. L'Excellent. Multifrontal parallel distributed symmetric and unsymmetric solvers. *Comput. Meth. Appl. Mech. Eng.*, 184 (2000), 501–520.

[2] O. Axelsson. *Iterative Solution Methods.* Cambridge University Press, Cambridge, 1994.

[3] T.F. Chan and T.P. Mathew. Domain decomposition algorithms. *Acta Numerica*, 1994, 61–143.

[4] Y.-H. De Roeck and P. Le Tallec. Analysis and test of a local domain decomposition. In *Fourth International Symposium on Domain Decomposition Methods for Partial Differential Equations*, R. Glowinski et al., eds., pp. 112–128, SIAM, Philadelphia, 1991.

[5] M. Dryja, B.F. Smith, and O.B. Widlund. Schwarz analysis of iterative substructuring algorithms for elliptic problems in three dimensions. *SIAM J. Numer. Anal.*, 31 (1994), 1662–1694.

[6] L. Formaggia, A. Scheinine, and A. Quarteroni. A numerical investigation of Schwarz domain decomposition techniques for elliptic problems on unstructured grids. *Math. Comput. Simulations*, 44 (1994), 313–330.

[7] G.H. Golub and C.F. van Loan. *Matrix Computations.* 2^{nd} ed., The Johns Hopkins University Press, Baltimore, 1989.

[8] M. Hestenes and E. Steifel. Methods of conjugate gradients for solving linear systems. *J. Res. Nat. Bur. Stand.*, 49 (1952), 409–436.

[9] L. Jenkins, T. Kelley, C.T. Miller, and C.E. Kees. An aggregation-based domain decomposition preconditioner for groundwater flow. Technical Report TR00–13, Department of Mathematics, North Carolina State University, 2000.

[10] G. Karypis and V. Kumar. Multilevel k-way partitioning scheme for irregular graphs *J. Parallel Distrib. Comput.*, 48 (1998), 96–129.

[11] C.E. Kees, C.T. Miller, E.W. Jenkins, and C.T. Kelley. Versatile multilevel Schwarz preconditioners for multiphase flow. Technical Report CRSC-TR01-32, Center for Research in Scientific Computation, North Carolina State University, 2001.

[12] C. Lasser and A. Toselli. An overlapping domain decomposition preconditioner for a class of discontinuous Galerkin approximations of advection-diffusion problems. *Math. Comp.*, 72 (2003), 1215–1238.

[13] C. Lasser and A. Toselli. Convergence of some two-level overlapping domain decomposition preconditioners with smoothed aggregation coarse spaces. In *Recent Developments in Domain Decomposition Methods*, L.F. Pavarino and A. Toselli, eds., pp. 95–117, 2002.

[14] P. Le Tallec. Domain decomposition methods in computational mechanics. *Comput. Mech. Adv.*, 1 (1994), 121–220.

[15] J. Mandel. Balancing domain decomposition. *Comm. Appl. Numer. Methods*, 9 (1993), 233–241.

[16] B.B. Mandelbrot. *The Fractal Geometry of Nature.* W.H. Freeman and Company, San Francisco, 1982.

[17] L. Paglieri, A. Scheinine, L. Formaggia, and A. Quarteroni. Parallel conjugate gradient with Schwarz preconditioner applied to fluid dynamics problems. In *Parallel Computational Fluid Dynamics. Algorithms and Results Using Advanced Computer*, P. Schiano et al., eds., pp. 21–30, 1997.

[18] A. Quarteroni and A. Valli. *Numerical Approximation of Partial Differential Equations.* Springer-Verlag, Berlin, 1994.

[19] A. Quarteroni and A. Valli. *Domain Decomposition Methods for Partial Differential Equations.* Oxford University Press, Oxford, 1999.

[20] Y. Saad. *Iterative Methods for Sparse Linear Systems.* PWS, Boston, 1996.

[21] M. Sala. Analysis of two-level domain decomposition preconditioners based on aggregation. Submitted to *M2AN Math. Model. Numer. Anal.*

[22] M. Sala. *Domain Decomposition Preconditioners: Theoretical Properties, Application to the Compressible Euler Equations, Parallel Aspects.* PhD Thesis, École Polytechnique Fédérale de Lausanne, Switzerland, 2003.

[23] M. Sala and L. Formaggia. Parallel Schur and Schwarz based preconditioners and agglomeration coarse corrections for CFD problems. Technical Report 15, DMA-EPFL, Lausanne, 2001.

[24] M. Sala and L. Formaggia. Algebraic coarse grid operators for domain decomposition based preconditioners. In *Parallel Computational Fluid Dynamics – Practice and Theory*, P. Wilders et al., eds., pp. 119–126., Elsevier, The Netherlands, 2002.

[25] B.F. Smith, P. Bjorstad, and W.D. Gropp. *Domain Decomposition. Parallel Multilevel Methods for Elliptic Partial Differential Equations.* Cambridge University Press, Cambrige, 1996.

[26] H. Van der Vorst and C. Vuik. GMRESR: a family of nested GMRES methods. *Numer. Linear Algebra Appl.*, 1 (1994), 369–386.

[27] J. Xu and J. Zou. Some nonoverlapping domain decomposition methods. *SIAM Rev.*, 40 (1998), 857–914.