



MOX–Report No. 33/2010

**Multilevel Schwarz Methods for Elliptic Partial  
Differential Equations**

GIOVANNI MIGLIORATI, ALFIO QUARTERONI

MOX, Dipartimento di Matematica “F. Brioschi”  
Politecnico di Milano, Via Bonardi 9 - 20133 Milano (Italy)

[mox@mate.polimi.it](mailto:mox@mate.polimi.it)

<http://mox.polimi.it>



# Multilevel Schwarz Methods for Elliptic Partial Differential Equations

Giovanni Migliorati<sup>‡</sup>, Alfio Quarteroni<sup>†‡</sup>

October 18, 2010

<sup>‡</sup> MOX - Dipartimento di Matematica "Francesco Brioschi", Politecnico di Milano,  
via Bonardi 9, 20133 Milano, Italy

`giovanni.migliorati@mail.polimi.it`

<sup>†</sup> SB-MATHICSE-CMCS, Ecole Polytechnique Fédérale de Lausanne,  
1015 Lausanne, Switzerland

`alfio.quarteroni@epfl.ch`

**Keywords:** elliptic equations, finite element method, domain decomposition, overlapping Schwarz, multilevel preconditioners, aggregative or interpolative coarse level

**AMS Subject Classification:** 65F08, 65N55

## Abstract

We investigate multilevel Schwarz domain decomposition preconditioners, to efficiently solve linear systems arising from numerical discretizations of elliptic Partial Differential Equations by the finite element method. In our analysis we deal with unstructured mesh partitions and with subdomain boundaries resulting from using the mesh partitioner. We start from two-level preconditioners with either aggregative or interpolative coarse level components, then we focus on a strategy to increase the number of levels. For all preconditioners, we consider the additive residual update and its multiplicative variants within and between levels. Moreover, we compare the preconditioners behaviour, regarding scalability and rate of convergence. Numerical results are provided for elliptic boundary-value problems, including a convection-diffusion problem when suitable stabilization becomes necessary.

## 1 Introduction

Domain decomposition (DD) preconditioners are essential tools toward efficient parallel implementation of finite dimensional discretization of partial differential equations (PDEs). For a general presentation, see the monographs [BS92], [QV99] and [TW05].

Ideal preconditioners are optimal and scalable. Optimality is achieved when the condition number of the preconditioned matrix is independent of  $h$  (the discretization grid-size). Scalability occurs when the same condition number is independent of the number of subdomains. A necessary condition for scalability is that the DD preconditioner include a coarse-level mechanism.

Two-level DD preconditioners are very commonly used in the DD community to numerically solve elliptic PDEs since two decades. Multilevel (featuring more than two levels) preconditioners have been used more seldom, see *e.g.* [CSZ96].

In this paper we address and investigate multilevel DD preconditioners of Schwarz type. We start from two-level preconditioners with either aggregative or interpolative coarse level components, then we focus on a strategy to increase the number of levels.

Quite recently, analysis of domain decomposition methods applied on partitions featuring low regularity has been carried out [DKW08], [KRW08]. More precisely subdomain partitions with nonuniformly Lipschitz continuous boundaries were considered. This new framework is interesting because these partitions are invariably generated by common mesh partitioners (like, *e.g.* *Metis* [KK98]).

In [DKW08] the behaviour of a nonstandard additive two-level Schwarz preconditioner is investigated, for partitions where subdomains need only to be domains of John type (see Def.2.4 in [DKW08]). In this paper we focus on uniformly Lipschitz continuous boundaries. However, the number of patches required to cover the boundary of the region, in each of which the boundary is the graph of a Lipschitz continuous function, still depends on the finite element mesh size. The same kind of partitionings are employed in [DKW08, Figure 5.5], but on structured meshes.

We analyze all combinations of two-level preconditioners, with either additive or multiplicative link among subdomains in the first level, or on the interface between levels. In the literature (see *e.g.* [BS92]) these preconditioners are classified as two-level, pre-hybrid and post-hybrid. The coarse level can be the usual interpolative preconditioner or a simple aggregative one (without smoothing). Furthermore, we investigate how coarse parameters affect the speed of convergence. We show scalability for all preconditioner variants, when the first mesh level is refined and the number of subdomains increases properly. Numerical results indicate that classical upper and lower convergence bounds proposed in [Cai94], [Bre00] still hold in this general framework too.

Next we consider multilevel preconditioners. We restrict our analysis to nested meshes and additive interfaces among levels. In a first instance, levels are added from above to reach a five-level preconditioner, and a suitable choice of partitioning and spacing in each level is given. Afterwards, we focus on different alternative strategies to add levels.

In [CZ96] a convergence estimate is proposed, based on number of levels and their spacing. We show that, when adding from above new coarser levels, the number of iterations grows and that estimate is satisfied. Numerical results on the multilevel behaviour, with regular partitioning were pointed out

in [Zha92]. On the other hand, we can add new intermediate levels among the older ones. Here the number of iterations decreases, because the preconditioner must be faster at least, to justify its larger assembling cost. Apparently, this may seem to contradict previous statements. We show later that indeed there is no contradiction, since when adding new levels from above we are changing partitioning and size of local problems.

Then, the behaviour of multilevel preconditioners is studied for a non-symmetric convection-diffusion problem, and when using incomplete factorizations. The last numerical test that we propose addresses a problem in a domain with a complex shape, in which adapted meshes are employed.

## 2 Multilevel Schwarz preconditioner theory

Let us consider an elliptic operator  $L$ , provided with suitable boundary conditions. The strong and weak formulation of the overlapping Schwarz method were analyzed in [Lio88], when applied to the standard Poisson problem with Dirichlet conditions. For a more general kind of operator  $L$  the analysis in the differential framework is harder to achieve. Here we introduce the Schwarz method starting from the discrete problem.

The weak formulation in the abstract functional space  $V$  for the boundary-value problem associated to the operator  $L$  is:

$$\text{find } u \in V \quad : \quad a(u, v) = F(v) \quad \forall v \in V,$$

where  $a : V \times V \rightarrow \mathbb{R}$  is a bilinear form associated with  $L$ , while  $F \in V'$  is a functional that depends on problem's data. Approximating this problem by the usual *Galerkin method*, gives the linear system

$$Au = f, \tag{2.1}$$

where  $A$  is the stiffness matrix associated to the bilinear form  $a(\cdot, \cdot)$  and  $f$  the vector associated to the functional  $F(\cdot)$  (see [QV99]). Now we can partition the mesh underlying problem (2.1) in smaller local meshes. This operation corresponds to partition at the differential level the global domain in local subdomains. We call local problem the original (global) problem appropriately discretized on a subdomain. Each local problem can be described by a local projector acting on global problem (2.1). A detailed description of this procedure can be found in [QV94], [TW05].

On the right of Figure 1 we display square domain partitioned in 16 local subdomains. The mesh is partitioned in smaller local meshes, partially overlapped. The elements of the mesh that belong to the overlap are marked in red. We consider only minimal overlaps, *i.e.* neighboring local meshes are extended by just one strip of elements. Further details about partitioning will be provided at a later stage.

We adopt the following notation for multilevel Schwarz methods. See [QV94], [TW05] for a comprehensive description of the Schwarz method and well-posedness issues.

Denote with  $l$ ,  $1 \leq l \leq L$ , the index that identifies the level. Then, in the  $l$ th level, we denote by:

- $h_l$  the finite element gridsize parameter;
- $H_l$  the characteristic subdomain dimension;
- $M_l$  the number of subdomains;
- $\delta_l$  the fraction of the overlap.

We sort the levels in such a way that

$$h_1 < \dots < h_L.$$

Then, for brevity we may refer to the first and last level as the finest and coarsest, respectively. The coarsest level ( $l = L$ ) is not partitioned ( $M_L = 1$ ), consequently  $H_L$  is equal to the (non partitioned) original domain, and  $\delta_L$  is not defined. Figure 1 shows how this notation applies to describe an instance of a three-level decomposition.

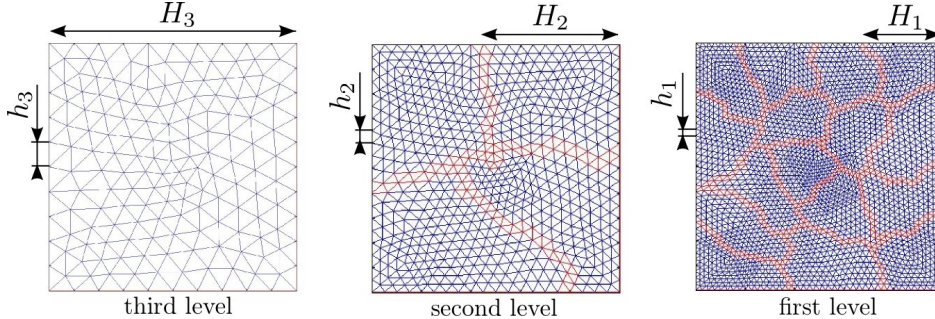


Figure 1: Graphical example: multilevel notation for a three-level preconditioner.

In this example we have:

$$H_2 \simeq \frac{H_3}{2}, \quad h_2 \simeq \frac{h_3}{2}, \quad H_1 \simeq \frac{H_3}{4}, \quad h_1 \simeq \frac{h_3}{4}.$$

We can associate gridsize parameters to the minimum or maximum element edge (in each level). Regardless of the choice, we see from Tables 1 and Table 10 that gridsize parameters halves exactly when refining. Conversely, we are not able to guarantee that subdomain characteristic dimension halves so precisely, because subdomains can be very irregularly shaped. However, as indicated by

the following figures, even when elongated shapes are generated, we may assume that

$$H_l \propto (M_l)^{-1/2}, \quad (2.2)$$

for the generic  $l$ th level. The exponent would be  $-1/d$  when working in a  $d$ -dimensional space.

From now on  $i$  will denote a generic index on subdomains. So in the  $l$ th level  $i$  ranges from 1 to  $M_l$ .

Let us define the restriction (or extension, when transposed) operators  $R_{li}$ . With  $l = 1$  the operator  $R_{1i}$  restricts the global residual on the  $i$ th subdomain of the first level. For  $2 \leq l \leq L$  an intralevel operator between first level mesh and  $l$ th level mesh is also accomplished, and entries for  $R_{li}$  are

$$R_{li}(r, c) = \varphi_r(x_c), \quad (2.3)$$

whereas  $\{\varphi_r\}_r$  are finite element basis functions whose support belong to the  $i$ th subdomain of the  $l$ th level, and  $\{x_c\}_c$  are the nodes of first level mesh.

Only when  $l = L = 2$  a particular kind of operator can be used, with unitary constant basis functions over each subdomain. We name it as aggregating  $R_{agg}$  and its entries are

$$R_{agg}(r, c) = \begin{cases} 1, & \text{if the node } x_c \text{ belongs to the } r\text{th subdomain,} \\ 0, & \text{otherwise.} \end{cases} \quad (2.4)$$

Restricting and extending the initial stiffness matrix  $A$  we obtain local matrices:

$$A_{li} = R_{li} A R_{li}^T, \quad l = 1 \div L, i = 1 \div M_l. \quad (2.5)$$

When using  $R_{agg}$  we denote the local matrix with  $A_{agg}$ .

We introduce the additive projector  $\mathcal{P}_{add}^l$  on the  $l$ th level:

$$\mathcal{P}_{add}^l = \sum_{i=1}^{M_l} R_{li}^T A_{li}^{-1} R_{li}, \quad 1 \leq l \leq L - 1. \quad (2.6)$$

This projector acts as in Algorithm 1. It takes a global residual and returns the preconditioned (global) residual.

The multiplicative variant  $\mathcal{P}_{mul}^l$  acts as in Algorithm 2, where a colouring among subdomains is required. Here  $C_j$  is the set of subdomains featuring the  $j$ th color, and  $L_U$  is the upper limit on the number of colours. In the multiplicative variant, operations in the *for*-cycle are serial.

Projectors  $\mathcal{P}_{add}^l$  and  $\mathcal{P}_{mul}^l$  are defined for  $l = 1 \div L - 1$ . For the  $L$ th level ( $L \geq 2$ ) we need a monolithic aggregating projector

$$\mathcal{P}_{agg} = R_{agg}^T A_{agg}^{-1} R_{agg}, \quad (2.7)$$

as well as the interpolating projector

$$\mathcal{P}_{int} = R_{L1}^T A_{L1}^{-1} R_{L1}. \quad (2.8)$$

---

**Algorithm 1** The action of additive projector  $\mathcal{P}_{add}^l$  on the global residual.

---

**INPUT** residual  $\mathbf{v}$

**OUTPUT** preconditioned residual  $\mathbf{z}$

$$\mathbf{z} \leftarrow \sum_{i=1}^M R_{li}^T A_{li}^{-1} R_{li} \mathbf{v}$$


---

---

**Algorithm 2** The action of multiplicative projector  $\mathcal{P}_{mul}^l$  on the global residual.

---

**INPUT** residual  $\mathbf{v}$

**OUTPUT** preconditioned residual  $\mathbf{z}$

$$\begin{aligned} \mathbf{z} &\leftarrow \sum_{i \in C_1} R_{li}^T A_{li}^{-1} R_{li} \mathbf{v} \\ \text{for } 2 \leq j \leq L_U \text{ do} \\ &\quad \mathbf{z} \leftarrow \mathbf{z} + \sum_{i \in C_j} R_{li}^T A_{li}^{-1} R_{li} (\mathbf{v} - A\mathbf{z}) \\ \text{end for} \end{aligned}$$


---

Now we are ready to define generic Schwarz multilevel preconditioners. Say  $\mathcal{P}_l$  the generic projector on the  $l$ th level, with  $1 \leq l \leq L$ . For instance, it might be either  $\mathcal{P}_l = \mathcal{P}_{add}^l$  or  $\mathcal{P}_l = \mathcal{P}_{mul}^l$  for  $l = 1 \div L - 1$ , and either  $\mathcal{P}_l = \mathcal{P}_{agg}$  or  $\mathcal{P}_l = \mathcal{P}_{int}$  for  $l = L$ .

We denote with  $P_L$  the generic  $L$ -level preconditioner

$$P_L = \left( \sum_{l=1}^L \mathcal{P}_l \right)^{-1}. \quad (2.9)$$

Now, depending on the choice of projectors  $\{\mathcal{P}_l\}_{l=1}^L$ , we can build several kind of multilevel preconditioners. The simplest is the one-level additive preconditioner  $P_1$ , which corresponds to taking  $L = 1$  and  $\mathcal{P}_1 = \mathcal{P}_{add}^1$  in (2.9). To obtain the one-level multiplicative preconditioner just take  $\mathcal{P}_1 = \mathcal{P}_{mul}^1$ .

For preconditioners composed of more than one level we can choose either a parallel or a serial interface between levels. The serial interface differs from the parallel for accomplishing a residual update interposed between level projectors. Only for the case of two-level preconditioners ( $L = 2$ ) we analyze all possible combinations of first level projectors, second level projectors and interfaces, as shown in Figure 2.

We denote with  $P_2$  the two-level preconditioner with a parallel interface between levels, and with  $P_{2hy}^{pre}, P_{2hy}^{post}$  hybrid variants obtained with a serial interface. In pre-hybrid variant the first level projector is applied, then the residual is updated and lastly the second level projector is applied. In post-hybrid variant the order is reversed. Correct orders in which first and second level projectors are applied are explicitly highlighted in Algorithms 3, 4, 5.

For preconditioners with more than two levels a serial interface becomes unattractive, so for  $L \geq 3$  we build the multilevel preconditioner as shown in Figure 3. We can choose projectors  $\mathcal{P}_l = \mathcal{P}_{add}^l$  (for  $1 \leq l \leq L - 1$ ) and  $\mathcal{P}_L = \mathcal{P}_{int}$  to obtain a multilevel preconditioner  $P_L$  that is additive between and within levels (Algorithm 6). Otherwise, choosing  $\mathcal{P}_l = \mathcal{P}_{mul}^l$  (for  $1 \leq l \leq L - 1$ ) and  $\mathcal{P}_L = \mathcal{P}_{int}$ , we obtain  $P_L$  that is additive between levels but multiplicative within each level (Algorithm 7).



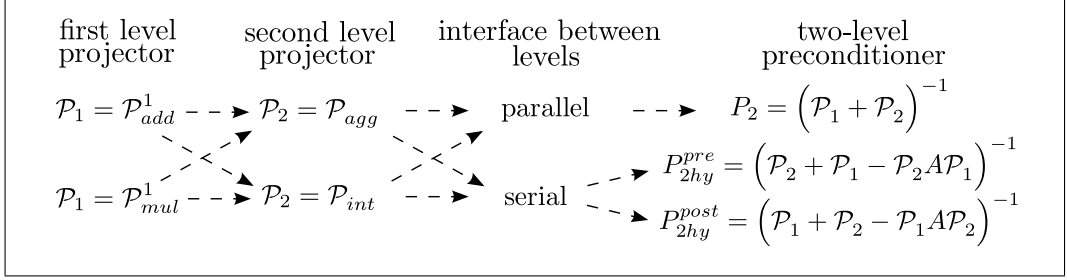


Figure 2: Construction of two-level preconditioners. Choose the projector  $\mathcal{P}_1$  for the first level (either additive or multiplicative), then choose the projector  $\mathcal{P}_2$  for the second level (aggregative or interpolative), and finally choose the interface between levels (parallel or serial). Choosing parallel interface yields the two-level preconditioners  $P_2$ . The serial interface gives hybrid variants: pre-hybrid  $P_{2hy}^{pre}$  when applying first level projector followed by second level projector, post-hybrid  $P_{2hy}^{post}$  otherwise.

---

**Algorithm 3** Two-level Schwarz preconditioner  $P_2$  with projectors  $\mathcal{P}_1, \mathcal{P}_2$  linked by parallel interface.

---

**INPUT** residual  $\mathbf{v}$

**OUTPUT** preconditioned residual  $\mathbf{z}$

$$\begin{aligned} \mathbf{w} &\leftarrow \mathcal{P}_1 \mathbf{v} \\ \mathbf{z} &\leftarrow \mathbf{w} + \mathcal{P}_2 \mathbf{v} \end{aligned}$$


---

---

**Algorithm 4** Pre-hybrid Schwarz preconditioner  $P_{2hy}^{pre}$ . Apply the first level projector  $\mathcal{P}_1$ , then update residual and apply second level projector  $\mathcal{P}_2$ .

---

**INPUT** residual  $\mathbf{v}$

**OUTPUT** preconditioned residual  $\mathbf{z}$

$$\begin{aligned} \mathbf{w} &\leftarrow \mathcal{P}_1 \mathbf{v} \\ \mathbf{z} &\leftarrow \mathbf{w} + \mathcal{P}_2 (\mathbf{v} - A \mathbf{w}) \end{aligned}$$


---

---

**Algorithm 5** Post-hybrid Schwarz preconditioner  $P_{2hy}^{post}$ . Apply the second level projector  $\mathcal{P}_2$ , then update residual and apply first level projector  $\mathcal{P}_1$ .

---

**INPUT** residual  $\mathbf{v}$

**OUTPUT** preconditioned residual  $\mathbf{z}$

$$\begin{aligned} \mathbf{w} &\leftarrow \mathcal{P}_2 \mathbf{v} \\ \mathbf{z} &\leftarrow \mathbf{w} + \mathcal{P}_1 (\mathbf{v} - A \mathbf{w}) \end{aligned}$$


---

## 2.1 Convergence estimates

We recall now classical estimates that hold for some preconditioners. We denote with the lower or upper case letter  $C$  a generic positive constant independent of other quantities figuring in each estimate. A preconditioner is said to be symmetric if the starting problem is symmetric. Nonsymmetric preconditioners can be possibly symmetrized [HV97]. All the following estimates hold for the usual

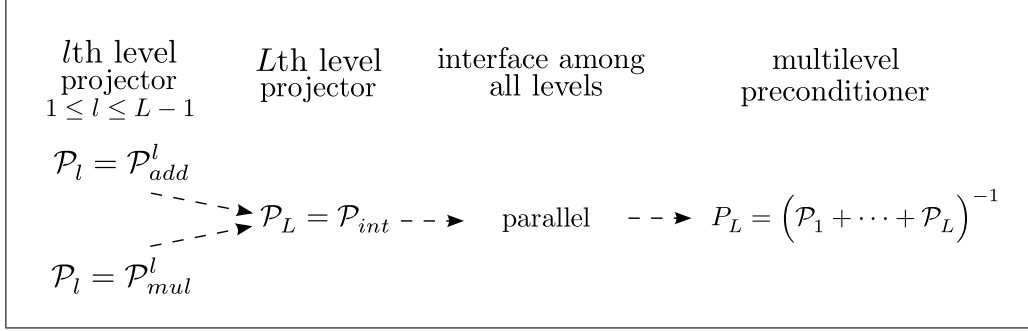


Figure 3: Multilevel preconditioners. Choose projectors  $\mathcal{P}_l$  on levels  $l = 1 \div L - 1$ . These projectors can be additive or multiplicative in each level, but all of the same type. The projector  $\mathcal{P}_L$  on the  $L$ th level can only be interpolative. The interface among all levels is always parallel. So we obtain a multilevel preconditioner that is always additive among levels, whereas it can be either additive or multiplicative in each level.

---

**Algorithm 6** Multilevel Schwarz preconditioner  $P_L$  with additive projectors  $\mathcal{P}_l = \mathcal{P}_{add}^l$ ,  $l = 1 \div L - 1$  and  $\mathcal{P}_L = \mathcal{P}_{int}$ .

---

**INPUT** residual  $\mathbf{v}$

**OUTPUT** preconditioned residual  $\mathbf{z}$

```

 $\mathbf{z} \leftarrow R_{L1}^T A_{L1}^{-1} R_{L1} \mathbf{v}$ 
for  $1 \leq l \leq L-1$  do
   $\mathbf{z} \leftarrow \mathbf{z} + \sum_{i=1}^{M(l)} R_{li}^T A_{li}^{-1} R_{li} \mathbf{v}$ 
end for

```

---



---

**Algorithm 7** Multilevel Schwarz preconditioner  $P_L$  with multiplicative projectors  $\mathcal{P}_l = \mathcal{P}_{mul}^l$ ,  $l = 1 \div L - 1$  and  $\mathcal{P}_L = \mathcal{P}_{int}$ .

---

**INPUT** residual  $\mathbf{v}$

**OUTPUT** preconditioned residual  $\mathbf{z}$

```

 $\mathbf{z} \leftarrow R_{L1}^T A_{L1}^{-1} R_{L1} \mathbf{v}$ 
for  $1 \leq l \leq L-1$  do
   $\mathbf{z}_l \leftarrow \sum_{i \in C_1} R_{li}^T A_{li}^{-1} R_{li} \mathbf{v}$ 
  for  $2 \leq j \leq L_U$  do
     $\mathbf{z}_l \leftarrow \mathbf{z}_l + \sum_{i \in C_j} R_{li}^T A_{li}^{-1} R_{li} (\mathbf{v} - A \mathbf{z}_l)$ 
  end for
   $\mathbf{z} \leftarrow \mathbf{z} + \mathbf{z}_l$ 
end for

```

---

symmetric elliptic problem with Dirichlet boundary conditions. Additional hypotheses are given, whenever required.

For the one-level additive (symmetric) preconditioner  $P_1$  given by  $\mathcal{P}_1 = \mathcal{P}_{add}^1$  we have [QV99]

$$K(P_1^{-1}A) \leq C \frac{1}{\delta_1^2 H_1^2}. \quad (2.10)$$

The dependence on  $1/H_1^2$  was pointed out in [Wid88] and "holds for any domain decomposition method for which the interaction is only through next neighboring subdomains, cf.[DW90]". The estimate is sharp, then to improve the convergence speed additional shrewdness is required.

In the two-level additive (symmetric) case, *i.e.*  $P_{2L}$  with  $\mathcal{P}_1 = \mathcal{P}_{add}^1$  and

$\mathcal{P}_2 = \mathcal{P}_{int}$  the following estimate holds [Cai94]

$$K(P_2^{-1}A) \leq C \min \left( 1 + \frac{h_2^2}{\delta_1^2}, 1 + \frac{H_1}{\delta_1} + \frac{h_2}{\delta_1} \frac{h_2}{H_1} \right). \quad (2.11)$$

We remark that the first term does not depend on  $H_1$ , allowing to employ arbitrary shape domains. The second term reduces to  $H_1/\delta_1$  when  $H_1 \sim h_2$ , so its importance increases for small  $\delta_1$ . The importance of a small overlap has been discussed in [DW94], [Wid92]. We focus on the minimal overlap choice, *i.e.* subdomains in the  $l$ th level, with  $1 \leq l \leq L-1$ , are overlapped only by an element strip, corresponding to take  $\delta_l = h_l$ . In the minimal overlap case a lower bound holds too [Bre00], and combining this with (2.11) we obtain

$$c \left( \frac{H_1}{h_1} \right) \leq K(P_2^{-1}A) \leq C \left( 1 + \frac{H_1}{h_1} \right). \quad (2.12)$$

The upper bound in (2.12) is reported in [DW94], [Wid92], and both the upper and lower bounds are recalled in [Bre00].

For multilevel additive preconditioner  $P_L$ , with  $\mathcal{P}_l = \mathcal{P}_{add}^l$  for  $l = 1 \div L-1$  and  $\mathcal{P}_L = \mathcal{P}_{int}$ , the following convergence estimate holds [CZ96]

$$K(P_L^{-1}A) \leq \rho^2 L^2, \quad (2.13)$$

with

$$\rho = \max_{1 \leq l \leq L-1} \frac{h_l + h_{l+1}}{\delta_l}. \quad (2.14)$$

The condition number worsens at most quadratically when the number of levels  $L$  increases. When convex domains are employed the dependence on  $L$  becomes linear and this upper estimate is sharp [DW90].

**Proposition 1** *In the case of minimal overlap, *i.e.*  $\delta_l = h_l$  ( $1 \leq l \leq L-1$ ),  $\rho$  is minimized if the ratio  $h_{l+1}/h_l$  is kept fixed among levels.*

**Proof.** We show it for a three-level preconditioner. Let us denote with  $h_1$  and  $h_3$  two f.e.gridsizes such that  $h_1 < h_3$ . We look for  $h_2$  that satisfies

$$h_2 = \arg \min_{h_1 < x < h_3} \rho(x). \quad (2.15)$$

Rewriting (2.14) with the unknown  $h_2$  renamed to  $x$ , and dropping constant terms gives

$$\rho(x) = \max \left( \frac{x}{h_1}, \frac{h_3}{x} \right). \quad (2.16)$$

By equating the two terms we obtain:

$$\frac{x}{h_1} = \frac{h_3}{x} \quad \Rightarrow \quad x = \sqrt{h_1 h_3}.$$

□

In the end,  $h_2$  is the geometric average between  $h_1$  and  $h_3$ . Figure 4 resumes graphically the whole proof.

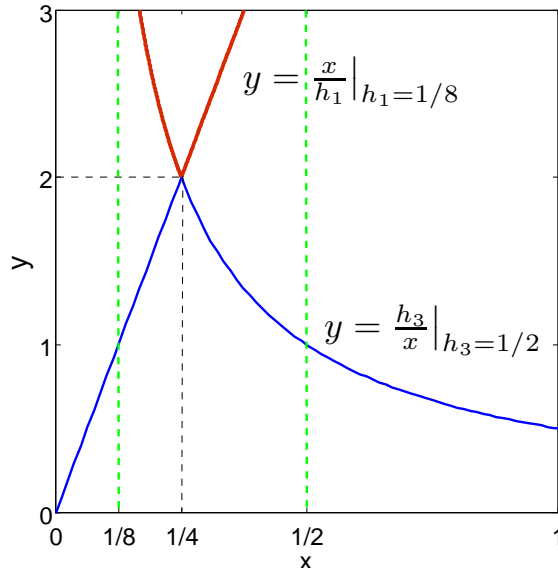


Figure 4: Geometrical solution for the problem (2.15), given  $h_3 = 1/2$  and  $h_1 = 1/8$ . Admissible values for  $x$  are between green lines. Blue lines plot terms in (2.16) and their maximum  $\rho$  is highlighted in red. Then take the minimum on  $\rho$  to obtain  $x = 1/4$ .

### 3 Two-level preconditioners for a symmetric problem

Let us start with a Poisson problem in  $\Omega = (0, 1) \times (0, 1)$ :

$$\begin{cases} -\Delta u = f, & \text{in } \Omega, \\ u = g, & \text{on } \partial\Omega, \end{cases} \quad (3.1)$$

with  $f$  and  $g$  chosen in such a way that the exact solution be  $u(x, y) = -x \exp(y)$ . Given an initial mesh in  $\Omega$ , we refine and create local partitions with *Metis*. Then we discretize it by piecewise linear finite elements, see *e.g.* [Qua09]. We associate every regular grid refinement with a color (Table 1).

color	gridsize parameter $h_1$	nb. of elements	nb. of nodes	maximum elem. side	minimum elem. side
red	$h$	5504	2833	0.0265	0.0133
blue	$h/2$	22016	11169	0.0133	0.0067
green	$h/4$	88064	44353	0.0066	0.0033

Table 1: Progressive refinement for first level meshes. Given  $h = 0.0265$ , we start from the initial mesh (red) with f.e. gridsize  $h_1 = h$ , and refining it we obtain meshes with f.e. gridsize equal to  $h_1 = h/2$  (blue) and  $h_1 = h/4$  (green).

In the whole section we choose a Cholesky complete factorization, see *e.g.*

[QSS07], to solve the linear algebraic system associated with the coarse problem as well as each local problem.

In the following figures each star refers to a simulation, and the continuous line highlights the upper part of their convex envelope.

If the problem is symmetric (as in the current case) we can solve the global problem by CG iterations, preconditioned with one of the (symmetric) Schwarz preconditioners previously introduced. For all numerical tests, those in following sections too, in the iterative method we use a stopping criterion based on relative residual error, with a tolerance fixed to  $tol = 1e - 6$ .

To start with, let us solve problem (3.1) with the one-level additive preconditioner  $P_1$  and  $\mathcal{P}_1 = \mathcal{P}_{add}^1$ . As shown in Figure 5 the number of required iterations grows when subdomains number  $M_1$  grows and  $h_1$  decreases, highlighting the need for a second level to achieve scalability. We observe that the number of iterations increases regularly, even if partitionings related to consecutive number of subdomains may totally differ one from the other.

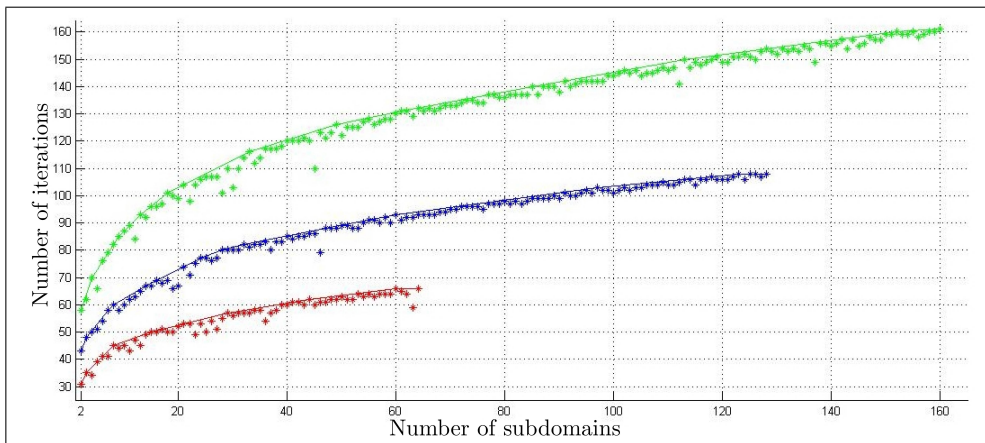


Figure 5: Results for one-level additive preconditioner  $P_1$  with  $\mathcal{P}_1 = \mathcal{P}_{add}^1$ , for problem (3.1). Colors refer to the mesh in Table 1.

In Table 2, which contains data picked from Figure 5, we see that the estimate (2.10) is sharply satisfied, *i.e.* if the product  $h_1 H_1$  is kept constant then the number of iterations is constant too.

Let us mention two other features about these one-level results. Considering that the number of CG iterations, say  $N_{CG}$ , is proportional to the square root of the condition number of the preconditioned matrix, the estimate (2.10) yields

$$N_{CG} \leq \frac{C}{h_1 H_1}, \quad (3.2)$$

for a suitable constant  $C$ . This is confirmed from the results of Table 2. In addition, the role of the overlap has been analyzed too: we obtain the same results of Figure 5 with the red mesh by taking the blue mesh but with a doubled overlap amplitude.

		$h_1$		
		$h$	$h/2$	$h/4$
$H_1$	$H$	34	50	70
	$H/2$	50	69	96
	$H/4$	66	94	132

Table 2: Number of CG iterations extracted from Figure 5. Values for  $h_1$  correspond to mesh colors. Values for  $H_1$  are calculated by (2.2) taking the number of subdomains  $M_1$  from Figure 5; here we took  $M_1$  equal to 4, 16, 64, so we have  $H = 1/2$  and  $H_1$  is equal to  $H, H/2, H/4$ . The number of iterations is constant along secondary diagonals, corresponding to keep constant the product  $h_1 H_1$ .

From Figure 6 we see that a second aggregative level ( $P_2$  with  $\mathcal{P}_1 = \mathcal{P}_{add}^1$  and  $\mathcal{P}_2 = \mathcal{P}_{agg}$ ) is enough to start taking advantage of the subdomain number. For low  $M_1$  the preconditioner efficacy decays, due to its artificial nature.

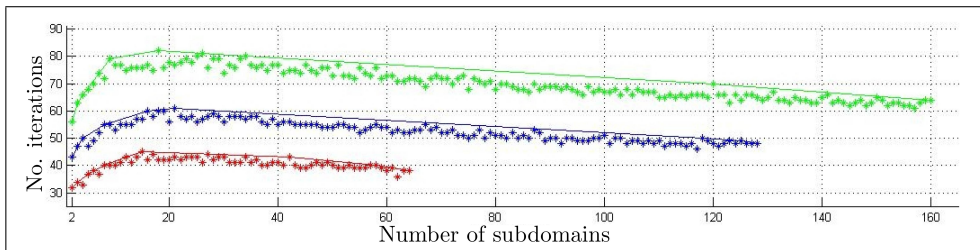


Figure 6: Results for two-level additive aggregative preconditioner  $P_2$  with  $\mathcal{P}_1 = \mathcal{P}_{add}^1$  and  $\mathcal{P}_2 = \mathcal{P}_{agg}$ , for problem (3.1). Colors refer to the mesh in Table 1.

The interpolative second level exhibits a better efficiency than the aggregative one. Let us consider the two-level preconditioner  $P_2$  with  $\mathcal{P}_1 = \mathcal{P}_{add}^1$  and  $\mathcal{P}_2 = \mathcal{P}_{int}$ . We choose a fixed coarse mesh composed of 344 elements, with maximum diameter  $h_2 = 0.1061$ . In this way we analyze only the effect due to  $\delta_1 = h_1$  in the upper bound of (2.12). The number of iterations increases when  $h_1$  is halved (Figure 7). The number of subdomains changes from 2 to 160, consequently their characteristic dimension decreases from 0.707 to 0.079.

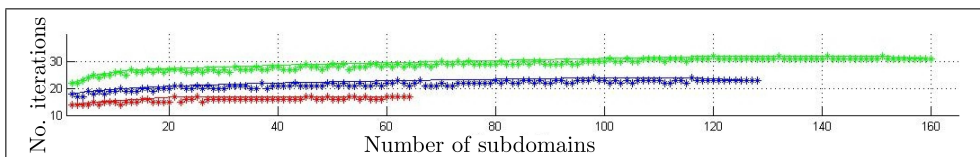


Figure 7: Results for two-level additive interpolative preconditioner  $P_2$  with  $\mathcal{P}_1 = \mathcal{P}_{add}^1$  and  $\mathcal{P}_2 = \mathcal{P}_{int}$ , with a fixed coarse mesh for problem (3.1). Colors refer to the mesh in Table 1.

In another test the coarse mesh is changed, *i.e.* tuning the f.e.gridsize  $h_2$ . Meshes for first level are the same as those in Table 1 with  $h_1$  equal to  $h, h/2, h/4$ . Coarse meshes (Table 3) are chosen so that the maximum element diameter approximates the characteristic dimension of subdomains (Table 4). In Figures 8 and 9 some meshes with the coarse level (blue) are shown. In Table 5 we see that the number of iterations remains constant if the ratio  $h_2/h_1$  is kept constant.

gridsize parameter $h_2$	no. coarse elements	maximum elem. side
$h_C$	86	0.2122
$h_C/2$	344	0.1061
$h_C/4$	1376	0.0531
$h_C/8$	5504	0.0265

Table 3: Progressive refinement for second level meshes. Given  $h_C = 0.2122$ , we start from the initial coarse mesh with f.e. gridsize  $h_2 = h_C$ , and refining it we obtain coarse meshes with f.e.gridsize equal to  $h_2 = h_C/2$ ,  $h_2 = h_C/4$  and  $h_2 = h_C/8$ .

no. subdom. $M_1$	$H_1 = \sqrt{1/M_1}$	ratios among no. subdomains
16	0.250	$M = 16$
64	0.125	$4 \cdot M$
256	0.0625	$16 \cdot M$
1024	0.0312	$64 \cdot M$

Table 4: First level mesh partitions. In the first partition there are  $M_1 = 16$  subdomains. In the following partitions the number of subdomains is multiplied by 4, so  $M_1$  takes 64, 256, 1024 values. For each partition the characteristic dimension  $H_1$  related to subdomains is given.

	$M_1 = 16$ $h_2 = h_C$	$M_1 = 64$ $h_2 = h_C/2$	$M_1 = 256$ $h_2 = h_C/4$	$M_1 = 1024$ $h_2 = h_C/8$
$h_1 = h$	21	16	-	-
$h_1 = h/2$	27	21	17	-
$h_1 = h/4$	36	29	21	17

Table 5: Number of CG iterations for the two-level additive interpolative preconditioner with coarse meshes shown in Table 3, for problem (3.1). Labels on the left ( $h_1$ ) refer to 1st level f.e. gridsize. Labels on top ( $M_1$  and  $h_2$ ) refer to the number of subdomains in 1st level and 2nd level f.e. gridsize. The number of iterations is constant along principal diagonals; this means that the algorithm is weakly scalable. Quantities  $h$  and  $h_C$  refer to meshes in Table 1 and Table 3.

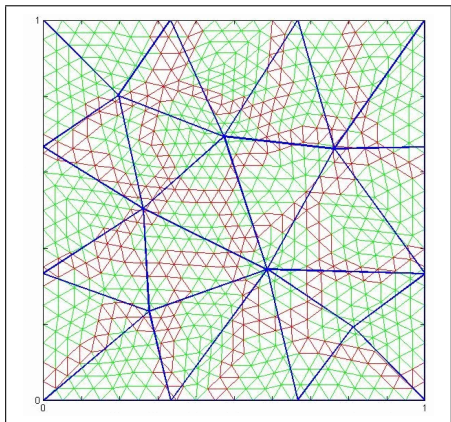


Figure 8: A coarse mesh with 24 elements (in blue) superimposed on a fine mesh (in green) with 1376 elements partitioned into 16 subdomains. Red elements are those belonging to the overlap.

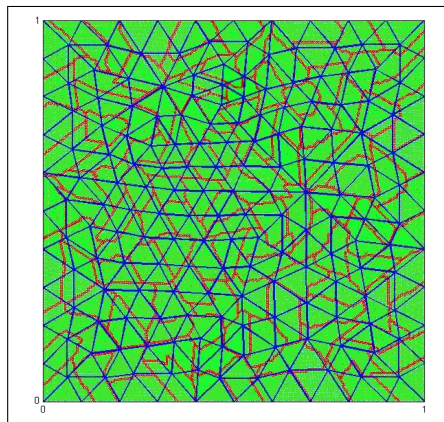


Figure 9: A coarse mesh with 344 elements (in blue) superimposed on a fine mesh (in green) with 88064 elements partitioned into 128 subdomains. Red elements are those belonging to the overlap.

## 4 Two-level preconditioners for a nonsymmetric problem

Let us consider the following problem in  $\Omega = (0, 1) \times (0, 1)$ :

$$\begin{cases} -\Delta u + (1, 0) \cdot \nabla u = f, & \text{in } \Omega, \\ u = g, & \text{on } \partial\Omega, \end{cases} \quad (4.1)$$

with  $f$  and  $g$  chosen such that the exact solution is  $u(x, y) = x^2 + y^2$ . We solve the associated linear system by BiCGstab (because of lack of symmetry) with different kind of preconditioners. The local problems (including the coarse problem) are solved by the LU complete factorization. We use the same meshes displayed in Table 1. The coarse meshes are those from Table 3. In Table 6 results with an interpolative coarse level are shown. In all cases the number of iterations decreases when the number of subdomains increases. If we keep fixed the ratio  $h_2/h_1$  (on principal diagonals) the number of iterations is constant (again, this yields weak scalability). From simulations, there seem to be no advantages in using the multiplicative interface, when the first level is multiplicative. In Table 7 some results for aggregative second level are shown.



			$M_1 = 16$ $h_2 = h_C$	$M_1 = 64$ $h_2 = h_C/2$	$M_1 = 256$ $h_2 = h_C/4$	$M_1 = 1024$ $h_2 = h_C/8$
preconditioner	$P_2$ with $\mathcal{P}_1 = \mathcal{P}_{add}^1$ and $\mathcal{P}_2 = \mathcal{P}_{int}$	$h_1 = h$	13	11	-	-
		$h_1 = h/2$	18	14	10	-
		$h_1 = h/4$	23	17	13	10
	$P_{2hy}^{pre}$ with $\mathcal{P}_1 = \mathcal{P}_{add}^1$ and $\mathcal{P}_2 = \mathcal{P}_{int}$	$h_1 = h$	12	8	-	-
		$h_1 = h/2$	16	12	8	-
		$h_1 = h/4$	25	15	12	9
	$P_{2hy}^{post}$ with $\mathcal{P}_1 = \mathcal{P}_{add}^1$ and $\mathcal{P}_2 = \mathcal{P}_{int}$	$h_1 = h$	11	8	-	-
		$h_1 = h/2$	15	12	8	-
		$h_1 = h/4$	22	17	12	8
	$P_2$ with $\mathcal{P}_1 = \mathcal{P}_{mul}^1$ and $\mathcal{P}_2 = \mathcal{P}_{int}$	$h_1 = h$	7	5	-	-
		$h_1 = h/2$	9	6	-	-
		$h_1 = h/4$	13	10	-	-
$P_{2hy}^{pre}$ with $\mathcal{P}_1 = \mathcal{P}_{mul}^1$ and $\mathcal{P}_2 = \mathcal{P}_{int}$	$h_1 = h$	7	5	-	-	
	$h_1 = h/2$	9	6	-	-	
	$h_1 = h/4$	13	10	-	-	

Table 6: Number of BiCGstab iterations for several preconditioners with an interpolative second level (*i.e.*  $\mathcal{P}_2 = \mathcal{P}_{int}$ ). The interface between levels is either additive or multiplicative. The residual adjournment within the first level is either additive or multiplicative (*i.e.*  $\mathcal{P}_1 = \mathcal{P}_{add}^1$  or  $\mathcal{P}_1 = \mathcal{P}_{mul}^1$ ). Labels on the left refer to preconditioners. In each row we specify the 1st level f.e. gridsize ( $h_1$ ). Labels on top ( $M_1$  and  $h_2$ ) refer to the number of subdomains in 1st level and 2nd level f.e. gridsize. The number of iterations is constant along principal diagonals (yielding weak scalability). Quantities  $h$  and  $h_C$  refer to meshes in Table 1 and Table 3, respectively.

			$M_1 = 16$	$M_1 = 64$	$M_1 = 256$	$M_1 = 1024$
preconditioner	$P_2$ with $\mathcal{P}_1 = \mathcal{P}_{add}^1$ and $\mathcal{P}_2 = \mathcal{P}_{agg}$	$h_1 = h$	26	23	-	-
		$h_1 = h/2$	34	30	24	-
		$h_1 = h/4$	55	39	33	26
	$P_2$ with $\mathcal{P}_1 = \mathcal{P}_{mul}^1$ and $\mathcal{P}_2 = \mathcal{P}_{agg}$	$h_1 = h$	16	11	-	-
		$h_1 = h/2$	18	15	-	-
		$h_1 = h/4$	25	28	-	-

Table 7: Number of BiCGstab iterations for several preconditioners with an aggregative second level (*i.e.*  $\mathcal{P}_2 = \mathcal{P}_{agg}$ ). The interface between levels is additive. The residual adjournment within the first level is either additive or multiplicative (*i.e.*  $\mathcal{P}_1 = \mathcal{P}_{add}^1$  or  $\mathcal{P}_1 = \mathcal{P}_{mul}^1$ ). Labels on the left refer to preconditioners. In each row we specify the 1st level f.e. gridsize ( $h_1$ ). Labels on top ( $M_1$ ) refer to the number of subdomains in 1st level. Quantities  $h$  and  $h_C$  refer to meshes in Table 1 and Table 3, respectively.

## 5 Multilevel preconditioners for a symmetric problem

Let us consider again problem (3.1), which we now solve using several multilevel variants, that differ in terms of residual adjournments. We want to analyze preconditioner behaviours with respect to:

- number of levels;
- total degrees of freedom in the problem;
- additive versus multiplicative WITHIN levels (but always additive BETWEEN levels).

First, there are several ways to change the number of levels, *e.g.* to add coarser levels while the older are fixed, or to add properly new levels between older levels.

Here we choose the former strategy, *i.e.* to keep the first level (the finer) fixed and to add further levels by successive derefinement. In this way our levels are *nested*, and the last level in the preconditioner will depend on the number of total levels. In practice, we will start from the coarsest level and build further levels by refinements, because this is much easier to handle from the computational standpoint. Anyway, we will always obtain the finest level with the same discretization and the same partitioning.

In Table 8 a five-level preconditioner is shown. We leave values of  $h_i$  and  $H_i$  expressed in symbolic notation to highlight their ratios. Ratios between discretization parameters and characteristic dimensions of subdomains for consecutive domains are constant. Discretization parameters for each level are recoverable from Table 10, *e.g.* we obtain a first level with 39233 nodes (refinement label equal to 5) starting from the fourth level with maximum element side of length  $h = 0.0505$  (refinement label equal to 2). Values for  $H_i$  are calculated by (2.2) from the number of subdomains  $M_i$ . Here the initial mesh is the unitary square, so when  $M_L = 1$  we have  $H_L = H = 1$ . Below we refer to meshes in Table 10 by refinement labels. Figures 10, 11, 12, 13, 14, 15 show meshes with some refinement and partitions. Elements belonging to the overlap are marked in red.

To decrease the number of levels we observe that, with nested meshes, the set of admissible  $h$  is discrete (it is a geometric progression). Then, if we want to keep the ratio  $H_l/h_l$  fixed, we can only delete a given level, losing the uniform spacing  $h_{l+1}/h_l$  among levels. It would be wrong to delete the finest level too, because this would yield a dramatic change of the degrees of freedom in the preconditioner. In the end, we are not able to keep constant the ratio  $H_l/h_l$  among levels.

Following these ideas we obtain the four-level preconditioner outlined in Table 9.

level nb.	$h_l$	$H_l$	$M_l$
$l = 5$	$h$	$H$	1
$l = 4$	$h/2$	$H/2$	4
$l = 3$	$h/4$	$H/4$	16
$l = 2$	$h/8$	$H/8$	64
$l = 1$	$h/16$	$H/16$	256

Table 8: Five-level preconditioner: grid-size parameter, characteristic subdomain dimension, number of subdomains in each level.

level nb.	$h_l$	$H_l$	$M_l$
$l = 4$	$h/2$	$H$	1
$l = 3$	$h/4$	$H/2.5$	6
$l = 2$	$h/8$	$H/6$	36
$l = 1$	$h/16$	$H/16$	256

Table 9: Four-level preconditioner: grid-size parameter, characteristic subdomain dimension, number of subdomains in each level.

refinement lbl.	no. nodes	no. elements	min. elem. side	max elem. side
0	49	76	0.1093	0.2022
1	173	304	0.0547	0.1011
2	649	1216	0.0273	0.0505
3	2513	4864	0.0137	0.0253
4	9889	19456	0.0068	0.0126
5	39233	77284	0.0034	0.0063
6	156289	311296	0.0017	0.0032

Table 10: Mesh used to build multilevel preconditioners.

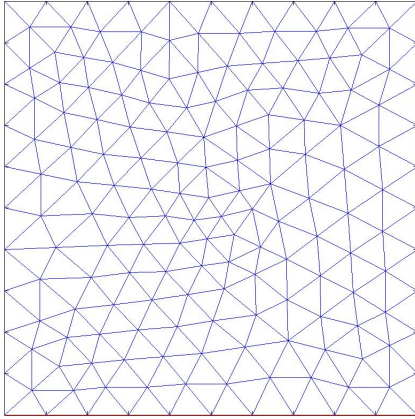


Figure 10: Mesh with a refinement label equal to 1. This mesh is the coarsest level of our five-level preconditioner, when we want a mesh with 39233 nodes on the first level.

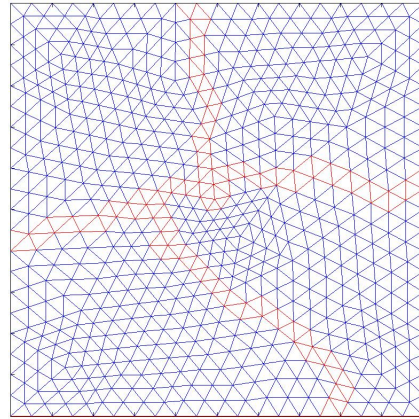


Figure 11: Mesh with a refinement label equal to 2 partitioned into 4 subdomains.

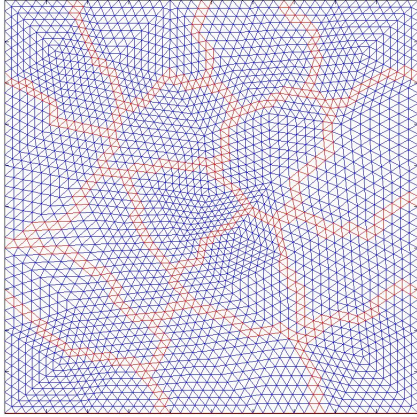


Figure 12: Mesh with a refinement label equal to 3 partitioned into 16 subdomains.

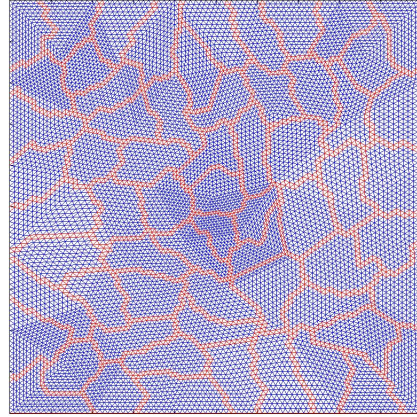


Figure 13: Mesh with a refinement label equal to 4 partitioned into 64 subdomains.

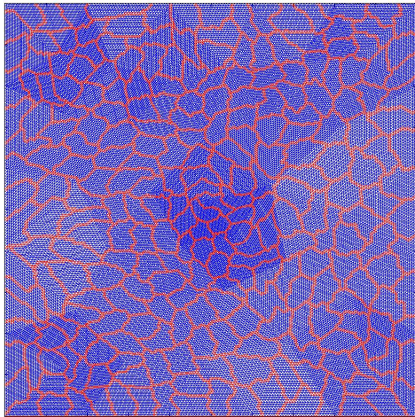


Figure 14: Mesh with a refinement label equal to 5 partitioned into 256 subdomains.

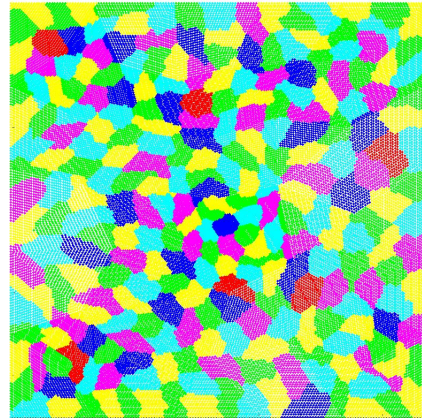


Figure 15: The same mesh of Figure 14 represented in terms of internal nodes, with subdomains coloured by a greedy algorithm (*sequential vertex coloring* in *Boost libraries*, [CM83]). When using a minimal overlap the representation with internal nodes avoids to overlap nodes belonging to different subdomains.

We have kept unchanged the discretization parameters of the finest level, and we removed the coarsest level, so to minimize the variation in the total number of elements of the preconditioner. The first (finest) level is completely identical. In the following levels we reduce the number of subdomains. We interpolated linearly the number of subdomains on intermediate levels, so that the coarsest level always has only one domain.

With the same argument we obtain preconditioners with 3 levels (Table 11) and with 2 levels (Table 12).

level nb.	$h_l$	$H_l$	$M_l$
$l = 3$	$h/4$	$H$	1
$l = 2$	$h/8$	$H/4$	16
$l = 1$	$h/16$	$H/16$	256

Table 11: Three-level preconditioner: gridsize parameter, characteristic subdomain dimension, number of subdomains in each level.

level nb.	$h_l$	$H_l$	$M_l$
$l = 2$	$h/8$	$H$	1
$l = 1$	$h/16$	$H/16$	256

Table 12: Two-level preconditioner: gridsize parameter, characteristic subdomain dimension, number of subdomains in each level.

In Table 13 we report the numerical results obtained by using CG and several kind of multilevel additive preconditioners. In the first row we use a simple CG without the domain decomposition preconditioner. In following lines we use  $\mathcal{P}_l = \mathcal{P}_{add}^l$  for levels  $l = 1 \div L - 1$ , and  $\mathcal{P}_L = \mathcal{P}_{int}$ . We pick  $L = 2 \div 5$ , corresponding to test several multilevel additive preconditioners, with level going from 2 to 5. In Tables 14 and 16 we report results obtained in the same way, but using BiCGstab. We recall that the multiplicative preconditioner is not symmetric, hence we would not be allowed to use CG. However, to combine nonsymmetric preconditioners with symmetric problems and vice versa may be beneficial [HV97]. Our multiplicative multilevel preconditioner performs well with CG too (Table 15) and its behaviour with respect to the additive variant is the same as with BiCGstab.

		no. 1st level nodes		
		9889	39233	156289
precondit.	CG monolithic	363	717	1417
	five-level additive	32	29	28
	four-level additive	29	27	26
	three-level additive	26	25	24
	two-level additive	23	22	-

Table 13: Number of iterations for several solvers, on progressively refined meshes. CG with additive variant is used, *i.e.*  $\mathcal{P}_l = \mathcal{P}_{add}^l$  for levels  $l = 1 \div L - 1$ , and  $\mathcal{P}_L = \mathcal{P}_{int}$ .

		no. 1st level nodes		
		9889	39233	156289
precondit.	BiCGstab monolithic	273	511	1197
	five-level additive	17	17	16
	four-level additive	16	15	15
	three-level additive	16	15	14
	two-level additive	13	12	-

Table 14: Number of iterations for several solvers, on progressively refined meshes. BiCGstab with additive variant is used, *i.e.*  $\mathcal{P}_l = \mathcal{P}_{add}^l$  for levels  $l = 1 \div L - 1$ , and  $\mathcal{P}_L = \mathcal{P}_{int}$ .

		no. 1st level nodes		
		9889	39233	156289
precondit.	CG monolithic	363	717	1417
	five-level multiplicative	24	25	26
	four-level multiplicative	22	24	24
	three-level multiplicative	19	20	20
	two-level multiplicative	14	14	-

Table 15: Number of iterations for several solvers, on progressively refined meshes. CG with multiplicative variant is used, *i.e.*  $\mathcal{P}_l = \mathcal{P}_{mul}^l$  for levels  $l = 1 \div L - 1$ , and  $\mathcal{P}_L = \mathcal{P}_{int}$ .

		no. 1st level nodes		
		9889	39233	156289
precondit.	BiCGstab monolithic	273	511	1197
	five-level multiplicative	11	11	15
	four-level multiplicative	11	14	13
	three-level multiplicative	11	12	11
	two-level multiplicative	8	8	-

Table 16: Number of iterations for several solvers, on progressively refined meshes. BiCGstab with multiplicative variant is used, *i.e.*  $\mathcal{P}_l = \mathcal{P}_{mul}^l$  for levels  $l = 1 \div L - 1$ , and  $\mathcal{P}_L = \mathcal{P}_{int}$ .



In all cases the number of iterations grows, when the number of levels grows. This applies when the mesh is fixed, corresponding to read tables by columns. If we read tables by rows we can investigate scalability properties with respect to the discretization parameter. The number of iterations when preconditioning decreases, while iterations for not preconditioned (we refer to this as monolithic) grows proportionally to  $h^{-2}$ .

Obiously, multiplicative approach is always more efficient than additive, even if iterations decrease so much that it is hard to distinguish contributions due to partitioning from those due to not regular BiCGstab convergence.

Last, we point out the difference from additive to multiplicative variant: when refining the mesh the number of iterations for the additive decreases, whereas that for the multiplicative increases, because coloring becomes less significant.

Let us now address a critical issue, that is: *why are iterations rising when the number of levels increases?* Apparently, it would seem that the computational work per iteration becomes larger when adding new levels, and at least the number of iterations should decrease.

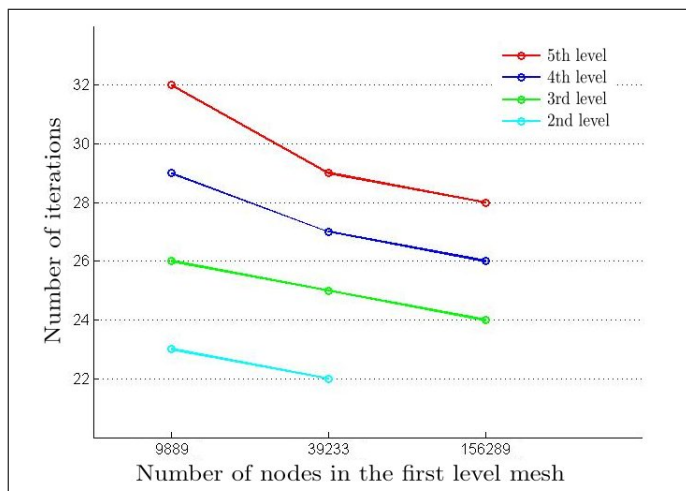


Figure 16: Data from Table 13. Number of iterations for CG with multilevel preconditioner  $P_L$ ,  $L = 2, 3, 4, 5$ . Projectors are  $\mathcal{P}_l = \mathcal{P}_{add}^l$  and  $\mathcal{P}_L = \mathcal{P}_{int}$ . The parameters of discretization vary according to Table 13. For the most refined mesh the coarse problem in two-level preconditioner is too large to be factored.

Let us see what happens when a new level is added, *e.g.* if we want to switch from two levels to three levels. When the coarse problem with two levels becomes too large (because of progressive refinement), we divide it in subdomains and build up the preconditioner as for the finer level. Then we add the third level, whose problem will be much smaller than the previous coarse problem. So in Figure 16, for a given mesh, we see that the number of iterations linearly grows with levels but the coarse problem get smaller (and problems in intermediate

levels too). In the end, multilevel preconditioners allow to refine the mesh and keep bounded the size of local problems among all levels. This result is mentioned in a less general framework in [SBG96], and is named *h*-scalability.

## 6 Multilevel preconditioners for stabilized problems

Let us consider the elliptic convection-diffusion problem in the domain  $\Omega = (0, 1) \times (0, 1)$ :

$$\begin{cases} -\nu\Delta u + \beta \cdot \nabla u = f, & \text{in } \Omega, \\ u = g, & \text{on } \partial\Omega, \end{cases} \quad (6.1)$$

with  $f$  and  $g$  chosen such that the exact solution is  $u(x, y) = \cos(x) \exp(2y)$ . As before, we use piecewise linear finite elements for the discretization. Characteristic dimensions for domains and elements are named  $L$  and  $\bar{h}$ , respectively. We define local and global *Péclet number* respectively by:

$$Pe_{loc} = \frac{|\beta|\bar{h}}{2\nu}, \quad Pe_{glob} = \frac{|\beta|L}{2\nu}. \quad (6.2)$$

Given  $\nu = 1$  and  $\beta = 10^5(y, -x)$  we have  $Pe_{glob} = 10^5/\sqrt{2}$  and a stabilization is needed. Thus we add the usual SUPG stabilization term in the weak formulation:

$$a_S(u, v) = \gamma \sum_T \int_{\partial T} \frac{h_T}{\|\beta\|} (\beta \cdot \nabla u) (\beta \cdot \nabla v) dT, \quad (6.3)$$

with  $\gamma = 1/3$  (there are no constraints on  $\gamma$  for P1 finite elements, [QV94]). Without either preconditioner and stabilization the finite element solution exhibits oscillations. In addition, when solving with multilevel preconditioner the stabilization is required to converge.

We discretize the problem (6.1) with the same meshes and partitionings of the previous section (Table 8, 9, 10, 9, 11, 12, Figure 10, 11, 12, 13, 14, 15). The preconditioner proved to be robust for a wide range of Péclet number ( $Pe_{glob} = (10^5 \div 10^{11})/\sqrt{2}$ ). In Table 17 we report results obtained when  $Pe_{glob} = 10^5/\sqrt{2}$ . Table 18 indicates  $Pe_{loc}$  in each level. In Table 20 and Table 19 we report the results when  $Pe_{glob} = 10^{11}/\sqrt{2}$ . The overall performance of the preconditioners for the stabilized problem (6.1) is completely identical to those for the model problem (3.1) given in the previous section. A moderate enhancement in the number of iterations is due to the increased stiffness of the problem. A graphical resume of data in Table 20 is given in Figure 17.



		no. 1st level nodes		
		9889	39233	156289
preconditioner	BiCGstab monolithic	352	638	1186
	five-level additive	92	91	77
	four-level additive	69	67	68
	three-level additive	48	43	37
	two-level additive	36	33	-
	five-level mutliplicative	47	84	73
	four-level mutliplicative	30	32	36
	three-level mutliplicative	24	26	23
	two-level mutliplicative	17	16	-

Table 17: Number of BiCGstab iterations for some preconditioners with different number of levels, on progressively refined meshes, with  $Pe_{glob} = 10^5/\sqrt{2}$ . Here we use additive ( $\mathcal{P}_l = \mathcal{P}_{add}^l$  and  $\mathcal{P}_L = \mathcal{P}_{int}$ ) and multiplicative ( $\mathcal{P}_l = \mathcal{P}_{mul}^l$  and  $\mathcal{P}_L = \mathcal{P}_{int}$ ) multilevel variants, with complete LU factorizations.

		no. 1st level nodes		
		9889	39233	156289
preconditioner	BiCGstab monolithic	400	817	1652
	five-level additive	95	94	83
	four-level additive	69	68	67
	three-level additive	49	45	40
	two-level additive	34	32	-
	five-level mutliplicative	48	56	63
	four-level mutliplicative	32	34	30
	three-level mutliplicative	24	27	25
	two-level mutliplicative	16	17	-

Table 19: Number of BiCGstab iterations for some preconditioners with different number of levels, on progressively refined meshes, with  $Pe_{glob} = 10^{11}/\sqrt{2}$ . Here we use additive ( $\mathcal{P}_l = \mathcal{P}_{add}^l$  and  $\mathcal{P}_L = \mathcal{P}_{int}$ ) and multiplicative ( $\mathcal{P}_l = \mathcal{P}_{mul}^l$  and  $\mathcal{P}_L = \mathcal{P}_{int}$ ) multilevel variants, with complete LU factorizations.

preconditioner level	$Pe_{loc}$
level 1	1096
level 2	2192
level 3	4377
level 4	8761
level 5	17522

Table 18: Local Péclet number in all levels, when  $Pe_{glob} = 10^5/\sqrt{2}$  and the first level mesh has 9889 nodes.

preconditioner level	$Pe_{loc}$
level 1	276
level 2	544
level 3	1096
level 4	2192
level 5	4377

Table 20: Local Péclet number in all levels, when  $Pe_{glob} = 10^5/\sqrt{2}$  and the first level mesh has 156289 nodes.

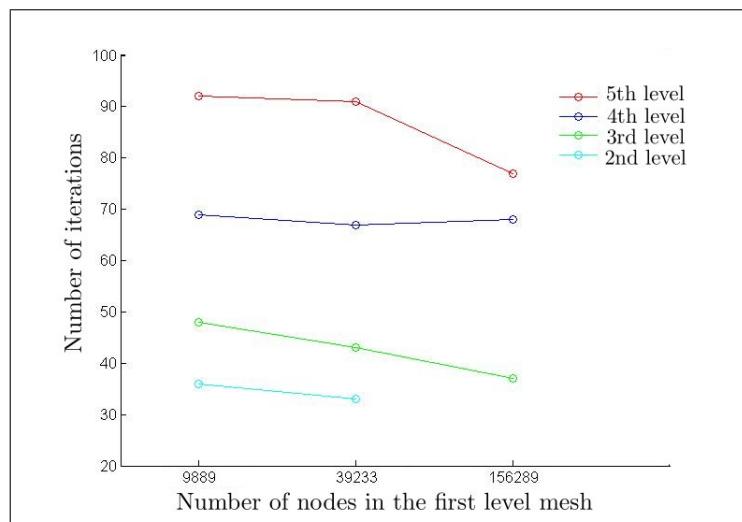


Figure 17: Data from Table 17. Number of iterations for BiCGstab with multi-level preconditioner  $P_L$ ,  $L = 2, 3, 4, 5$ . Projectors are  $\mathcal{P}_l = \mathcal{P}_{add}^l$  and  $\mathcal{P}_L = \mathcal{P}_{int}$ .

## 7 Incomplete local factorizations

In previous tests we may choose incomplete factorizations instead of complete ones to solve the local problems, because former are faster and cheaper to compute. Results show that, with a reasonable *drop tolerance*, convergence behaviour is not significantly affected. However, on the coarsest level we will always use a complete factorization. In Table 21 we solve (3.1) with CG and some five-level preconditioners, using different local factorizations.

solver	local factorizations	9889 nodes	39233 nodes	156289 nodes
five-level additive	LU	32	29	28
five-level additive	ILU no fill in	32	30	28
five-level additive	ILU <i>drop.tol</i> = 0.1	33	30	29
five-level multiplicative	LU	24	25	26
five-level multiplicative	ILU no fill in	19	20	22
five-level multiplicative	ILU <i>drop.tol</i> = 0.1	19	19	21

Table 21: Problem (3.1). Number of CG iterations with five-level additive or multiplicative preconditioner. Local factorizations may be complete or incomplete LU, without *fill in* or with a given drop tolerance (*drop.tol*).

In Table 22 we solve (6.1) with  $P_{e_{glob}} = 10^8/\sqrt{2}$ . We change the drop tolerance to reduce the number of total non-zero elements (in Table 23 we report those associated to all local factorizations in fifth level).

solver	local factorizations	9889 nodes
five-level add.	LU	93
five-level add.	ILU $drop\_tol=0.001$	93
five-level add.	ILU $drop\_tol=0.01$	92
five-level add.	ILU $drop\_tol=0.05$	97
five-level add.	ILU $drop\_tol=0.1$	120
five-level add.	ILU no fill in	$\infty$
five-level mult.	LU	48
five-level mult.	ILU $drop\_tol=0.001$	48
five-level mult.	ILU $drop\_tol=0.01$	52
five-level mult.	ILU $drop\_tol=0.05$	81
five-level mult.	ILU $drop\_tol=0.1$	$\infty$
five-level mult.	ILU no fill in	$\infty$

Table 22: Problem (6.1) with  $Pe_{glob} = 10^8/\sqrt{2}$ . Number of iterations for BiCGstab with five-level additive or multiplicative preconditioner. Local factorizations may be complete or incomplete LU, without *fill in* or with a given  $drop\_tol$ .

local factorizations	nnz level 5
LU	350355
ILU $drop\_tol=0.001$	349947
ILU $drop\_tol=0.01$	188143
ILU $drop\_tol=0.05$	135707
ILU $drop\_tol=0.1$	108477
ILU no fill in	115542

Table 23: Number of non-zero elements in local factorizations of fifth level, refer to Table 22.

## 8 A computational domain with arbitrary shape

Let us consider a Poisson problem on the domain  $D \subset \mathbb{R}^2$  whose (funny) geometry is shown in Figure 18:

$$\begin{cases} -\Delta u = f, & \text{in } D, \\ u = g, & \text{on } \partial D. \end{cases} \quad (8.1)$$

The physical boundary  $\partial D$  is uniformly Lipschitz continuous. Data  $g$  and  $f$  are chosen such that exact solution is  $u(x, y) = x^2 + y^2$ .

solver	preconditioner variant	no. subdomains within levels	local factorization within levels	refinement between levels	no. nodes within levels	CG iterations	BiCGstab iterations
monolithic	-	1	-	-	405631	1459	1158
one-level	additive	1024	LU	-	405631	155	115
two-level	additive	1/1024	LU/LU	4	2116/405631	39	22
two-level	additive	1/1024	LU/LU	3	7331/405631	30	19
two-level	additive	1/1024	LU/LU	2	27055/405631	27	17
three-level	additive	1/64/1024	LU/LU/LU	2/2	2116/27055/405631	35	19
three-level	additive	1/64/1024	LU/ILU/ILU*	2/2	2116/27055/405631	35	21
three-level	additive	1/64/1024	LU/ILU/ILU**	2/2	2116/27055/405631	35	19
one-level	multiplicative	1024	LU	-	405631	77	46
two-level	multiplicative	1/1024	LU/LU	4	2116/405631	23	14
two-level	multiplicative	1/1024	LU/LU	3	7331/405631	18	10
two-level	multiplicative	1/1024	LU/LU	2	27055/405631	19	11
three-level	multiplicative	1/64/1024	LU/LU/LU	2/2	2116/27055/405631	22	13
three-level	multiplicative	1/64/1024	LU/ILU/ILU*	2/2	2116/27055/405631	22	12
three-level	multiplicative	1/64/1024	LU/ILU/ILU**	2/2	2116/27055/405631	22	13

Table 24: Problem (8.1). Number of CG/BiCGstab iterations for some preconditioners. \* $drop\_tol = 1e - 3$ , \*\* $drop\_tol = 1e - 6$ .

In this test we have a domain with a complex shape, being meshed with adaptivity criteria. Therefore, subdomain dimensions may differ significantly. In Figure 18 we see that isolines match correctly through artificial interfaces. Figures 19 and 20 are levels used to build preconditioners in Table 24. Figure 21 shows the finer level colouring used when solving with multiplicative variants.

The aim of this test is to show differences among strategies used to change levels. In Table 24 we test many preconditioners. To build them all we just use four progressively refined meshes, with 2116, 7331, 27055 and 405631 nodes. The

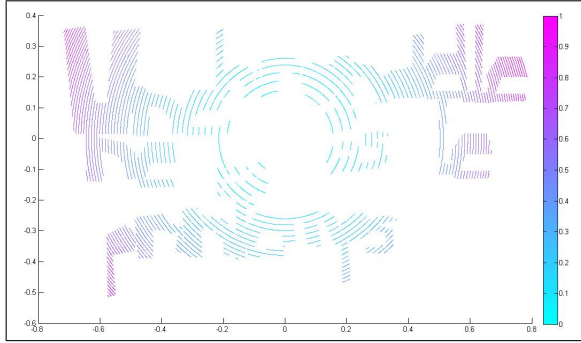


Figure 18: Isolines for the solution of problem (8.1) solved on the domain  $D$  with the three-level additive preconditioner from Table 24.



Figure 19: Mesh on  $D$  with 27055 nodes, partitioned into 64 subdomains, with minimal overlap.

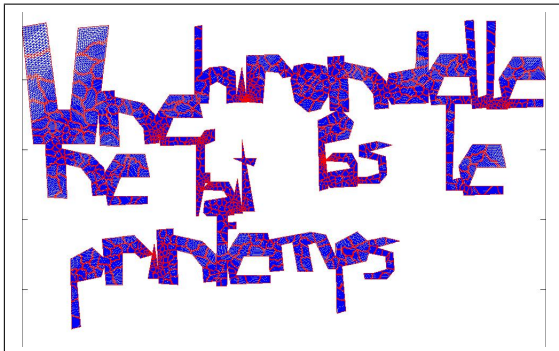


Figure 20: Mesh on  $P$  with 103679 nodes, partitioned into 1024 subdomains, with minimal overlap.

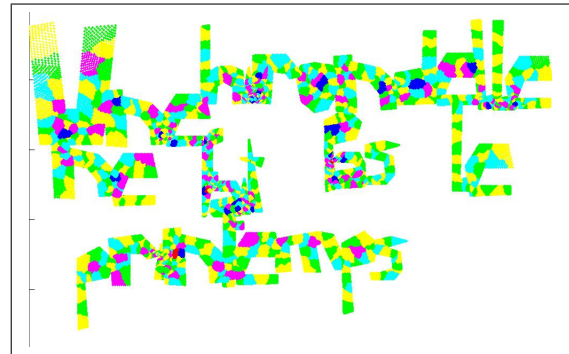


Figure 21: Mesh in Figure 18 with a greedy colouring on the 1024 subdomains.

last is always used (with the same partition) on the first level. For the second level of the two-level preconditioner we can choose one of remaining meshes. So we are able to test three different two-level preconditioners, and we observe that a coarser second level is cheaper to compute, but requires more iterations too. However, here the coarsest mesh is the best, because its coarse problem dimension is the nearest to local problem dimension of the first level.

Now, let us build a three-level preconditioner with 2116, 27055, and 405631 node meshes. If we consider these three-level and two-level preconditioners, we are able to compare different strategies to choose levels. We see that, when adding an intermediate level (with 27055 nodes), the number of iterations for CG decreases from 39 to 35. In this way we do not change partitioning in levels. The same behaviour when adding intermediate levels was pointed out in [PS08, Table 7.3].

If we add the level from above the number of iterations increases from 27 to 35, and we recover the standard theory behaviour. We remark once again that to add levels from above requires to change partitions in levels. So the number of iterations increases but local factorization cost decreases because local problems get smaller.

Multiplicative variants perform likewise. Incomplete local factorizations on finer levels still does not impact on performances.

## 9 Conclusion

In this paper we have introduced a broad class of multilevel Schwarz preconditioners (with an arbitrary number of levels), for subdomains resulting from mesh partitioners. When theoretical estimates exist (like in [DKW08] for two-level preconditioners), our numerical results confirm scalability properties as predicted by theory.

For multiple (greater than two) levels, several strategies can be adopted. One possibility consists of starting from one (fine) level and then adding coarser and coarser ones. In this paper we have shown that this strategy yields a linear growth of the number of iterations.

Another strategy consists of proceeding in the dual way, namely to start from one (coarse) level and then adding finer and finer ones. In this case the number of iterations is bounded with respect to the number of levels (see *e.g.* [PS08]).

A further strategy, that we have pursued in this paper, consists of starting from two (one fine, one coarse) levels, and increasing the number of levels by adding more and more intermediate ones. We show that also in this case the number of iterations remains bounded.

Several numerical tests have been performed for classical Laplace equation, for convection-diffusion problems with dominating convection, in complex shape domains, and using inexact algebraic solvers for the local subproblems.

## Acknowledgements

The authors would like to thank professors Olof Widlund and Clark Dohrmann for many useful comments and advises on an early version of this paper.

## References

- [Bre00] Susanne C. Brenner, *Lower bounds for two-level additive Schwarz preconditioners with small overlap*, SIAM J. Sci. Comput. **21** (2000), no. 5, 1657–1669 (electronic), Iterative methods for solving systems of algebraic equations (Copper Mountain, CO, 1998). MR MR1771050 (2001j:65181)
- [BS92] P. E. Bjørstad and M. D. Skogen, *Domain decomposition algorithms of Schwarz type, designed for massively parallel computers*, Fifth International Symposium on Domain Decomposition Methods for Partial Differential Equations (Norfolk, VA, 1991), SIAM, Philadelphia, PA, 1992, pp. 362–375. MR MR1189587
- [Cai94] X. Cai, *A non-nested coarse space for schwarz type domain decomposition methods*, Tech. report, Department of Computer Science University of Colorado, 1994.
- [CM83] Thomas F. Coleman and Jorge J. Moré, *Estimation of sparse Jacobian matrices and graph coloring problems*, SIAM J. Numer. Anal. **20** (1983), no. 1, 187–209. MR MR687376 (84f:05043)
- [CSZ96] Tony F. Chan, Barry F. Smith, and Jun Zou, *Overlapping Schwarz methods on unstructured meshes using non-matching coarse grids*, Numer. Math. **73** (1996), no. 2, 149–167. MR MR1384228 (97h:65135)
- [CZ96] Tony F. Chan and Jun Zou, *A convergence theory of multilevel additive Schwarz methods on unstructured meshes*, Numer. Algorithms **13** (1996), no. 3-4, 365–398 (1997). MR MR1430525 (98d:65155)
- [DKW08] Clark R. Dohrmann, Axel Klawonn, and Olof B. Widlund, *Domain decomposition for less regular subdomains: overlapping Schwarz in two dimensions*, SIAM J. Numer. Anal. **46** (2008), no. 4, 2153–2168. MR MR2399412 (2009a:65314)
- [DW90] Maksymilian Dryja and Olof B. Widlund, *Towards a unified theory of domain decomposition algorithms for elliptic problems*, Third International Symposium on Domain Decomposition Methods for Partial Differential Equations (Houston, TX, 1989), SIAM, Philadelphia, PA, 1990, pp. 3–21. MR MR1064335 (91m:65294)

- [DW94] ———, *Domain decomposition algorithms with small overlap*, SIAM J. Sci. Comput. **15** (1994), no. 3, 604–620, Iterative methods in numerical linear algebra (Copper Mountain Resort, CO, 1992). MR MR1273155 (95d:65102)
- [HV97] Michael Holst and Stefan Vandewalle, *Schwarz methods: to symmetrize or not to symmetrize*, SIAM J. Numer. Anal. **34** (1997), no. 2, 699–722. MR MR1442935 (98c:65196)
- [KK98] George Karypis and Vipin Kumar, *A fast and high quality multilevel scheme for partitioning irregular graphs*, SIAM J. Sci. Comput. **20** (1998), no. 1, 359–392 (electronic). MR MR1639073 (99f:68158)
- [KRW08] Axel Klawonn, Oliver Rheinbach, and Olof B. Widlund, *An analysis of a FETI-DP algorithm on irregular subdomains in the plane*, SIAM J. Numer. Anal. **46** (2008), no. 5, 2484–2504. MR MR2421044 (2010c:65049)
- [Lio88] P.-L. Lions, *On the Schwarz alternating method. I*, First International Symposium on Domain Decomposition Methods for Partial Differential Equations (Paris, 1987), SIAM, Philadelphia, PA, 1988, pp. 1–42. MR MR972510 (90a:65248)
- [PS08] Luca F. Pavarino and Simone Scacchi, *Multilevel additive Schwarz preconditioners for the bidomain reaction-diffusion system*, SIAM J. Sci. Comput. **31** (2008), no. 1, 420–445. MR MR2460784 (2010d:92044)
- [QSS07] Alfio Quarteroni, Riccardo Sacco, and Fausto Saleri, *Numerical mathematics*, second ed., Texts in Applied Mathematics, vol. 37, Springer-Verlag, Berlin, 2007. MR MR2265914 (2007f:65001)
- [Qua09] Alfio Quarteroni, *Numerical models for differential problems*, MS&A. Modeling, Simulation and Applications, vol. 2, Springer-Verlag Italia, Milan, 2009. MR MR2522375
- [QV94] Alfio Quarteroni and Alberto Valli, *Numerical approximation of partial differential equations*, Springer Series in Computational Mathematics, vol. 23, Springer-Verlag, Berlin, 1994. MR MR1299729 (95i:65005)
- [QV99] ———, *Domain decomposition methods for partial differential equations*, Numerical Mathematics and Scientific Computation, The Clarendon Press Oxford University Press, New York, 1999, Oxford Science Publications. MR MR1857663 (2002i:65002)
- [SBG96] Barry F. Smith, Petter E. Bjørstad, and William D. Gropp, *Domain decomposition*, Cambridge University Press, Cambridge, 1996, Parallel multilevel methods for elliptic partial differential equations. MR MR1410757 (98g:65003)



- [TW05] Andrea Toselli and Olof Widlund, *Domain decomposition methods—algorithms and theory*, Springer Series in Computational Mathematics, vol. 34, Springer-Verlag, Berlin, 2005. MR MR2104179 (2005g:65006)
- [Wid88] Olof B. Widlund, *Iterative substructuring methods: algorithms and theory for elliptic problems in the plane*, First International Symposium on Domain Decomposition Methods for Partial Differential Equations (Paris, 1987), SIAM, Philadelphia, PA, 1988, pp. 113–128. MR MR972514 (90c:65138)
- [Wid92] ———, *Some Schwarz methods for symmetric and nonsymmetric elliptic problems*, Fifth International Symposium on Domain Decomposition Methods for Partial Differential Equations (Norfolk, VA, 1991), SIAM, Philadelphia, PA, 1992, pp. 19–36. MR MR1189560 (93j:65202)
- [Zha92] Xuejun Zhang, *Multilevel Schwarz methods*, Numer. Math. **63** (1992), no. 4, 521–539. MR MR1189535 (93h:65047)

## MOX Technical Reports, last issues

Dipartimento di Matematica “F. Brioschi”,  
Politecnico di Milano, Via Bonardi 9 - 20133 Milano (Italy)

- 33/2010** GIOVANNI MIGLIORATI, ALFIO QUARTERONI:  
*Multilevel Schwarz Methods for Elliptic Partial Differential Equations*
- 32/2010** A. CRISTIANO I. MALOSSO, PABLO J. BLANCO,  
SIMONE DEPARIS, ALFIO QUARTERONI:  
*Algorithms for the partitioned solution of weakly coupled fluid models  
for cardiovascular flows*
- 31/2010** ANDREA MANZONI, ALFIO QUARTERONI, GIANLUIGI ROZZA:  
*Shape optimization for viscous flows by reduced basis methods and free-  
form deformation*
- 30/2010** PABLO J. BLANCO, MARCO DISCACCIATI,  
ALFIO QUARTERONI:  
*Modeling dimensionally-heterogeneous problems: analysis,  
approximation and applications*
- 29/2010** MATTEO LESINIGO, CARLO D’ANGELO, ALFIO QUARTERONI:  
*A multiscale Darcy-Brinkman model for fluid flow in fractured porous  
media*
- 28/2010** PAOLO CROSETTO, PHILIPPE REYMOND, SIMONE DEPARIS,  
DIMITRIOS KONTAXAKIS, NIKOLAOS STERGIOPULOS,  
ALFIO QUARTERONI:  
*Fluid Structure Interaction Simulations of Physiological Blood Flow in  
the Aorta*
- 27/2010** MATTEO BRUGGI, MARCO VERANI:  
*An adaptive algorithm for topology optimization with goal-oriented  
error control*
- 26/2010** FRANCESCA IEVA, ANNA MARIA PAGANONI:  
*Designing and mining a multicenter observational clinical registry  
concerning patients with Acute Coronary Syndromes*
- 25/2010** G. PENA, C. PRUD’HOMME, ALFIO QUARTERONI:  
*High Order Methods for the Approximation of the  
Incompressible Navier-Stokes Equations in a Moving Domain*

**24/2010** LORENZO TAMELLINI, LUCA FORMAGGIA,  
EDIE MIGLIO, ANNA SCOTTI:  
*An Uzawa iterative scheme for the simulation of floating boats*