An h-multigrid method for Hybrid High-Order discretizations

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

AN H-MULTIGRID METHOD FOR HYBRID HIGH-ORDER DISCRETIZATIONS *

PIERRE MATALON[†§¶‖], DANIELE A. DI PIETRO[†], FRANK HÜLSEMANN[‡], PAUL MYCEK[§], ULRICH RÜDE[§¶], AND DANIEL RUIZ[‖]

Abstract. We consider a second order elliptic PDE discretized by the Hybrid High-Order method, for which globally coupled unknowns are located at faces. To efficiently solve the resulting linear system, we propose a geometric multigrid algorithm that keeps the degrees of freedom on the faces at every grid level. The core of the algorithm lies in the design of the prolongation operator that passes information from coarse to fine faces through the reconstruction of an intermediary polynomial of higher degree on the cells. High orders are natively handled by the use of the same polynomial degree at every grid level. The proposed algorithm requires a hierarchy of meshes, non necessarily nested, such that the faces (and not only the elements) are successively coarsened. Numerical tests on homogeneous and heterogeneous diffusion problems show fast convergence, scalability in the mesh size and polynomial order, and robustness with respect to heterogeneity of the diffusion coefficient.

Key words. Partial Differential Equations, Hybrid High-Order, Multigrid, static condensation.

AMS subject classifications. 35J15, 65N55, 65N22, 65F50

1. Introduction. We address in this work the solution of large sparse linear systems arising in Hybrid High-Order (HHO) methods. Originally introduced in [12] (see also [13] and the monograph [11]), HHO methods hinge on discrete unknowns that are broken polynomials on the mesh and its skeleton, and are designed so that element-based unknowns are not directly coupled with each other. As a result, the corresponding degrees of freedom (DoFs) can be efficiently eliminated from the linear system by computing a Schur complement element by element, a procedure known in the mechanical literature as *static condensation*. The discrete solution can then be obtained in two steps: first, the Schur complement system is solved, yielding the values of the face unknowns; second, element unknowns are recovered element-wise by solving a small local system. This second step is inexpensive inasmuch as it can be parallelized, leaving the first step as the costliest operation. Consequently, the problem matrix in the context of hybridized methods is usually the Schur complement matrix obtained after static condensation, also called *trace, statically condensed*, or sometimes *Lagrange multiplier system* (referring to the interpretation of face unknowns as Lagrange multipliers enforcing a discrete flux continuity constraint, see [11, Section 5.4.6]). For a more detailed introduction to hybridization, we refer the reader to the first pages of [9] and also [11, Appendix B.3.2]. The defining feature of HHO methods is the embedding of a high-order potential reconstruction into the definition of the discrete bilinear form. As a result, up to one order of convergence is gained with respect to other hybrid methods [7, 10]; see, e.g., the discussion in [6] and also [11,

*Submitted to the editors June 11th, 2020.

Funding: ANR project Fast4HHO under contract ANR-17-CE23-0019.

[†]IMAG, Univ Montpellier, CNRS, Montpellier, France (pierre.matalon@etu.umontpellier.fr, daniele.di-pietro@umontpellier.fr).

[‡]EDF R&D, Paris-Saclay, France (frank.hulsemann@edf.fr)

[§]CERFACS, Toulouse, France (paul.mycek@cerfacs.fr)

[¶]FAU, Erlangen-Nürnberg, Germany (ulrich.ruede@fau.de)

[‖]IRIT, Toulouse, France (daniel.ruiz@enseeiht.fr)

Section 5.1.6].

The main difficulty in designing a geometric h-multigrid algorithm for trace systems lies in the fact that functional spaces on the mesh skeleton may be non-nested when coarsening. This prevents the straightforward construction of a multigrid algorithm based on standard ingredients. Although no existing geometric h-multigrid method has specifically targeted HHO so far, a few trace system solvers have been designed over the last years. In [8], the authors propose a geometric multigrid solver for general Hybridizable Discontinuous Galerkin discretizations of elliptic equations and low order approximation, where the trace functions are recast into functions defined over the elements in order to make use of a known efficient solver, typically a standard piecewise linear continuous Finite Element multigrid solver. This special multigrid method actually takes its origin from the one previously designed for hybridized versions of the Raviart–Thomas and Brezzi–Douglas–Marini methods in [15], from which the intergrid transfer operators are borrowed. A variation using an underlying algebraic multigrid method instead of a geometric one was experimented in [19]. A different approach is then considered in [28], where an hp-multigrid algorithm based on trace functions at every level is proposed: it handles unstructured polyhedral meshes and is based on the use of Dirichlet-to-Neumann maps to preserve energy from coarse to fine levels. The management of high orders is carried out in the traditional way of putting a p-multigrid algorithm on top of a multigrid iteration in h. Two p-multigrid algorithms can also be cited [14, 24] as solvers used on condensed systems. Additionally, besides multigrid methods, we can also refer to domain decomposition methods [23, 27] and nested dissection [20].

In this paper, we develop a novel geometric h-multigrid algorithm (i) based on approximation spaces supported by the mesh skeleton at every level, (ii) targeting HHO discretizations by making use of the underlying high-order potential reconstruction, (iii) natively managing higher orders (as opposed to, e.g., putting a p-multigrid on top of an h- one). Our algorithm development is based on the systematic approach proposed in the seminal guide to multigrid development [4]. The method consists in identifying the individual difficulties and obstacles that may inhibit the optimal performance of a multigrid algorithm. For each of the difficulties, appropriate multigrid components are developed. Here, in particular, we start from the Laplace problem discretized on the skeleton of a simple Cartesian mesh to first develop a multigrid method that is scalable in the number of unknowns and robust with respect to the polynomial degree. With this algorithm, we then proceed to work on more general problems and meshes. One consequence of this approach is that we focus in this article on multigrid as a solver, not as a preconditioner. When used only as preconditioner, this tends to obscure misconceptions in the design of the multigrid components. The multigrid algorithms developed here can serve as efficient stand-alone solvers, but they can also serve as preconditioners, as we will explore in future research.

In our multigrid method, the polynomial order of approximation is preserved at every level at the sole cost of using a blockwise smoother instead of a pointwise one. This approach originates from the remark that a high-order finite element discretization yields a block matrix, whose diagonal blocks are formed by the degrees of freedom connected to the same cell. This configuration usually destroys the desirable M- or H-matrix structure and, along with it, the convergence of pointwise smoothers; on the other hand, the block structure paves the way to using block versions of similar smoothers. In a more functional way of thinking, relaxing together the DoFs related to the same polynomial comes as intuitive. The robustness of the multigrid algorithms using block smoothers for high-order methods has been experimentally illustrated in

[17] and later used in practical solvers such as [21].

The rest of this work is organized as follows. Section 2 summarizes the basics of the HHO method. Section 3 is devoted to the construction of the multigrid algorithm and illustrates how it takes advantage of the HHO potential reconstruction operator. Numerical results for various polynomial degrees are presented in section 4, considering both homogeneous and heterogeneous diffusion problems in two and three space dimensions. The numerical experiments show that the number of iterations is nearly independent of the mesh size and of the presence of jumps in the diffusion coefficient. Finally, future research directions are discussed in conclusion.

2. HHO formulation.

2.1. Notation. Let $d \in \{2,3\}$ be the space dimension and Ω a bounded polyhedral domain of $\mathbb{R}^d$. We consider a mesh $(\mathcal{T}_h, \mathcal{F}_h)$ of Ω in the sense of [11, Definition 1.4], with $\mathcal{T}_h$ denoting the set of polyhedral elements, $\mathcal{F}_h$ the set of faces, and $h := \max_{T\in\mathcal{T}_h} \text{diameter}(T)$ the mesh size. The set $\mathcal{F}_h$ is partitioned as $\mathcal{F}_h^{\mathrm{I}} \cup \mathcal{F}_h^{\mathrm{B}}$, where $\mathcal{F}_h^{\mathrm{I}}$ denotes the set of internal faces and $\mathcal{F}_h^{\mathrm{B}}$ the set of boundary faces. For all $T \in \mathcal{T}_h$, $\mathcal{F}_T$ collects the mesh faces lying in the boundary of T. Reciprocally, given a face $F \in \mathcal{F}_h$, $\mathcal{T}_F$ collects the elements which F is a face of. (Note that $\text{card}(\mathcal{T}_F) = 2$ for internal faces and $\text{card}(\mathcal{T}_F) = 1$ for boundary faces.) For all $T \in \mathcal{T}_h$ and $F \in \mathcal{F}_T$, $\mathbf{n}_{TF}$ denotes the unit vector normal to F pointing out of T. For $X \subset \Omega$, $L^2(X)$ denotes the Hilbert space of square-integrable functions over X, equipped with its usual inner product $(u,v)_X := \int_X uv$. The same notation is also used for the inner product of $[L^2(X)]^d$ (the exact meaning can be inferred from the context). Additionally, we denote by $H^1(X)$ the space spanned by functions of $L^2(X)$ whose partial derivatives are also square-integrable, and by $H^1_0(X)$ its subspace with vanishing trace on the boundary ∂X of X. Finally, $\mathbb{P}^m(X)$ is the space spanned by the restriction to X of d-variate polynomials of degree at most $m \in \mathbb{N}$ (in short, polynomials of degree m).

2.2. Model problem. Given a source function $f \in L^2(\Omega)$, we consider the following diffusion problem with homogeneous Dirichlet boundary conditions:

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

where the diffusion tensor $\mathbf{K}\colon \Omega \to \mathbb{R}^{d\times d}_{\mathrm{sym}}$ (with $\mathbb{R}^{d\times d}_{\mathrm{sym}}$ collecting symmetric $d\times d$ real matrices) is assumed uniformly elliptic and piecewise constant over a fixed partition of Ω into polyhedra. The variational formulation of problem (2.1) reads

$$\begin{gathered} \text{Find } u \in H^1_0(\Omega) \text{ such that} \\ a(u,v) = \int_\Omega fv \quad \forall v \in H^1_0(\Omega), \end{gathered} \tag{2.2}$$

where the bilinear form $a\colon H^1(\Omega)\times H^1(\Omega) \to \mathbb{R}$ is such that, for all $v, w \in H^1(\Omega)$,

$$a(v,w) := (\mathbf{K}\nabla v, \nabla w)_\Omega = \int_\Omega \mathbf{K}\nabla u\cdot\nabla v.$$

2.3. Discrete spaces and operators. We briefly recall the standard HHO discretization of problem (2.2) and refer to [11, Section 3.1] for a more comprehensive presentation. From this point on, we assume that $\mathcal{T}_h$ partitions Ω in such a way that the diffusion tensor is constant inside each element, and we let $\mathbf{K}_T := \mathbf{K}|_T$ for all $T \in \mathcal{T}_h$.

The HHO method is based on discrete unknowns at cells and faces, and the adjective *hybrid* refers to their union, in spite of their different natures, to form one set of unknowns. Let an integer $k \geq 0$ be fixed. We introduce the following broken polynomial spaces, respectively supported by the mesh and its skeleton:

$$U_{\mathcal{T}_h}^m := \{v_{\mathcal{T}_h} := (v_T)_{T\in\mathcal{T}_h} \mid v_T \in \mathbb{P}^m(T) \ \ \forall T \in \mathcal{T}_h\} \qquad \text{for } m \in \{k, k+1\},$$
$$U_{\mathcal{F}_h}^k := \left\{v_{\mathcal{F}_h} := (v_F)_{F\in\mathcal{F}_h} \mid v_F \in \mathbb{P}^k(F) \ \ \forall F \in \mathcal{F}_h\right\},$$

from which we build the global space of hybrid variables

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

The homogeneous Dirichlet boundary condition is strongly enforced in the following subspaces of $U_{\mathcal{F}_h}^k$ and $\underline{U}_h^k$:

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

For any $X \in \mathcal{T}_h \cup \mathcal{F}_h$, denote by $\pi_X^k \colon L^2(X) \to \mathbb{P}^k(X)$ the L^2-orthogonal projector on $\mathbb{P}^k(X)$ (cf., e.g., [11, Definition 1.36]). According to [11, Eq. (2.34)], the element of $\underline{U}_h^k$ associated to a function $v \in H^1(\Omega)$ through its interpolator is $\big((\pi_T^k v)_{T\in\mathcal{T}_h}, (\pi_F^k v)_{F\in\mathcal{F}_h}\big)$, showing that the element and face unknowns of the HHO method (2.6) can be interpreted as local L^2-orthogonal projections of the exact solution.

For all $T \in \mathcal{T}_h$, denote by $\underline{U}_T^k$ the restriction of $\underline{U}_h^k$ to T, that is,

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

We define the *local potential reconstruction* $p_T^{k+1} \colon \underline{U}_T^k \to \mathbb{P}^{k+1}(T)$ 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

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

$$(p_T^{k+1}\underline{v}_T, 1)_T = (v_T, 1)_T. \tag{2.4b}$$

It can be checked that, for any $v \in H^1(T)$, applying p_T^{k+1} to the interpolate of v yields the local oblique elliptic projection of v on $\mathbb{P}^{k+1}(T)$; see [11, Section 3.1.2].

2.4. HHO discretization of the model problem. The global bilinear form $a_h \colon \underline{U}_h^k \times \underline{U}_h^k \to \mathbb{R}$ 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 $a_T \colon \underline{U}_T^k \times \underline{U}_T^k \to \mathbb{R}$ is defined as

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

In this expression, the first term is responsible for consistency while the second, involving the bilinear form $s_T \colon \underline{U}_T^k \times \underline{U}_T^k \to \mathbb{R}$, is required to ensure stability of the scheme. The global discrete problem then reads

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

REMARK 1 (Stabilization bilinear form s_T). *Design conditions for the stabilization bilinear form s_T are provided in [11, Assumption 3.9]. These conditions imply, in particular, that s_T must depend on its argument only through the* difference operators $\delta_T^k : \underline{U}_T^k \to \mathbb{P}^k(T)$ *and, for all $F \in \mathcal{F}_T$, $\delta_{TF}^k : \underline{U}_T^k \to \mathbb{P}^k(F)$ such that, for all $\underline{v}_T \in \underline{U}_T^k$,*

$$\delta_T^k \underline{v}_T := \pi_T^k(p_T^{k+1}\underline{v}_T - v_T) \text{ and } \delta_{TF}^k \underline{v}_T := \pi_F^k(p_T^{k+1}\underline{v}_T - v_F) \text{ for all } F \in \mathcal{F}_T.$$

These operators capture the higher-order correction that the reconstruction p_T^{k+1} adds to the element and face unknowns, respectively. A classical expression for s_T is the following:

$$s_T(\underline{v}_T, \underline{w}_T) := \sum_{F\in\mathcal{F}_T} \frac{K_{TF}}{h_F}((\delta_{TF}^k - \delta_T^k)\underline{v}_T, (\delta_{TF}^k - \delta_T^k)\underline{w}_T)_F,$$

where $K_{TF} := \mathbf{K}_T\mathbf{n}_{TF} \cdot \mathbf{n}_{TF}$ for all $F \in \mathcal{F}_T$.

2.5. Assembly and static condensation. The local contributions corresponding to the representations, in the selected basis for $\underline{U}_{h,0}^k$, 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

$$\mathbf{A}_T := \begin{pmatrix} \mathbf{A}_{TT} & \mathbf{A}_{T\mathcal{F}_T} \\ \mathbf{A}_{\mathcal{F}_T T} & \mathbf{A}_{\mathcal{F}_T\mathcal{F}_T} \end{pmatrix}, \qquad \mathbf{B}_T := \begin{pmatrix} \mathbf{b}_T \\ \mathbf{0} \end{pmatrix}, \tag{2.7}$$

in which the unknowns have been numbered so that cell unknowns come first and face unknowns come last; see [11, Appendix B] 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

$$\begin{pmatrix} \mathbf{A}_{\mathcal{T}_h\mathcal{T}_h} & \mathbf{A}_{\mathcal{T}_h\mathcal{F}_h^{\mathrm{I}}} \\ \mathbf{A}_{\mathcal{F}_h^{\mathrm{I}}\mathcal{T}_h} & \mathbf{A}_{\mathcal{F}_h^{\mathrm{I}}\mathcal{F}_h^{\mathrm{I}}} \end{pmatrix} \begin{pmatrix} \mathbf{v}_{\mathcal{T}_h} \\ \mathbf{v}_{\mathcal{F}_h^{\mathrm{I}}} \end{pmatrix} = \begin{pmatrix} \mathbf{b}_{\mathcal{T}_h} \\ \mathbf{0} \end{pmatrix}. \tag{2.8}$$

Since cell-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 cell-DoFs: it goes by expressing $\mathbf{v}_{\mathcal{T}_h}$ in terms of $\mathbf{v}_{\mathcal{F}_h^{\mathrm{I}}}$ in the first equation of (2.8):

$$\mathbf{v}_{\mathcal{T}_h} = -\mathbf{A}_{\mathcal{T}_h\mathcal{T}_h}^{-1}\mathbf{A}_{\mathcal{T}_h\mathcal{F}_h^{\mathrm{I}}}\mathbf{v}_{\mathcal{F}_h^{\mathrm{I}}} + \mathbf{A}_{\mathcal{T}_h\mathcal{T}_h}^{-1}\mathbf{b}_{\mathcal{T}_h}, \tag{2.9}$$

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

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

thus yielding a smaller system, involving only face unknowns. The main advantage of this technique is the reduction of the problem size, especially for high polynomial degrees k.

3. Multigrid algorithm. In this section, we present a geometric multigrid algorithm to efficiently solve the condensed system (2.10). The method we propose hinges on face-defined functions at every grid level, and works in synergy with the discretization through intergrid transfer operators leveraging the potential reconstruction (2.4). The algorithm applies to any polynomial degree k without resorting to an additional p-multigrid, which, in practice, can be seen as a valuable reduction of the implementation cost.

3.1. Coarsening strategy. The levels of the multigrid method are numbered from 1 to L, L being the finest and 1 the coarsest. In what follows, we denote by ℓ the generic level and by h_ℓ the corresponding mesh size. To simplify the notation, from this point on h_ℓ is replaced by ℓ in subscripts so we write, e.g., $\mathcal{T}_\ell$ instead of $\mathcal{T}_{h_\ell}$, $\mathcal{F}_\ell$ instead of $\mathcal{F}_{h_\ell}$, and so on.

Relative to those levels, we consider a hierarchy of nested polyhedral meshes $(\mathcal{T}_\ell, \mathcal{F}_\ell)_{\ell=1\ldots L}$. We assume the hierarchy to successively coarsen not only elements, but also faces. This means that, for all $\ell = 1\ldots L$, letting $h_{\mathcal{T}_\ell} := \max_{T\in\mathcal{T}_\ell} h_T$ and $h_{\mathcal{F}_\ell} := \max_{F\in\mathcal{F}_\ell} h_F$, it holds:

$$h_{\mathcal{T}_{\ell-1}} > h_{\mathcal{T}_\ell}, \qquad h_{\mathcal{F}_{\ell-1}} > h_{\mathcal{F}_\ell}.$$

Standard coarsening of structured Cartesian and triangular meshes, as well as unstructured meshes obtained from successive structured refinements of an initial coarse mesh fall under the scope of these assumptions; examples of admissible coarsening strategies are illustrated in Figure 3.1. Requiring that the faces be coarsened is justified by our algorithm being face-defined at every level. Indeed, the smoother applies to faces the same way it applies to elements in a classical element-defined multigrid method: once the high frequencies of the error have been annihilated on the fine mesh, the smoother requires coarser elements to reach the low frequencies on the coarse mesh. For the same reason, a multigrid working on the mesh skeleton needs the faces to be coarsened: the consequence of a face not being coarsened between a fine and a coarse mesh would be to keep the smoother working on the same range of frequencies, leaving it unable to efficiently reduce the lowest ones; see Figure 4.10 below.

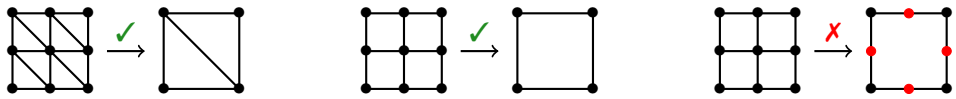


Fig. 3.1: Coarsening examples. The first two are admissible, whereas the third one is not: edges have been removed, but none of the remaining ones has been coarsened.

We will also assume that, for every $\ell = 1\ldots L$, the diffusion coefficient is piecewise constant on $\mathcal{T}_\ell$, so that jumps can occur across faces but not inside elements.

3.2. Prolongation. We consider two successive levels ℓ (fine) and $\ell-1$ (coarse). In this algorithm, *faces* support the functions at every level. To prolongate a coarse function onto the fine mesh skeleton, which includes some faces that are not present in the coarse mesh, we propose an intermediary step that passes through the cells (Figure 3.2). Following this idea, the prolongation operator $P\colon U^k_{\mathcal{F}_{\ell-1},0} \to U^k_{\mathcal{F}_\ell,0}$ is defined as the composition

$$P = \Pi^\ell_{\ell-1} \circ \Theta_{\ell-1}, \tag{3.1}$$

where the coarse level potential reconstruction operator $\Theta_{\ell-1}\colon U^k_{\mathcal{F}_{\ell-1},0} \to U^{k+1}_{\mathcal{T}_{\ell-1}}$ reconstructs a broken polynomial of degree $k+1$ on $\mathcal{T}_{\ell-1}$ from face unknowns; then, the trace prolongation operator $\Pi^\ell_{\ell-1}\colon U^{k+1}_{\mathcal{T}_{\ell-1}} \to U^k_{\mathcal{F}_\ell,0}$ maps the polynomials of degree $k+1$ defined on the coarse cells to a broken polynomial function of degree k on the fine skeleton.

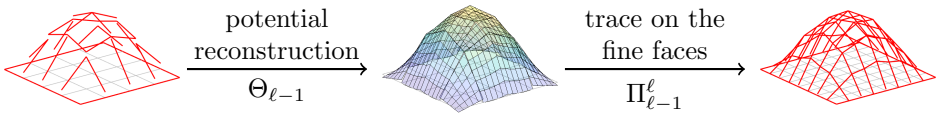


Fig. 3.2: Prolongation from coarse to fine edges.

3.2.1. Θ_ℓ: from faces to cells. This operator is at the core of the algorithm and is what makes it original. Given a trace error function $e_{\mathcal{F}_\ell} \in U^k_{\mathcal{F}_\ell,0}$ as the operand of Θ_ℓ, we retrieve the associated cell-defined error function $e_{\mathcal{T}_\ell} \in U^{k+1}_{\mathcal{T}_\ell}$ by first reversing the static condensation process, then take advantage of the potential reconstruction operator defined by (2.4) to gain one order of approximation inside the cells.

As these operations are local, the process will be outlined for a generic mesh element $T \in \mathcal{T}_\ell$. Defining $e_{\mathcal{F}_T} := (e_F)_{F\in\mathcal{F}_T}$, we let $\mathbf{e}_T$ and $\mathbf{e}_{\mathcal{F}_T} := (\mathbf{e}_F)_{F\in\mathcal{F}_T}$ denote the algebraic representations of e_T and $e_{\mathcal{F}_T}$, respectively, as vectors of coefficients in the selected polynomial bases. If we denote by $(\mathbf{x}_{\mathcal{T}_\ell}^\top, \mathbf{x}_{\mathcal{F}_\ell}^\top)^\top$ the vector obtained completing the solution vector of the global system (2.8) with boundary unknowns equal to zero, then its restriction to T, namely $(\mathbf{x}_T^\top, \mathbf{x}_{\mathcal{F}_T}^\top)^\top$, is the solution of the local system defined by (2.7), i.e.

$$\begin{pmatrix} \mathbf{A}_{TT} & \mathbf{A}_{T\mathcal{F}_T} \\ \mathbf{A}_{\mathcal{F}_T T} & \mathbf{A}_{\mathcal{F}_T\mathcal{F}_T} \end{pmatrix} \begin{pmatrix} \mathbf{x}_T \\ \mathbf{x}_{\mathcal{F}_T} \end{pmatrix} = \begin{pmatrix} \mathbf{b}_T \\ \mathbf{0} \end{pmatrix},$$

from which the static condensation process expresses $\mathbf{x}_T$ in terms of $\mathbf{x}_{\mathcal{F}_T}$ as

$$\mathbf{x}_T = -\mathbf{A}_{TT}^{-1}\mathbf{A}_{T\mathcal{F}_T}\mathbf{x}_{\mathcal{F}_T} + \mathbf{A}_{TT}^{-1}\mathbf{b}_T. \tag{3.2}$$

We now introduce the local face-defined approximate solution vector $\tilde{\mathbf{x}}_{\mathcal{F}_T}$ such that $\mathbf{e}_{\mathcal{F}_T} = \mathbf{x}_{\mathcal{F}_T} - \tilde{\mathbf{x}}_{\mathcal{F}_T}$, and, inspired by (3.2), we define the associated cell-based approximate vector $\tilde{\mathbf{x}}_T$ by

$$\tilde{\mathbf{x}}_T := -\mathbf{A}_{TT}^{-1}\mathbf{A}_{T\mathcal{F}_T}\tilde{\mathbf{x}}_{\mathcal{F}_T} + \mathbf{A}_{TT}^{-1}\mathbf{b}_T. \tag{3.3}$$

Definition (3.3) ensures consistency in the sense that for $\tilde{\mathbf{x}}_{\mathcal{F}_T} = \mathbf{x}_{\mathcal{F}_T}$, it yields $\tilde{\mathbf{x}}_T = \mathbf{x}_T$ by (3.2). We can finally define the error on the cell by setting $\mathbf{e}_T := \mathbf{x}_T - \tilde{\mathbf{x}}_T$, and replace $\mathbf{x}_T$ and $\tilde{\mathbf{x}}_T$ with their respective expressions (3.2) and (3.3), thus cancelling the terms involving $\mathbf{b}_T$ and giving

$$\mathbf{e}_T = -\mathbf{A}_{TT}^{-1}\mathbf{A}_{T\mathcal{F}_T}(\mathbf{x}_{\mathcal{F}_T} - \tilde{\mathbf{x}}_{\mathcal{F}_T}) = -\mathbf{A}_{TT}^{-1}\mathbf{A}_{T\mathcal{F}_T}\mathbf{e}_{\mathcal{F}_T}. \tag{3.4}$$

Once e_T is retrieved from (3.4), the local potential reconstruction p_T^{k+1} defined in subsection 2.3 is applied to the hybrid vector $(e_T, e_{\mathcal{F}_T})$ to obtain an approximate error of degree $k+1$ on the cell:

$$(\Theta_\ell e_{\mathcal{F}_\ell})_{|T} := p_T^{k+1}(e_T, e_{\mathcal{F}_T}) \qquad \forall T \in \mathcal{T}_h. \tag{3.5}$$

Figure 3.3 summarizes the process.

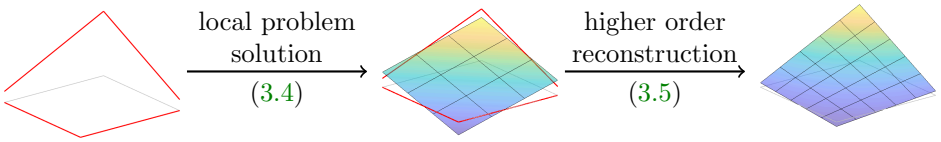


Fig. 3.3: Reconstruction of a polynomial of degree $k+1$ from polynomials of degree k (here for $k=1$) on the four edges of a 2D square element.

3.2.2. $\Pi_{\ell-1}^{\ell}$: from cells to faces. For any $v \in U_{\mathcal{T}_{\ell-1}}^{k+1}$ and any $F \in \mathcal{F}_\ell$, $(\Pi_{\ell-1}^{\ell} v)_{|F}$ is built as the L^2-orthogonal projection on $\mathbb{P}^k(F)$ of the weighted average of the traces of v on both sides of F if F is an internal face, while $(\Pi_{\ell-1}^{\ell} v)_{|F}$ is set equal to zero if F is a boundary face, i.e.,

$$(\Pi_{\ell-1}^{\ell} v)_{|F} := \begin{cases} w_{T_1F}\, \pi_F^k(v_{|T_1})_{|F} + w_{T_2F}\, \pi_F^k(v_{|T_2})_{|F} & \text{if } F \in \mathcal{F}_\ell^{\mathrm{I}}, \\ 0 & \text{otherwise,} \end{cases} \tag{3.6}$$

where T_1, T_2 denote the distinct elements in $\mathcal{T}_F \subset \mathcal{T}_\ell$, π_F^k is the L^2-projector on $\mathbb{P}^k(F)$, and the weights satisfy

$$w_{T_1F} + w_{T_2F} = 1 \tag{3.7a}$$

$$\frac{w_{T_1F}}{w_{T_2F}} = \frac{K_{T_1F}}{K_{T_2F}}, \tag{3.7b}$$

where we remind the reader that, for $i \in \{1,2\}$, $K_{T_iF} := \mathbf{K}_{T_i} \mathbf{n}_{T_iF} \cdot \mathbf{n}_{T_iF}$. Enforcing both constraints (3.7) yields, for $i \in \{1,2\}$,

$$w_{T_iF} := \frac{K_{T_iF}}{K_{T_1F} + K_{T_2F}}. \tag{3.8}$$

3.3. Multigrid components. The prolongation operator P is defined by (3.1). The restriction operator R is defined as the adjoint of P in the usual way. Interpreted algebraically as matrices (using the notations $\mathbf{R}$ and $\mathbf{P}$), it means $\mathbf{R} = \mathbf{P}^\top$. Note that $\Pi_{\ell-1}^{\ell}$ does not make a distinction between the fine faces contained in the skeleton of the coarse grid and those that are not; consequently, the polynomials on coarse faces are not transferred identically to the fine grid, but instead take on new values coming from the (weighted) average of the reconstructed cell-polynomials on each side. The alternative way of prolongating coarse functions from coarse faces to their respective identical fine faces, namely keeping them unchanged, has also been tested (cf. subsection 4.4) and yields a less efficient algorithm. This observation is consistent with the fact that solving the local problems produces additional information that the coarse polynomials do not possess. In addition, the reconstruction using higher degree polynomials also results in higher accuracy in the case where two fine faces are agglomerated into a single coarse one: the polynomial of degree $k+1$ on the coarse cell can induce two different polynomials of degree k on the two corresponding fine faces, which would not be the case if the reconstruction were only of degree k.

The coarse grid operator at level $\ell-1$ can be chosen either as the discretized operator on the respective coarse mesh, or as the Galerkin construction: $A_{\ell-1} := RA_\ell P$. The numerical tests in the next section show equivalent performances.

In order to relax the DoFs related to the same polynomial function together, block versions of standard fixed-point smoothers are chosen, whose block size corresponds to the number of DoFs on each face.

4. Numerical results.

4.1. Experimental setup. The numerical tests have been performed on the diffusion problem (2.1) in various d-dimensional domains $\Omega \subset \mathbb{R}^d$, $d \in \{2, 3\}$. The unit square/cube $\Omega := (0, 1)^d$ is used to study the algorithm on structured meshes, whereas more complicated geometries shall be used for unstructured ones. The source function f is chosen so that the analytical solution of the homogeneous problem corresponds to $(x, y) \mapsto \sin(4\pi x)\sin(4\pi y)$ in 2D and $(x, y, z) \mapsto \sin(4\pi x)\sin(4\pi y)\sin(4\pi z)$ in 3D. For structured cases, given an integer $N > 0$, the domain is discretized by a Cartesian grid composed of N^d square/cubic elements of side length $1/N$. Each of them is respectively decomposed into 2 triangles or 6 tetrahedra if the mesh is simplicial. In what follows, k denotes the polynomial degree on the faces (meaning that the HHO method ultimately yields an approximation of degree $k+1$). Our multigrid algorithm is used to solve the statically condensed linear system (2.10). The mesh is successively coarsened until the coarse system reaches a size with less than 1000 unknowns. On the coarsest level, the system is solved by a direct solver. The operators on the coarser levels are constructed directly as the discretization of the equation on the respective coarse meshes. The smoother is a block Gauss–Seidel method, in which the block size corresponds to the number of face-DoFs. In pre-smoothing, the iteration is performed in lexicographic order, while in post-smoothing, it is performed in anti-lexicographic order to ensure the symmetry of the overall iteration. The multigrid cycles will vary depending on the test. An L^2-orthogonal Legendre basis is chosen to represent the local polynomials on cells and faces. The stopping criterion is set to $\|\mathbf{r}\|_2/\|\mathbf{b}\|_2 < 10^{-8}$, where $\mathbf{r}$ denotes the residual vector, $\mathbf{b}$ the right-hand side of the linear system, and $\|\cdot\|_2$ the Euclidean norm on the vector space of coordinates.

4.2. Homogeneous diffusion on structured meshes. The diffusion tensor field is constant across the domain and equals the identity matrix. The model problem is discretized using four structured meshes: Cartesian and triangular in 2D, Cartesian and tetrahedral in 3D. The mesh hierarchies are constructed from the fine mesh by standard coarsening. This strategy ensures the hierarchical nestedness as well as geometrically similar elements at every level. Note that the tetrahedral meshes are built from the Cartesian ones, where each cube is divided into six geometrically similar tetrahedra (cf [3, Figure 9]).

4.2.1. Preliminary tests using classical multigrid cycles. Figure 4.1 presents performance results of the multigrid algorithm as a solver, using the cheapest symmetric V-cycle that ensures convergence in a reasonable number of iterations as well as good scalability, namely V(1,1) for 2D meshes and the 3D Cartesian one, V(2,2) for the tetrahedral mesh. Leaving the lowest order case aside for the moment, these results are consistent with the desired multigrid property of a convergence rate that is independent of the mesh size and the number of levels, provided that a sufficient (yet reasonable) number of smoothing steps are performed. Moreover, although the number of iterations may increase moderately with the polynomial degree, the algorithm still exhibits the same desirable properties for high approximation orders.

For the lowest order case $k = 0$, the results are plotted throughout the tests in dashed lines. Here the results are less clear. Although in case of $k = 0$ we still observe good scalability on Cartesian meshes, the convergence on the triangular meshes deteriorates with growing mesh size. For the tetrahedral mesh in 3D and $k = 0$ no data is shown in Figure 4.1 since this version does not converge with the V(2,2) cycle. Here, more smoothing steps would be needed to ensure convergence.

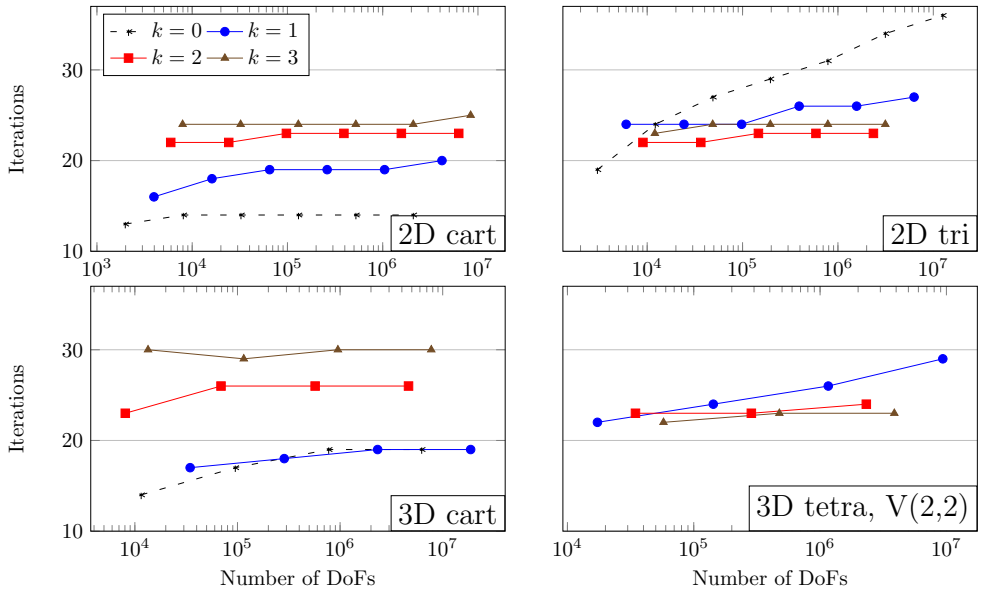


Fig. 4.1: Number of iterations to achieve convergence for the homogeneous problem on structured meshes: 2D Cartesian (top left), 2D triangular (top right), 3D Cartesian (bottom left), 3D tetrahedral (bottom right). The first three are solved using the V(1,1) cycle, while the last one (tetrahedral mesh) is solved using the V(2,2) cycle.

4.2.2. Multigrid cycle optimization. While the above results demonstrate the asymptotic optimality of our new multigrid algorithm, we now proceed to studying how the inherent design options of multigrid can be used to further improve the real-life efficiency. In particular, we identify the most efficient cycle structure and how much pre- and post-smoothing should be performed. To assess the performance impact of these choices, it is necessary to define a criterion modeling the trade-off between convergence rate and iteration cost. We emphasize that the sole number of iterations is not sufficient to assess the solver's overall efficiency, because the cost of each iteration must be taken into account. Hence, Figure 4.2 compares the performance, measured in total computational work to reach convergence, of different multigrid cycles on a 2D test problem (triangular mesh, $N = 512$, $k = 1$). In the left plot, the numerical values have been obtained by taking the theoretical computational work (in flops) of the multigrid algorithm, using the following simplifying rules: (i) the asymptotic value of the work count is used, meaning that only the dominant term (in the matrix size or non-zero entries) is kept; (ii) the work of the direct solver on the coarsest grid is neglected. The total number of iterations required to achieve convergence is displayed for information in the right plot. Recalling that all tests stop upon reaching the same convergence criterion, we consider all the solutions produced to be equivalent: for instance, V(1,1) is about 50% more computationally expensive than V(0,2) for the same quality result. Note that V- and W-cycles have been tested, and exhibit, for the same numbers of smoothing steps, the same convergence rate. Since W-cycles are more computationally expensive by definition, the corresponding results are not presented in further detail. The comparisons have also been made in terms of CPU time in order to compare the estimates of computational work with

respect to a hard practical criterion. Again, these results show a similar ranking and allow to draw the same conclusions; so they are not displayed in detail either.

Now, we can comment on the importance of post-smoothing: for example, amongst V(1,2), V(2,1) and V(0,3), although they all have the same total number of smoothing steps, and consequently the same cost per cycle iteration, V(0,3) is found to be the most efficient. More generally, among all cycles V(ν_1,ν_2) with $\nu_1+\nu_2=\nu$, the option $\nu_1=0$ and $\nu_2=\nu$ appears to be the most efficient. Moreover, we find that the extra cost of more post-smoothing is compensated to a great extent by a better convergence rate. Particularly, moving away from the sweet spot V(0,ν), e.g. by taking V(0,$\nu+1$) instead, only induces a minor overhead, which grants a pragmatic flexibility in the actual choice of the number of post-smoothing steps.

Figure 4.3 presents the same tests in 3D on the tetrahedral mesh. They also clearly show the superiority of cycles with post-smoothing only. Since both the lexicographic and the antilexicographic Gauss-Seidel smoothers depend on the numbering of the DoFs, we have checked whether these observations depend on a particular numbering of the unknowns. To this end, we have additionally performed experiments with the damped block Jacobi smoother with $\omega=2/3$ as the under-relaxation parameter (see Figure 4.4). This test leads to the same qualitative conclusions, but with a milder quantitative effect. Based on these measures, we settle for V(0,3) in 2D and V(0,6) in 3D as the most efficient cycles when using block Gauss-Seidel. In Figure 4.5 we present the results of the same scalability tests as in Figure 4.1 using these "optimized" cycles. Besides the expected improved convergence rates, we point out that the iteration count for different polynomial degrees are now almost the same: in all cases, the number of iterations lies in a narrow interval regardless of the polynomial degree. Again the lowest order on the tetrahedral mesh constitutes an exception, since the method still diverges in that case.

The multigrid algorithm can be used as a preconditioner for Krylov-space methods and in particular for the Conjugate Gradient (CG) method. In the latter case, the algorithm requires formally a symmetric positive definite preconditioner to ensure convergence. Consequently, the choice of a symmetric and therefore suboptimal multigrid cycle seems to be necessary, unless the conditions of [16] are met. A thorough investigation of the multigrid algorithm of this article as preconditioner is outside the scope of this article.

4.3. Heterogeneous diffusion. The domain is split into four quadrants as illustrated in Figure 4.6a. The heterogeneity pattern is such that each pair of opposite quadrants have the same, homogeneous, diffusion coefficient. On each homogeneous part Ω_i, $i=1,2$, the diffusion tensor is defined as $\mathbf{K}|_{\Omega_i}:=\kappa_i\mathbf{I}_d$, where κ_i is a positive scalar constant and $\mathbf{I}_d$ denotes the identity matrix of size d.

Our first test evaluates the convergence rate for varying values of the coefficient ratio $\rho_{\mathbf{K}}:=\kappa_1/\kappa_2$ in the range $1\leq\rho_{\mathbf{K}}\leq 10^8$. The results demonstrate robustness of the algorithm with respect to the heterogeneity. Regardless of the magnitude of the coefficient ratio, the convergence rate remains unchanged and matches that of the homogeneous case.

In [18], Kellogg studied the analytical solution of a specific case of such a configuration. The source function is set $f\equiv 0$ and non-homogeneous Dirichlet boundary conditions are imposed. The particular solution u exhibits a singularity at the center of the square and has reduced regularity $u\in H^{1+\epsilon}$, $0<\epsilon\leq 1$. Since the strength of the singularity and thus the regularity $1+\epsilon$ can be adjusted via the coefficient ratio, this problem is often used to benchmark discretizations and solvers. Here we set the

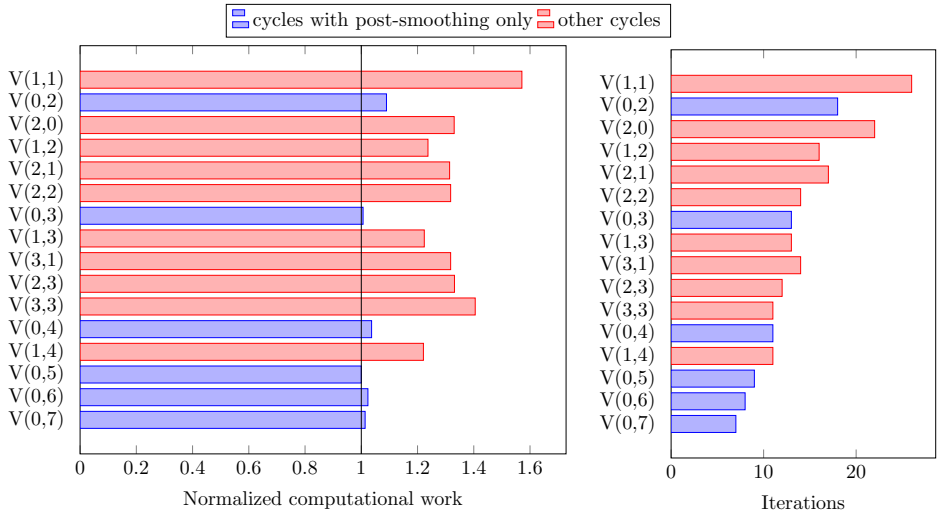


Fig. 4.2: Cycle comparison on the 2D test problem, triangular mesh, $N = 512$, $k = 1$ ($\approx 1.6 \times 10^6$ DoFs). The pre-smoother is the lexicographic block Gauss-Seidel, the post-smoother is the antilexicographic block Gauss-Seidel. In the first plot, the numerical values in flops of the computational work are normalized by the lowest one. V(0,1) and V(1,0), being very inefficient in comparison to the others, are not presented here.

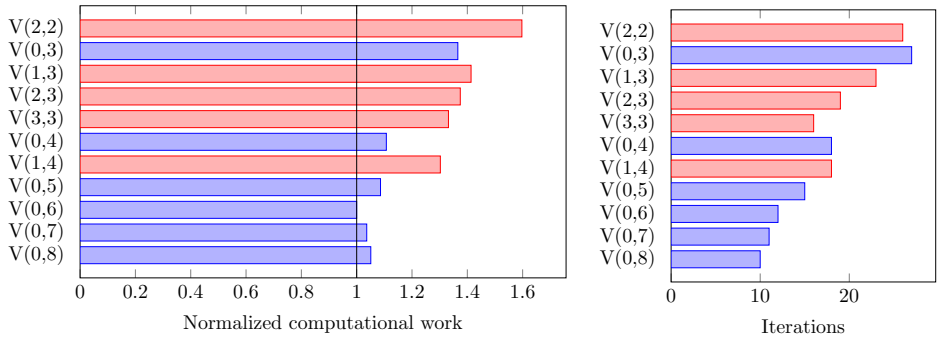


Fig. 4.3: Cycle comparison on the 3D test problem, tetrahedral mesh, $N = 32$, $k = 1$ ($\approx 1.2 \times 10^6$ DoFs). The pre-smoother is the lexicographic Block Gauss-Seidel, the post-smoother is the antilexicographic Gauss-Seidel. In the first plot, the numerical values in flops of the computational work are normalized by the lowest one. The first cycles, with less than 3 or 4 total iterations are not efficient and therefore not presented here.

parameters of the Kellogg problem such that we have a strong singularity of $\epsilon = 0.1$, corresponding to $\rho_{\mathbf{K}} \approx 161$. The analytical solution u is illustrated in Figure 4.6b, and Figure 4.7 shows the scalability of the multigrid solver and its robustness with respect to the polynomial degree. Here, the V(1,1) cycle is used, but other cycle types exhibit the same properties.

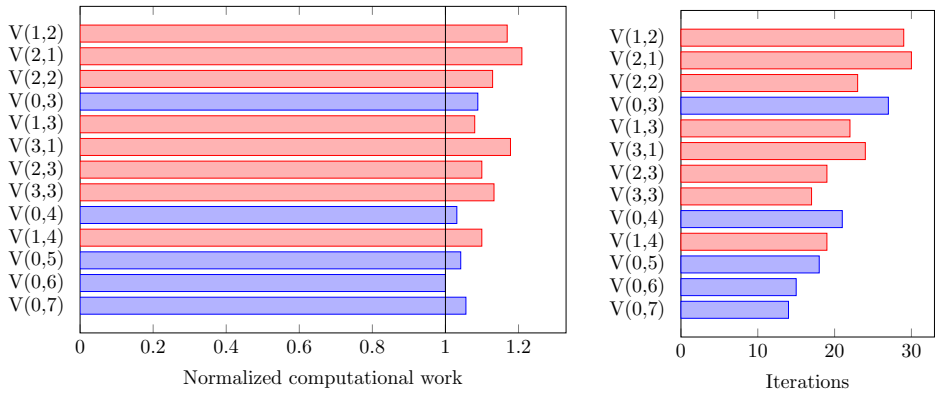


Fig. 4.4: Cycle comparison on the 2D test problem, triangular mesh, $N = 512$, $k = 1$ ($\approx 1.6 \times 10^6$ DoFs). The pre- and post-smoothers are the damped block Jacobi smoother with the relaxation parameter 2/3. In the first plot, the numerical values in flops of the computational work are normalized by the lowest one. The first cycles, with less than 3 or 4 total smoothing steps are not efficient and therefore not presented here.

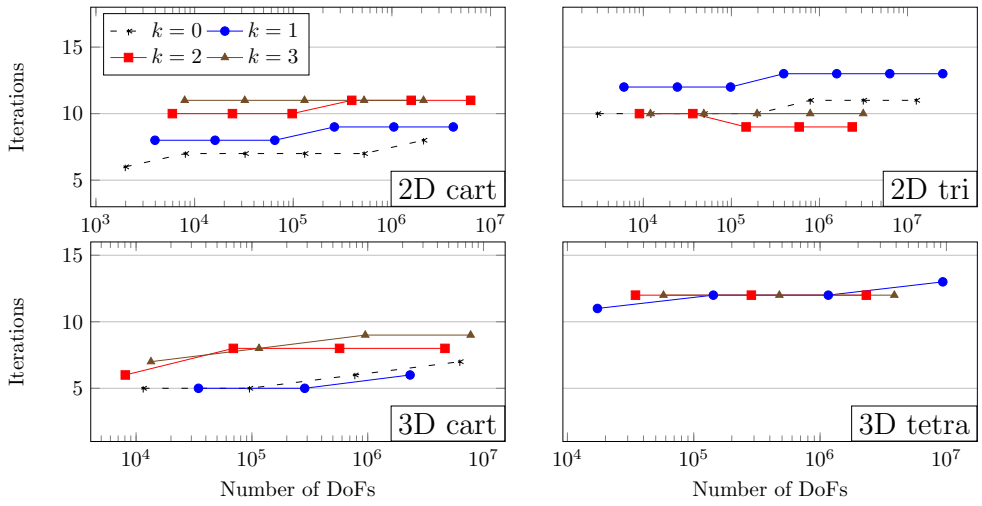
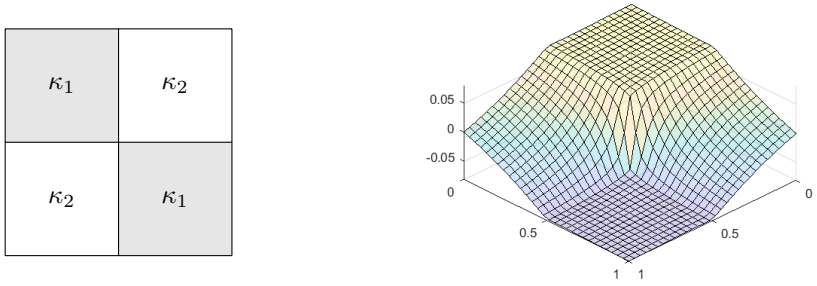


Fig. 4.5: Number of iterations to achieve convergence for the homogeneous problem on structured meshes: 2D Cartesian (top left), 2D triangular (top right), 3D Cartesian (bottom left), 3D tetrahedral (bottom right). The V(0,3) cycle is used for 2D problems, the V(0,6) for 3D. The absence of the lowest order case on the tetrahedral mesh is due to the divergence of the multigrid method.

4.4. Impact of different choices in the algorithm.

4.4.1. Alternative prolongation operators. Here we discuss alternatives in the coarse reconstruction of the cell-defined polynomial and in the trace prolongation on the fine faces. Especially, in the definition of $\Theta_{\ell-1}$, reconstructing a polynomial of higher degree may be optional. As a matter of fact, only solving the cell unknowns by



(a) Chiasmus heterogeneity pattern. (b) Kellogg's solution.

Fig. 4.6: (a) Square domain partitioned into four quadrants defining an heterogeneity pattern in the shape of a chiasmus. (b) Analytical solution of the Kellogg problem.

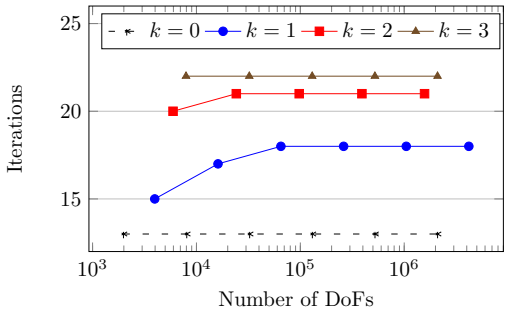


Fig. 4.7: Scalability results on the Kellogg problem. Number of V(1,1) iterations to achieve convergence for a growing number of DoFs.

(3.4) and skipping the higher order reconstruction (3.5) could be enough to construct a suitable cell-based polynomial from which to take the trace on the fine faces. In that second step of the prolongation, namely $\Pi_{\ell-1}^{\ell}$, we could also rely on the nestedness of the meshes to identically transfer the polynomials on the coarse faces to the fine grid using the canonical injection[1], instead of taking the average of the traces of the cell-based polynomials on both sides (see (3.6)). Table 4.1 summarizes these options. In order to quantify the impact of the choices made, Figure 4.8 compares the performance in term of scalability of the four option combinations applied to a homogeneous test problem. With the optimal V(0,3) cycle (left plot), the results show the good scalability of all options, with a better convergence rate for the final algorithm (about 15% better than with any other option combination). However, the results with the V(1,2) cycle (right plot) indicate that the differences between options may amplify when using non-optimal cycles. Indeed, in this case, it seems that taking the average on both sides instead of using the canonical injection on the faces geometrically shared by the coarse and fine meshes becomes an important criterion

[1]Here, the canonical injection refers to the linear operator that identically transfers the elements from one space to a larger one. It is not to be mistaken with the *straight injection* designating, in multigrid terminology, a special type of restriction operator.

to achieve the most scalable behaviour. On the other hand, the reconstruction of higher degree seems to simply improve the convergence rate, with no visible impact on scalability.

	Option label	Description
$\Theta_{\ell-1}$	cell $k+1$	Formula (3.5).
	cell k	Formula (3.4), the higher-order reconstruction (using p_T^{k+1}) is skipped.
$\Pi_{\ell-1}^{\ell}$	average	Formula (3.6).
	injection	The polynomials on the coarse faces are identically transferred to the fine grid using the canonical injection.

Table 4.1: Summary of the 4 options defined for the algorithm.

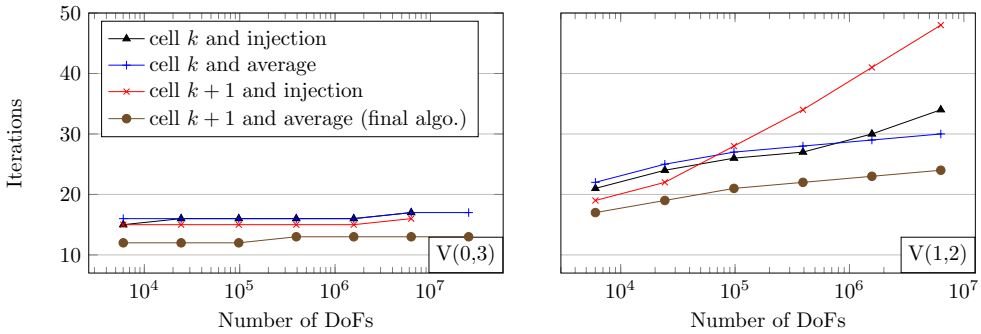
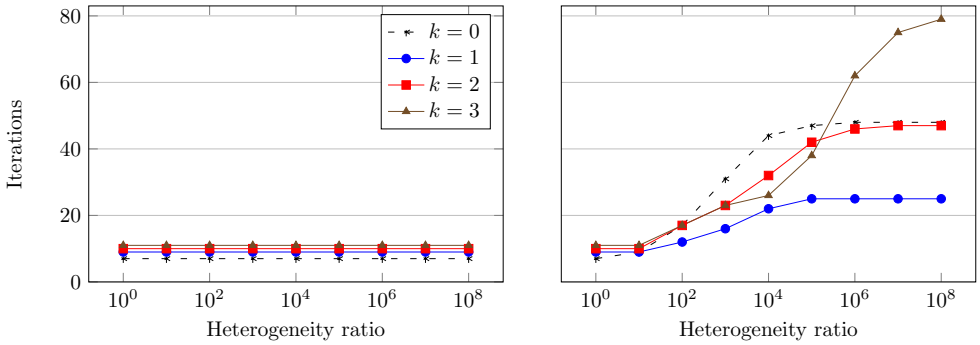


Fig. 4.8: Scalability comparison of different versions of the algorithm, applied on the 2D homogeneous problem discretized with the structured triangular mesh for $k = 1$. On the left, the V(0,3) cycle is used, on the right, V(1,2). The various option combinations are labeled according to Table 4.1.

4.4.2. Weighting strategy for heterogeneous problems. In the same way, in the case of a heterogeneous problem, we quantify the impact of weighting the cell contribution in $\Pi_{\ell-1}^{\ell}$ proportionally to its diffusion coefficient via (3.8). This strategy is hereafter called *heterogeneous weighting*, as opposed to the alternative that is, given a non-boundary face, to take from each cell of which it is the interface an equally weighted contribution (i.e. to use weighting factors of 1/2). The first thing to be noted is that, if we do not use the heterogeneous weighting (3.8), our algorithm diverges when the heterogeneity ratio $\rho_{\mathbf{K}} \geq 50$. Now, if we use the Galerkin operator instead of the discretized operator on the coarse grids, the algorithm becomes much more robust to high heterogeneity ratios and allows for a quantitative comparison. Figure 4.9 illustrates the differences in the weighting strategies for an increasing heterogeneity ratio, using the Galerkin operator. The heterogeneous weighting ensures perfect robustness with respect to $\rho_{\mathbf{K}}$ regardless of the polynomial degree. But without it, the convergence rate of the algorithm clearly becomes sensitive to the strength of the discontinuity. Moreover, this sensitivity intensifies with the increase of the polynomial degree.



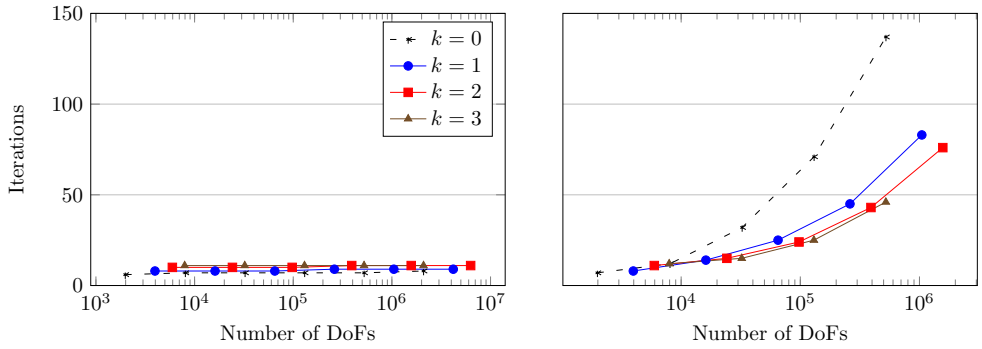
(a) Heterogeneous weighting (3.8)

(b) Homogeneous weighting ($w_{T_1F} = w_{T_2F} = \frac{1}{2}$)

Fig. 4.9: Robustness of our multigrid algorithm for the heterogeneous 2D test problem with (a) and without (b) the heterogeneous weighting strategy (3.8), in terms of number of iterations to achieve convergence for various orders of magnitude of the heterogeneity ratio $\rho_{\mathbf{K}}$. The square domain is discretized by a Cartesian mesh with $N = 64$, partitioned in four quadrants as described in 4.3. The multigrid algorithm uses the Galerkin operator and the V(0,3) cycle.

4.4.3. Role of the face coarsening. We now investigate the need for coarsening the faces in the coarsening strategy and show that without doing so, the solver's performance degrades rapidly. In our standard coarsening of uniform Cartesian meshes, two edges in 2D (or four faces in 3D) are ideally combined to become a single one. Consequently, each mesh cell has four edges (or six faces in 3D), and this on all levels. Alternatively, we also have the option that each coarse cell is represented with eight edges, colinear by pairs (see the last (inadmissible) coarsening strategy described by Figure 3.1). One of the effects of this alternative coarsening is that the number of unknowns per level is reduced less aggressively. Indeed, asymptotically, the number of unknowns is only decreasing by a factor of two per level (whether in 2D or 3D), whereas the standard coarsening rate is four in 2D and eight in 3D. For this nonstandard coarsening, the coarse grid spaces are then enlarged. In a Galerkin setting, the usual coarse grid spaces are subspaces of these enlarged coarse grid spaces, and as a consequence, we expect the two-grid convergence rates to improve. This is indeed verified, under the condition that the Galerkin operator is invertible (which is not necessarily the case with this coarsening strategy). However, this improvement cannot be observed in V-cycles with more levels, and not at all if we step out of the Galerkin setting to use the rediscretized operator. This observation is caused by the neglect of one important condition in multigrid convergence: as the smoother only efficiently reduces the high-frequency components of the error, it is crucial that the remaining low frequencies be seen as higher frequencies in the coarser spaces. And this can happen only if the geometric entities on which the DoFs lie are coarsened in between levels. As we work on the condensed system, whose unknowns rest on the mesh skeleton, this condition means that the *faces* should be coarsened. Only in this appropriate setting can the smoother at each level successfully target its own range of frequencies. If the faces are not coarsened, the smoother on the coarser grids

spends most of its effort to compute irrelevant solution modes, which causes the convergence rate to deteriorate. Figure 4.10 illustrates this convergence degradation as the number of DoFs (and therefore the number of levels) grows. Note that the overall performance of the solver also reduces due to the increased amount of work and the slower coarsening.



(a) With face coarsening

(b) Without face coarsening

Fig. 4.10: On the right, scalability test of the algorithm using a coarsening strategy which does not coarsen faces. The test problem is the homogeneous problem on the 2D Cartesian mesh, solved by the V(0,3) cycle of our multigrid algorithm, using the coarsening strategy described on the right-hand side of Figure 3.1. In the left plot, standard coarsening is used in comparison.

4.5. Unstructured meshes. The convergence of a geometric multigrid method relies on the approximation properties of the underlying discretization scheme through its coarse grid correction step. Furthermore, most discretization schemes are sensitive to the quality of the mesh, degrading in presence of flattened or stretched elements. As a direct consequence, the convergence of a multigrid method is often also sensitive to the mesh quality. The reader may refer to [1] for further details about the sensitivity of the HHO method to the element shapes. Moreover, even when starting from a good-quality coarse (resp. fine) mesh, the construction of a suitable mesh hierarchy may be difficult. In such an unstructured mesh hierarchy, the distortions of the elements must be kept under control when refining (resp. coarsening) the mesh. This is less a problem in 2D, but it remains difficult in 3D [25, 5]. Tetrahedral mesh refinement preserving mesh quality is still a topic of recent research [26, 22]. These difficulties help explain why more costly cycles may be required for highly unstructured 3D meshes.

In all the following tests, the mesh hierarchy is built by successive refinement of a coarse mesh. In 2D, we find that the convergence on unstructured meshes is qualitatively and quantitatively comparable to the convergence on structured meshes. Here the V(0,3) cycle is found to be sufficient. In 3D, the lower quality of tetrahedral meshes forces us to use costlier cycles. First of all, since the meshing method used to discretize the different refinements of the cubic domain (described in 4.2) is not applicable on general geometries, we investigate the impact of another, more generally applicable, tetrahedral refinement method. The method used is inspired by Bey's refinement algorithm [3]. Figure 4.11 shows that trading the refinement strategy for

one that does not conserve the topology of the tetrahedra causes a serious performance degradation, which can be mitigated at the cost of substantially more smoothing steps. In order to quantify the loss of performance with respect to the loss of mesh quality upon refinement, we use the regularity indicator ϱ_h defined as

$$\varrho_h := \max_{T\in\mathcal{T}_h} \varrho_T \qquad \text{with} \qquad \varrho_T := \frac{d_T}{h_T} \quad \forall T \in \mathcal{T}_h, \tag{4.1}$$

where d_T denotes the diameter of the largest ball included in T. A good regularity parameter is close to 1 while a bad one may be close to 0. As a reference, a cube has a regularity parameter of 0.64. Now, the original coarse mesh, namely the cubic domain divided into 6 geometrically identical tetrahedra, has a ϱ_h of 0.21. The so-called Cartesian tetrahedral refinement method described in 4.2 does not change the geometry of the refined tetrahedra, so the regularity parameter is conserved on the refined meshes, and we have seen that the V(0,3) cycle exhibits scalable behaviour and fast convergence. On the other hand, our custom Bey's method degrades the mesh quality during the first refinement (but not during the next ones), yielding in this case a regularity parameter of 0.14, which corresponds to a loss of 1/3. Figure 4.11 shows that over three times more smoothing steps are required to compensate for the poorer mesh quality and to recover comparable performance.

Using this time the custom Bey's refinement method, a cycle comparison in the model of Figure 4.3 finds V(0,10) as the most efficient cycle in this context. V(0,10) is therefore used for the test of the highly unstructured 3D mesh presented in Figure 4.12. In this test case, the initial coarse mesh has $\varrho_h = 0.10$, which degrades to 0.06 after refinement. The poor initial mesh quality and the further degradation of 40% result in sub-optimal performance: in spite of the large number of smoothing steps, the convergence degrades for larger meshes. Thus, the desired h-independent convergence cannot be confirmed. Table 4.2 summarizes the impact of the mesh quality and the refinement method on the performance of the multigrid algorithm.

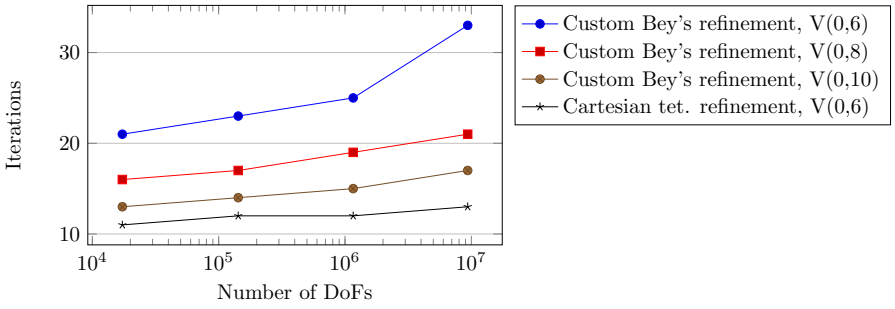


Fig. 4.11: Comparison on the cubic domain of the Cartesian tetrahedral refinement method described in 4.2 and the custom Bey's refinement method, with different cycles, in terms of scalable performance.

Having stated the sensitivity of the algorithm to the mesh quality, along with the known problem of refining (resp. coarsening) unstructured tetrahedral meshes without (too much) degradation, it is important to recall HHO as a polyhedral method. In this context, taking advantage of the flexibility of general polyhedral meshes is one way of overcoming these difficulties and keep the mesh quality under control. For the

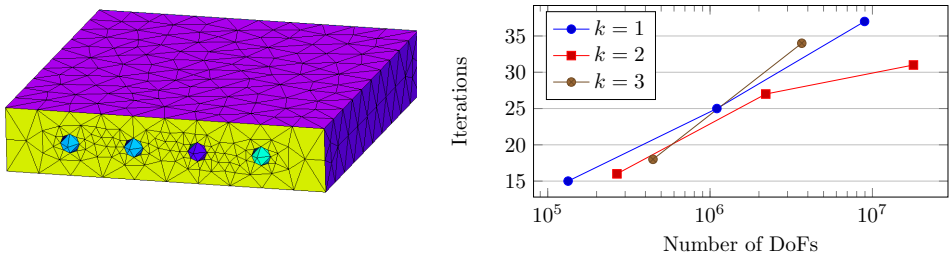


Fig. 4.12: On the left: tetrahedral mesh of a plate with four polyhedral holes going through the object from one side to the other. On the right: Scalability results of the multigrid algorithm, using the V(0,10) cycle and Bey's tetrahedral refinement. The case $k = 0$ is divergent.

Domain	Mesh	Refinement method	ϱ_1	ϱ_L	Quality loss	Required cycle
Cube	Cartesian	Standard	0.64	0.64	0%	V(0,3)
	Tetrahedral	Cartesian tet.	0.21	0.21	0%	V(0,6)
	Tetrahedral	Custom Bey	0.21	0.14	33%	V(0,10)
Plate w/ holes	Tetrahedral	Custom Bey	0.10	0.06	40%	$>$ V(0,10)

Table 4.2: Numerical values of the mesh quality and the quality loss caused by the refinement method, along with their consequences on the minimum cycle required to retrieve a close-to-optimal convergence rate. ϱ_1 (resp. ϱ_L) corresponds to the quality indicator (4.1) of the initial coarse mesh (resp. of the fine mesh obtained by the application of the refinement method). The domain "Plate w/ holes" refers to Figure 4.12.

same purpose, the use of non-nested meshes, discussed in the next paragraph, can constitute an additional tool.

4.6. Non-nested meshes. In this section we relax the requirement that the meshes must be nested. Although no rigorous redefinition of the algorithm is made here, we will explain the changes that are necessary for non-nested meshes.

- Coarsening strategy: The hierarchy $(\mathcal{T}_\ell)_{\ell=1\dots L}$ is now non-nested, in which we consider two successive levels ℓ (fine) and $\ell - 1$ (coarse). Figure 4.13 presents an example of two such levels. For a fine element $T \in \mathcal{T}_\ell$, we define its associated coarse element $T^{\ell-1} \in \mathcal{T}_{\ell-1}$ as the one coarse element that contains "most of" T, and we make the assumption that the definition used for "most of" ensures existence and uniqueness of $T^{\ell-1}$ for all $T \in \mathcal{T}_\ell$. In our implementation, we have considered that a coarse element contains "most of" the fine element T if it contains its centroid. Note that when the meshes are nested, $T^{\ell-1}$ simply reduces to the coarse element that geometrically embeds T. Now, given a fine face $F \in \mathcal{F}_\ell$, F is either absent from the coarse mesh (black dotted edges in Figure 4.13), or still geometrically present in a coarsened form, which we denote by $F^{\ell-1}$ (solid edges). Non-nestedness implies that F may no longer be geometrically embedded in $F^{\ell-1}$ (like blue

dotted edges are not embedded in the coarse red ones). If so, we talk of *non-nested faces*. The assumption that discontinuities in the diffusion coefficient do not happen inside elements consequently implies that the faces describing such discontinuities must still be nested.

- Trace on the fine faces: Although the reconstruction from the coarse faces to the coarse cells remains the same, inasmuch as the fine mesh is not involved, taking the trace of a coarse cell-defined polynomial on a fine face which is not fully included in the closure of the coarse cell is no longer possible. As example, consider in Figure 4.13 one coarse triangle with a red edge and pointing to the center of the figure: the blue edges corresponding to the refinement of the red edge are fully outside the coarse triangle, while two interior fine edges (which are simply no present on the coarse mesh in a coarsened form) are only partially included in the coarse triangle. In this case, we consider that the cell-defined polynomial is extended outside of the cell boundaries to overlap the targeted fine cells, which allows to take the trace on those faces. Considering $v \in U_{\mathcal{T}_{\ell-1}}^{k+1}$ and a face $F \in \mathcal{F}_\ell^{\mathrm{I}}$, formula (3.6) then becomes

$$(\Pi_{\ell-1}^{\ell} v)_{|F} := w_{T_1F}\, \pi_F^k(\mathfrak{v}_{|T_1})_{|F} + w_{T_2F}\, \pi_F^k(\mathfrak{v}_{|T_2})_{|F},$$

  where $\mathfrak{v}$ is the extension of $v_{|T^{\ell-1}}$ to $T_1 \cup T_2$.

These remarks are equivalent to stating that non-nested faces correspond to slight perturbations of nested ones.

Our algorithm is tested on a hierarchy of non-nested 2D meshes obtained by successive refinements for a domain containing a disk. The curved boundary of this disk is approximated more accurately with each refinement, see Figure 4.14. The results of Figure 4.15 show that the algorithm does not suffer from the non-nestedness in this form.

This observation is important regarding the design of coarsening strategies for unstructured meshes. In this case, agglomerating faces that are close to being coplanar (close to colinear in 2D) to form coarser ones seems possible. However, the non-nested meshes must be employed with great care. Although this leads to efficient algorithms for the homogeneous diffusion case when approximating complex fine shapes, it loses its efficiency when used to approximate the fine geometry of the coefficient discontinuities. Indeed, as said above, the faces describing the discontinuities must be nested. Indeed, regarding the test case of Figure 4.14, if we set the diffusion tensor to $\kappa_1 \mathbf{I}_d$ inside the circle approximation at every level, and to $\kappa_2 \mathbf{I}_d$ outside, the multigrid fails to stay robust to the heterogeneity ratio κ_2/κ_1: it quicky diverges with the discretized operator or yields very poor convergence (equivalent to the right plot of Figure 4.9) with the Galerkin operator.

5. Conclusion. The multigrid solver proposed and developed in this article is fast, scalable with respect to the mesh size, and robust to heterogeneity for elliptic problems discretized with HHO, provided a good enough mesh is used. Moreover, these desirable properties hold also for higher polynomial order. The algorithm proposed works for general meshes. However, the need to coarsen the faces can make the design of admissible coarsening strategies more difficult. While excellent convergence rates are observed for canonical nested mesh hierarchies, additional complexity must be expected when the faces are not co-planar. We have shown that fine faces that are sufficiently close to being co-planar may be approximated on coarse meshes by straight coarse faces without loss of performance. This can be exploited to construct coarse

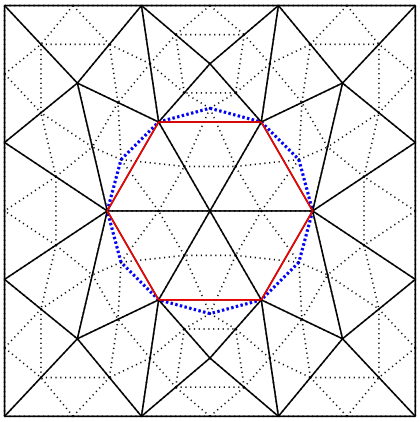


Fig. 4.13: Example of admissible non-nested coarsening. The fine mesh is represented in dotted lines, the coarse mesh in solid lines. The non-nestedness is highlighted by colors: the blue fine edges are coarsened into the red ones.

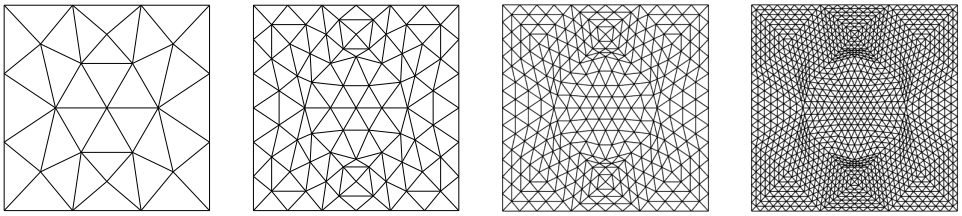


Fig. 4.14: Geometry of a disk embedded in a square, coarsely meshed and successively refined by a splitting method. Each refinement approximates more accurately the disk's shape, consequently yielding a mesh hierarchy that is non-nested at the disk's boundary. The non-nestedness of the first two meshes is highlighted in Figure 4.13.

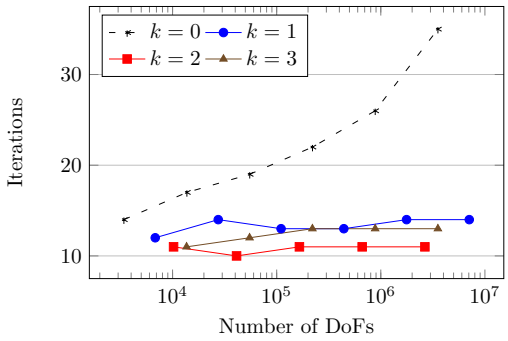


Fig. 4.15: Number of iterations to achieve convergence for the homogeneous problem on the non-nested mesh hierarchy described by Figure 4.14, solved by the V(0,3) cycle of our multigrid algorithm.

meshes in which the faces are also coarsened. Additionally, using non-nested meshes may also offer ways to avoid the degradation of the mesh quality upon coarsening or refinement.

The work on non-nested, unstructured meshes is motivated by the aim to provide a solver for the free, open-source CFD software *code_saturne* [2], developed and released by EDF.

REFERENCES

[1] J. Aghili, D. A. Di Pietro, and B. Ruffini, *An hp-Hybrid High-Order Method for Variable Diffusion on General Meshes*, Computational Methods in Applied Mathematics, 17 (2017), pp. 359–376, https://doi.org/10.1515/cmam-2017-0009.

[2] F. Archambeau, N. Méchitoua, and M. Sakiz, *Code Saturne: A Finite Volume Code for the computation of turbulent incompressible flows - Industrial Applications*, International Journal on Finite Volumes, 1 (2004), http://ijfv.math.cnrs.fr/spip.php?article3.

[3] J. Bey, *Tetrahedral grid refinement*, Computing, 55 (1995), pp. 355–378, https://doi.org/10.1007/BF02238487.

[4] A. Brandt and O. E. Livne, *Multigrid techniques: 1984 guide with applications to fluid dynamics*, SIAM, 2011.

[5] P. R. Brune, M. G. Knepley, and L. R. Scott, *Unstructured Geometric Multigrid in Two and Three Dimensions on Complex and Graded Meshes*, SIAM Journal on Scientific Computing, 35 (2013), pp. A173–A191, https://doi.org/10.1137/110827077.

[6] B. Cockburn, D. A. Di Pietro, and A. Ern, *Bridging the Hybrid High-Order and Hybridizable Discontinuous Galerkin methods*, ESAIM: Math. Model Numer. Anal., 50 (2016), pp. 635–650, https://doi.org/10.1051/m2an/2015051.

[7] B. Cockburn, B. Dong, and J. Guzmán, *A superconvergent LDG-hybridizable Galerkin method for second-order elliptic problems*, Mathematics of Computation, 77 (2008), pp. 1887–1916, https://doi.org/10.1090/S0025-5718-08-02123-6.

[8] B. Cockburn, O. Dubois, J. Gopalakrishnan, and S. Tan, *Multigrid for an HDG method*, IMA Journal of Numerical Analysis, 34 (2014), pp. 1386–1425.

[9] B. Cockburn, J. Gopalakrishnan, and R. Lazarov, *Unified Hybridization of Discontinuous Galerkin, Mixed, and Continuous Galerkin Methods for Second Order Elliptic Problems*, SIAM Journal on Numerical Analysis, 47 (2009), pp. 1319–1365, https://doi.org/10.1137/070706616.

[10] B. Cockburn, J. Guzmán, and H. Wang, *Superconvergent discontinuous Galerkin methods for second-order elliptic problems*, Mathematics of Computation, 78 (2009), pp. 1–1, https://doi.org/10.1090/S0025-5718-08-02146-7.

[11] D. A. Di Pietro and J. Droniou, *The Hybrid High-Order method for polytopal meshes*, no. 19 in Modeling, Simulation and Application, Springer International Publishing, 2020, https://doi.org/10.1007/978-3-030-37203-3.

[12] D. A. Di Pietro and A. Ern, *A hybrid high-order locking-free method for linear elasticity on general meshes*, Comput. Meth. Appl. Mech. Engrg., 283 (2015), pp. 1–21, https://doi.org/10.1016/j.cma.2014.09.009.

[13] D. A. Di Pietro, A. Ern, and S. Lemaire, *An arbitrary-order and compact-stencil discretization of diffusion on general meshes based on local reconstruction operators*, Comput. Meth. Appl. Math., 14 (2014), pp. 461–472, https://doi.org/10.1515/cmam-2014-0018. Open access (editor's choice).

[14] M. Franciolini, K. Fidkowski, and A. Crivellini, *Efficient discontinuous Galerkin implementations and preconditioners for implicit unsteady compressible flow simulations*, arXiv:1812.04789 [physics], (2018), http://arxiv.org/abs/1812.04789.

[15] J. Gopalakrishnan and S. Tan, *A convergent multigrid cycle for the hybridized mixed method*, Numerical Linear Algebra with Applications, 16 (2009), pp. 689–714, https://doi.org/10.1002/nla.636.

[16] M. Holst and S. Vandewalle, *Schwarz methods: to symmetrize or not to symmetrize*, SIAM Journal on Numerical Analysis, 34 (1997), pp. 699–722.

[17] G. Kanschat, *Robust smoothers for high-order discontinuous Galerkin discretizations of advection–diffusion problems*, Journal of Computational and Applied Mathematics, 218 (2008), pp. 53–60.

[18] R. B. Kellogg, *Singularities in interface problems*, in Numerical Solution of Partial Differential Equations–II, B. Hubbard, ed., Academic Press, Jan. 1971, pp. 351–400.

[19] M. Kronbichler and W. Wall, *A Performance Comparison of Continuous and Discontinuous Galerkin Methods with Fast Multigrid Solvers*, SIAM Journal on Scientific Computing, 40 (2018), pp. A3423–A3448, https://doi.org/10.1137/16M110455X.

[20] S. Muralikrishnan, T. Bui-Thanh, and J. N. Shadid, *A Multilevel Approach for Trace System in HDG Discretizations*, arXiv:1903.11045 [cs, math], (2019), http://arxiv.org/abs/1903.11045.

[21] L. N. Olson and J. B. Schroder, *Smoothed aggregation multigrid solvers for high-order discontinuous Galerkin methods for elliptic problems*, Journal of Computational Physics, 230 (2011), pp. 6959–6976.

[22] M. S. Petrov and T. D. Todorov, *Refinement strategies related to cubic tetrahedral meshes*, Applied Numerical Mathematics, 137 (2019), pp. 169 – 183, https://doi.org/10.1016/j.apnum.2018.11.006.

[23] J. Schoeberl and C. Lehrenfeld, *Domain Decomposition Preconditioning for High Order Hybrid Discontinuous Galerkin Methods on Tetrahedral Meshes*, in Lecture Notes in Applied and Computational Mechanics, vol. 66, Jan. 2013, pp. 27–56, https://doi.org/10.1007/978-3-642-30316-6_2.

[24] J. Schütz and V. Aizinger, *A hierarchical scale separation approach for the hybridized discontinuous Galerkin method*, Journal of Computational and Applied Mathematics, 317 (2017), pp. 500–509, https://doi.org/10.1016/j.cam.2016.12.018.

[25] L. R. Scott and S. Zhang, *Higher-Dimensional Nonnested Multigrid Methods*, Mathematics of Computation, 58 (1992), pp. 457–466, https://doi.org/10.2307/2153196.

[26] T. D. Todorov, *The optimal refinement strategy for 3-D simplicial meshes*, Computers & Mathematics with Applications, 66 (2013), pp. 1272–1283, https://doi.org/10.1016/j.camwa.2013.07.026.

[27] X. Tu and B. Wang, *A BDDC algorithm for second-order elliptic problems with hybridizable discontinuous Galerkin discretizations*, Electronic Transactions on Numerical Analysis, 45 (2016).

[28] T. Wildey, S. Muralikrishnan, and T. Bui-Thanh, *Unified Geometric Multigrid Algorithm for Hybridized High-Order Finite Element Methods*, SIAM Journal on Scientific Computing, 41 (2019), pp. S172–S195, https://doi.org/10.1137/18M1193505.