



**HAL**  
open science

# Toward robust, fast solutions of elliptic equations on complex domains through HHO discretizations and non-nested multigrid methods

Pierre Matalon, Daniele Di Pietro, Frank Hülsemann, Paul Mycek, Ulrich Råde, Daniel Ruiz

## ► To cite this version:

Pierre Matalon, Daniele Di Pietro, Frank Hülsemann, Paul Mycek, Ulrich Råde, et al.. Toward robust, fast solutions of elliptic equations on complex domains through HHO discretizations and non-nested multigrid methods. 2021. hal-03163476v1

**HAL Id: hal-03163476**

**<https://hal.science/hal-03163476v1>**

Preprint submitted on 9 Mar 2021 (v1), last revised 31 Jul 2021 (v2)

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

## RESEARCH ARTICLE

# Toward robust, fast solutions of elliptic equations on complex domains through HHO discretizations and non-nested multigrid methods

D. A. Di Pietro<sup>1</sup> | F. Hülsemann<sup>2</sup> | P. Matalon<sup>\*1,3,4,5</sup> | P. Mycek<sup>3</sup> | U. Råde<sup>3,4</sup> | D. Ruiz<sup>5</sup><sup>1</sup>IMAG, University of Montpellier, France<sup>2</sup>EDF R&D, Paris-Saclay, France<sup>3</sup>CERFACS, Toulouse, France<sup>4</sup>FAU Erlangen-Nürnberg, Germany<sup>5</sup>IRIT, Toulouse, France**Correspondence**

\*Pierre Matalon. Email: matalon@cerfacs.fr

**Summary**

The use of modern discretization technologies such as Hybrid High-Order (HHO) methods, coupled with appropriate linear solvers, allow for the robust and fast solution of Partial Differential Equations (PDEs). Although efficient linear solvers have recently been made available for simpler cases, complex geometries remain a challenge for large scale problems. To address this problem, we propose in this work a geometric multigrid algorithm for unstructured non-nested meshes. The non-nestedness is handled in the prolongation operator through the use of the  $L^2$ -orthogonal projection from the coarse elements onto the fine ones. However, as the exact evaluation of this projection can be computationally expensive, we develop a cheaper approximate implementation that globally preserves the approximation properties of the  $L^2$ -orthogonal projection. Additionally, as the multigrid method requires not only the coarsening of the elements, but also that of the faces, we leverage the geometric flexibility of polytopal elements to define an abstract non-nested coarsening strategy based on element agglomeration and face collapsing. Finally, the multigrid method is tested on homogeneous and heterogeneous diffusion problems in two and three space dimensions. The solver exhibits near-perfect asymptotic optimality for moderate degrees of approximation.

**KEYWORDS:**Partial Differential Equations, Hybrid High-Order, Multigrid, non-nested meshes,  $L^2$ -orthogonal projection

## 1 | INTRODUCTION

Hybrid High-Order (HHO) discretizations<sup>1</sup> have gained growing interest in recent years for the solution of Partial Differential Equations (PDEs). Amongst their seminal features, we can list their application to general polytopal meshes, their yielded approximate as a broken (i.e. piecewise) polynomial of any arbitrary degree, and their optimal orders of convergence. Especially, the flexibility of polytopal methods with respect to the nature and characteristics of the mesh makes HHO a tool of choice to tackle complex geometries: instead of resorting to local refinement to capture a complex region of the mesh, thereby adding a possibly large number of small elements — and thus increasing the computational cost of the solution —, introducing a small number of polyhedral elements can achieve the same geometric accuracy without sensibly altering the mesh granularity. For

similar structural reasons, HHO natively handles hanging nodes and, more generally, non-conforming meshes, by viewing usual element shapes, like tetrahedra or hexahedra, as identically shaped polyhedra with additional vertices. Another built-in and defining feature of the HHO methods is the use, in the formulation of the bilinear form, of a higher-order potential reconstruction operator, which allows the gain of one additional order of approximation compared to similar hybrid methods, like Hybridizable Discontinuous Galerkin (HDG).<sup>2,3</sup> Finally, the capability of HHO methods to adapt their design to the underlying physics allows for more robust solutions with respect to the problem. Up to this day, HHO methods have been successfully derived for a large variety of problems in fluid dynamics (heterogeneous anisotropic diffusion,<sup>4</sup> incompressible Navier-Stokes,<sup>5</sup> creeping flows of non-Newtonian fluids<sup>6</sup>) and structural mechanics (linear and nonlinear elasticity<sup>7,8</sup> and poroelasticity<sup>9,10</sup>).

HHO methods hinge on degrees of freedom (DoFs) located in elements and on faces, which can be globally viewed as broken polynomials respectively on the mesh and its skeleton. The element-defined DoFs being only locally coupled, they can be expressed, element by element, in function of the DoFs on the faces, and subsequently eliminated from the global HHO linear system. This gives rise to a Schur complement of smaller size where only face unknowns remain. This process is known as *static condensation* in the mechanical literature, and the resulting system as a *statically condensed* system, or *trace* system, in reference to the mesh skeleton as the support for the set of globally coupled unknowns. The solution of the trace system, yielding the face unknowns, remains the costliest operation, after which the values of the element unknowns can be inexpensively recovered by solving small independent linear systems.

As a consequence, the practical usefulness of HHO discretizations in an industrial context, where reaching very high scales of resolution is critical, depends on the existence of efficient linear solvers for the condensed system. In this paper, we focus on scalar elliptic equations, whose HHO discretizations give rise to trace matrices that are sparse, symmetric and positive-definite. We endeavour to solve large systems of this type by means of a multigrid method. The main difficulty in the design of a geometric multigrid algorithm for a trace system resides in the location of the DoFs associated to the set of unknowns that remain after static condensation, namely, the face unknowns. Supported by the mesh skeleton, the broken polynomials defined by these DoFs are not fitted for standard intergrid transfer operators, applicable to *element*-defined functions. The classical multigrid methods that are widely used with standard Finite Elements methods (or any discretization whose DoFs are element-based) are therefore excluded. A literature review of trace system solvers shows the variety of paths one can follow to tackle this particular setting. The first such multigrid algorithm<sup>11</sup> aims at converting the face-defined polynomials to element-defined ones via the adjoint of the trace operator in order to make use of a known standard geometric multigrid. A variant<sup>12</sup> using this time an algebraic multigrid instead of a geometric one was also experimented. Later, a different approach<sup>13</sup> is considered through the design of an *hp*-multigrid algorithm for unstructured meshes based, for the first time, on trace functions at every level, where the intergrid operators depend on Dirichlet-to-Neumann maps to preserve energy between levels. Finally, also conserving face-defined polynomials at every level, the latest multigrid algorithm,<sup>14</sup> developed by the present authors and which this work follows up on, is based on a prolongation operator that internally reverses the static condensation to recover coarse element-defined polynomials, before computing their traces on the fine faces. Prior to detailing this solver, we also point out the efforts made to design *p*-multigrid preconditioners for non-elliptic equations,<sup>15,16,17</sup> as well as other techniques such as domain decomposition<sup>18,19</sup> and nested dissection.<sup>20</sup>

Although applicable to other hybrid methods with static condensation, the solver developed in our previous work<sup>14</sup> is, to this day, the only multigrid method specifically targeting HHO discretizations of elliptic problems, by taking advantage of the higher-order potential reconstruction operator to improve its convergence rate. Additionally, rather than managing higher degrees of approximations by the mean of a *p*-multigrid on top of an *h*-one, high-order polynomials are preserved at every level at the cost of a block smoother relaxing all the unknowns related to the same polynomial. Used as a solver, a near-perfect *h*-independent convergence rate is exhibited on structured 2D and 3D meshes, as well as robustness with respect to the polynomial degree. However, on complex 3D geometries requiring highly unstructured meshes, the average performance may degrade rapidly with the problem size. Indeed, the algorithm requires a hierarchy of nested meshes (although it is experimentally shown that slightly non-nested meshes do not affect the global performance), and such hierarchies are usually obtained from successive refinements of an initial coarse mesh. This way of building embedded grids has a number of known limitations due to the loss of control over the resulting fine grid: firstly, part of the flexibility offered by unstructured meshes to accurately approximate complex regions may be lost in the implicit geometrical conservation in the fine mesh of the large coarse elements, which imposes that the initial coarse mesh already be a good approximation of the geometry. Having stated this, provided an efficient coarse solver, starting from an initial coarse grid that is nonetheless fine enough to adequately capture the geometry can still offer possibilities of very efficient multigrid implementations, such as Hierarchical Hybrid Grids.<sup>21,22</sup> And to help in this approach, recent techniques<sup>23</sup> based on polynomial mappings of the elements allow to better approximate curved boundaries upon each refinement. Secondly,

in 3D, it is well-known<sup>24,25,16,26</sup> that most methods of tetrahedral subdivision generate elements of degraded shape quality, which most often affects the accuracy of the approximate solution as well as the performance of the linear solvers. The nested multigrid method<sup>14</sup> indeed shows high sensitivity to the mesh regularity and therefore poor convergence on unstructured meshes obtained by tetrahedral refinement. Exploring non-nested grids as an alternative to hierarchies produced by mesh refinement is the purpose of the present work. To that end, and before describing our adaptation of the nested algorithm<sup>14</sup> to non-nested grids, we state that existing non-nested multigrid methods have been devised for the same motives as ours, although none of them applicable to trace systems of hybrid discretizations: continuous Finite Element Methods (FEM) are mostly targeted<sup>27,28,29,30,31</sup> and a recent one has been designed for Discontinuous Galerkin (DG) methods,<sup>32</sup> closer to our setting.

Compliance to non-nested settings is performed by inserting an additional step in the definition of the prolongation operator: starting with polynomials lying on the coarse faces, the nested version<sup>14</sup> begins with the reconstruction of a broken *element*-defined polynomial on the coarse mesh, which we propose (as already performed in the context of DG discretizations<sup>32</sup>) to orthogonally project in  $L^2$ -norm onto the non-nested fine mesh. In a next step, the trace of the result can be computed on the fine faces. The numerical evaluation of this  $L^2$ -orthogonal projection operator hinges on the projection of the local coarse basis functions onto the fine bases, i.e. on the computation of the  $L^2$ -inner products of the coarse and fine basis functions over the fine elements. As a direct consequence, the local definition of the functional bases makes the intersections of coarse and fine elements the respective integration supports to these inner products. The prerequisite of computing the geometric intersections between coarse and fine elements can occur as computationally prohibitive. So instead of this exact computation, we propose the implementation of an approximate operator that does not require intersections. We also refer to a previous work<sup>33</sup> that tackles the practical computation of  $L^2$ -orthogonal projections as well.

In a face-defined multigrid method, as the smoother operates on the mesh skeleton, the efficient reduction of the low-frequency components of the error relies on accessing coarse representations of the face-defined polynomials. This implies that *faces* must be coarsened between levels,<sup>14, §4.4.3</sup> which is a new constraint imposed to any suited coarsening strategy. Unfortunately, usual agglomeration methods,<sup>34</sup> typical candidates for the coarsening of unstructured meshes, do not meet this requirement. More generally, methods that conserve embedded meshes do so by conserving fine faces on the coarse mesh, which goes against this new constraint. This observation points out non-nested methods as suitable alternatives. As such, strategies which build coarse tetrahedra from the fine tetrahedra's vertices<sup>35</sup> seem well-adapted to our problem, as long as the shape quality of the coarse elements is sufficient. This condition drives us towards *polytopal* coarse elements in order to make use of their shape flexibility to construct high-quality coarse meshes. Methods based on Voronoi diagrams<sup>36,37</sup> fulfill that purpose, as well as our face coarsening constraint. Nonetheless, we choose to propose another option based on element agglomeration. Indeed, additionally to being easy to understand and implement, these methods also benefit from simple derivations to anisotropic grids.<sup>38</sup> Thus, we suggest to add to any chosen agglomeration process a step of *face collapsing*, already used in different fashions.<sup>39,40</sup> Moreover, we will show that agglomeration-based methods ease the construction of our approximate  $L^2$ -orthogonal projection.

Having established the reasons driving us to non-nested settings, this work is organized around the adaptation of the existing nested solver<sup>14</sup> and its efficient implementation for the purpose of its practical use for the solution of condensed linear systems arising from the HHO discretization of scalar elliptic equations on complex geometries. Thus, we begin by describing in Section 2 the HHO discretization of the diffusion equation as model problem. In Section 3, we recall the ingredients in the construction of the nested multigrid method,<sup>14</sup> and extend its range of application to non-nested meshes through the use of the  $L^2$ -orthogonal projection. This approach is validated by numerical tests using independent retriangulations at every levels. Insofar as the newly used  $L^2$ -orthogonal projection depends, in its exact evaluation, on the expensive computation of the intersections between coarse and fine elements, we devote Section 4 to the definition of a cheaper approximate operator. We also propose in Section 5 a suited non-nested and polytopal coarsening strategy based on element agglomeration and face collapsing. Finally, numerical tests are presented in Section 6, which demonstrate the optimality of our multigrid method, used as a solver, on 2D and 3D diffusion problems.

## 2 | HHO FORMULATION OF THE DIFFUSION EQUATION

### 2.1 | Model problem

Let  $\Omega \subset \mathbb{R}^d$ , with  $d \in \{2, 3\}$ , be a bounded polyhedral domain with boundary  $\partial\Omega$ . For  $X$  a measured subset of  $\Omega$ ,  $L^2(X)$  denotes the space of square-integrable functions over  $X$ , equipped with the inner product  $(u, v)_X := \int_X uv$ , for  $u, v \in L^2(X)$ . Likewise,  $[L^2(X)]^d$  is equipped with the inner product  $(\mathbf{u}, \mathbf{v})_X := \int_X \mathbf{u} \cdot \mathbf{v}$ , for  $\mathbf{u}, \mathbf{v} \in [L^2(X)]^d$ . Classically, we denote by  $H^1(X)$  the space

spanned by functions of  $L^2(X)$  whose partial derivatives are also square-integrable, and by  $H_0^1(X)$  its subspace of functions with vanishing trace on the boundary  $\partial X$  of  $X$ . We consider the following boundary value problem:

$$\begin{cases} -\nabla \cdot (\mathbf{K}\nabla u) = f & \text{in } \Omega, \\ u = 0 & \text{on } \partial\Omega, \end{cases} \quad (2.1)$$

where  $f \in L^2(\Omega)$  is a given source term and  $\mathbf{K} : \Omega \rightarrow \mathbb{R}^{d \times d}$  is the diffusion tensor field, which is assumed to be real, symmetric, uniformly positive definite and piecewise constant over a fixed partition of  $\Omega$  into polyhedra. The variational formulation of problem (2.1) reads

$$\begin{aligned} & \text{Find } u \in H_0^1(\Omega) \text{ such that} \\ & a(u, v) = (f, v)_\Omega \quad \forall v \in H_0^1(\Omega), \end{aligned} \quad (2.2)$$

where the bilinear form  $a : H^1(\Omega) \times H^1(\Omega) \rightarrow \mathbb{R}$  is defined by

$$\forall v, w \in H^1(\Omega), \quad a(v, w) := (\mathbf{K}\nabla v, \nabla w)_\Omega.$$

## 2.2 | HHO Discretization

This section briefly recalls the standard HHO discretization<sup>1, Section 3.1</sup> of problem (2.2).

### 2.2.1 | Mesh definition

We consider a mesh  $(\mathcal{T}_h, \mathcal{F}_h)$  of  $\Omega$  in the sense of<sup>1, Definition 1.4</sup>, i.e.,  $\mathcal{T}_h$  denotes the set of polyhedral elements,  $\mathcal{F}_h$  denotes the set of faces, and  $h := \max_{T \in \mathcal{T}_h} \text{diameter}(T)$  is the meshsize<sup>1, Def. 1.4</sup>. In the context of problem (2.2), we further assume that the diffusion tensor  $\mathbf{K}$  is constant over every element  $T \in \mathcal{T}_h$ , and we denote  $\mathbf{K}_T := \mathbf{K}|_T$  for all  $T \in \mathcal{T}_h$ . The faces that lie on the boundary of  $\Omega$  are collected in  $\mathcal{F}_h^B$  and are referred to as boundary faces. Then,  $\mathcal{F}_h$  may be partitioned as  $\mathcal{F}_h := \mathcal{F}_h^B \cup \mathcal{F}_h^I$ , with  $\mathcal{F}_h^B \cap \mathcal{F}_h^I = \emptyset$ , where  $\mathcal{F}_h^I$  denotes the set of internal faces. For  $T \in \mathcal{T}_h$ ,  $\mathcal{F}_T$  denotes the set of faces that lie on the boundary of  $T$ . Reciprocally, for any face  $F \in \mathcal{F}_h$ ,  $\mathcal{T}_F := \{T \in \mathcal{T}_h \mid F \in \mathcal{F}_T\}$  denotes the set of elements that have  $F$  as one of their faces.

### 2.2.2 | Discrete setting

With  $m \geq 0$  given integer, we denote by  $\mathbb{P}^m(X)$  the space spanned by the restriction of  $d$ -variate polynomials of total degree  $\leq m$  (in short, of degree  $m$ ) to  $X \subset \Omega$ . Denote by  $k \geq 0$  the polynomial degree of the skeletal unknowns in the HHO scheme. We introduce the following broken polynomial spaces, respectively supported by the mesh and its skeleton:

$$\begin{aligned} \underline{U}_{\mathcal{T}_h}^m &:= \{v_{\mathcal{T}_h} := (v_T)_{T \in \mathcal{T}_h} \mid v_T \in \mathbb{P}^m(T) \quad \forall T \in \mathcal{T}_h\} \quad \text{for } m \in \{k, k+1\}, \\ \underline{U}_{\mathcal{F}_h}^k &:= \{v_{\mathcal{F}_h} := (v_F)_{F \in \mathcal{F}_h} \mid v_F \in \mathbb{P}^k(F) \quad \forall F \in \mathcal{F}_h\}, \end{aligned}$$

from which we build the global space of hybrid variables

$$\underline{U}_h^k := \left\{ \underline{v}_h = (v_{\mathcal{T}_h}, v_{\mathcal{F}_h}) \in \underline{U}_{\mathcal{T}_h}^k \times \underline{U}_{\mathcal{F}_h}^k \right\}.$$

For all  $T \in \mathcal{T}_h$ , its local counterpart is

$$\underline{U}_T^k := \left\{ \underline{v}_T = (v_T, (v_F)_{F \in \mathcal{F}_T}) \mid v_T \in \mathbb{P}^k(T), v_F \in \mathbb{P}^k(F) \quad \forall F \in \mathcal{F}_T \right\}. \quad (2.3)$$

The homogeneous Dirichlet boundary condition is strongly accounted for in the following subspaces:

$$\underline{U}_{\mathcal{F}_h,0}^k := \left\{ v_{\mathcal{F}_h} \in \underline{U}_{\mathcal{F}_h}^k \mid v_F = 0 \quad \forall F \in \mathcal{F}_h^B \right\}, \quad \underline{U}_{h,0}^k := \underline{U}_{\mathcal{T}_h}^k \times \underline{U}_{\mathcal{F}_h,0}^k.$$

Given the exact solution  $u$  of (2.2), the element and face unknowns of the HHO method (2.6) can be interpreted as the local  $L^2$ -orthogonal projections of  $u$  onto the respective mesh elements and faces (refer to<sup>1, Eq. (2.34)</sup>). For all  $T \in \mathcal{T}_h$ , the *local potential reconstruction*  $p_T^{k+1} : \underline{U}_T^k \rightarrow \mathbb{P}^{k+1}(T)$  is defined such that, for all  $\underline{v}_T = (v_T, (v_F)_{F \in \mathcal{F}_T}) \in \underline{U}_T^k$ ,  $p_T^{k+1} \underline{v}_T$  satisfies

$$\begin{cases} (\mathbf{K}_T \nabla p_T^{k+1} \underline{v}_T, \nabla w)_T = -(v_T, \nabla \cdot (\mathbf{K}_T \nabla w))_T + \sum_{F \in \mathcal{F}_T} (v_F, \mathbf{K}_T \nabla w \cdot \mathbf{n}_{TF})_F & \forall w \in \mathbb{P}^{k+1}(T), \end{cases} \quad (2.4a)$$

$$\begin{cases} (p_T^{k+1} \underline{v}_T, 1)_T = (v_T, 1)_T, \end{cases} \quad (2.4b)$$

where  $\mathbf{n}_{TF}$  denotes the unit vector normal to  $F$  pointing out of  $T$ .

### 2.2.3 | Discretization of the model problem

The global bilinear form  $a_h : \underline{U}_h^k \times \underline{U}_h^k \rightarrow \mathbb{R}$  is assembled from elementary contributions as follows:

$$a_h(\underline{u}_h, \underline{v}_h) := \sum_{T \in \mathcal{T}_h} a_T(\underline{u}_T, \underline{v}_T),$$

where for all  $T \in \mathcal{T}_h$ , the local bilinear form  $a_T : \underline{U}_T^k \times \underline{U}_T^k \rightarrow \mathbb{R}$  is defined as

$$a_T(\underline{u}_T, \underline{v}_T) := (\mathbf{K}_T \nabla p_T^{k+1} \underline{u}_T, \nabla p_T^{k+1} \underline{v}_T)_T + s_T(\underline{u}_T, \underline{v}_T). \quad (2.5)$$

In this expression, the first term is responsible for consistency while the second, involving the bilinear form  $s_T : \underline{U}_T^k \times \underline{U}_T^k \rightarrow \mathbb{R}$ , is required to ensure stability of the scheme. The design of the stabilization bilinear form  $s_T$  is standard and is not detailed here for the sake of conciseness; see<sup>1, Assumption 3.9</sup> on this subject. The global discrete problem then reads

$$\text{Find } \underline{u}_h \in \underline{U}_{h,0}^k \text{ such that } a_h(\underline{u}_h, \underline{v}_h) = \sum_{T \in \mathcal{T}_h} (f, v_T)_T \quad \forall \underline{v}_h \in \underline{U}_{h,0}^k. \quad (2.6)$$

### 2.3 | Assembly and static condensation

After fixing bases for the spaces  $U_T^k$ ,  $U_T^{k+1}$  and  $U_F^k$  for all  $(T, F) \in \mathcal{T}_h \times \mathcal{F}_h$ . Considering these bases, the local algebraic contributions of the bilinear form  $a_T$  (cf. (2.5)) and of the linear form  $\underline{U}_T^k \ni \underline{v}_T \mapsto (f, v_T)_T \in \mathbb{R}$  are, respectively, the matrix  $\mathbf{A}_T$  and the vector  $\mathbf{B}_T$  such that

$$\mathbf{A}_T := \begin{pmatrix} \mathbf{A}_{TT} & \mathbf{A}_{TF_T} \\ \mathbf{A}_{F_T T} & \mathbf{A}_{F_T F_T} \end{pmatrix}, \quad \mathbf{B}_T := \begin{pmatrix} \mathbf{b}_T \\ \mathbf{0} \end{pmatrix}, \quad (2.7)$$

in which the unknowns have been numbered so that element unknowns come first and face unknowns come last; see<sup>1, Appendix B</sup> for further details. After assembling the local contributions and eliminating the boundary unknowns by a strong enforcement of the Dirichlet boundary condition, we end up with a global linear system of the form

$$\begin{pmatrix} \mathbf{A}_{\mathcal{T}_h \mathcal{T}_h} & \mathbf{A}_{\mathcal{T}_h \mathcal{F}_h^i} \\ \mathbf{A}_{\mathcal{F}_h^i \mathcal{T}_h} & \mathbf{A}_{\mathcal{F}_h^i \mathcal{F}_h^i} \end{pmatrix} \begin{pmatrix} \mathbf{v}_{\mathcal{T}_h} \\ \mathbf{v}_{\mathcal{F}_h^i} \end{pmatrix} = \begin{pmatrix} \mathbf{b}_{\mathcal{T}_h} \\ \mathbf{0} \end{pmatrix}. \quad (2.8)$$

Since element-DoFs are coupled with each other only through face-DoFs,  $\mathbf{A}_{\mathcal{T}_h \mathcal{T}_h}$  is block-diagonal, therefore inexpensive to invert. The static condensation process takes advantage of this property to locally eliminate the element-DoFs: it goes by expressing  $\mathbf{v}_{\mathcal{T}_h}$  in terms of  $\mathbf{v}_{\mathcal{F}_h^i}$  in the first equation of (2.8):

$$\mathbf{v}_{\mathcal{T}_h} = -\mathbf{A}_{\mathcal{T}_h \mathcal{T}_h}^{-1} \mathbf{A}_{\mathcal{T}_h \mathcal{F}_h^i} \mathbf{v}_{\mathcal{F}_h^i} + \mathbf{A}_{\mathcal{T}_h \mathcal{T}_h}^{-1} \mathbf{b}_{\mathcal{T}_h}, \quad (2.9)$$

and then replacing  $\mathbf{v}_{\mathcal{T}_h}$  with its expression (2.9) in the second equation:

$$\left( \mathbf{A}_{\mathcal{F}_h^i \mathcal{F}_h^i} - \mathbf{A}_{\mathcal{F}_h^i \mathcal{T}_h} \mathbf{A}_{\mathcal{T}_h \mathcal{T}_h}^{-1} \mathbf{A}_{\mathcal{T}_h \mathcal{F}_h^i} \right) \mathbf{v}_{\mathcal{F}_h^i} = -\mathbf{A}_{\mathcal{F}_h^i \mathcal{T}_h} \mathbf{A}_{\mathcal{T}_h \mathcal{T}_h}^{-1} \mathbf{b}_{\mathcal{T}_h}, \quad (2.10)$$

thus yielding the so-called statically condensed system. After solving (2.10) for the face unknowns  $\mathbf{v}_{\mathcal{F}_h^i}$ , the element unknowns are locally recovered by the local counterpart of (2.9):

$$\mathbf{v}_T = -\mathbf{A}_{TT}^{-1} \mathbf{A}_{TF_T} \mathbf{v}_{F_T} + \mathbf{A}_{TT}^{-1} \mathbf{b}_T, \quad (2.11)$$

where  $\mathbf{v}_T$  denotes the restriction to  $T$  of  $\mathbf{v}_{\mathcal{T}_h}$ , and  $\mathbf{v}_{F_T}$  the restriction of  $\mathbf{v}_{\mathcal{F}_h^i}$  to the faces of  $T$  interior to the domain, completed by zeros for boundary faces.

## 3 | MULTIGRID METHOD ON NON-NESTED MESHES

The multigrid method presented in this section aims at solving the statically condensed system (2.10). It extends the nested algorithm of our previous work<sup>14</sup> to non-nested mesh hierarchies. The core of the algorithm lies in the definition of the prolongation operator, which transfers broken polynomials lying on the coarse skeleton to broken polynomials on the fine one.

Consistently with the multigrid literature, we use the subscript  $\ell \in \{1, \dots, L\}$  to index the levels in the mesh hierarchy, and we denote by  $h_\ell$  the meshsize at level  $\ell$ . We sort the levels so that  $L$  refers to the finest mesh, whereas 1 corresponds to the coarsest one, i.e.  $h_L < h_{L-1} < \dots < h_1$ . Importantly, we further assume that, from a level  $\ell > 1$  to the immediately coarser one

$(\ell - 1)$ , not only are the elements coarsened, but so are the faces. To ease the notation, the subscripts  $h_\ell$  may be replaced with  $\ell$ , so that the hierarchy of non-nested polyhedral meshes may be denoted by  $(\mathcal{T}_\ell, \mathcal{F}_\ell)_{\ell=L \dots 1}$ .

Now, considering two successive levels  $\ell > 1$  (fine) and  $\ell - 1$  (coarse), we define the prolongation operator  $P : U_{\mathcal{F}_{\ell-1},0}^k \rightarrow U_{\mathcal{F}_\ell,0}^k$  as the composition of three operators,

$$P := \Pi_\ell \circ J_{\mathcal{F}_{\ell-1}}^\ell \circ \Theta_{\ell-1}, \quad (3.1)$$

where the coarse level potential reconstruction operator  $\Theta_{\ell-1} : U_{\mathcal{F}_{\ell-1},0}^k \rightarrow U_{\mathcal{T}_{\ell-1}}^{k+1}$  reconstructs, from the face unknowns, a broken polynomial of degree  $k + 1$  on  $\mathcal{T}_{\ell-1}$ ; then,  $J_{\mathcal{F}_{\ell-1}}^\ell : U_{\mathcal{T}_{\ell-1}}^{k+1} \rightarrow U_{\mathcal{T}_\ell}^{k+1}$  is now taken equal to the  $L^2$ -orthogonal projection to include the non-nested case; finally the trace operator  $\Pi_\ell : U_{\mathcal{T}_\ell}^{k+1} \rightarrow U_{\mathcal{F}_\ell,0}^k$  maps the polynomials of degree  $k + 1$  defined on the coarse elements to a broken polynomial of degree  $k$  on the fine skeleton.

### 3.1 | The potential reconstruction operator $\Theta_\ell$

Given a trace error function  $e_{\mathcal{F}_\ell} \in U_{\mathcal{F}_\ell,0}^k$  as the operand of  $\Theta_\ell$ , we recover a element-defined error function in  $U_{\mathcal{T}_\ell}^k$  by locally reversing the static condensation process. For all  $T \in \mathcal{T}_\ell$ , let us denote by  $\theta_T : U_{\mathcal{F}_T}^k \rightarrow U_T^k$  the local operator performing this operation, as well as  $e_{\mathcal{F}_T} := (e_F)_{F \in \mathcal{F}_T}$ . For all  $(v_T, v_{\mathcal{F}_T}) \in U_T^k \times U_{\mathcal{F}_T}^k$ , we let  $\mathbf{v}_T$  and  $\mathbf{v}_{\mathcal{F}_T} := (\mathbf{v}_F)_{F \in \mathcal{F}_T}$  denote their respective algebraic representations as vectors of coefficients in the selected polynomial bases. With these notations, for all  $v_{\mathcal{F}_T} \in U_{\mathcal{F}_T}^k$ , the algebraic representation of  $v_T := \theta_T v_{\mathcal{F}_T}$  is given by the algebraic counterpart of (2.11):

$$\mathbf{v}_T := -\mathbf{A}_{TT}^{-1} \mathbf{A}_{T\mathcal{F}_T} \mathbf{v}_{\mathcal{F}_T}. \quad (3.2)$$

Note that the removal of the affine term of (2.11) is justified<sup>14, §3.2.1</sup> by  $\theta_T$  being applied to error functions.

Once the element-defined error  $e_T := \theta_T e_{\mathcal{F}_T}$  is retrieved from (3.2), its accuracy can be improved by one order of approximation through the use of the higher-order reconstruction operator  $p_T^{k+1}$  defined by (2.4). Introducing  $\underline{\theta}_T : U_{\mathcal{F}_T}^k \rightarrow \underline{U}_T^{k+1}$  such that for all  $v_{\mathcal{F}_T} \in U_{\mathcal{F}_T}^k$ ,

$$\underline{\theta}_T v_{\mathcal{F}_T} := (\theta_T v_{\mathcal{F}_T}, v_{\mathcal{F}_T}),$$

the local contribution to  $\Theta_\ell$  is finally given by

$$(\Theta_\ell e_{\mathcal{F}_\ell})|_T := p_T^{k+1} \underline{\theta}_T e_{\mathcal{F}_T}. \quad (3.3)$$

Note that, although this step of higher-order reconstruction is optional for the convergence and algorithmic scalability of the multigrid method, it improves the convergence rate<sup>14, §4.4.1</sup>.

### 3.2 | The trace operator $\Pi_\ell$

The trace operator is also locally defined. Let  $v \in U_{\mathcal{T}_\ell}^{k+1}$  be an element-defined broken polynomial, and  $F \in \mathcal{F}_\ell$ . If  $F$  is a boundary face, then  $(\Pi_\ell v)|_F$  is set to be zero; else,  $(\Pi_\ell v)|_F$  is built as the weighted average of the traces of  $v$  on both sides of  $F$ , which is subsequently orthogonally projected onto  $\mathbb{P}^k(F)$  in order to lower the degree, i.e.

$$(\Pi_\ell v)|_F := \begin{cases} w_{T_1 F} \pi_F^k(v|_{T_1})|_F + w_{T_2 F} \pi_F^k(v|_{T_2})|_F & \text{if } F \in \mathcal{F}_\ell^1, \\ 0 & \text{otherwise,} \end{cases} \quad (3.4)$$

where  $T_1, T_2$  denote the distinct elements in  $\mathcal{T}_F \subset \mathcal{T}_\ell$ , and  $\pi_F^k$  is the  $L^2$ -orthogonal projector on  $\mathbb{P}^k(F)$ . Introducing for  $i \in \{1, 2\}$  the scalar coefficients  $K_{T_i F} := \mathbf{K}_{T_i} \mathbf{n}_{T_i F} \cdot \mathbf{n}_{T_i F}$ , the weights are defined as

$$w_{T_i F} := \frac{K_{T_i F}}{K_{T_1 F} + K_{T_2 F}}. \quad (3.5)$$

### 3.3 | Multigrid components

Having defined the prolongation operator, the restriction is set to its adjoint in the usual fashion, i.e. with respect to the coarse and fine face-defined global inner products. As such, the matrix of one transfer operator in the chosen polynomial basis is the transpose of the other. Coarse grid operators come from the rediscrization of the problem on the coarse meshes. In order to relax together all the unknowns related to the same local polynomial, block versions of standard fixed point iterations (like

damped Jacobi or SOR) should be used, with a block size corresponding to the number of DoFs per face. For instance, for a 3D problem with the polynomial degree  $k = 1$  chosen for the face-defined polynomials, the blocks will be of size  $3 \times 3$ . For our numerical tests, the relaxation method is set to the block Gauss-Seidel iteration. In order to keep the computational cost as low as possible, we use the post-smoothing-only cycles that were found to be the most efficient<sup>14, §4.2.2</sup> in terms of total theoretical work or CPU time. In particular, we use V(0,3) in 2D and V(0,6) in 3D. Finally, on the coarsest level, the system is exactly solved by a direct method.

## 4 | APPROXIMATION OF THE $L^2$ -ORTHOGONAL PROJECTION

In the second step of the prolongation, the global broken polynomial, reconstructed on the coarse mesh from the DoFs on the coarse faces, is projected onto the fine mesh via the  $L^2$ -orthogonal projection operator denoted  $J_{\ell-1}^\ell : \mathcal{T}_{\ell-1} \rightarrow \mathcal{T}_\ell$ . The construction of this projection requires the computation of the  $L^2$ -inner products of all pairs of coarse/fine basis functions over the fine elements. Locally, one such integral can only be non-zero if the supports of the fine and coarse basis functions intersect, i.e. if the respective fine and coarse elements overlap. In this case, the non-zero part of the integral is restricted to the intersection of the fine and coarse elements.

For any element  $T$ , we denote by  $\mathcal{B}_T$  the selected basis for  $\mathbb{P}^k(T)$ . For all  $T^c \in \mathcal{T}_{\ell-1}$  and  $T^f \in \mathcal{T}_\ell$ , we then have

$$(\varphi^c, \varphi^f)_{T^f} = (\varphi^c, \varphi^f)_{T^c \cap T^f} \quad \forall (\varphi^c, \varphi^f) \in \mathcal{B}_{T^c} \times \mathcal{B}_{T^f}. \quad (4.1)$$

Consequently, the exact construction of  $J_{\ell-1}^\ell$  preliminary requires the computation of the geometric intersection of every pair of overlapping fine/coarse elements. We can see two drawbacks to this method: firstly, the computation of the intersections adds a costly load to the computational burden. Although the actual overhead depends on the efficiency of the algorithm and its implementation, our numerical tests based on the state-of-the-art geometric library CGAL<sup>41</sup> find it to heavy for practical use. Secondly, the intersection of usual element shapes, even plain simplices, generally yields a polytope. In practice, if only simplicial meshes are used, one might want to avoid introducing other shapes, over which integral computation may be more expensive.

Given these practical limitations, we introduce a mechanism to compute the operator  $J_{\ell-1}^\ell$  in a cheaper and simpler way, though approximately, by avoiding computing element intersections altogether. It is based on the following approximation (exact when the meshes are nested): For any pair of coarse and fine elements  $(T_c, T_f)$ ,

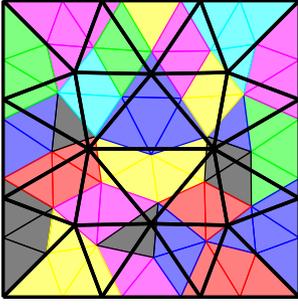
$$T_c \cap T_f \approx \begin{cases} T_f & \text{if } T_c \text{ is the coarse element which "embeds } T_f \text{ the most",} \\ \emptyset & \text{otherwise.} \end{cases} \quad (4.2)$$

According to the element shapes, determining which coarse element embeds the largest part of a given fine one may have multiple interpretations, which may also be implemented in different fashions. For simplicial elements, it suffices to choose the coarse element containing the barycenter of the fine one.

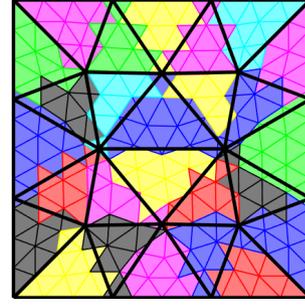
Following this model, each integral over an intersection is either discarded or computed over the whole fine element, in which case the support of the coarse basis function is implicitly extended to include the entire fine element. The number of integrals to compute is then reduced to the number of fine elements.

The validity of this method hinges on the assumption that, if the function to project is *smooth enough* and if the fine elements *do not stick too much out* of the coarse element they are associated to, global approximation properties are preserved. Note that the smoothness requirement emphasizes the restriction not to have elements crossing the boundaries of physical subdomains, i.e. where coefficient discontinuities can occur. Figure 1a shows on a triangulated 2D example how the fine elements are distributed among the coarse ones and how much they can stick out. Graphically, the more the colored clusters of fine elements resemble their respective coarse elements, the less the coarse basis functions must be extended outside of their supports, thus losing accuracy, and therefore the better the approximation of the integrals.

Our numerical tests show that in 2D, on the triangulated square, the approximate operator constructed by this method exhibits good enough accuracy to reproduce the results obtained with the exact computation only for  $k \leq 1$ , and fails to be a good approximate for higher orders. In 3D, the method is unsuccessful for all orders. The method is improved the following way: instead of approximating intersections by an all-or-nothing result (either the whole fine element or the empty set) as in (4.2), we can increase the granularity by subdividing the fine elements, and distribute the subshapes among the coarse elements the same way as before. Assuming, for any fine element  $T_f \in \mathcal{T}_\ell$  a given subdivision  $\text{Sub}(T_f)$ , and denoting by  $\text{closest}(\cdot)$  the function



(a) Distribution of the fine elements to their “closest” coarse element.



(b) Distribution of the fine elements’ sub-triangles to their “closest” coarse element.

**FIGURE 1** (a) shows the superposition of coarse and fine triangular meshes. The coarse edges are represented by thick black segments. The fine triangles are colored according the coarse triangle they are “closest” to, i.e. the one containing their barycenters. (b) shows the same colored partitioning, this time for the fine elements’ sub-triangles.

associating to any element or subelement its “closest” coarse element, we then define the new approximate intersection as

$$T_c \cap T_f \approx \bigcup_{\substack{t \in \text{Sub}(T_f) \\ \text{closest}(t) = T_c}} t. \quad (4.3)$$

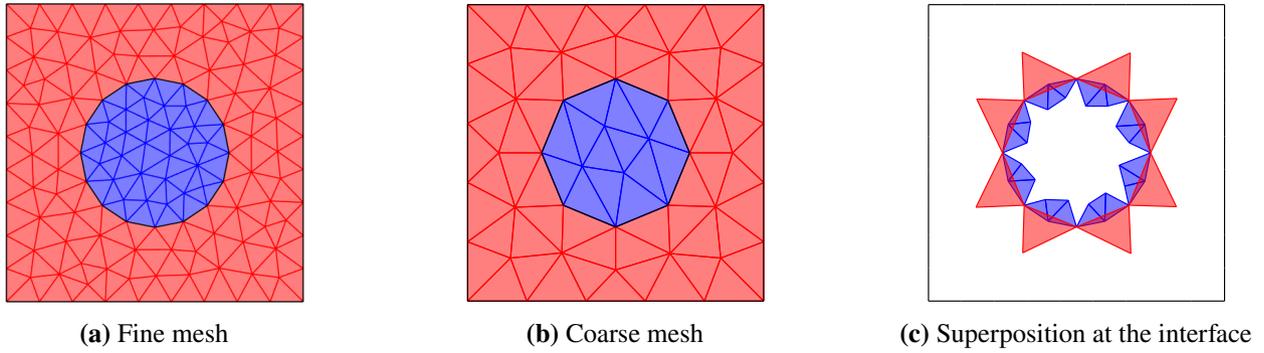
Figure 1b shows how sub-triangles obtained by connecting the middle-edges of the fine elements now approach the coarse ones.

The fine elements can be refined multiple times for even better accuracy, though at the cost of an increasing number of integrals to compute. However, the approximate operator derived from the fine elements being subdivided only once is found to be sufficient to achieve good multigrid results in 3D and for low polynomial orders. We refer to Section 6.2 for the details of the numerical tests.

*Remark 1.* (Heterogeneous case with curved region boundary) If the geometry is partitioned into multiple physical regions with curved boundary, the different levels of grid may approximate this boundary in a non-nested way. Then, imposing that jumps in the coefficient do not occur inside elements inevitably leads to coarse elements overlapping fine ones of different regions. Figure 2 illustrates a two-region domain separated by a circular interface. The approximation of the circle and therefore the discrete delimitation of the regions depend of the granularity of the mesh. Figures 2a and 2b, respectively, show a fine and coarse mesh obtained by Delaunay triangulation, while Figure 2c shows how coarse elements belonging to the red region overlap fine triangles of the blue region. These cases must be handled with great care. In such a configuration, it is crucial that values of DoFs belonging to one region should never be transferred to another one. The  $L^2$ -orthogonal projection must then keep this separation. Indeed, the solution cannot be locally represented with accuracy unless it is kept smooth inside elements. In other words, the intersection between a coarse element and a fine one that belong to two different regions must never be computed, and be assimilated to the empty set, even though, in fact, they geometrically overlap. Infringing this guideline results in a quick deterioration of multigrid convergence, and divergence can even occur for small-size problem provided a large jump in the coefficient.

## 5 | AGGLOMERATION-BASED COARSENING STRATEGY WITH FACE COLLAPSING

In this section, we lay the foundations of an abstract coarsening strategy for polytopal meshes that also coarsens faces. The steps are described by Algorithm 1. We recall that in this abstract setting, the generic term ‘face’ refers to the interface between elements (it being an actual face in 3D or an edge in 2D), and the term ‘neighbours’ refers to a pair of elements sharing a face. Step 1 is standard and defines any agglomeration method. The face coarsening comes from Step 2, where multiple fine faces are collapsed into a single coarse one. Figure 3 illustrates the result of successive coarsenings in 2D. Recall that the created polygons are not necessarily convex. Also, note that this algorithm does not ensure that every fine face is either removed (being interior to an agglomerate) or coarsened (by face collapsing): some fine faces may find themselves unaltered by the process. However,



**FIGURE 2** (a) (resp. (b)) presents a fine (resp. coarse) mesh obtained by Delaunay triangulation of a square domain embedding a round physical region. In (c), the red coarse elements overlapping fine blue ones are plotted.

numerical experiments show that the number of unaltered faces between two successive levels decreases with the number of times the coarsening strategy is applied, i.e. the number of levels built.

---

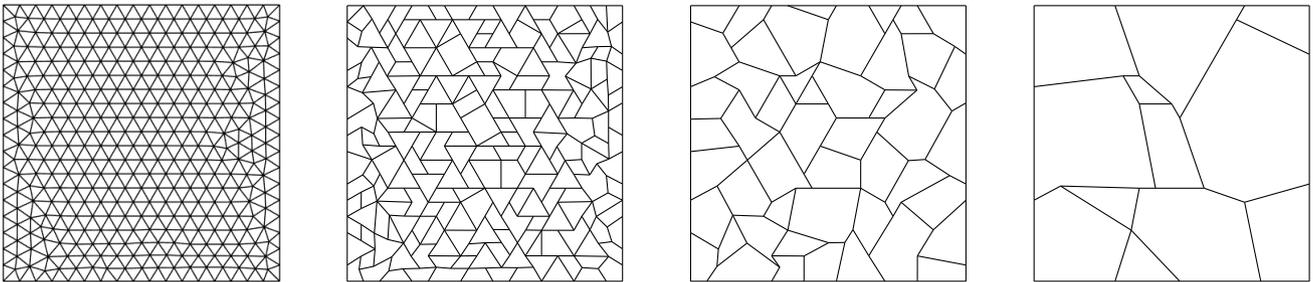
**Algorithm 1** Abstract coarsening strategy with face collapsing

---

**Step 1.** Agglomerate each fine element with its non-already agglomerated neighbours to form one polytopal coarse element.

**Step 2.** Collapse into one single coarse face the interfaces between two neighbouring coarse elements that are composed of multiple fine faces.

---



**FIGURE 3** Successive coarsenings of a triangulated square.

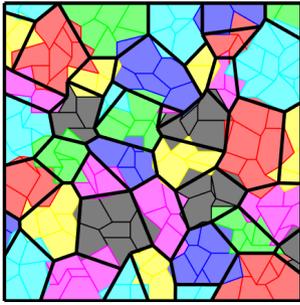
*Remark 2.* (Preventing domain erosion) The face collapsing step removes vertices from the mesh. In order to keep the domain from “eroding” at its corners (whether at domain boundaries or at interfaces between inner regions), one must make sure that the vertices that describe the domain geometry do not vanish in a face collapsing operation. To do so, it suffices to prevent the faces sharing such vertices from collapsing into one coarse face. Applied to a square domain or inner region, it means that at each of the four corners, the two orthogonal corner edges should never collapse together, so that the global squared shape would be preserved.

## 5.1 | Coarsening in 2D

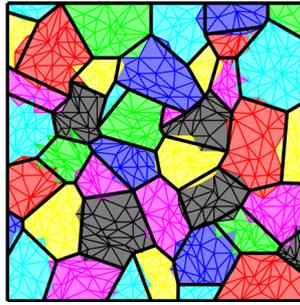
In 2D, the abstract Step 2 can be detailed as such: for all interfaces composed of multiple fine edges, remove all vertices interior to that interface and connect the boundary vertices to form a new coarse edge. This way, we ensure *node-nestedness*, i.e. that the

set of vertices of the coarse mesh is a subset of the fine vertices. Although having a node-nested hierarchy is not mandatory to achieve good multigrid performance, it may help: assuming that the fine nodes adequately capture the geometry, using subsets of those nodes at coarse levels should present some desirable properties with respect to the geometric approximation.

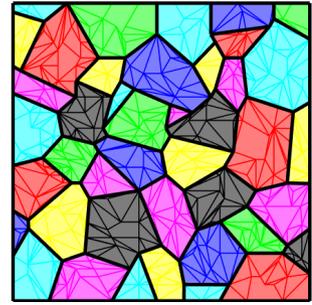
Regarding the computation of the  $L^2$ -orthogonal projection developed in the preceding section, numerical experiments show that the rough approximation given by (4.2) is not viable; see Figure 4a for a graphical illustration. The polygons must be subdivided to improve precision and, clearly, the method of subdivision influences the ultimate accuracy of the approximation. While a barycentric triangulation (Figures 4b and 4e) only offers practical usability for  $k \leq 1$ , an optimal triangulation (i.e. yielding an exact approximation; see Figures 4c and 4f) can be derived in the context of a coarsening strategy, inasmuch as the intergrid relationships are explicitly formed and can therefore be exploited. In the present coarsening strategy, the agglomeration step does not cause non-nestedness, which can only occur during the edge collapsing phase. Firstly, only fine elements possessing an edge that has been collapsed and that crosses the coarse collapsed edge must be subject to a careful subdivision; the other fine elements are fully embedded in a coarse one, and therefore do not need to be subdivided at all. Then, for the relevant fine elements, it suffices to generate triangles keeping on one side of the coarse edge. Algorithm 2 presents the details of the optimal subdivision we have used. Note that it applies to convex polygons; the non-convex ones require a preliminary step of convex partitioning before Algorithm 2 can be applied. This algorithm starts by triangulating the polygon independently of the coarse edges. Each triangle is then subtriangulated in order not to cross the coarse edges, following Algorithm 3. Note that this algorithm relies on evaluations of intersections between coarse and fine edges, which node-nestedness can certainly ease, insofar as a large part of the crossings between coarse and fine edges will then occur at mesh vertices.



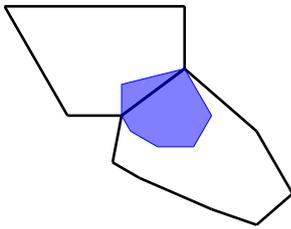
(a) Distribution of the fine elements to their “closest” coarse element.



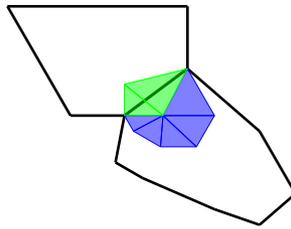
(b) The fine elements are subtriangulated by a barycentric method.



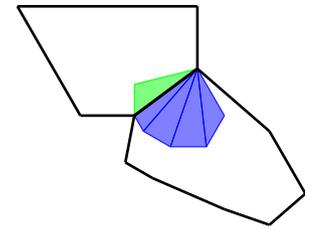
(c) The fine elements are subtriangulated such that they do not cross any coarse element's edge.



(d) Zoom-in on a fine element overlapping two coarse ones.



(e) Barycentric triangulation.



(f) Adapted triangulation preventing subtriangles from overlapping two coarse elements.

**FIGURE 4** The top figures show how the fine polygons (in (a)) or their subtriangulations (in (b),(c)) are clustered in the process of approximating the coarse/fine intersection involved in the computation of the  $L^2$ -orthogonal projection. The coarse edges are represented by thick black segments. In (b), the fine elements are triangulated by a barycentric method. In (c), the triangulation is adapted to prevent subtriangles to overlap multiple coarse elements. The bottom figures zoom in on a fine element overlapping two coarse ones. In (d), the whole fine element is affected to one of them for the computation of the approximate  $L^2$ -orthogonal projection. In (e) and (f), the subtriangles are dispatched on one or the other coarse element according to the location of their barycenters.

**Algorithm 2**  $\text{SubdivideConvexPolygon}(V, E)$ **Input:**  $V$ : set of vertices sorted in direct order, representing a convex polygon.  $E$ : set of coarse edges not to cross.**Output:**  $T$ : set of triangles partitioning the polygon and not crossing the coarse edges.

```

1: Let  $(v_1, v_2, v_3)$  be the first 3 vertices in  $V$ 
2:  $T := \text{SubdivideTriangle}(v_1, v_2, v_3, E)$ 
3: if  $\text{card}(V) > 3$  then
4:    $T := T \cup \text{SubdivideConvexPolygon}(V \setminus \{v_2\}, E)$            // see Figure 5a
5: end if

```

**Algorithm 3**  $\text{SubdivideTriangle}(v_1, v_2, v_3, E)$ **Input:**  $v_1, v_2, v_3$ : the vertices of the triangle sorted in direct order.  $E$ : set of coarse edges not to cross.**Output:**  $T$ : set of triangles partitioning the triangle and not crossing the coarse edges.

```

1: if  $E = \emptyset$  then  $T := \{(v_1, v_2, v_3)\}$            // no need to subtriangulate
2: else
3:   Let  $e \in E$            // take one coarse edge in the list, the others will be handled by recursion
4:    $\text{noTriangleEdgeCrossesTheCoarseEdge} := \text{true}$ 
5:   for  $i = 1, 2, 3$  do
6:     Given  $v_i$ , let  $v_j, v_k$  be the other two s.t.  $(v_i, v_j, v_k)$  are in direct order.
7:      $W := e \cap [v_i, v_j]$ 
8:     if  $W = \emptyset$  or  $W = \{v_i\}$  or  $W = \{v_j\}$  or  $W = [v_i, v_j]$  then continue for loop
9:     else
10:       $\text{noTriangleEdgeCrossesTheCoarseEdge} := \text{false}$ 
11:      Define  $w$  s.t.  $W = \{w\}$ .
12:      // division into two triangles and recursion; see Figure 5b
13:       $T := \text{SubdivideTriangle}(v_i, w, v_k, E)$ 
14:       $T := T \cup \text{SubdivideTriangle}(w, v_j, v_k, E)$ 
15:      break for loop
16:     end if
17:   end for
18:   if  $\text{noTriangleEdgeCrossesTheCoarseEdge}$  then
19:      $T := \text{SubdivideTriangle}(v_1, v_2, v_3, E \setminus \{e\})$ 
20:   end if
21: end if

```

## 5.2 | Coarsening in 3D

We have not generalized the preceding coarsening strategy to 3D. Instead, we briefly state the issue and propose directions.

In the 3D setting, it is generally not possible to collapse neighbouring faces into a single one as straightforwardly as in 2D, because the edges framing the fine faces do not usually describe a planar region that could define a new face. However, allowing the addition of new vertices, one can derive a variety of face collapsing methods. A simple one consists in choosing three non-colinear vertices amongst those of the fine faces, thus defining a 2D plane, and then orthogonally projecting the frame's vertices onto that plane to obtain the coarse face. However, in an attempt to preserve, on the coarse mesh, the approximation of the geometry given by the fine one, a more suitable choice for the plane defining the collapsed face would be one minimizing the distance to the vertices lying on the edge frame of the fine faces.

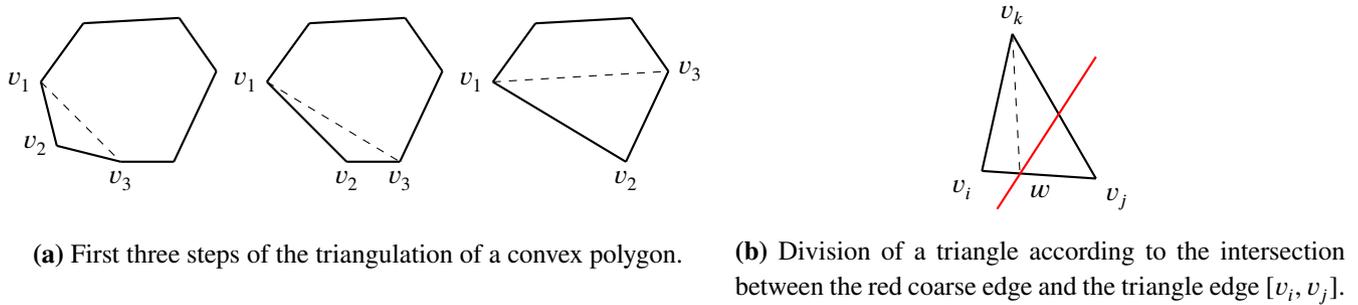


FIGURE 5 Respective illustrations for Algorithm 2 and Algorithm 3.

## 6 | NUMERICAL RESULTS

### 6.1 | Experimental setup

The numerical tests presented in this section have been performed on the diffusion problem (2.1), on 2D and 3D domains, where the source  $f \in L^2(\Omega)$  is a discontinuous piecewise constant function. The unit square and cube shall be used as preliminary tests before moving on to more complicated domains. The problems are discretized by the HHO method described in Section 2, in which the polynomial degree  $k$  of the element- and face-defined polynomials is taken between 0 and 3. We recall that the discrete solution ultimately provided by the method after higher-order reconstruction is a broken polynomial of degree  $k + 1$ . The local polynomial bases in elements and on faces are  $L^2$ -orthogonal Legendre bases.

The multigrid method defined in Section 3.3 is used as a solver for the solution of the statically condensed system (2.10), and our main goal is to study its asymptotic behaviour so that it can be used for large scale problems. The fine mesh is obtained by Delaunay triangulation of the domain, and coarse meshes are built until the system reaches a maximum size of 1000 unknowns or if the mesh cannot be coarsened anymore. Two different strategies are employed to build the coarse meshes: (i) independent remeshing: letting  $h$  be the meshsize of the fine simplicial mesh, the coarse mesh is obtained independently by retriangulation of the domain, enforcing a meshsize  $H \approx 2h$ ; (ii) for 2D problems, the agglomeration-based coarsening strategy with face collapsing, described in Section 5. Table 1 summarizes, for each strategy, the various methods evaluated in the numerical tests to compute the  $L^2$ -orthogonal projection. The stopping criterion is based on the backward error  $\|\mathbf{r}\|_2/\|\mathbf{b}\|_2$ , where  $\mathbf{r}$  denotes the residual of the algebraic system,  $\mathbf{b}$  the right-hand side, and  $\|\cdot\|_2$  the standard Euclidean norm on the vector space of coordinates. In all tests, we say that convergence is achieved when the criterion  $\|\mathbf{r}\|_2/\|\mathbf{b}\|_2 < 10^{-8}$  is reached.

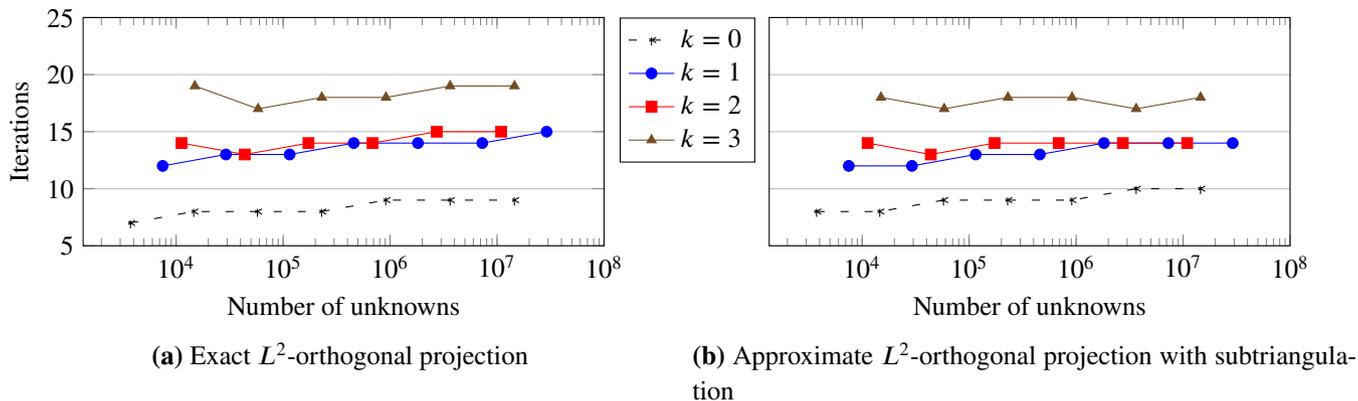
Coarsening strategy	$L^2$ -orthogonal projection
Independent simplicial remeshing	Exact, by computing intersections (cf (4.1))
	Approximate, w/o subtriangulation (cf (4.2))
	Approximate, w/ subtriangulation (cf (4.3)) by middle-edge connection in 2D, Bey's method in 3D
Agglomeration w/ face collapsing	Exact, by computing intersections (cf (4.1))
	Approximate, w/ barycentric subtriangulation (cf (4.3) and Figure 4e)
	Exact = Approximate w/ optimal subtriangulation (cf (4.3) and Figure 4f)

TABLE 1 Summary of the testing combinations.

### 6.2 | Assessment of the approximate $L^2$ -projection

We want to assess how the loss of accuracy implied by the approximate  $L^2$ -orthogonal projection (see Section 4) affects the convergence of the multigrid solver. In order not to add other difficulties that would interfere with the results, we use the unit

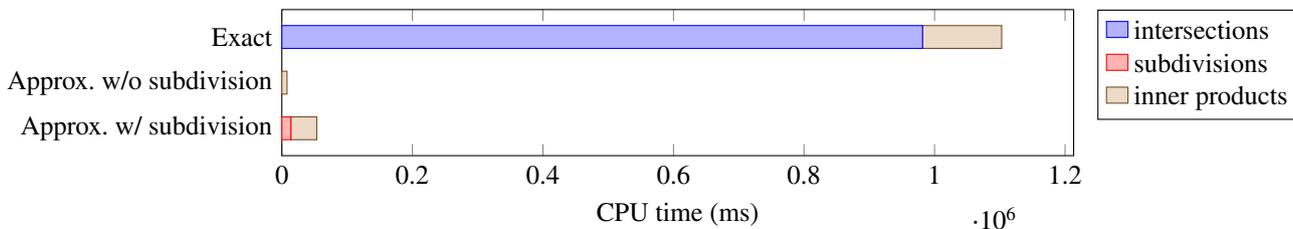
square as domain of study. The coarse meshes are built independently from each other by retriangulation of the domain, ensuring good quality at every level. As a reference to the best achievable result, a first test is made with the  $L^2$ -orthogonal projection operator computed exactly, meaning that the intersections of fine and coarse elements are actually computed, over which the required inner products are evaluated (as per (4.1)). Our implementation uses the CGAL library<sup>41</sup> to compute intersections. In this first setting, Figure 6a shows, for all polynomial degrees  $k$ , the scalable behaviour of the solver, whose convergence rate appears to be independent of the number of unknowns. The number of V(0,3)-cycles required to achieve convergence remains moderate (below 20), although higher than with nested meshes.<sup>14, Fig. 4.5</sup> This first experiment shows the validity of our non-nested approach to multigrid.



**FIGURE 6** Number of V(0,3)-cycle iterations to achieve convergence according to the number of unknowns in the system. The geometry is the unit square, and the hierarchy of meshes is obtained by independent remeshing. Each caption states the method of computation of the  $L^2$ -orthogonal projection.

It is important to mention that — in our implementation — the computation of the intersections consumes for  $k = 1$  over 80% of the CPU time used during the setup phase. The approximate  $L^2$ -orthogonal projection we propose allows to reduce the setup cost by a factor of 10 to 40, depending on the granularity of the fine element subdivisions. As an example, Figure 7 compares, for a test problem with  $k = 1$ , the CPU time consumed to compute the  $L^2$ -orthogonal projection, according to the method used: exact evaluation, approximate without subdivision (given by (4.2)), approximate with subdivision (given by (4.3)). One can clearly see that the largest share is spent in the exact evaluation of the intersection, which makes approximate methods remarkably more effective.

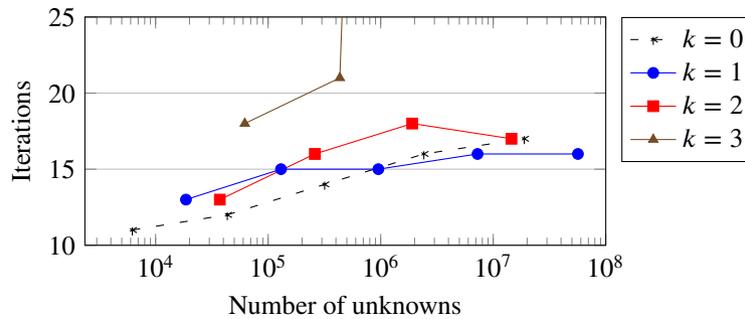
In order to assess the usability of our approximate methods, we now compare to the first scalability results the performance of the solver delivered by our approximations. The approximate  $L^2$ -orthogonal projection without subdivision (i.e. (4.2)) shows, in 2D, a lack of robustness w.r.t. the polynomial degree. Although the approximation does not seem to degrade the convergence rate for  $k \leq 1$ , it worsens with  $k = 2$ , and the solver finally diverges for  $k = 3$ . Moreover, the same test performed in 3D on the



**FIGURE 7** Comparison in CPU time of different methods for the computation of the  $L^2$ -orthogonal projection during the setup phase of the multigrid. Each bar is divided into sections corresponding to subtasks. The test problem is the unit square meshed by  $10^5$  triangles,  $k = 1$ .

unit cube causes the solver to diverge for all values of  $k$ , and so does the use of the 2D polygonal coarsening strategy defined in Section 5. These limitations make us discard this simple method.

We next focus on the finer approximation (4.3), where the fine elements are subdivided into subshapes to increase the granularity of the fine-coarse associations. In our tests, we use standard refinement methods to subdivide simplicial elements: connection of the middle-edges in 2D, and Bey’s tetrahedral refinement<sup>42</sup> in 3D. Figure 6b presents the performance of the multigrid solver on the unit square using this refined  $L^2$ -orthogonal projection. We can see that the results given by the exact method are now reproduced. On the unit cube (Figure 8), the solver exhibits good performance and scalable behaviour for  $k \leq 2$ , but diverges for  $k = 3$ . Note that these 3D results cannot be compared to those of the exact method, as the latter has not been implemented.



**FIGURE 8** Number of  $V(0,6)$ -cycle iterations to achieve convergence according to the number of unknowns in the system. The geometry is the unit cube, which is independently retriangulated at each level. The  $L^2$ -orthogonal projection is computed approximately with subtriangulation of the fine elements via Bey’s method.

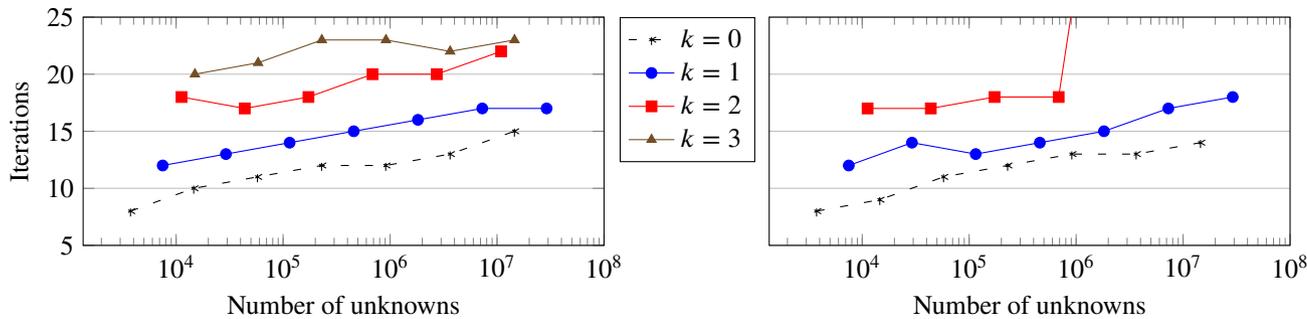
### 6.3 | Assessment of the agglomeration coarsening strategy with face collapsing

We now evaluate how the multigrid method responds to the coarsening strategy described in Section 5, especially when coupled to the approximate  $L^2$ -orthogonal projection. Figure 9 then presents the same scalability tests as the previous section, this time using the agglomeration coarsening with face collapsing to successively build the coarse meshes from an initial fine triangulation. As a reference, Figure 9a shows the results when the  $L^2$ -orthogonal projection is computed exactly. We can already remark that the convergence rate is slightly worse and the trend slightly less flat than with the independent remeshing (cf Figure 6a). Various reasons can explain these differences, the main one probably being the simplicity of implementation of our coarsening strategy, where we do not control and subsequently improve the element shapes and sizes.

Algorithmic scalability tests with the  $L^2$ -orthogonal projection approximately computed without subdivision of the fine elements are not presented here: except for  $k = 0$ , the solver quickly diverges. In Figure 9b, we use the barycentric triangulation to subdivide the fine elements (cf Figure 4e) and implement the approximation (4.3). We can see that the good performance of  $k = 1$  is now recovered, while for  $k = 2$ , the solver still diverges at moderate problem sizes. Finally, we stress that in the context of the coarsening strategy, the determination of an optimal fine subtriangulation (i.e. whose subelements do not overlap the limits of the coarse elements) is facilitated, which yields an exact implementation of the  $L^2$ -orthogonal projection, and therefore leads to results equivalent to Figure 9a.

### 6.4 | Highly unstructured test cases

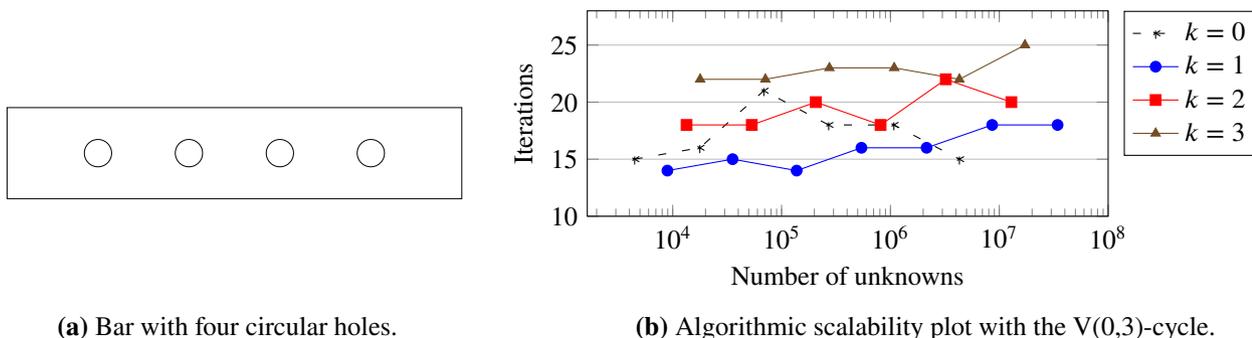
We now extend the experiments to geometries requiring highly unstructured meshes in order to validate the method on a wider class of problems. Figures 10a and 11a respectively draw similar geometries in 2D and 3D containing circular (resp. cylindrical) holes, thus requiring highly unstructured meshes in their discrete settings. Still focusing on the capability of the multigrid solver to handle large scale problems, Figures 10b and 11b present the respective algorithmic scalability plots of the solver. Starting from a fine simplicial mesh obtained by Delaunay triangulation, the coarse meshes are built using the coarsening strategy with face collapsing for the 2D tests, and by independent remeshing for the 3D tests. Notice that, whichever the method, the holes



(a) Exact  $L^2$ -proj., or approximate w/ optimal subtriangulation (b) Approximate  $L^2$ -proj. w/ barycentric subtriangulation

**FIGURE 9** Algorithmic scalability plots: number of V(0,3)-cycle iterations to achieve convergence according to the problem size. The geometry is the unit square. The coarse meshes are built using the agglomeration coarsening strategy with face collapsing. The captions state the method of computation for the  $L^2$ -orthogonal projection. The absence of the curve  $k = 3$  in (b) means a very large number of iterations or divergence of the solver.

are approximated, at each level, with respect to the mesh granularity, i.e. by polygons/polyhedra with less and less edges/faces as the mesh grows coarser. The  $L^2$ -orthogonal projection is computed exactly in 2D through the construction of an optimal subtriangulation of the elements. In 3D, the  $L^2$ -orthogonal projection is computed approximately with tetrahedral subdivision by Bey’s method. The 2D results presented in Figure 10b show that our multigrid method still exhibits the desired scalable behaviour for all  $k \leq 3$ . In 3D, the results of Figure 11b consistently show the same scalable behaviour for expected polynomial degrees, namely  $k = 1$  and 2. The case  $k = 0$  is a special case<sup>14</sup> which is not expected to necessarily work on unstructured problems, while the approximation of the  $L^2$ -orthogonal projection explains the divergence of the solver for  $k = 3$  (cf Figure 8 for the test on the unit cube). Although the displayed datapoints have been obtained with Dirichlet boundary conditions on the holes, the results with Neumann conditions on the holes, not reported here for the sake of brevity, indicate the same behaviour.

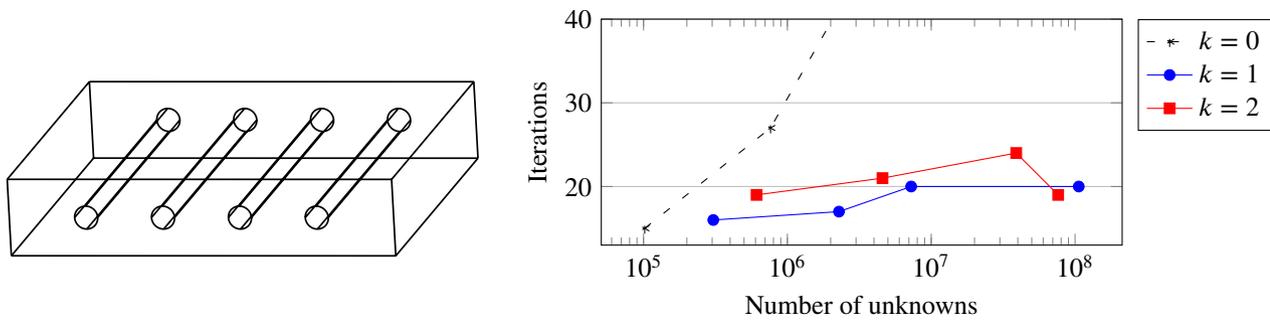


(a) Bar with four circular holes. (b) Algorithmic scalability plot with the V(0,3)-cycle.

**FIGURE 10** 2D complex geometry and associated algorithmic scalability plot for the multigrid solver. The coarse meshes are built using the agglomeration coarsening strategy with face collapsing. The  $L^2$ -orthogonal projection is computed exactly.

### 6.5 | Heterogeneous test case

The algorithm is finally tested on a heterogeneous domain, plotted in Figure 12a. This test case, provided by Electricité de France (EDF), is a proxy application for an industrial setting, which is characterized by jumps in the diffusion coefficients of several orders of magnitude in an otherwise relatively straightforward geometry. The proxy differs from the industrial setting in its geometry and its diffusion coefficients. However, the most salient feature, the ratio between the largest and the smallest diffusion coefficient, is close to the relevant regime. The domain is composed of four homogeneous and isotropic subdomains.



(a) Plate with four cylindric holes.

(b) Algorithmic scalability plot with the V(0,6)-cycle.

Elements	Hybrid DoFs	Face unknowns (system size)	Multigrid levels	Iterations	Asymptotic convergence rate
17,848,483	177,511,492	106,117,560	5	20	0.45

(c) Details on the last datapoint of  $k = 1$ .

**FIGURE 11** 3D test case using the geometry (a). The algorithmic scalability plot (b) of the solver is obtained with the coarse meshes independently retriangulated at each level, and the  $L^2$ -orthogonal projection computed approximately using Bey's subdivision.

Having assigned a color to each of those regions, their respective diffusion tensors  $\kappa_{\text{color}}I$ , where  $I$  denotes the identity matrix of dimension 2, define a global discontinuous tensor. The values of the coefficients  $\kappa_{\text{color}}$  are given in the caption of Figure 12a and lead to a maximum jump of  $10^8$  located at the interface between the gray and blue regions. When the coarse meshes come from independent remeshing of the domain, the scalability plot of Figure 12b demonstrates a convergence rate that is near-independent to the mesh size for all degrees except the lowest order. With the coarsening strategy, on the other hand, (cf. Figure 12c), algorithmic scalability is achieved for all degrees. Moreover, consistently with the results on nested meshes<sup>14, §4.3</sup>, the convergence rate of the multigrid method is found to be independent of the size of the discontinuities in the coefficient.

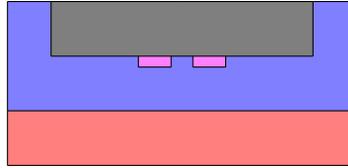
*Remark 3.* (Corner singularities) Heterogeneity in such a domain creates corner singularities which make the discretization lose its optimal convergence rate in the general case. Consequently, although the linear solver converges fast, it converges towards a solution that is not necessarily accurate. It is therefore not imperative to impose the linear solver to reach such a low algebraic error. In order to recover the optimal convergence rate of the discretization, techniques of local mesh refinement or energy correction<sup>43</sup> must be used.

## 7 | CONCLUSION

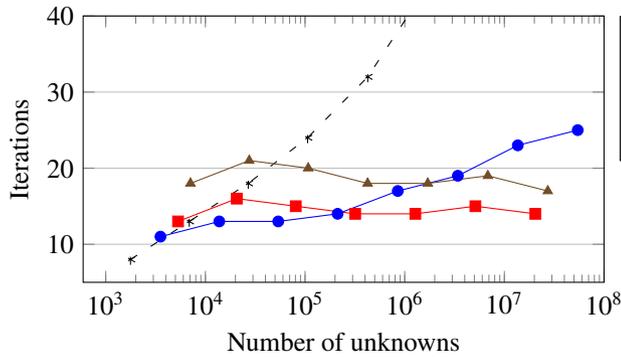
In this work, we have successfully extended to non-nested mesh hierarchies an efficient nested multigrid solver. Indeed, without requiring stronger smoothing, our adaptation allows to preserve the optimality of the convergence rate on a wider class of problems. The extra cost implied by the numerical evaluation of the  $L^2$ -orthogonal projection is also kept to a minimum thanks to an efficient approximation enabling us to discard the expensive computation of intersections between elements. The agglomeration coarsening strategy with face collapsing that we have developed, although naively implemented here, gives promising results, thus offering leads to more advanced methods fulfilling two purposes: coarsening the faces, and preparing the subtriangulation of the fine elements for the purpose of the construction of an accurate approximation of the  $L^2$ -orthogonal projection operator.

## ACKNOWLEDGEMENTS

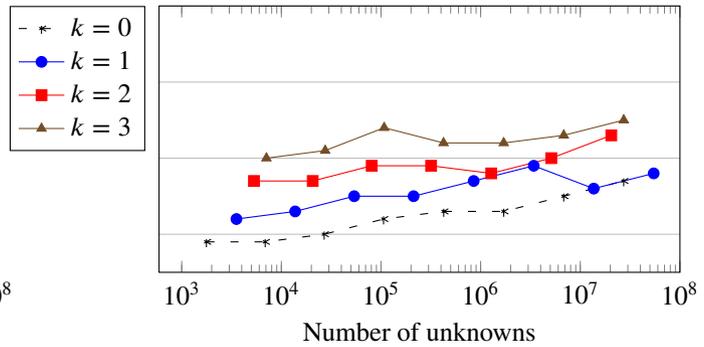
The authors acknowledge the support of *Agence Nationale de la Recherche* grant fast4hho (ANR-17-CE23-0019).



(a) Heterogeneous domain with the coefficients  $\kappa_{\text{red}} = 30$ ,  $\kappa_{\text{blue}} = 1$ ,  $\kappa_{\text{gray}} = 10^8$ ,  $\kappa_{\text{pink}} = 100$ .



(b) Algorithmic scalability plot using independent remeshing.



(c) Algorithmic scalability plot using the coarsening strategy with face collapsing.

Elements	Hybrid DoFs	Face unknowns (system size)	Multigrid levels	Iterations	Asymptotic convergence rate
18,170,130	109,008,748	54,498,358	9	18	0.40

(d) Details on the last datapoint of (c)  $k = 1$ .

**FIGURE 12** The heterogeneous domain described by (a) is composed of homogeneous, isotropic regions. Their respective diffusion coefficients are given in the caption. In (b), the coarse meshes are built by retriangulation and the  $L^2$ -orthogonal projection is computed approximately with subtriangulation of the fine elements. In (c), the coarsening strategy is used.

## References

1. Di Pietro DA, Droniou J. *The Hybrid High-Order method for polytopal meshes. Design, analysis, and applications*. No. 19 in Modeling, Simulation and ApplicationSpringer International Publishing . 2020
2. Cockburn B, Di Pietro DA, Ern A. Bridging the Hybrid High-Order and Hybridizable Discontinuous Galerkin methods. *ESAIM: Math. Model Numer. Anal.* 2016; 50(3): 635–650. doi: 10.1051/m2an/2015051
3. Nguyen N, Peraire J, Cockburn B. An implicit high-order hybridizable discontinuous Galerkin method for linear convection-diffusion equations. *Journal of Computational Physics* 2009; 228(9): 3232–3254. doi: 10.1016/j.jcp.2009.01.030
4. Di Pietro DA, Ern A. Arbitrary-order mixed methods for heterogeneous anisotropic diffusion on general meshes. *IMA Journal of Numerical Analysis* 2016; 37(1): 40–63. doi: 10.1093/imanum/drw003
5. Botti L, Di Pietro DA, Droniou J. A Hybrid High-Order method for the incompressible Navier–Stokes equations based on Temam’s device. *Journal of Computational Physics* 2019; 376: 786 - 816. doi: 10.1016/j.jcp.2018.10.014
6. Botti M, Castanon Quiroz D, Di Pietro DA, Harnist A. A Hybrid High-Order method for creeping flows of non-Newtonian fluids. <https://arxiv.org/abs/2003.13467>; 2020.

7. Botti M, Di Pietro DA, Sochala P. A Hybrid High-Order Method for Nonlinear Elasticity. *SIAM Journal on Numerical Analysis* 2017; 55(6): 2687-2717. doi: 10.1137/16M1105943
8. Di Pietro DA, Ern A. Equilibrated tractions for the Hybrid High-Order method. *Comptes Rendus Mathématique* 2015; 353(3): 279 - 282. doi: 10.1016/j.crma.2014.12.009
9. Boffi D, Botti M, Di Pietro DA. A Nonconforming High-Order Method for the Biot Problem on General Meshes. *SIAM Journal on Scientific Computing* 2016; 38(3): A1508-A1537. doi: 10.1137/15M1025505
10. Botti M, Di Pietro DA, Sochala P. A Hybrid High-Order Discretization Method for Nonlinear Poroelasticity. *Computational Methods in Applied Mathematics* 01 Apr. 2020; 20(2): 227-249. doi: 10.1515/cmam-2018-0142
11. Cockburn B, Dubois O, Gopalakrishnan J, Tan S. Multigrid for an HDG method. *IMA Journal of Numerical Analysis* 2014; 34(4): 1386–1425.
12. Kronbichler M, Wall W. A Performance Comparison of Continuous and Discontinuous Galerkin Methods with Fast Multigrid Solvers. *SIAM Journal on Scientific Computing* 2018; 40(5): A3423–A3448. doi: 10.1137/16M110455X
13. Wildey T, Muralikrishnan S, Bui-Thanh T. Unified Geometric Multigrid Algorithm for Hybridized High-Order Finite Element Methods. *SIAM Journal on Scientific Computing* 2019; 41(5): S172–S195. doi: 10.1137/18M1193505
14. Di Pietro DA, Hülsemann F, Matalon P, Mycek P, Råde U, Ruiz D. An  $h$ -multigrid method for Hybrid High-Order discretizations. Submitted. HAL preprint hal-02434411; 2020.
15. Franciolini M, Fidkowski K, Crivellini A. Efficient discontinuous Galerkin implementations and preconditioners for implicit unsteady compressible flow simulations. *arXiv:1812.04789 [physics]* 2018.
16. Schütz J, Aizinger V. A hierarchical scale separation approach for the hybridized discontinuous Galerkin method. *Journal of Computational and Applied Mathematics* 2017; 317: 500–509. doi: 10.1016/j.cam.2016.12.018
17. Botti L, Di Pietro DA.  $p$ -Multilevel preconditioners for HHO discretizations of the Stokes equations with static condensation. Submitted. HAL preprint hal-02951823; 2020.
18. Schoeberl J, Lehrenfeld C. Domain Decomposition Preconditioning for High Order Hybrid Discontinuous Galerkin Methods on Tetrahedral Meshes. In: Apel T, Steinbach O., eds. *Advanced Finite Element Methods and Applications*. 66 of *Lecture Notes in Applied and Computational Mechanics*. Berlin, Heidelberg: Springer. 2013 (pp. 27–56)
19. Tu X, Wang B. A BDDC algorithm for second-order elliptic problems with hybridizable discontinuous Galerkin discretizations. *Electronic Transactions on Numerical Analysis* 2016; 45.
20. Muralikrishnan S, Bui-Thanh T, Shadid JN. A Multilevel Approach for Trace System in HDG Discretizations. *arXiv:1903.11045 [cs, math]* 2019.
21. Bergen BK, Hülsemann F. Hierarchical hybrid grids: data structures and core algorithms for multigrid. *Numerical Linear Algebra with Applications* 2004; 11(2-3): 279–291. doi: <https://doi.org/10.1002/nla.382>
22. Kohl N, Thönnies D, Drzisga D, Bartuschat D, Råde U. The HyTeG finite-element software framework for scalable multigrid solvers. *International Journal of Parallel, Emergent and Distributed Systems* 2019; 34(5): 477–496. doi: 10.1080/17445760.2018.1506453
23. Bauer S, Mohr M, Råde U, Weismüller J, Wittmann M, Wohlmuth B. A two-scale approach for efficient on-the-fly operator assembly in massively parallel high performance multigrid codes. *Applied Numerical Mathematics* 2017; 122: 14–38. doi: 10.1016/j.apnum.2017.07.006
24. Scott LR, Zhang S. Higher-Dimensional Nonnested Multigrid Methods. *Mathematics of Computation* 1992; 58(198): 457–466. doi: 10.2307/2153196
25. Brune PR, Knepley MG, Scott LR. Unstructured Geometric Multigrid in Two and Three Dimensions on Complex and Graded Meshes. *SIAM Journal on Scientific Computing* 2013; 35(1): A173–A191. doi: 10.1137/110827077

26. Todorov TD. The optimal refinement strategy for 3-D simplicial meshes. *Computers & Mathematics with Applications* 2013; 66(7): 1272–1283. doi: 10.1016/j.camwa.2013.07.026
27. John V, Knobloch P, Matthies G, Tobiska L. Non-Nested Multi-Level Solvers for Finite Element Discretisations of Mixed Problems. *Computing* 2002; 68(4): 313–341. doi: 10.1007/s00607-002-1444-2
28. Braess D, Dryja M, Hackbush W. A Multigrid Method for Nonconforming FE-Discretisations with Application to Non-Matching Grids. *Computing* 1999; 63(1): 1–25. doi: 10.1007/s006070050048
29. Adams MF. Parallel multigrid solvers for 3D unstructured finite element problems in large deformation elasticity and plasticity. *International Journal for Numerical Methods in Engineering* 2000; 48(8): 1241–1262. doi: [https://doi.org/10.1002/\(SICI\)1097-0207\(20000720\)48:8<1241::AID-NME946>3.0.CO;2-R](https://doi.org/10.1002/(SICI)1097-0207(20000720)48:8<1241::AID-NME946>3.0.CO;2-R)
30. Feng YT, Perić D, Owen DRJ. A non-nested Galerkin multi-grid method for solving linear and nonlinear solid mechanics problems. *Computer Methods in Applied Mechanics and Engineering* 1997; 144(3): 307–325. doi: 10.1016/S0045-7825(96)01183-8
31. Fish J, Pandheeradi M, Belsky V. An efficient multilevel solution scheme for large scale non-linear systems. *International Journal for Numerical Methods in Engineering* 1995; 38(10): 1597–1610. doi: <https://doi.org/10.1002/nme.1620381002>
32. Antonietti PF, Pennesi G. V-cycle Multigrid Algorithms for Discontinuous Galerkin Methods on Non-nested Polytopic Meshes. *Journal of Scientific Computing* 2019; 78(1): 625–652. doi: 10.1007/s10915-018-0783-x
33. Dickopf T, Krause R. Evaluating Local Approximations of the L2-Orthogonal Projection Between Non-Nested Finite Element Spaces. *Numerical Mathematics: Theory, Methods and Applications* 2014; 7(3): 288–316. doi: 10.1017/S100489790000012X
34. Koobus B, Lallemand MH, Dervieux A. Unstructured volume-agglomeration MG: Solution of the Poisson equation. *International Journal for Numerical Methods in Fluids* 1994; 18(1). doi: 10.1002/flid.1650180103
35. Morano E, Mavriplis DJ, Venkatakrisnan V. Coarsening Strategies for Unstructured Multigrid Techniques with Application to Anisotropic Problems. *SIAM Journal on Scientific Computing* 1998; 20(2): 393–415. doi: 10.1137/S1064827595287638
36. Talischi C, Paulino GH, Pereira A, Menezes IFM. Polygonal finite elements for topology optimization: A unifying paradigm. *International Journal for Numerical Methods in Engineering* 2010; 82(6): 671–698. doi: 10.1002/nme.2763
37. Lévy B, Liu Y. Lp Centroidal Voronoi Tessellation and Its Applications. *ACM Trans. Graph.* 2010; 29(4). doi: 10.1145/1778765.1778856
38. Francescatto J, Dervieux A. A semi-coarsening strategy for unstructured multigrid based on agglomeration. *International Journal for Numerical Methods in Fluids* 1998; 26(8): 927–957. doi: 10.1002/(SICI)1097-0363(19980430)26:8<927::AID-FLD679>3.0.CO;2-0
39. Ollivier-Gooch C. Coarsening unstructured meshes by edge contraction. *International Journal for Numerical Methods in Engineering* 2003; 57(3): 391–414. doi: 10.1002/nme.682
40. Salinas D, Lafarge F, Alliez P. Structure-Aware Mesh Decimation. *Computer Graphics Forum* 2015: 20.
41. Giezeman GJ, Wesselink W. 2D Polygons. In: CGAL Editorial Board. 5.1 ed. 2020.
42. Bey J. Tetrahedral grid refinement. *Computing* 1995; 55(4): 355–378. doi: 10.1007/BF02238487
43. Egger H, Rüde U, Wohlmuth B. Energy-Corrected Finite Element Methods for Corner Singularities. *SIAM Journal on Numerical Analysis* 2014; 52(1): 171–193. doi: 10.1137/120871377

**How to cite this article:** D. A. Di Pietro, F. Hülsemann, P. Matalon, P. Mycek U. Rüde, and D. Ruiz (2021), Toward robust, fast solutions of elliptic equations on complex domains through HHO discretizations and non-nested multigrid methods.