
This version is available at https://strathprints.strath.ac.uk/52160/

Strathprints is designed to allow users to access the research output of the University of Strathclyde. Unless otherwise explicitly stated on the manuscript, Copyright © and Moral Rights for the papers on this site are retained by the individual authors and/or other copyright owners. Please check the manuscript for details of any other licences that may have been applied. You may not engage in further distribution of the material for any profitmaking activities or any commercial gain. You may freely distribute both the url (https://strathprints.strath.ac.uk/) and the content of this paper for research or private study, educational, or not-for-profit purposes without prior permission or charge.

Any correspondence concerning this service should be sent to the Strathprints administrator: strathprints@strath.ac.uk
Two-level domain decomposition methods for highly heterogeneous Darcy equations.
Connections with multiscale methods

Victorita Dolean, Pierre Jolivet, Frédéric Nataf, Nicole Spillane and Hua Xiang

1 Laboratoire J.L. Lions, UPMC et CNRS UMR 7598
2 Laboratoire J.A. Dieudonné, Université de Nice
3 School of Mathematics and Statistics, Wuhan University, 430072 Wuhan, P. R. China

e-mail: name@ann.jussieu.fr — dolean@unice.fr — hxiang@whu.edu.cn

*Corresponding author

1 INTRODUCTION

Multiphase, compositional porous media flow models, used in reservoir simulations or basin modeling, lead to the solution of complex non linear systems of Partial Differential Equations (PDEs). These PDEs are typically discretized using a cell-centered finite volume scheme and a fully implicit Euler integration in time in order to allow for large time steps. After Newton type linearization, one ends up with the solution of a linear system at each Newton iteration which represents up to 90 percents of the total simulation elapsed time. The corresponding pressure block matrix is related to the discretization of a Darcy equation with high contrasts and anisotropy in the coefficients. We focus on overlapping Schwarz type methods on parallel computers and on multiscale methods.

Coarse spaces are instrumental in obtaining scalability for domain decomposition methods. For matrices arising from problems with smooth coefficients, it is possible to build efficient coarse spaces based on domain wise constant vectors, see and references therein. For problems with high heterogeneities, these simple coarse spaces do not work well. Here, we present a recent coarse space.
that is robust even when there are such heterogeneities. We achieve this by solving local generalized eigenvalue problems which isolate the terms responsible for slow convergence. Building efficient coarse spaces is closely related to multiscale methods which also aim to reduce the computational cost of large scale problems.

An outline of the paper is as follows. Section 2 consists in an introduction to one level Schwarz methods. Material is basic but the presentation is quite new. In §3 we present a recent spectral coarse space which adapts automatically to the heterogeneities of the problem. In §4 we present results of large scale computations. In §5 it is compared to multiscale methods. In section 6 we conclude and present prospects on adaptation of the spectral coarse space to finite volume discretizations.

2 SCHWARZ METHODS

We start with the original Schwarz algorithm \[27\] at the continuous (i.e. partial differential equations) level whose parallel version is named Jacobi-Schwarz method (JSM). We introduce two variants that are at the origin of the popular additive Schwarz method (ASM) and restricted additive Schwarz (RAS \[4\]) algorithms. The first one has been the subject of hundreds of papers (see \[29\] and references therein). The second one is the default parallel solver of the parallel package software PETSc \[2\]. This presentation shows in a unified setting the connections between these three algorithms.

2.1 Three Schwarz Algorithms at the continuous level

Hermann Schwarz was a German analyst of the 19th century. He was interested in proving existence and uniqueness of the Poisson problem. At his time, there were no Sobolev spaces nor Lax-Milgram theorem. The only available tool of the domain \(\Omega\) was the Fourier transform, limited by its very nature to simple geometries. In order to consider more general situations, we need extension operators, \(H^1(\Omega)\), the space in which problem (1) is naturally posed, and thus the well-posedness of the Poisson problem in complex geometries.

With the advent of digital computers, this method also acquired a practical interest as an iterative linear solver. Subsequently, parallel computers became available and a small modification of the algorithm makes it suited to these architectures. It is sufficient to solve concurrently in all subdomains, \(i = 1, 2\):

\[
\begin{align*}
-\Delta(u_i^{n+1}) &= f & \text{in } \Omega_i \\
\frac{\partial u_i^{n+1}}{\partial \Omega} &= 0 & \text{on } \partial \Omega_i \\
u_i^{n+1} &= u_{i-1}^n & \text{on } \partial \Omega_i \cap \Omega_{i-1}.
\end{align*}
\]

It is easy to see that if the algorithm converges, the solutions \(u_i^n, \, i = 1, 2\) in the intersection of the subdomains take the same values. Indeed, in the overlap \(\Omega_{12} := \Omega_1 \cap \Omega_2\), let \(e^{n} := u_1^n - u_2^n\). By the last line of (3), we know that \(e^{n} = 0\) on \(\partial \Omega_{12}\). By linearity of the equation, we also have that \(e^{\infty}\) is harmonic. Thus, \(e^{n}\) solves the homogeneous well posed BVP:

\[
\begin{align*}
-\Delta(e^{n}) &= 0 & \text{in } \Omega_{12} \\
e^{n} &= 0 & \text{on } \partial \Omega_{12}
\end{align*}
\]

and thus \(e^{\infty} = 0\).

Algorithms (2) and (3) act on the local functions \((u_i)_{i=1,2}\). In order to write algorithms that act on global functions in \(H^1(\Omega)\), the space in which problem (1) is naturally posed, we need extension operators, \(E_i\), so that for a function \(w_i: \Omega_i \mapsto \mathbb{R}\), \(E_i(w_i): \Omega \mapsto \mathbb{R}\) is the extension of \(w_i\) by zero outside \(\Omega_i\). We also need partition of unity functions \(\chi_i: \Omega \mapsto [0,1]\).
\( \Omega \mapsto \mathbb{R}, \chi_i \geq 0 \) and \( \chi_i(x) = 0 \) for \( x \in \partial \Omega_i \) and such that:

\[
w = \sum_{i=1}^{2} E_i(\chi_i w|_{\Omega_i})
\]

for any function \( w : \Omega \mapsto \mathbb{R} \). This definition of a partition of unity is closer to the computer implementation than the classical definition of a partition of unity functions.

There are two ways to write related algorithms that act on functions \( \mathbf{u}^n \in H^1(\Omega) \). The first possibility is to use \( \mathbf{u}^n \) as an approximation to the solution to the Poisson problem \( (1) \). 

\[
\begin{align*}
-\Delta (\mathbf{u}^{n+1}) &= f & \text{in } \Omega \\
\mathbf{u}^{n+1} &= 0 & \text{on } \partial \Omega \cap \partial \Omega_i \\
\mathbf{u}^{n+1} &= \mathbf{u}^n & \text{on } \partial \Omega \cap \overline{\Omega}_{3-i}.
\end{align*}
\]

and then gluing them together using the partition of unity functions.

\[
\mathbf{u}^{n+1} := \sum_{i=1}^{2} E_i(\chi_i \mathbf{u}^{n+1}_i).
\]

A second possibility consists in replacing the above formula by a simpler formula not based on the partition of unity:

\[
\mathbf{u}^{n+1} := \sum_{i=1}^{2} E_i(\mathbf{u}^{n+1}_i).
\]

### 2.2 Schwarz Algorithms at the algebraic level

So far, we have given a continuous presentation of domain decomposition methods. Actually, these methods are used in their algebraic form to solve linear systems arising from the discretization of partial differential equations. We now give the matrix counterpart of these algorithms.

For this, we first give a kind of dictionary to go from the continuous level to the discrete one:

- the counterparts of a domain \( \Omega \) and of an overlapping decomposition \( \Omega = \bigcup_{i=1}^{N} \Omega_i \) are a set of degrees of freedom (d.o.f.) \( \mathbf{N} \) and a decomposition in subsets \( \mathbf{N} = \bigcup_{i=1}^{N} \mathbf{N}_i \).
- a function \( u : \Omega \rightarrow \mathbb{R} \) corresponds a vector \( \mathbf{u} \in \mathbb{R}^{\mathbf{N}} \).
- the restriction of a function \( u : \Omega \rightarrow \mathbb{R} \) to a subdomain \( \Omega_i, 1 \leq i \leq N \) is analog to the restriction \( R_i \mathbf{u} \) of a vector \( \mathbf{u} \in \mathbb{R}^{\mathbf{N}} \) to subset \( \mathbf{N}_i \). Matrix \( R_i \) is a Boolean rectangular of size \( \# \mathbf{N}_i \times \# \mathbf{N} \).
- similarly, \( E_i(\mathbf{u}) \) the extension by zero of a function \( \mathbf{u} : \Omega_i \rightarrow \mathbb{R} \) to a function \( \mathbf{u} : \Omega \rightarrow \mathbb{R} \) corresponds at the algebraic level to \( R_i^T \mathbf{u}_i \) where \( R_i^T \) is the transpose of matrix \( R_i \) and \( \mathbf{u}_i \in \mathbb{R}^{\# \mathbf{N}_i} \) is a local vector.
- the counterparts of partition of unity functions \( \chi_i, 1 \leq i \leq N \) are diagonal matrices with positive entries, of size \( \# \mathbf{N}_i \times \# \mathbf{N}_i \) s. t. \( \mathbf{1} \mathbf{d} = \sum_{i=1}^{N} R_i^T D_i R_i \).
- After discretization, solving Poisson problem \( (1) \) amounts to solving a SPD linear system

\[
A \mathbf{u} = \mathbf{f}.
\]

- Solving a local subproblem in a subdomain \( \Omega_i \) such as in equations \( (3) \) or \( (5) \) corresponds at the algebraic level to solving linear systems of the form \( R_i A R_i^T \mathbf{u}^{n+1}_i = \mathbf{f}_i \).

We now define, at the algebraic level, the RAS and ASM algorithms and not JSM since it is seldom used and is more complex to define. As for the counterpart of the RAS algorithms \( (5)-(6) \), we give the following definition

\[
M_{\text{RAS}}^{-1} := \sum_{i=1}^{N} R_i^T D_i (R_i A R_i^T)^{-1} R_i
\]

so that the iterative RAS algorithm reads:

\[
U^{n+1} = U^n + M_{\text{RAS}}^{-1} \mathbf{r}^n
\]

where \( \mathbf{r}^n := \mathbf{f} - A U^n \).

As for the counterpart of the ASM algorithm \( (5)-(7) \), we give the following definition

\[
M_{\text{ASM}}^{-1} := \sum_{i=1}^{N} R_i^T (R_i A R_i^T)^{-1} R_i
\]

so that the iterative ASM algorithm reads:

\[
U^{n+1} = U^n + M_{\text{ASM}}^{-1} \mathbf{r}^n
\]

As is well known, such fixed point methods are out performed by Krylov based iterative solvers such as the conjugate gradient (CG) algorithm of the generalized minimal residual method (GMRES), see the book by Y. Saad [26] and references therein. In our context, using these methods amounts to solve the linear system \( (8) \) by a CG algorithm.
preconditioned by the symmetric preconditioner \( M_{\mathrm{ASM}} \) or by a GMRES algorithm preconditioned by the unsymmetric preconditioner \( M_{\mathrm{RAS}} \). In both cases, the convergence properties are related to the spectral properties of the preconditioned operator \( M^{-1} \). The restricted additive Schwarz method (RAS, see [3]) is the default parallel solver in the PETSc package. For the additive Schwarz method (ASM) many theoretical results have been derived, see [29] and references therein.

3 ADAPTIVE SPECTRAL COARSE SPACE

The domain decomposition methods presented so far were written for a two subdomain decomposition. Their extension to an arbitrary number \( N \) of subdomains \( (\Omega_i)_{i \in S_N} \) is only a matter of notation. It is sufficient in definitions of the previous section to sum over all subdomains from \( i = 1 \) to \( i = N \). But, when the number of subdomains is large, plateaus appear in the convergence of Schwarz domain decomposition methods. This is the case even for a regular problem such as the Poisson problem (1). The problem comes from the fact the preconditioner lacks of a global mechanism for exchange of information. Preconditioners RAS and ASM defined in the previous sections are called one-level methods. Data are exchanged only from one subdomain to its direct neighbors. But the solution in each subdomain depends on the right handside in all subdomains. Let us call \( N_d \) the number of subdomains in one direction. Then, for instance, the leftmost domain of Figure 3 needs at least \( N_d \) iterations before knowing something about the value of the right handside \( f \) in the rightmost subdomain. The length of the plateau is thus typically related to the number of subdomains in one direction and therefore to the notion of scalability met in the context of high performance computing.

In order to achieve scalability of the domain decomposition (DD) method, we introduce two-level domain decomposition methods via a coarse space correction. The precise motivation and linear algebra setting are given in § 3.1 for a problem with smooth coefficients. A new approach § 3.2 introduced in [22, 23] is necessary to achieve scalability for arbitrary highly heterogeneous coefficients. A condition number estimate theorem supports the approach. The method is tested in § 3.4 on difficult heterogeneous test cases including channelized medium. In practice, the coarse space seems to be optimal, see Table 10 in § 3.4.1.

3.1 Need for a two-level method

When the number of subdomains is large, plateaus appear in the convergence of Schwarz domain decomposition methods. The remedy will consist in the introduction of a two-level preconditioner via a coarse space correction.

The problem and its cure are well illustrated in Figure 2 for a domain decomposition into 64 strips. The one level method has a long plateau in the convergence whereas with a coarse space correction convergence is quite fast. For instance, in Figure 2 we consider a 2D problem decomposed into \( 2 \times 2 \), \( 4 \times 4 \) and \( 8 \times 8 \) subdomains. For each domain decomposition, we have two curves: one with a one-level method and the second with a coarse grid correction which is denoted by M2. We see that for the one-level curves, the plateau has a size proportional to the number of subdomains in one direction. In two-level methods, a small problem of size typically the number of subdomains couples all subdomains at each iteration. It is through this mechanism that scalability can be achieved.

From a condition number point of view, stagnation corresponds to a few very low eigenvalues in the spectrum of the preconditioned problem. Using preconditioners \( M_{\mathrm{ASM}} \) or \( M_{\mathrm{RAS}} \), we can remove the influence of very large eigenvalues of the coefficient matrix, which correspond to high frequency modes. Indeed, it has been proved that for a SPD matrix, the largest eigenvalue of the preconditioned system by \( M_{\mathrm{ASM}} \) is bounded by the number of colors needed to color the overlapping subdomains with different colors.
colors for adjacent subdomains, see [29] or [25] for instance. But the small eigenvalues still exist and hamper the convergence. These small eigenvalues correspond to low frequency modes and represent certain global information. We need a suitable coarse grid space to efficiently deal with them.

A classical remedy consists in the introduction of a coarse grid or coarse space correction that couples all subdomains at each iteration of the iterative method. This is closely related to deflation technique classical in linear algebra, see Nabben and Vuik’s paper [21] and references therein. Suppose we have identified the modes corresponding to the slow convergence of the iterative method used to solve the linear system:

$$ Ax = b $$

with a preconditioner $M$, in our case a domain decomposition method. That is, we have some a priori knowledge on the small eigenvalues of the preconditioned system $M^{-1}A$. For a Poisson problem, these slow modes correspond to constant functions that are in the null space (kernel) of the Laplace operators. For a homogenous elasticity problem, they correspond to the rigid body motions. Let us call $Z$ the rectangular matrix whose columns correspond to these slow modes. There are algebraic ways to incorporate these informations to accelerate the domain decomposition method. We give here the presentation that is classical in domain decomposition methods. In the case where $A$ is SPD, the starting point is to consider the minimization problem

$$ \min_{\beta} \| Ay + Z\beta - b \|_{A^{-1}}. $$

It corresponds to finding the best correction to an approximate solution $y$ by a vector $Z\beta$ in the vector space spanned by the $n_c$ columns of $Z$. This problem is equivalent to

$$ \min_{\beta \in \mathbb{R}^{n_c}} 2(Ay - b, Z\beta)_2 + (AZ\beta, AZ\beta)_2 $$

and whose solution is:

$$ \beta = (Z^TAZ)^{-1}Z^T(b - Ay). $$

Thus, the correction term is:

$$ Z\beta = Z(Z^TAZ)^{-1}Z^T(b - Ay). $$

Let $R_0 := Z^T$ and $r = b - Ay$ be the residual associated to the approximate solution $y$, the best correction that belongs to the vector space spanned by the columns of $Z$ reads:

$$ R_0^i (R_0 AR_0^T)^{-1} R_0 r. $$

When using such an approach with an additive Schwarz method (ASM), it is natural to introduce an additive correction to the additive Schwarz method:

$$ M_{ASM,2}^{-1} := R_0^i (R_0 AR_0^T)^{-1} R_0 + \sum_{i=1}^{N} R_0^i (R_i AR_i^T)^{-1} R_i $$

where the $R_i$’s ($1 \leq i \leq N$) are the restriction operators to the overlapping subdomains. The structure of the two level preconditioner $M_{ASM,2}^{-1}$ is thus the same than in the one level method. Compared to the one level Schwarz method where only local subproblems have to be solved in parallel, the two-level method adds the solution of a linear system in a sequential way with the matrix $R_0 AR_0^T$. This problem couples all subdomains at each iteration. But this matrix is a small $O(N \times N)$ square matrix and the extra cost is negligible compared to the gain. Indeed, in Table [1] we display the iteration counts for a decomposition of the domain in an increasing number of subdomains. In figure [4] we see that without a coarse grid correction, the convergence curve of the one level Schwarz method has a very long plateau that can be bypassed by a two-level method.

<table>
<thead>
<tr>
<th>N subdomains</th>
<th>Schwarz</th>
<th>With coarse grid</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>18</td>
<td>25</td>
</tr>
<tr>
<td>8</td>
<td>37</td>
<td>22</td>
</tr>
<tr>
<td>16</td>
<td>54</td>
<td>24</td>
</tr>
<tr>
<td>32</td>
<td>84</td>
<td>25</td>
</tr>
<tr>
<td>64</td>
<td>144</td>
<td>25</td>
</tr>
</tbody>
</table>

TABLE 1
Iteration counts for a Poisson problem on a domain decomposed into strips. The number of unknowns is proportional to the number of subdomains (weak scalability).

Figure 4
Convergence curves with and without a coarse space correction for a decomposition into 64 strips

We give here a precise definition to $Z$ for a Poisson problem. This construction was introduced in Nicolaides [24]. We take $Z$ so that it has a domain decomposition structure. $Z$ is defined by vectors which have local support in the sub-
domains and so that the constant function $1$ belongs to the vector space spanned by $Z$. Recall that we have a partition of unity in the following sense: let $D_i$, $1 \leq i \leq N$, be matrices
\[ D_i : \mathbb{R}^{\dim(N_i)} \to \mathbb{R}^{\dim(N_i)} \]
so that we have:
\[ \sum_{i=1}^{N} R_i^T D_i R_i = 1d. \]
We define $Z$ such that the $i$-th column of $Z$ is:
\[ Z_i := R_i^T D_i R_i 1 \text{ for } 1 \leq i \leq N \]
where $1$ is the vector full of ones. The structure of $Z$ is thus the following:
\[ Z_{Nico} = \begin{bmatrix} D_1 R_1 1 & 0 & \cdots & 0 \\ \vdots & D_2 R_2 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & D_N R_N 1 \end{bmatrix}. \]

The results of Figures 5 and Table 1 were obtained using this method.

For problems of the type $-\text{div}(\alpha \nabla u) = f$ with smooth coefficients $\alpha$, this coarse space gives good results. But for highly heterogeneous coefficients, there is still a plateau in the convergence of the solver. The results in Figure 5 correspond to a domain which has two layers with high values of $\alpha$. The computational domain has a stripwise decomposition into 64 subdomains. Two Schwarz methods are tested with either no coarse space correction, a Nicolaides coarse space or a spectral coarse space defined in § 3.2 so a total of six curves. The curves with very long plateaus are one level Schwarz methods. The curves $Z_{Nico}$ (pink curves) correspond to two level Schwarz methods with a Nicolaides coarse space, equation (13). The plateau in the convergence is not as large but still exists. With the spectral coarse space of the next section, we automatically select two modes per subdomain and get the convergence curves $Z_{DDN}$ (black curves).

### 3.2 Spectral coarse space for highly heterogeneous problems

We now propose a construction of the coarse space that will be suitable for parallel implementation and efficient for accelerating the convergence for problems with highly heterogeneous coefficients of the type
\[ \begin{align*}
-\text{div}(\alpha \nabla u) &= f \text{ in } \Omega, \\
B(u) &= 0 \text{ on } \partial\Omega
\end{align*} \]
with $\alpha$ a positive function. We still choose $Z$ such that it has the form
\[ Z = \begin{bmatrix} W^1 & 0 & \cdots & 0 \\ \vdots & W^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & W^N \end{bmatrix}, \]
where $N$ is the number of overlapping subdomains. But $W$ is now a rectangular matrix whose columns are based on the harmonic extensions of the eigenvectors corresponding to small eigenvalues of the Dirichlet-to-Neumann (DtN) map in each subdomain $\Omega_i$. Remark that the sparsity of the coarse operator $E = Z' A Z$ is a result of the sparsity of $Z$. The nonzero components of $E$ correspond to adjacent subdomains.

More precisely, let us consider first at the continuous level the Dirichlet to Neumann map $\text{DtN}_{\Omega_i}$. Let $u : \Gamma_i \mapsto \mathbb{R}$, $(\Gamma_i := \partial\Omega_i \setminus \partial\Omega)$
\[ \text{DtN}_{\Omega_i}(u) = \alpha \frac{\partial v}{\partial n} \bigg|_{\Gamma_i}, \]
where $v$ satisfies
\[ \begin{cases}
\mathcal{L}(v) := -\text{div}(\alpha \nabla v) = 0, & \text{in } \Omega_i, \\
v = u, & \text{on } \Gamma_i,
\end{cases} \]
and $\Gamma_i$ is the interface boundary. If the subdomain is not a floating one (i.e. $\partial\Omega_i \cap \partial\Omega \neq \emptyset$), we use on the part of the global boundary, the boundary condition from the original problem $B(u) = 0$. To construct the coarse grid subspace, we use the low frequency modes associated with the DtN operator:
\[ \text{DtN}_{\Omega_i}(u) = \lambda u \]
with
\[ \lambda < 1 / \text{diam}(\Omega_i) \]
where $\text{diam}(\Omega_i)$ is the diameter of subdomain $\Omega_i$. The rationale for this choice is that the condition number estimate of Theorem 3.2 is then similar to the one of Theorem 3.1 for the Poisson problem. Note the term $\alpha$ in the generalized eigenvalue problem (15).

We first motivate our choice of a coarse space based on DtN map. We write the original Schwarz method at the continuous level, where the domain $\Omega$ is decomposed in a one-way partitioning, see Figure 6. The error $e^0$ between the
current iterate at step $n$ of the algorithm and the solution $u_{\Omega}(e_{i}^{n} := u^{n} - u_{\Omega})$ in subdomain $\Omega_i$ at step $n$ of the algorithm satisfies:

$$L(e_{i}^{n+1}) = 0 \quad \text{in} \quad \Omega_i,$$

$$e_{i}^{n+1} = e_{j}^{n} \quad \text{on} \quad \Omega_i \cap \partial \Omega_j.$$  

On the 1D example sketched in Figure 6, we see that the rate of convergence of the algorithm is related to the decay of the harmonic functions $e_{i}$ in the vicinity of $\partial \Omega_i$ via the subdomain boundary condition. Indeed, a small value for this boundary condition leads to a smaller error in the entire subdomain thanks to the maximum principle.

Moreover a fast decay for this value corresponds to a large eigenvalue of the DtN map whereas a slow convergence of the Schwarz algorithm. With block notations, we have

$$A^{(i)}_{\Gamma_i} = A_{\Gamma_i} - \lambda M_{\alpha,\Gamma_i},$$

where $A_{\Gamma_i}$ refers to the matrix prior to assembly with the neighboring subdomains and thus cannot be simply extracted from the coefficient matrix $A$. In problem (17), we use Dirichlet boundary conditions. Let $U \in \mathbb{R}^{n_{I_i}}$ and $u := \sum_{k \in \Gamma_i} U_k \phi_k.$ Let $v := \sum_{k \in \Gamma_i} V_k \phi_k.$ Let $V_l = (V_l)_{k \in \Gamma_i}$ be the finite element approximation of the solution of (17). Let $V_l = (V_l)_{k \in \Gamma_i}$, we have with obvious notations:

$$A_{\Gamma_i} V_l + A_{\Gamma_i} U = 0.$$  

A variational definition of the flux reads

$$\int_{\Gamma_i} \alpha \frac{\partial v}{\partial n} \phi_k = \int_{\Omega_i} \alpha \nabla v \cdot \nabla \phi_k$$

for all $\phi_k, k \in \Gamma_i.$ So the variational formulation of the eigenvalue problem (18) reads

$$\int_{\Gamma_i} \alpha \nabla v \cdot \nabla \phi_k = \lambda \int_{\Gamma_i} \operatorname{tr}(\alpha) v \phi_k$$

for all $\phi_k, k \in \Gamma_i$ and where $\operatorname{tr}(\alpha)$ is the restriction of $\alpha_{\Omega}$ to $\Gamma_i.$ Let $M_{\alpha,\Gamma_i}$ be the weighted mass matrix

$$(M_{\alpha,\Gamma_i})_{kl} := \int_{\Gamma_i} \operatorname{tr}(\alpha) \phi_k \phi_l, \quad \forall k, l \in \Gamma_i.$$  

The compact form of equation (21) is

$$A^{(i)}_{\Gamma_i} V_l + A_{\Gamma_i} U = \lambda M_{\alpha,\Gamma_i} U.$$  

With (22), the discrete form of (18) is a generalized eigenvalue problem

$$(A^{(i)}_{\Gamma_i} - A_{\Gamma_i} A^{-1}_{\Gamma_i} A_{\Gamma_i} V_l) = \lambda M_{\alpha,\Gamma_i} U.$$  

Let $(U, \lambda)$ be an eigenpair, we need its harmonic extension to the degrees of freedom of domain $\Omega_i$, that is the vector:

$$v := [V_l \quad A_{\Gamma_i}^{-1} U].$$  

Actually, there is more practical way to directly compute these eigenpairs. For subdomain $\Omega_i$, let

$$v := [V_l \quad A_{\Gamma_i}^{-1} U], \quad A^{(i)} := A_{\Gamma_i} A_{\Gamma_i}^{-1} A_{\Gamma_i}^{-1} A_{\Gamma_i}.$$
we compute the lowest eigenvalues of the sparse generalized eigenvalue problem:
\[
A^{(i)} v_{dn} = \lambda \begin{bmatrix} 0 & 0 \\ 0 & M_{E,i} \end{bmatrix} v_{dn}.
\]

This can be done using standard linear algebra library such as ARPACK.

The step by step procedure on how to construct the rectangular matrices \( W^i \) in the coarse space matrix \( Z \), see [16], is summed up in Algorithm 1. We call this procedure Algorithm 1 Construction of the spectral coarse space

In parallel for all subdomains \( 1 \leq i \leq N \),

1. Compute eigenpairs of \((V_1^i, \lambda_1^i), (V_2^i, \lambda_2^i), \ldots, (V_{m_i}^i, \lambda_{m_i}^i)\) such that
   \[
   \lambda_1^i \leq \ldots \leq \lambda_{m_i}^i < 1 / \text{diam}(\Omega_i) \leq \lambda_{m_i+1}^i \leq \ldots
   \]

2. Let \( Z \) defined as in [16] with for each \( 1 \leq i \leq N \), \( W^i \) the rectangular matrix with \( m_i \) columns defined by
   \[
   W^i = [ D_i V_1^i | \ldots | D_i V_{m_i}^i ].
   \]

3. Note \( R_0 := Z^T \) and compute the coarse matrix \( E \):
   \[
   E := R_0^T A R_0^T.
   \]

4. The two-level preconditioner is given by eq. (11):
   \[
   M_{ASM}^{-1} := R_0^T E^{-1} R_0 + \sum_{i=1}^{N} R_i^T ( R_i A R_i^T )^{-1} R_i
   \]

It has the \( Z_{DDN} \) method. We also use \( Z_{DDN} \) to denote the coarse space constructed by this method. Its construction is fully parallel. Similarly we call \( Z_{Nico} \) the method of Nicolaides or the corresponding coarse space. Let us remark that when the subdomain does not touch the boundary of \( \Omega \), the lowest eigenvalue of the DtN map is zero and the corresponding eigenvector is a constant vector. Thus, \( Z_{Nico} \) and \( Z_{DDN} \) coincide. As we shall see in the next section, when a subdomain has several jumps of the coefficient, taking \( Z_{Nico} \) is not efficient and it is necessary to take \( Z_{DDN} \) with more than one mode per subdomain.

This construction has been analyzed in [17]. We first recall a classical result. Let \( Z \) be a “Nicolaides type” coarse space
\[
Z := ( R_i^T D_i R_i 1 )_{1 \leq i \leq N}.
\]
We have, see [29]:

**Theorem 3.1** Let \( M_{ASM} \) be the two-level additive Schwarz method with the “Nicolaides” coarse space, we have for \( \alpha = 1 \) the following condition number estimate:

\[
\kappa(M_{ASM}^{-1} A) \leq C(1 + \frac{H}{\delta})
\]

where \( \delta \) is the size of the overlap between the subdomains and \( H \) the subdomain size and \( C \) does not depend on the number of subdomains.

But, for \( \alpha \) discontinuous, \( C \) would depend on the jumps of \( \alpha \).

Let \( Z \) be the coarse space built via Algorithm 1, we prove under technical assumptions on \( \alpha \)

**Theorem 3.2** Under the monotonicity of \( \alpha \) in the overlapping regions, we have the following condition number estimate:

\[
\kappa(M_{ASM}^{-1} A) \leq C(1 + \max_{1 \leq i \leq N} \frac{1}{\delta_i r_i})
\]

where \( \delta_i \) is the size of the overlap of domain \( \Omega_i \) and \( C \) is independent of the jumps of \( \alpha \) and of the number of subdomains.

Note that if \( \alpha = 1 \) and we take only one mode per subdomain \( (m_i = 1) \), we have for a regular interface \( \lambda_{m_i+1}^i \approx 1/H_i \) (see [12]) and we recover the “classical” estimate. Now in the general case, if the number of modes associated to subdomain \( \Omega_i \), \( m_i \) is chosen so that, \( \lambda_{m_i+1}^i \geq 1/H_i \), the convergence rate will be analogous to the constant coefficient case.

### 3.3 Comparison with a volumic spectral coarse space

The DtN spectral coarse space makes use of eigenvectors of the local Dirichlet to Neumann maps. There is thus a clear relationship with recent works by Galvis and Efendiev [9, 10, 13, 15] where the coarse space is based on eigenvalues of the volumic operator

\[
- \text{div}(\alpha \nabla u_i) = \lambda \alpha u_i \text{ in } \Omega_i.
\]

The drawback of their approach is that the coarse space is too large. This is easy to see in 1D. In Figure 7 we represent the function \( \alpha \) in a subdomain \( \Omega_i \). We have many discontinuities inside the domain. But, whatever the number of discontinuities is, our DtN map is a two by two matrix. The number of eigenvectors of the DtN map is two. Thus, the coarse space is made of two vectors per subdomain at most. But, the size of a volumic spectral coarse space is equal to the number of high heterogeneities islands. This phenomenon also holds in the 2D case. In Figure 8 we show permeability field with high heterogeneities islands. For this case, only 4 eigenvalues of the DtN map are smaller than 1.3e - 4 whereas the other ones are larger than 0.9. Whereas twenty...
volumic eigenvalues of eq. (24) are smaller than $3.8e-3$ and the others are larger than 150. Numerical tests show that in this case only four eigenvalues are enough for having an efficient coarse space. In the papers by Galvis and Efendiev, they noted this fact and they have a complex procedure to get rid of the useless eigenvectors. In our case, the method adapts automatically to the permeability field. In Figure 9, we show typical DtN and volumic eigenvectors.

In Table 2, we test robustness w.r.t the heterogeneities. The domain $\Omega$ contains layers with jumps in the coefficients ranging from 1 to $10^6$. We have 32 subdomains. The iteration counts depend weakly on the size of the jump in the coefficients. In Figure 10, we show the permeability field, domain decomposition (regular or METIS) into 16 subdomains and the solution corresponding to convergence curves of Figures 11 and 12. In Table 3, we show how many eigenvalues were selected in the coarse space. In Table 4, we vary the domain decomposition for the same permeability field.

We now present a selection of difficult test cases in a more systematic way, with so called inclusions and channels. We solve two test cases with known difficulties. The diffusion coefficient $\alpha$ is highly heterogeneous and takes values between 1 and approximately $2 \times 10^6$ and contains both high-permeability inclusions and channels. First of all we will analyze the performance of the method by increasing the number of channels and then by increasing the number of inclusions.

We use a uniform triangulation with $160 \times 160$ nodes and a partition into 16 (irregular) subdomains (see Figure 13). Each subdomain is extended by one layer, leading to an overlap of 2 layers, such that $\delta_j = 2h$ for all $j = 1, \ldots, N$. 

### Table 2

<table>
<thead>
<tr>
<th>Jumps in coeff</th>
<th>1</th>
<th>$10^1$</th>
<th>$10^2$</th>
<th>$10^3$</th>
<th>$10^4$</th>
<th>$10^5$</th>
<th>$10^6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iteration counts</td>
<td>15</td>
<td>24</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>11</td>
<td>11</td>
</tr>
</tbody>
</table>

### Table 3

<table>
<thead>
<tr>
<th>Coefficient range</th>
<th>1</th>
<th>$10^1$</th>
<th>$10^2$</th>
<th>$10^3$</th>
<th>$10^4$</th>
<th>$10^5$</th>
<th>$10^6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iteration counts</td>
<td>15</td>
<td>24</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>11</td>
<td>11</td>
</tr>
</tbody>
</table>
Channels and inclusions: $1 \leq \alpha \leq 1.5 \cdot 10^6$. Top left: permeability field, top right: the solution, bottom left: regular partition and permeability field, bottom right: Metis partition and permeability field.

<table>
<thead>
<tr>
<th>Subdomain $i$</th>
<th>$nsmeig(i)$</th>
<th>Total number of eigenval.($i$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>135</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>109</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>175</td>
</tr>
<tr>
<td>10</td>
<td>4</td>
<td>174</td>
</tr>
<tr>
<td>11</td>
<td>2</td>
<td>71</td>
</tr>
<tr>
<td>12</td>
<td>2</td>
<td>128</td>
</tr>
<tr>
<td>13</td>
<td>3</td>
<td>166</td>
</tr>
<tr>
<td>14</td>
<td>3</td>
<td>127</td>
</tr>
<tr>
<td>15</td>
<td>3</td>
<td>188</td>
</tr>
<tr>
<td>16</td>
<td>3</td>
<td>106</td>
</tr>
</tbody>
</table>

**TABLE 3**
Number of small eigenvalues ($nsmeig(i)$) satisfying criterion (19) for subdomain $i$ – Metis 4 by 4 decomposition

<table>
<thead>
<tr>
<th>Subdomain</th>
<th>ASM</th>
<th>Nico</th>
<th>DtN</th>
<th>RAS</th>
<th>Nico</th>
<th>DtN</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 x 2</td>
<td>103</td>
<td>110</td>
<td>22</td>
<td>70</td>
<td>70</td>
<td>14</td>
</tr>
<tr>
<td>2 x 2 Metis</td>
<td>76</td>
<td>76</td>
<td>22</td>
<td>57</td>
<td>57</td>
<td>18</td>
</tr>
<tr>
<td>4 x 4</td>
<td>603</td>
<td>722</td>
<td>26</td>
<td>169</td>
<td>165</td>
<td>15</td>
</tr>
<tr>
<td>4 x 4 Metis</td>
<td>483</td>
<td>425</td>
<td>36</td>
<td>148</td>
<td>142</td>
<td>23</td>
</tr>
<tr>
<td>8 x 8</td>
<td>461</td>
<td>141</td>
<td>34</td>
<td>205</td>
<td>95</td>
<td>21</td>
</tr>
<tr>
<td>8 x 8 Metis</td>
<td>600</td>
<td>542</td>
<td>31</td>
<td>240</td>
<td>196</td>
<td>19</td>
</tr>
</tbody>
</table>

**TABLE 4**
Convergence results for the test case of Figure 10

**Figure 11**
RAS convergence for channels and inclusions – Regular partitioning

**Figure 12**
RAS convergence for channels and inclusions – Regular decomposition – Metis partitioning

**Figure 13**
Subdomain partitioning into 16 subdomains using METIS.
We use the ASM preconditioner within conjugate gradients (CG) and the RAS preconditioner within GMRES, and in each case we stop the iteration process, when the relative residual is smaller than $10^{-6}$.

![Figure 14](image)

Test Problem 1: Successively adding channels.

<table>
<thead>
<tr>
<th></th>
<th>ASM</th>
<th>+Nico.</th>
<th>+DtN</th>
<th>RAS</th>
<th>+Nico.</th>
<th>+DtN</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 ch.</td>
<td>529</td>
<td>1000</td>
<td>57</td>
<td>243</td>
<td>245</td>
<td>41</td>
</tr>
<tr>
<td>1 ch.</td>
<td>619</td>
<td>520</td>
<td>64</td>
<td>227</td>
<td>228</td>
<td>46</td>
</tr>
<tr>
<td>2 ch. &gt; 1000</td>
<td>516</td>
<td>68</td>
<td>226</td>
<td>226</td>
<td>47</td>
<td></td>
</tr>
<tr>
<td>3 ch.</td>
<td>585</td>
<td>697</td>
<td>76</td>
<td>212</td>
<td>213</td>
<td>44</td>
</tr>
</tbody>
</table>

**TABLE 5**
Number of iterations for Test Problem 1 (additive coarse grid correction).

We start with only inclusions and add the channels one by one as shown in Figure 14 (Test Problem 1). When there are no channels, $\alpha$ varies between 1 and $10^6$, as indicated by the colors in Figure 14. With all three channels present, $\alpha$ varies between 1 and $2.8 \times 10^6$. The corresponding convergence results are given in Table 5. Our algorithm performs significantly better. The piecewise constant coarse space has virtually no effect on the performance of either ASM or RAS, leading to iteration numbers that differ little from the results without any coarse grid in all four cases. Our new coarse space, on the other hand, is fully robust to the coefficient variation and to the addition of channels, and it leads to a gain of at least a factor 8 compared to the one-level method in all cases. The situation is similar, if we use deflation-based coarse grid correction [21] with the same coarse spaces (see Table 6). However, the absolute numbers of iterations are reduced almost by a factor 2 in this case. Our theory applies equally to this case (see e.g. [16] for details), but we will not include any further numerical results with deflation-based coarse grid correction.

Table 7 gives some information on the size of the coarse space that we build with our automatic selection strategy: for each number of channels we give min $m_j$, max $m_j$, as well as the global coarse space size $n_H = \sum_j m_j$ and the average number of modes included per subdomain $n_H/N$. For comparison, we also include information on the total number $n_{1-H}$ of eigenmodes of the discrete DtN operator on each subdomain. We note that adding channels does not have a big influence on the size of the coarse space; we only need three additional eigenvectors in the case of three channels compared to the case of no channels.

<table>
<thead>
<tr>
<th></th>
<th>ASM</th>
<th>+Nico.</th>
<th>+DtN</th>
<th>RAS</th>
<th>+Nico.</th>
<th>+DtN</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 ch.</td>
<td>529</td>
<td>656</td>
<td>39</td>
<td>243</td>
<td>231</td>
<td>25</td>
</tr>
<tr>
<td>1 ch.</td>
<td>619</td>
<td>538</td>
<td>41</td>
<td>227</td>
<td>215</td>
<td>28</td>
</tr>
<tr>
<td>2 ch. &gt; 1000</td>
<td>808</td>
<td>47</td>
<td>226</td>
<td>211</td>
<td>27</td>
<td></td>
</tr>
<tr>
<td>3 ch.</td>
<td>585</td>
<td>641</td>
<td>47</td>
<td>212</td>
<td>199</td>
<td>28</td>
</tr>
</tbody>
</table>

**TABLE 6**
Number of iterations for Test Problem 1 (deflation-based coarse grid correction).

Then, using the same domain and the same partition we successively add inclusions without any channels present as shown in Figure 15 (Test Problem 2). The results are in Table 8. Again, the piecewise constant coarse space is not working at all for this test problem. The DtN-based coarse space is almost completely robust to an increase in the number of inclusions and requires again significantly less iterations than the one-level method in all cases. Note that the subdomain partition (cf. Figure 15) is not aligned with the inclusions at all (cf. Figure 15). In Table 9 we see that also in this test problem, the coarse space size grows only very slowly with the number of inclusions and even in the hardest case $n_H$ is only 53 (cf. the dimension $n$ of $V_{h,0}$, and thus of $A$ is 25600).

### 3.4.1 Practical optimality of the spectral coarse space

The last series of tests, in Table 10 aims to prove that the number $m_j$ of eigenvectors per subdomain chosen by our au-
Automatic algorithm is indeed optimal in some sense. For Test Problem 1 with one channel (see Figure 14), we first reduce the number of coarse basis functions per subdomain by one, this has a huge influence on the iteration count. Then we add one basis function per subdomain and notice that this has much less effect. This suggests that the selection process we have designed is indeed the best compromise between enriching the coarse grid and solving a reasonably sized coarse problem.

<table>
<thead>
<tr>
<th># incl.</th>
<th>ASM</th>
<th>+Nico</th>
<th>+DtN</th>
<th>RAS</th>
<th>+Nico</th>
<th>+DtN</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 × 2</td>
<td>108</td>
<td>80</td>
<td>51</td>
<td>100</td>
<td>81</td>
<td>41</td>
</tr>
<tr>
<td>3 × 3</td>
<td>194</td>
<td>342</td>
<td>58</td>
<td>154</td>
<td>135</td>
<td>46</td>
</tr>
<tr>
<td>5 × 5</td>
<td>529</td>
<td>no cv.</td>
<td>57</td>
<td>243</td>
<td>245</td>
<td>41</td>
</tr>
<tr>
<td>6 × 6</td>
<td>835</td>
<td>823</td>
<td>71</td>
<td>266</td>
<td>267</td>
<td>51</td>
</tr>
</tbody>
</table>

TABLE 8
Number of iterations for Test Problem 2 (additive coarse grid correction) vs. number of inclusions.

<table>
<thead>
<tr>
<th>Over N = 16 subdomains</th>
<th># eigen. on Γ₀</th>
<th># local coarse space modes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0 ch.</td>
<td>1 ch.</td>
</tr>
<tr>
<td>Minimum</td>
<td>70</td>
<td>1</td>
</tr>
<tr>
<td>Maximum</td>
<td>191</td>
<td>3</td>
</tr>
<tr>
<td>Average</td>
<td>138.8</td>
<td>1.6</td>
</tr>
<tr>
<td>Sum</td>
<td>2220</td>
<td>26</td>
</tr>
</tbody>
</table>

TABLE 9
Size of the coarse space for Test Problem 2.

4 ADAPTIVE COARSE SPACE ON HPC PLATFORMS

Results in this section are based on a related method to the DtN coarse space method namely the Geneo method. The principle of this coarse space construction is similar in that the coarse space is built after solving local eigenvalues problems. It suffices to change the right hand side in the generalized eigenvalue problem \( (23) \). The new eigenvalue problem is of the form

\[
A^{(i)} v_{dn} = \lambda D_i R_i A R_i^T D_i v_{dn},
\]

see [28] for more details. The Geneo coarse space is in practice quite close to the DtN coarse space. Its main advantage is to work not only for scalar PDEs but also for systems of PDEs as the elasticity system for instance. When applied to scalar PDEs, DtN and Geneo coarse spaces are almost identical and give very similar results. As a result, in order to have a general purpose code, we focused in HPC developments and tests on the Geneo method. Results in this section were obtained on Curie, a Tier-0 system for PRACE2 (Partnership for Advanced Computing in Europe) composed of 5040 nodes made of 2 eight-core Intel Sandy Bridge processors clocked at 2.7 GHz. The interconnect is an InfiniBand QDR full fat tree. We want here to assess the capability of our framework to scale:

1. strongly: for a given global mesh, the number of subdomains increases while local mesh sizes are kept constant (i.e. local problems get smaller and smaller),
2. weakly: for a given global mesh, the number of subdomains increases while local mesh sizes are refined (i.e. local problems have a constant size).

We don’t time the generation of the mesh and partition of unity. Assembly and factorization of the local stiffness matrices, resolution of the generalized eigenvalue problems, construction of the coarse operator and time elapsed for the convergence of the Krylov method are the important procedures here. The Krylov method used is the GMRES, it is stopped when the relative residual error is inferior to $\varepsilon = 10^{-8}$ in 2D, and $10^{-9}$ in 3D. All the following results where obtained using a $LDL^T$ factorization of the local solvers $A_i^T$ and the coarse operator $E$ using MUMPS (with a MPI communicator set to respectively MPI_COMM_SELF or the communicator created by our library binding master processes).

First, the system of linear elasticity with highly heterogeneous elastic moduli is solved with a minimal geometric overlap of one mesh element. Its variational formulation reads:

$$
\int_\Omega \nabla \cdot \mu \nabla \cdot v + 2\mu \varepsilon(u)^T \varepsilon(v) + \int_\Omega f \cdot v + \int_\partial \Omega g \cdot v \quad (26)
$$

where

- $\lambda$ and $\mu$ are the Lamé parameters such that $\mu = \frac{E}{2(1 + \nu)}$ and $\lambda = \frac{E\nu}{(1 + \nu)(1 - 2\nu)}$ ($E$ being Young’s modulus and $\nu$ Poisson’s ratio). They are chosen to vary between two set of values, $(E_1, \nu_1) = (2 \cdot 10^{11}, 0.25)$, and $(E_2, \nu_2) = (10^6, 0.4)$.
- $\varepsilon$ is the linearized strain tensor and $f$ the volumetric forces (here, we just consider gravity).

Because of the overlap and the duplication of unknowns, increasing the number of subdomains means that the number of unknowns increases also, even though the number of mesh elements (triangles or tetrahedra in the case of FreeFem++) is the same. In 2D, we use piecewise cubic basis functions on an unstructured global mesh made of 110 million elements, and in 3D, piecewise quadratic basis functions on an unstructured global mesh made of 20 million elements. This yields a symmetric system of roughly 1 billion unknowns in 2D and 80 million unknowns in 3D. The geometry is a simple $[0; 1]^d \times [0; 10]$ beam ($d = 1$ or 2) partitioned with METIS.

Solving the 2D problem initially on 1 024 processes takes 227 seconds, on 8192 processes, this time is reduced to 31 seconds (superoptimal speedup). With that many subdomains, the coarse operator $E$ is of size $121\,935 \times 121\,935$. It is assembled and factorized in 7 seconds by 16 master processes.

The targeted number of unknowns per subdomains is kept constant at approximately 800 thousands in 2D, and 120 thousands in 3D (once again with $P_3$ and $P_2$ finite elements respectively).

Then, the coarse operator is of size $92\,160 \times 92\,160$ and is assembled and factorized by 16 master processes in 11 seconds.

Moving on to the weak scaling properties of our framework, the problem we now solve is a scalar equation of diffusivity with highly heterogeneous coefficients (varying from $1$ to $10^5$) on $[0; 1]^d$ ($d = 2$ or $3$). Its variational formulation reads:

$$
\int_\Omega \alpha \nabla u \cdot \nabla v + \int_\Omega f \cdot v \quad (27)
$$

The number of unknowns per subdomains is kept constant at approximately 800 thousands in 2D, and 120 thousands in 3D (once again with $P_3$ and $P_2$ finite elements respectively).

In 2D, the initial extended system (with the duplication of unknowns) is made of 800 million unknowns and is solved in 141 seconds. Scaling up to $12\,288$ processes yields a system of 10 billion unknowns solved in 172 seconds, hence an efficiency of $\frac{141}{172} \approx 82\%$. In 3D, the initial system is made of 130 million unknowns and is solved in 127 seconds. Scaling up to $8192$ processes yields a system of 1 billion unknowns solved in 152 seconds, hence an efficiency of $\frac{127}{152} \approx 83\%$.

5 CONNECTIONS WITH MULTISCALE METHODS

Multiscale methods are an active field of research, for finite element methods see [11] (and references therein) and for multiscale finite volume methods see for example [18].
In §5 we compare them with our two-level spectral coarse space. In §5.1 we first recall basic facts on multiscale discretizations and their difficulties with arbitrary channelized flows, §5.1.1. Although the goals of multiscale methods and DD methods are different, they have many related features that we compare in §5.2. In particular, both methods build coarse basis functions. The superiority of the spectral coarse space comes from the fact that the number and the shape of basis functions adapts automatically to the heterogeneities of the medium even for channelized media. This is not always the case for multiscale methods.

5.1 Presentation of multiscale methods

![Multiscale basis functions](Image)

Consider a problem set on a fine grid $\Omega_h$ (see Figure 18)

$$L_h(u_h) = f_h \text{ in } \Omega_h$$

(28)

that is too large to be solved. We approximate $u_h$ via a coarse problem set on a coarse mesh $\Omega_H$. Defining a multiscale methods involve three steps:

- pre-computation of a multiscale basis functions;
- global formulation at the coarse level;
- reconstruction of a fine scale solution.

There are of course many variants to deal with these topics and we don’t try to give a completer review on the subject. We present here basic materials in order to compare multiscale methods with our DtN coarse space. In particular, we shall see that the DtN approach is more general and systematic.

5.1.1 Multiscale basis functions

The preferred and most common technique is to use oversampling, see [11]. For simplicity, we start with the original non oversampling approach.

We consider a structured two-dimensional grid. A coarse element is typically denoted by $K$. Let $(x_i, y_i)$ be a coarse grid vertex. We recall the construction of the corresponding coarse basis function $\phi_{H,i,j}$. For both Multiscale Finite Element Method (MsFEM) and Multiscale Finite Volume (MsFV), a standard choice is to solve the fine scale equation on the four neighboring coarse elements $K_{i+1/2,j+1/2}$, see Figure 18

$$L_h(\phi_{i+1/2,j+1/2}) = 0 \text{ in } K_{i+1/2,j+1/2}$$

$$\phi_{i+1/2,j+1/2} = g_{i+1/2,j+1/2} \text{ on } \partial K_{i+1/2,j+1/2}$$

(29)

where $g_{i+1/2,j+1/2}$ is a piecewise affine function such that $g_{i+1/2,j+1/2}(x,y) = 1$ and is zero on the three other vertices of $\partial K_{i+1/2,j+1/2}$. Then, function $\phi_{H,i,j}$ is defined by taking restrictions of $\phi_{i+1/2,j+1/2}$ to the coarse elements adjacent to the coarse grid vertex $(x_i, y_i)$:

$$\phi_{H,i,j}(x,y) = \begin{cases} 
\phi_{i+1/2,j+1/2}(x,y) & \text{if } (x,y) \in K_{i+1/2,j+1/2}, \\
0 & \text{otherwise .} 
\end{cases}$$

(30)

This construction presents unwanted boundary layers effects. In order to fix this problem, functions $\phi_{i+1/2,j+1/2}$ are computed on a coarse cell $K^d_{i+1/2,j+1/2}$ enlarged with a few layers of fine elements, see Figure 18. Then a coarse basis function $\phi_{H,i,j}$ is computed as a linear combination of the restrictions of functions $\phi_{i+1/2,j+1/2}$ to $K_{i+1/2,j+1/2}$. This leads to a non conformal basis. When the coefficients of the operator $L_h$ are sufficiently smooth, this basis is adequate. This procedure is called oversampling.

When the coefficients are heterogeneous across these edges (left picture of Figure 19) the basis functions should see the heterogeneities. For this purpose, the piecewise linear Dirichlet boundary conditions are replaced by oscillatory boundary condition obtained by solving a reduced elliptic problem along the boundary of the coarse cell. The Dirichlet data must be in the kernel of the tangential part of the partial differential operator in eq. (29). An algebraic implementation of this construction was proposed in [25] and [31]. Note that for finite volume schemes for problems with high anisotropies, the cell problems [25] can also be modified by replacing Dirichlet boundary conditions (BC) by Neumann BC on some parts of $\partial K_{i+1/2,j+1/2}$, see § 6.3. of [18].

5.1.1.1 Multiscale basis functions for channelized permeability distributions

When the problem has strong heterogeneities, typically, three situations occur as shown in Figure 19.

1. Isolated heterogeneities

2. One heterogeneous channel
3. Several heterogeneous channels

The first case is well treated by multiscale methods as recalled above. For the second case, it has been noticed that it might not be sufficient: “It has been shown that the accuracy of purely local methods may be low if the permeability field has structures with very long correlation lengths” quoted from [18]. This is the case for instance with channels, see for example middle and right pictures of Figure 19. In order to fix this problem, iterative constructions of the coarse basis functions have been proposed, see [5] [6] [8]. Iterations take place between the coarse scale global flow and the fine scale local flow. A coarse space is first built with the oversampling method. It is used to obtain a coarse and then a fine grid solution. Then, the coarse basis functions are corrected by taking the coarse edge values of this solution as Dirichlet boundary conditions in equation (29). This procedure stops with some convergence criterion. To our knowledge, this technique is not supported by theoretical approximation results. The last case with several heterogeneous channels seems to be a concern even for this approach. Indeed, in this case the good coarse space function depend on the flow conditions: “The introduction of wells may additionally change global flow significantly and the coarse properties generated from the two generic global flows might lose accuracy in some cases. For such problems, the T can be recomputed, based on the actual well configuration and flow rates, using a local-global procedure analogous to that applied here. The overall issue of robustness with respect to global boundary conditions is complex and will be addressed in detail in a future paper.” quoted from [5]. The problem comes from the fact that the number of coarse basis functions attached to the cell should be at least equal to the number of channels crossing the cell, see [30]. But, in multiscale methods even in the more algebraic ones as [31], the number of degree of freedom per aggregate is prescribed in advance. For a scalar problem, only one coarse basis function is assigned to a coarse grid vertex. It is thus not possible to cover all possible flow configurations.

5.1.2 Coarse problem

This step consists in approximating the fine scale solution \( u_h \) by defining a suitable coarse space problem whose solution, denoted by \( u_H \), belongs to the space spanned by the coarse basis functions \( (\phi_{H,i,j})_i \). We consider first finite element formulation and then finite volume approximations.

For a finite element method, a Galerkin approach is usually used. For all \( i, j \), let us denote by \( Z_{i,j} \) the vector of the components of \( \phi_{H,i,j} \) on the basis of the fine FEM. We collect all these vectors in a rectangular matrix \( Z \). Let \( A_H \) denote the matrix associated to the fine FEM so that the matrix form of the fine FEM reads:

\[
A_h U_h = F_h
\]

where \( U_h \) are the components of the solution \( u_h \) on the fine FEM basis. Let us define \( A_H := Z^T A_h Z \) and the coarse problem by:

Find \( u_H := \sum_{i,j} U_{H,i,j} \phi_{H,i,j} \) such that

\[
A_H U_H = Z^T F_h.
\]

This way, the coarse approximation \( U_H \) satisfies a variational formulation in the coarse space spanned by the coarse basis functions \( \phi_{H,i,j} \).

For finite volume methods, the Galerkin approach can be used as well. But then, conservativity and monotonicity of the initial finite volume scheme are lost. In order to recover them, a dual coarse mesh is introduced, see Figure 18. The coarse grid problem consists in finding \( u_H := \sum_{i,j} U_{H,i,j} \phi_{H,i,j} \) such that conservativity is satisfied on the boundaries of the dual cells. Typically, a 9-point stencil is thus obtained and for anisotropic problems the monotonicity of the finite volume scheme on the fine mesh is lost on the coarse problem. Then a modified 7-point stencil is sought that still ensures conservativity, see [18].

5.1.3 Fine scale solution

This step is actually optional since \( U_H \) contains fine scale information via the coarse basis functions \( \phi_{H,i,j} \). In MsFEM, one can further improve the solution by solving local Dirichlet boundary value problems in each coarse element \( K \in \{1/2,j=1/2\} \):

\[
\mathcal{L}_h(u_h) = f_h \text{ in } K \in \{1/2,j=1/2\} \text{ and } u_h = U_H \text{ on } \partial K \in \{1/2,j=1/2\}.
\]

Thus \( U_H \) could be used in principle to solve for instance a transport equation at the fine level. But the method to compute \( U_H \) is not conservative which is then a big drawback. In multiscale finite volume methods, the reconstruction is based on solving Neumann problems in each coarse cell so that local conservativity is satisfied. As a result, the fine scale solution is not continuous at the edges of the coarse elements.

5.2 Comparison with the DtN two-level Schwarz method

The foremost difference between multiscale methods and two-level domain decomposition methods is the goal
In multiscale methods, they mimic finite element basis functions and they are harmonic in the aggregates. But, coarse basis functions are used to define a coarse problem.

5.2.2 Local and global effects

For both methods the fine mesh is decomposed into aggregates of fine elements. But,

- in MsFEM or MsFV, the aggregates consist of some dozens of elements
- whereas in DDM, subdomains may be quite large
  the construction of the coarse problem is essentially parallel. But,
  - in multiscale methods, we have a fine grain parallelism
  - in DDM, we have a coarse grain parallelism

Oversampling is very reminiscent of overlapping in DDM. In both approaches, coarse basis functions are harmonic functions in overlapping aggregates (subdomains in DDM and extended coarse cells in oversampling multiscale methods). In order to use them in a coarse problem, they have to be cast to functions defined in the whole domain \( \Omega \). In multiscale methods this was done via procedure which is somehow “brutal” since the resulting function is not even continuous on \( \Omega \). In DDM methods, the local coarse space functions are multiplied by a kind of partition of unity (the local matrices \( (D_{ij})_{j \in \mathcal{N}} \), see formula (12)) before the extension by zero in the whole domain \( \Omega \) so that the resulting function is continuous on \( \Omega \). In [10], the authors use partition of unity functions in MsFEM methods.

5.2.2 Local and global effects

Coarse basis functions are used to define a coarse problem and they are harmonic in the aggregates. But,

- in multiscale methods, they mimic finite element basis function: only one such function per aggregate.
- In the spectral coarse space of § 3.2 the number of coarse basis functions per aggregate is not prescribed a priori.

Moreover, the involved tools have some similarities.

5.2.1 Oversampling and overlapping

For both methods the fine mesh is decomposed into aggregates of fine elements. But,

- in MsFEM or MsFV, the aggregates consist of some dozens of elements
- whereas in DDM, subdomains may be quite large
  the construction of the coarse problem is essentially parallel. But,
  - in multiscale methods, we have a fine grain parallelism
  - in DDM, we have a coarse grain parallelism

Oversampling is very reminiscent of overlapping in DDM. In both approaches, coarse basis functions are harmonic functions in overlapping aggregates (subdomains in DDM and extended coarse cells in oversampling multiscale methods). In order to use them in a coarse problem, they have to be cast to functions defined in the whole domain \( \Omega \). In multiscale methods this was done via procedure which is somehow “brutal” since the resulting function is not even continuous on \( \Omega \). In DDM methods, the local coarse space functions are multiplied by a kind of partition of unity (the local matrices \( (D_{ij})_{j \in \mathcal{N}} \), see formula (12)) before the extension by zero in the whole domain \( \Omega \) so that the resulting function is continuous on \( \Omega \). In [10], the authors use partition of unity functions in MsFEM methods.

6 CONCLUSION AND PROSPECTS

After having introduced Schwarz domain decomposition methods, we have presented the spectral coarse space introduced in [23] and later analyzed in [7]. It is practically optimal in the sense that a larger coarse space does not bring much improvement while a smaller one has a poor performance, see § 3.4.1. Moreover, the method adapts automatically to the heterogeneities of the problem. If necessary, more than one coarse basis function is allowed per aggregate. This construction is supported by a theoretical condition number estimate independent of the heterogeneities of the physical problem, see Theorem 3.2. In coarse spaces built using multiscale methods, such a theorem cannot hold since only one degree of freedom is allowed per aggregate, see § 5.2. This is why these methods have problems with channelized permeability distributions. A cure proposed in [10] is to use a suitable spectral coarse space as a basis for a MsFEM method. Our DtN coarse space could be used in MsFEM methods in the same manner.

The spectral coarse space was developed, tested and analyzed in the finite element framework. In reservoir or basin simulations, finite volume discretizations are usually preferred to finite element discretizations. The extension of the spectral coarse space of § 3 to a finite volume discretization is thus mandatory for its use in subsurface modeling. As
explained in §3.2, the rationale behind this coarse space is written in terms of the original model i.e. in terms of partial differential equations. Thus the basis of the method does not depend on the discretization scheme. Therefore the definition and implementation of the spectral coarse space in a finite volume discretization will demand some work but can definitely be done. It would improve the method introduced in §3.1 by selecting in a sure (see Theorem 3.2) and optimal (see §3.4.1) manner more efficient coarse spaces when the channelized character of the permeability distribution makes it necessary.

REFERENCES


