

# PARALLEL ELEMENT-BASED ALGEBRAIC MULTIGRID FOR $H(\text{curl})$ AND $H(\text{div})$ PROBLEMS USING THE PARELAG LIBRARY

DELYAN Z. KALCHEV, PANAYOT S. VASSILEVSKI, AND UMBERTO VILLA

**ABSTRACT.** This paper presents the use of element-based algebraic multigrid (AMGe) hierarchies, implemented in the ParELAG (Parallel Element Agglomeration Algebraic Multigrid Upscaling and Solvers) library, to produce multilevel preconditioners and solvers for  $H(\text{curl})$  and  $H(\text{div})$  formulations. ParELAG constructs hierarchies of compatible nested spaces, forming an exact de Rham sequence on each level. This allows the application of hybrid smoothers on all levels and AMS (Auxiliary-space Maxwell Solver) or ADS (Auxiliary-space Divergence Solver) on the coarsest levels, obtaining complete multigrid cycles. Numerical results are presented, showing the parallel performance of the proposed methods. As a part of the exposition, this paper demonstrates some of the capabilities of ParELAG and outlines some of the components and procedures within the library.

**KEY WORDS.** algebraic multigrid (AMG), AMGe,  $H(\text{curl})$  solvers,  $H(\text{div})$  solvers, ADS, AMS, de Rham sequence, hybrid smoothers, finite element methods, ParELAG, MFEM

**MATHEMATICS SUBJECT CLASSIFICATION.** 65F08, 65F10, 65N22, 65N30, 65N55

## 1. INTRODUCTION

Partial differential equation (PDE) models involving the  $\text{curl}$  (rotation) and  $\text{div}$  (divergence) operators often arise in the numerical simulation of physical phenomena and engineering systems. For example, this includes models of electromagnetism using Maxwell equations (possibly as a part of larger multiphysics codes) [54, 62, 16], mixed finite element methods for second-order elliptic equations [18] and coupled systems [66, 9], first-order system least-squares (FOSLS) finite element methods [21, 60, 7, 8], certain formulations of the Stokes and Navier-Stokes equations [22, 23, 53], and radiation transport simulations [20].

Among other discretization techniques, the finite element method (FEM) is a particularly appealing technique for problems defined on complex geometries due to its ability to handle unstructured meshes. One major challenge in solving linear systems arising from (FEM) discretizations of  $H(\text{curl})$  and  $H(\text{div})$  forms (i.e., symmetric problems involving the  $\text{curl}$  or  $\text{div}$  operators) is the large null spaces of the curl and divergence. A variety of approaches have been developed, including geometric and algebraic multigrid (AMG) [34, 61, 15, 39, 33, 11, 28, 14, 67, 10, 68, 64], static condensation and hybridization [25], and domain decomposition methods [55, 69, 70, 58, 38]. Other techniques [36, 44] are based on reformulation of the governing equations that are attuned to (geometric) multigrid solvers and preconditioners. A fundamental contribution to the development of multilevel methods for  $H(\text{curl})$  and  $H(\text{div})$  problems is the work of Hiptmair and Xu [37], which proposed auxiliary space preconditioners employing stable regular decompositions; see also [17]. Based on those ideas quite

---

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344 (LLNL-JRNL-824368).

The work of the second author was partially supported by NSF under grant DMS-1619640.

successful parallel  $H(\text{curl})$  and  $H(\text{div})$  solvers were developed as a part of HYPRE [1]: AMS [45] and ADS [46]. Furthermore, recent developments of generic auxiliary space preconditioners [43, 42], utilizing nonconforming reformulations and static condensation, can potentially be employed to implement efficient preconditioners for  $H(\text{curl})$  and  $H(\text{div})$  problems.

This paper describes and demonstrates the utilization of an element-based AMG (AMGe) approach (see, e.g., [40, 63, 64]) for preconditioning conforming discrete  $H(\text{curl})$  and  $H(\text{div})$  formulations. While AMGe methods were originally developed in the context of symmetric positive definite (SPD) systems coming from  $H^1$ -conforming formulations [24, 50], they demonstrate the capacity for a broader applicability. An important role in the construction of the AMGe multilevel methods is played by the *de Rham complex*, a sequence of Sobolev spaces corresponding the domain and ranges of a chain of *exterior derivatives*. For example, the sequence  $H^1 \rightarrow H(\text{curl}) \rightarrow H(\text{div}) \rightarrow L^2$  form the three-dimensional de Rham complex, corresponding to the chain of  $\nabla$ ,  $\text{curl}$ ,  $\text{div}$  differential operators (exterior derivatives). For more detail regarding this elegant tool in the theory of finite elements, we refer to [35, 12]. Namely, a discrete version of the sequence of conforming finite element spaces, maintaining the *exactness* and *commutativity* properties, delivers numerical stability (inf-sup compatibility) for a variety of mixed finite element methods. These ideas are used and investigated in [51, 49, 59] for the construction of multilevel element-based algebraic hierarchies of de Rham sequences of spaces. The last constitutes the foundation for the current work expounded in this paper.

A fundamental idea in AMGe, as presented in this work and implemented in ParELAG [5], is the element-based construction of coarse levels that structurally resemble (fine) geometric levels composed of standard finite elements. This involves the identification of coarse meshes with properly established *coarse topologies* in the form of relations between coarse elements and coarse lower-dimensional mesh entities (facets, edges, and vertices), similarly to geometric levels. Consequently, utilizing the coarse topology, each coarse space is built via independent local coarse-element-by-coarse-element computations, whose combined effect is a conforming global coarse space. The independence of the local work makes the construction of AMGe hierarchies naturally attuned to parallel computing.

ParELAG is a parallel library that builds hierarchies of stable sequences of discrete spaces with approximation properties, to be utilized typically as discretization tools for numerical upscaling [65] of mixed finite element formulations. It also provides a set of respective preconditioners and solvers that can be used for solving the resulting problems or building composite solvers for more complex problems. ParELAG has been successfully applied, e.g., in upscaling for reservoir modeling [48] and multilevel Monte Carlo simulations [57, 56, 26, 27].

This paper discusses the construction of multilevel solvers for  $H(\text{curl})$  and  $H(\text{div})$  problems, using the hierarchies of spaces from ParELAG. To deliver a tidy and concrete presentation, ideas are conveyed for the three-dimensional case, while one can easily see how they would be applied in a two-dimensional setting. The availability of entire de Rham sequences, together with all necessary transfer operators, on all levels allows the utilization of *hybrid (Hiptmair) smoothers* [34, 64] on all levels, as well as AMS and ADS on the coarsest levels, producing complete multigrid cycles. An outline of the overall methodology is presented and the parallel performance of the proposed solvers is shown in numerical examples. Finally, to further demonstrate ParELAG's capabilities and increase its visibility, a mini application within MFEM [3], a massively parallel widely used finite element library, has been developed.

The contributions and goals of this work (including vis-à-vis [50, 49, 51]) are: (1) to provide a unified presentation of AMGe methodologies for constructing coarse de Rham sequences by use of an exterior calculus formalism; (2) to describe the ParELAG library and exhibit its major capabilities; (3) to outline the generic multilevel AMGe methodology as implemented in the library, applicable to the entire de Rham sequence for coarse and fine finite element spaces of arbitrary order; and (4) to present a competitive, against state-of-the-art methods, novel  $H(\text{curl})$  and  $H(\text{div})$  AMGe solver with all its constitutive components (space hierarchies, smoothers, coarse solvers) implemented or invoked using the features of the core ParELAG library. From a theoretical perspective, a contribution of the paper is the use of the exterior calculus formalism to provide a unified dimension-independent presentation of the AMGe techniques of [50, 49, 51]. In particular, by use of the external derivative operator, the construction presented here for two and three-dimensional de Rham sequences can be extended to higher dimensions (e.g., to four-dimensional cases; see [31]) or other complexes (such as the elasticity complex). From a software perspective, the utilization of the exterior calculus formalism allows for a drastic reduction of the volume of code in the implementation of the prolongator operators for both the two and three-dimensional cases. Thus, our concise yet novel presentation of the methodologies serves to highlight unique features and capabilities of the ParELAG library. Finally, the numerical results serve the purpose of illustrating the potential and scalability of the technique in ParELAG, as well as the use of the library, which is the central topic of this paper and the MFEM mini application. Notably, the numerical results will also show that the AMGe solver with deep cycles demonstrate superior scalability, which motivates possible future work in fully exploiting this potential.

The outline of the remainder of the paper is as follows. The notation, the  $H(\text{curl})$  and  $H(\text{div})$  problems of interest, and an overview of de Rham sequences and their finite element discretization are presented in Section 2. Section 3 is devoted to providing a succinct and unified (by use of exterior calculus formalism) presentation of the AMGe technique for the construction of a hierarchy of de Rham sequences in two or three space dimensions on agglomerated meshes, including the construction of prolongation operators, co-chain projectors, and coarse exterior derivatives. Those operators are useful for implementing the hybrid smoothers on all levels and the AMS and ADS coarse solvers, as described in Section 4. Numerical results, demonstrating the parallel performance of the proposed methods, are in Section 5. The conclusions and a discussion of possible future directions are left for the last Section 6.

## 2. PRELIMINARIES

This section presents the notation and function spaces used in this paper, as well as the formulation of the  $H(\text{curl})$  and  $H(\text{div})$  problems. It also provides an overview of finite element exterior calculus, including de Rham sequences of continuous and discrete (finite element) spaces.

**2.1.  $H(\text{curl})$  and  $H(\text{div})$  problems.** Let  $\Omega \subset \mathbb{R}^3$  be a bounded contractible<sup>1</sup> domain with a Lipschitz-continuous boundary. Let  $L^2(\Omega)$  and  $[L^2(\Omega)]^3$  be the spaces of square integrable scalar and, respectively, vector functions. For  $v \in L^2(\Omega)$  and  $\underline{v} \in [L^2(\Omega)]^3$ ,

---

<sup>1</sup>Intuitively, a domain  $\Omega$  can be continuously contracted to a point (i.e., is *contractible*) if it has no holes or tunnels. Formally, a contractible domain is homotopy equivalent to a ball and to a single point.

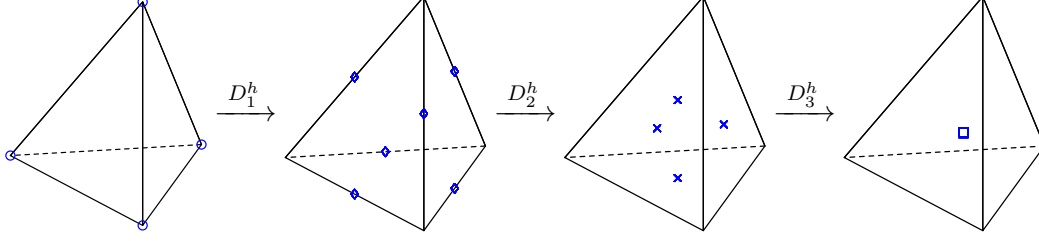


FIGURE 2.1. An illustration of the mapping between dofs in a tetrahedral element for the lowest order case.

denote with  $\|v\|_0^2 = (v, v)_0$  and  $\|\underline{v}\|_0^2 = (\underline{v}, \underline{v})_0$  the norm induced by the corresponding inner product.

Let  $D \in \{\text{grad}, \text{curl}, \text{div}, 0\}$  denote an exterior derivative operator. Then, the notation  $H(D; \Omega)$  is used to denote the usual function spaces  $H^1(\Omega) = \{v \in L^2(\Omega); \text{grad } v \in [L^2(\Omega)]^3\}$ ,  $H(\text{curl}; \Omega) = \{\underline{v} \in [L^2(\Omega)]^3; \text{curl } \underline{v} \in [L^2(\Omega)]^3\}$ ,  $H(\text{div}; \Omega) = \{\underline{v} \in [L^2(\Omega)]^3; \text{div } \underline{v} \in L^2(\Omega)\}$ , and  $L^2(\Omega)$ . Clearly,  $H(D; \Omega)$  is a Hilbert space endowed with the following inner product and induced norm:

$$(2.1) \quad \begin{aligned} (\cdot, \cdot)_D &= (\cdot, \cdot)_0 + (D\cdot, D\cdot)_0, \\ \|\cdot\|_D &= \sqrt{(\cdot, \cdot)_D}. \end{aligned}$$

For ease of notation, when the domain is omitted, it is understood that  $H(D) = H(D; \Omega)$ .

Next, for  $D \in \{\text{curl}, \text{div}\}$  consider the symmetric bilinear form

$$(2.2) \quad a_D(\underline{u}, \underline{v}) = (\alpha D\underline{u}, D\underline{v})_0 + (\beta \underline{u}, \underline{v})_0 \quad \text{for } \underline{u}, \underline{v} \in H(D; \Omega),$$

where  $\alpha, \beta \in L^\infty(\Omega)$ ,  $\alpha > 0$ ,  $\beta > 0$ . These bilinear forms, referred as  $H(\text{curl})$  and  $H(\text{div})$  forms hereafter, are positive definite and posses coefficient-dependent continuity in term of  $\|\cdot\|_D$ . If the coefficients are bounded away from zero, then the bilinear forms satisfy respective coefficient-dependent coercivity.

*Remark 2.1.* For simplicity, only the case  $\beta > 0$  is studied here. However, the semi-definite case  $\beta \geq 0$  is considered in [45, 46]. In general,  $\beta$  can be an essentially bounded symmetric positive (semi-)definite tensor. The bilinear forms are positive definite when  $\beta$  is positive definite, generally semi-definite when  $\beta$  is semi-definite, and coercivity depends on  $\beta$  being uniformly (on  $\Omega$ ) positive definite.

By means of a Galerkin projection onto  $H(D)$ -conforming discrete finite element spaces, the discrete version of the bilinear form in (2.2) can be represented by a symmetric positive (semi-)definite matrix. The goal of this paper is to construct multilevel preconditioners for linear systems with such matrices. The corresponding conforming finite element spaces, defined on a given fine mesh  $\mathcal{T}^h$ , are denoted by  $\mathcal{V}^h(D) \subset H(D)$ , for  $D \in \{\text{grad}, \text{curl}, \text{div}, 0\}$ . They are spaces of, respectively, continuous piecewise polynomial Lagrangian (nodal), Nédélec, Raviart–Thomas, and discontinuous piecewise polynomial finite elements [18]. In the case of lowest order finite elements, the degrees of freedom (dofs) in the spaces are associated with mesh entities of increasing dimensionality, one dof per entity. Namely, these are, respectively, point values at vertices, tangential flow along edges, normal flux across facets, constant values in elements (sometimes referred to as cells).

**2.2. De Rham sequences of continuous and discrete spaces.** Exterior calculus [12] is a key tool in the stability and convergence analysis of finite element

discretizations and solvers for the  $H(D)$  form in (2.2). This is based on the de Rham complex of Sobolev spaces on  $\Omega$  together with a respective *subcomplex* of conforming finite element spaces

$$(2.3) \quad \begin{array}{ccccccccc} \mathbb{R} & \xrightarrow{D_0=\mathfrak{J}} & H(D_1) & \xrightarrow{D_1=\text{grad}} & H(D_2) & \xrightarrow{D_2=\text{curl}} & H(D_3) & \xrightarrow{D_3=\text{div}} & H(D_4) & \xrightarrow{D_4=0} & \{0\} \\ & & \downarrow \Pi_1^h & & \downarrow \Pi_2^h & & \downarrow \Pi_3^h & & \downarrow \Pi_4^h & & \\ \mathbb{R} & \xrightarrow{D_0^h=\mathfrak{J}} & \mathcal{V}^h(D_1) & \xrightarrow{D_1^h} & \mathcal{V}^h(D_2) & \xrightarrow{D_2^h} & \mathcal{V}^h(D_3) & \xrightarrow{D_3^h} & \mathcal{V}^h(D_4) & \xrightarrow{D_4^h=0} & \{0\}. \end{array}$$

Above,  $\mathbb{R}$  represents the set of real numbers,  $\mathfrak{J}$  is the injection operator mapping a real number to the corresponding constant function on  $\Omega$ ,  $\Pi_i^h : H(D_i) \rightarrow \mathcal{V}^h(D_i)$  for  $i = 1, \dots, 4$  are appropriate (*cochain*) projection operators,  $D_i : H(D_i) \rightarrow H(D_{i+1})$  for  $i = 1, \dots, 3$  are differential operators (exterior derivatives) mapping between the Sobolev spaces, and  $D_i^h : \mathcal{V}^h(D_i) \rightarrow \mathcal{V}^h(D_{i+1})$  are the corresponding discrete versions.

In the finite-dimensional setting, functions in  $\mathcal{V}^h(D_i)$  ( $i = 1, \dots, 4$ ) can be identified with algebraic vectors collecting the coefficients in the respective finite element expansion (dofs). In what follows,  $d_i^h = \dim(\mathcal{V}^h(D_i))$  denotes the number of dofs of the space  $\mathcal{V}^h(D_i)$ . Hence,  $D_i^h$  for  $i = 1, \dots, 3$  can be viewed as matrices in  $\mathbb{R}^{d_{i+1}^h \times d_i^h}$  expressed in terms of the bases in  $\mathcal{V}^h(D_i)$ . They can be assembled via an *overwriting* finite element assembly<sup>2</sup> procedure from local, on elements (i.e., expressed in terms of shape functions), versions of the operators and their matrices. Note that, e.g., for the lowest order discretization (see Figure 2.1), the operators  $D_i^h$  ( $i = 1, \dots, 3$ ) map from mesh entities of lower dimensionality to those of higher dimensionality, i.e., vertices  $\rightarrow$  edges, edges  $\rightarrow$  facets, and facets  $\rightarrow$  elements, respectively.

It is assumed that  $\Pi_i^h$  for  $i = 1, \dots, 4$  are bounded operators, i.e.,  $\|\Pi_i^h\|_{D_i} < \infty$ , where  $\|\cdot\|_{D_i}$  denotes the corresponding induced operator norm from (2.1). This holds for the considered finite element spaces and implies the quasi-optimality property  $\|u - \Pi_i^h u\|_{D_i} \leq \|I - \Pi_i^h\|_{D_i} \inf_{v^h \in \mathcal{V}^h(D_i)} \|u - v^h\|_{D_i}$  for all  $u \in H(D_i)$ ; see [12].

Furthermore, for  $i = 1, \dots, 3$ , let  $\gamma_{\Gamma,i}$  denote the trace operator, that is the restriction of functions in  $H(D_i; \Omega)$  to  $\Gamma \subset \partial\Omega$ . Specifically,  $\gamma_{\Gamma,1}$  is the usual trace operator mapping  $H^1(\Omega)$  to  $H^{1/2}(\Gamma)$ ;  $\gamma_{\Gamma,2}$  restricts the tangential flow  $\underline{v} \times \underline{n}$  ( $\underline{v} \in H(D_2)$ ) to the surface  $\Gamma$ , and finally,  $\gamma_{\Gamma,3}$  restricts the normal flux  $\underline{v} \cdot \underline{n}$  ( $\underline{v} \in H(D_3)$ ) to  $\Gamma$ . Here,  $\underline{n}$  denotes the outward unit normal vector to  $\partial\Omega$ .

*Remark 2.2.* The diagram (2.3) corresponds to the so-called *natural* boundary conditions. It can be opportunely modified to the case of *essential* boundary conditions (that is, the case of vanishing traces on  $\partial\Omega$ ), as discussed in [12].

**2.2.1. Properties of the de Rham diagram.** Observe that the external derivative operators  $D_i$  are such that  $D_{i+1}D_i = 0$  (i.e.,  $\text{Range}(D_i) \subset \text{Ker}(D_{i+1})$ ) for  $i = 0, \dots, 3$ .

A de Rham sequence is called *exact* if and only if

$$(2.4) \quad \text{Range}(D_i) = \text{Ker}(D_{i+1}),$$

for  $i = 0, \dots, 3$ . Exactness depends on the topological characteristics of  $\Omega$ . In particular, the connectivity of  $\Omega$  is sufficient to demonstrate this property for  $i = 0$  and 3,

<sup>2</sup>*Overwriting* means that during the assembly the entries in the global matrix are overwritten by the values of the entries in the local matrices rather than accumulating (adding) them. See the implementation of `DiscreteOperator` in MFEM [3]. Since, as we will see in Section 3, the coarse spaces generated by ParELAG are conforming across agglomerated entities interfaces, overwriting entries in the global matrix causes no alterations as values coming from neighboring entities are the same. The overwriting procedure is also in ParELAG to assemble the coarse exterior derivative operators  $D_i^H$  and co-chain projectors  $\Pi_i^H$ , which are described below.

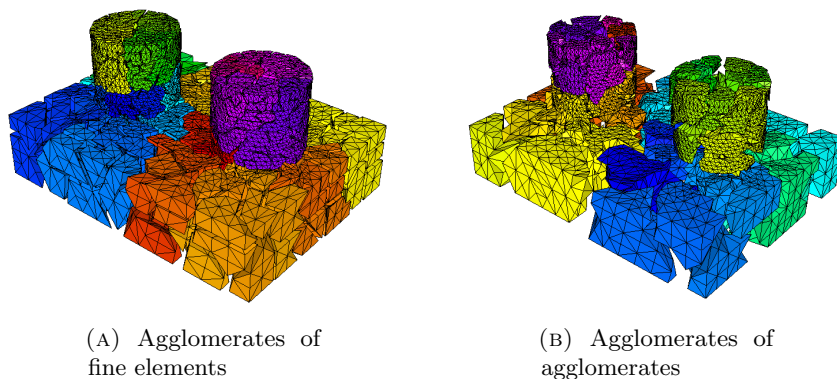


FIGURE 3.1. Examples of agglomerates. (The gaps between agglomerates are for illustration.)

whereas it holds for  $i = 1$  using that  $\Omega$  is simply-connected. The contractibility of  $\Omega$  provides the property for  $i = 2$  as a consequence of Poincaré's lemma; see, e.g., [35].

The de Rham diagram (2.3) is said to satisfy the *commutativity* property if and only if

$$D_i^h \circ \Pi_i^h = \Pi_{i+1}^h \circ D_i \quad \text{for } i = 1, \dots, 3.$$

Note that the commutativity property guarantees that, for a contractible domain  $\Omega$ , the exactness of the continuous de Rham sequence transfers to the discrete sequence [12].

The exactness of the continuous de Rham complex provides, e.g., stable decompositions (like the Helmholtz decomposition [54] and the so-called regular ones in [37]), while the commutativity of (2.3) and the exactness of the discrete subcomplex contribute to the inheritance of some important properties in the discrete setting, like the discrete stable decompositions in [37] and the provision of the (inf-sup) stability of certain mixed finite element methods; see [18, 35, 12]. Such stability, together with the approximation properties of the discrete spaces, is a sufficient and necessary condition for the convergence of those mixed finite element methods. As is discussed in the following section, ParELAG builds de Rham sequences of coarse spaces satisfying the same exactness and commutativity properties of the fine sequence to ensure stability of coarse level discretizations, as well as approximation properties [51, 49, 59].

### 3. OVERVIEW OF THE MULTILEVEL DE RHAM SEQUENCE

The fundamentals and notions associated with the coarsening of the de Rham sequence using AMGe techniques are now described. The key property of this approach, as articulated below, is that the de Rham sequence at each level of the hierarchy exhibits fine-like (geometric-like) finite element features. This means that coarse de Rham sequences are associated to generalized coarse meshes consisting of arbitrary shaped agglomerated entities (like elements, facets, edges, and vertices) and topological tables, and also maintain approximation properties as well as exactness (see (3.2)) and commutativity (see (3.3)) properties, resulting in stable coarse de Rham complexes. This is a significant property of AMGe utilizing agglomeration of elements, which, together with the algebraic nature of the approach, allows the recursive application of the coarsening procedure. Doing so, ParELAG produces

multilevel hierarchies of nested spaces forming exact and commutative de Rham complexes on all levels.

Starting with a given fine mesh the basic idea is to build a coarse mesh of coarse elements via *agglomeration* and identify lower-dimensional coarse mesh entities, like coarse facets, edges, and vertices, together with their relationships, forming the coarse mesh topology. The mesh topology is needed for the construction of the de Rham sequences as it reflects a natural structure within the sequence of spaces and exterior derivatives. Note that the coarse de Rham sequence is formed in terms of the fine one as a sequence of subspaces, i.e., coarse basis functions are linear combinations of fine basis functions. The construction of coarse bases is purely algebraic and entails the obtainment of *target traces*, or shortly *targets*, and an *extension* process. In particular, the extension process involves the solution of small local, on coarse entities, mixed finite element problems to produce the final coarse basis functions.

Specifically, consider the fine-level de Rham sequence (denoted with the superscript  $h$ ) and the coarser level one (denoted with the superscript  $H$ ), giving rise to the diagram

$$(3.1) \quad \begin{array}{ccccccccccc} \mathbb{R} & \xrightarrow{D_0^h=\mathfrak{I}} & \mathcal{V}^h(D_1) & \xrightarrow{D_1^h} & \mathcal{V}^h(D_2) & \xrightarrow{D_2^h} & \mathcal{V}^h(D_3) & \xrightarrow{D_3^h} & \mathcal{V}^h(D_4) & \xrightarrow{D_4^h=0} & \{0\} \\ & & \downarrow \Pi_1^H & & \downarrow \Pi_2^H & & \downarrow \Pi_3^H & & \downarrow \Pi_4^H & & \\ \mathbb{R} & \xrightarrow{D_0^H=\mathfrak{I}} & \mathcal{V}^H(D_1) & \xrightarrow{D_1^H} & \mathcal{V}^H(D_2) & \xrightarrow{D_2^H} & \mathcal{V}^H(D_3) & \xrightarrow{D_3^H} & \mathcal{V}^H(D_4) & \xrightarrow{D_4^H=0} & \{0\}, \end{array}$$

where  $D_i^H : \mathcal{V}^H(D_i) \rightarrow \mathcal{V}^H(D_{i+1})$  ( $i = 1, \dots, 3$ ) and  $\Pi_i^H : \mathcal{V}^h(D_i) \rightarrow \mathcal{V}^H(D_i)$  ( $i = 1, \dots, 4$ ) denote, respectively the coarse exterior derivative and co-chain projection matrices.

Assuming that the exactness (2.4) of the continuous de Rham sequence holds, the coarsening procedure must guarantee the exactness property of the coarse sequence, that is,

$$(3.2) \quad \text{Range}(D_i^H) = \text{Ker}(D_{i+1}^H) \quad \text{for } i = 1, \dots, 3,$$

and the commutativity property

$$(3.3) \quad D_i^H \circ \Pi_i^H = \Pi_{i+1}^H \circ D_i^h \quad \text{for } i = 1, \dots, 3.$$

These properties of the general approach to construct exact and commutative hierarchies via element-based algebraic multigrid techniques were proved in Pasciak and Vassilevski [59] (for the lowest order case) and in Lashuk and Vassilevski [51] (in the general case).

To this aim, ParELAG not only builds prolongation operators  $P_i : \mathcal{V}^H(D_i) \rightarrow \mathcal{V}^h(D_i)$  (which allows to transfer information between levels), but also the coarse exterior derivative operators  $D_i^H$  (which allows to define the hybrid smoothers in Section 4.1), and co-chain projectors  $\Pi_i^H$  (which play a fundamental role in applying the auxiliary space AMG preconditioners to coarse problems as described in Section 4.2). Moreover, ParELAG keeps track of element and facets attributes to ensure that material properties and essential boundary conditions can be properly applied to the discretized systems at every level of the hierarchy.

Note that the restriction,  $P_i^T : \mathcal{V}^h(D_i) \rightarrow \mathcal{V}^H(D_i)$ , and co-chain projection,  $\Pi_i^H$ , operators represent different actions and should not be confused; see Section 3.2.2.

Finally, it is important to note that ParELAG need not necessarily build the entire sequence of spaces, if the application does not require it. Instead, ParELAG builds spaces of the de Rham sequence in reverse order, that is from  $H(D_4) \equiv L^2$  to  $H(D_i)$  for any  $i \in [1, 4]$ .

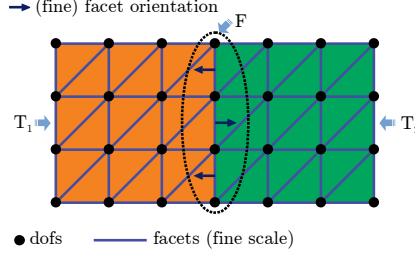


FIGURE 3.2. A two-dimensional illustration of the designation of a coarse facet  $F$  as a set of fine-scale facets, serving as an interface between agglomerates  $T_1$  and  $T_2$ . Arrows illustrate the (global) orientation of the fine-scale facets, i.e., the orientation of the vectors normal to the facets.

**3.1. Construction of a hierarchy of coarse meshes by agglomerating finer mesh entities.** The first step is the generation of a coarse mesh  $\mathcal{T}^H$ , from the given fine one  $\mathcal{T}^h$ , including all mesh entities: coarse elements, facets, edges, and vertices. The foundation of this is the construction of coarse elements as *agglomerates* (or *agglomerated elements*), which provide a non-overlapping partition of the fine elements; see Figure 3.1a. This is performed in a recursive manner to generate a hierarchy of nested meshes; see Figure 3.1b. One customary way to achieve that is via partitioning (e.g., using METIS [2]) of the *dual graph* of  $\mathcal{T}^h$ —a graph whose nodes are the elements in  $\mathcal{T}^h$  and any two nodes are connected in the graph when the respective mesh elements share a facet. It is not difficult to generate the agglomerates as contiguous partitions in terms of the dual graph, e.g., using METIS or simply identifying the connected components of the partitioning after it is generated. Moreover, ParELAG provides additional tools that, via weighting the dual graph and further splitting of agglomerates, can help improve the topological properties of the coarse elements, which are relevant if  $H(\text{curl})$  is utilized.

Using the partitioning of  $\mathcal{T}^h$  and viewing each agglomerate  $T \in \mathcal{T}^H$  as a collection of fine facets, an intersection procedure (see [64, Section 1.9]) over these collections provides the coarse facets as sets of fine facets (see Figure 3.2), which can be consistently interpreted as interface surfaces between coarse elements. Further viewing the obtained coarse facets as collections of fine edges, their intersection identifies coarse edges as sets of fine edges. Finally, the intersection of coarse edges in terms of fine vertices identifies the coarse vertices.

Coarse facets and edges additionally carry information about the orientation of their constituting fine entities; see Figure 3.2. Such a set of fine-scale orientations for a coarse entity represents the orientation of that coarse entity. More precisely, these are  $+1$  and  $-1$  data entries in the agglomerated topology relating each coarse entity to its comprising fine-scale ones, respectively representing the preservation or the reversal of the original orientation of the fine entity within the coarse-scale one, so that each agglomerated entity has a consistent orientation. For example, a coarse facet  $F$  has an associated vector of  $+1$  and  $-1$ , denoted by  $\varphi^F$ , that based on the orientation of the constituting fine facets devises a consistent orientation for  $F$ , so that the normal vector to  $F$  points everywhere from one of its adjacent agglomerates to the other, e.g., from  $T_1$  to  $T_2$  in Figure 3.2. Following the example in Figure 3.2, for the coarse facet  $F$  to be oriented so that its normal vector consistently points from  $T_1$  to  $T_2$ , one has  $\varphi^F = [-1, +1, -1]$ , indicating that the orientations of the



first and third constituting fine facets are flipped while the orientation of the second one is preserved.

Finally, coarse entities associated with the domain boundary, or portions of it, are identified, thus allowing to track boundary dofs and apply boundary conditions on coarse levels.

**3.1.1. ParELAG's implementation of agglomerated meshes hierarchy.** ParELAG contains a set of *partitioner* classes, which generate an element partitioning on the current level that composes the agglomerated elements. For example, the class `MFEMRefinedMeshPartitioner` constructs agglomerates in the form of geometric coarse elements by reverting previous refinements performed by MFEM, while the class `MetisGraphPartitioner` invokes METIS internally. Furthermore, the coarse elements can be optionally made to conform to material (coefficient) interfaces by splitting agglomerates that cross such interfaces.

Instances of `AgglomeratedTopology` collect coarse entities together with relationships between them in the form of a so called (*agglomerated*) *topology* of  $\mathcal{T}^H$ . Note that such a topology object in itself also represents a complex related to (3.1). It further contains relations between agglomerated entities and their constituting fine ones. Particularly, the `AgglomeratedTopology` object on the finest level is obtained from the given mesh  $\mathcal{T}^h$  (i.e., using MFEM). Relationships between fine and coarse entities are stored in `AgglomeratedEntity_Entity` tables within `AgglomeratedTopology`. Each row in `AgglomeratedEntity_Entity` table corresponds to a coarse entity and the non-zeros entries in each row represent the finer grid entities that form the coarse entity. Furthermore, `AgglomeratedEntity_Entity` tables for facets and edges also store orientation information.

Finally, having defined a topology on the current level, a new coarser agglomerated topology is generated by invoking the `CoarsenLocalPartitioning()` member function of `AgglomeratedTopology`, using the agglomerated elements produced on the current level by a partitioner class.

**3.2. The element-based construction of the components of the coarse sequences.** Here, we set the stage for the definition of the coarse basis functions in Section 3.3. We exhibit and motivate the fundamental utility of the process of building coarse bases by reviewing the element-based construction of the components that constitute the coarse sequences and coarse problems, once the coarse basis functions are obtained via the procedures in Section 3.3. This effectively reduces the considerations to the local agglomerate-by-agglomerate algebraic process of building locally-supported coarse bases presented in Section 3.3. Particularly, we review the coarse spaces formed via the choice of coarse bases, the construction of prolongation and coarse system matrices, including their local versions, as well as co-chain projectors and coarse exterior derivatives. In the end, we comment on the ParELAG classes implementing the multilevel de Rham sequences of spaces.

The procedures here are outlined to the minimal necessary extent following a concise constructive or algorithmic perspective and presented as implemented in ParELAG, since the goal is to describe the library and its capabilities. Therefore, the constructs are exposed and reviewed concisely for the purpose of presenting the software. Full details on the theoretical justification of the constructions explained in what follows are found in Pasciak and Vassilevski [59] (for the lowest order case) and in Lashuk and Vassilevski [51] (in the general case).

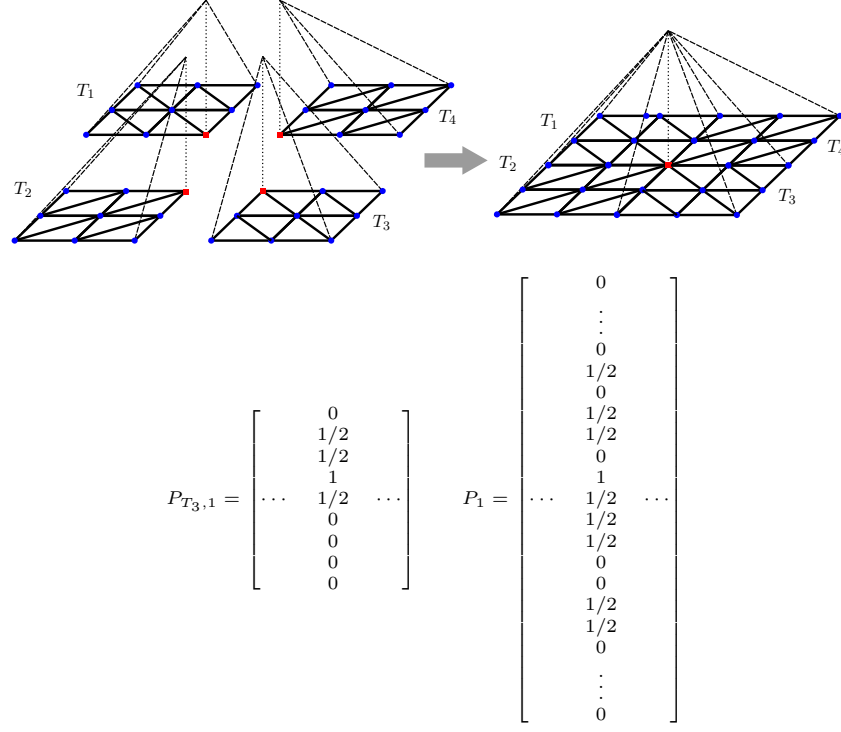


FIGURE 3.3. An illustration of coarse shape and basis functions and interpolation matrices. The example presents a piecewise linear  $H^1$ -conforming space on a two-dimensional mesh, where a single coarse basis function is supported on four agglomerates. The top left illustrates the separate shape functions that form the basis function. The final basis function (top right) is the result of conformingly combining the shape functions and the respective vectors defined on the fine dofs. The corresponding coarse dof is indicated by a small rectangle, while the fine dofs include the ones indicated by small circles and the rectangular one. The bottom shows a possible algebraic representation as column vectors of respective local (e.g., on  $T_3$ ) and global interpolation matrices under some dof indexing.

**3.2.1. Coarse spaces, prolongation matrices, and coarse system matrices.** The coarse spaces  $\mathcal{V}^H(D_i)$  for  $i = 1, \dots, 4$  are obtained via the construction of coarse bases in Section 3.3 as sets of algebraic vectors in terms of the respective dofs in  $\mathcal{V}^h(D_i)$ , i.e., coarse basis functions are linear combinations of fine basis functions. These algebraic vectors constitute the columns of corresponding *prolongation* matrices  $P_i \in \mathbb{R}^{d_i^H \times d_i^h}$ ,  $P_i : \mathcal{V}^H(D_i) \rightarrow \mathcal{V}^h(D_i)$ , with full column ranks, where  $d_i^H = \dim(\mathcal{V}^H(D_i))$ . Similarly to the fine level, the coarse basis functions are supported locally and built locally agglomerated entity by agglomerated entity (more details in Section 3.3). The  $\mathcal{V}^H(D_i)$ -dofs are identified with the columns of  $P_i$ , i.e., with the respective coarse basis functions. Moreover, the  $\mathcal{V}^H(D_i)$ -dofs associated with an agglomerate  $T \in \mathcal{T}^H$  are the ones whose basis functions have supports intersecting  $T$ , and the restrictions of those basis functions (i.e., the coarse *shape functions*) on  $T$  are precisely the restrictions of the respective algebraic vectors on the  $\mathcal{V}^h(D_i)$ -dofs of  $T$ , which are the  $\mathcal{V}^h(D_i)$ -dofs associated with the fine elements  $\tau \in \mathcal{T}^h$  such that  $\tau \subset T$ . Therefore,

local-on- $T$  prolongation matrices  $P_{T,i}$  can be defined and they are submatrices of  $P_i$  on the respective dofs in  $\mathcal{V}^h(D_i)$  and  $\mathcal{V}^H(D_i)$  on  $T$ ; see Figure 3.3.

Next, consider the construction of system matrices at coarse levels of the AMGe hierarchy. Let  $a_{ij}(\cdot, \cdot)$  be a bilinear form defined on  $H(D_i) \times H(D_j)$  for some  $i, j \in \{1, \dots, 4\}$ . By means of a Galerkin projection onto the finite element bases of the conforming discrete subspaces  $\mathcal{V}^h(D_i)$  and  $\mathcal{V}^h(D_j)$ , the bilinear form is represented by a (global) matrix  $A_{ij}^h \in \mathbb{R}^{d_j^h \times d_i^h}$  on the dofs in  $\mathcal{V}^h(D_i)$  and  $\mathcal{V}^h(D_j)$ . That is, for every entry of  $A_{ij}^h$  indexed  $(l, k)$ , it holds

$$(A_{ij}^h)_{lk} = a_{ij}(\phi_{i,k}^h, \phi_{j,l}^h) \quad \text{for } l = 1, \dots, d_j^h, k = 1, \dots, d_i^h,$$

where  $\{\phi_{i,k}^h\}_{k=1}^{d_i^h}$  denotes the basis of  $\mathcal{V}^h(D_i)$ . This global matrix is obtained via a standard assembly from local element matrices  $A_{\tau,ij}^h$  for the elements  $\tau \in \mathcal{T}^h$  formulated on the  $\mathcal{V}^h(D_i)$  and  $\mathcal{V}^h(D_j)$ -dofs associated with  $\tau$ . The coarse matrices are produced by standard ‘‘RAP’’ procedures. Indeed, the representations of  $a_{ij}(\cdot, \cdot)$  in terms of the bases of  $\mathcal{V}^H(D_i)$  and  $\mathcal{V}^H(D_j)$  is the matrix  $A_{ij}^H = P_j^T A_{ij}^h P_i \in \mathbb{R}^{d_j^H \times d_i^H}$ . Also, for  $T \in \mathcal{T}^H$ , using a standard assembly locally with  $A_{\tau,ij}^h$  for  $\tau \subset T$ , the local-on- $T$  fine-scale matrix  $A_{T,ij}^h$  is obtained on the  $\mathcal{V}^h(D_i)$  and  $\mathcal{V}^h(D_j)$ -dofs associated with  $T$ . Thus,  $A_{T,ij}^H = (P_{T,j})^T A_{T,ij}^h P_{T,i}$  forms the coarse element matrices, which can produce  $A_{ij}^H$  via a standard assembly.

**3.2.2. Co-chain projectors and coarse exterior derivatives.** Here, we comment on the co-chain projection operators as well as on the coarse derivative operators. Co-chain projectors  $\Pi_i^H$  are (right) inverses of the prolongation operators  $P_i$ , i.e., they satisfy

$$(3.4) \quad \Pi_i^H \circ P_i = I_i^H \text{ for } i = 1, \dots, 4,$$

where  $I_i^H$  denotes the identity operator in  $\mathcal{V}^H(D_i)$ . Their construction parallels that of the coarse basis functions in Section 3.3, starting from the projection of coarse dofs associated with local lower-dimensional entities and moving towards higher-dimensional local entities. Specifically, projection operators are obtained via independent local agglomerate-by-agglomerate procedures. The local projection operators can be pieced together to form the global  $\Pi_i^H$ , since, by construction, these local projectors agree on dofs shared between coarse entities. The independent production of each such local projection operator involves the inversion of a small coarse-scale local mass matrix on the respective coarse entity. More details can be found in [51]; see also [59, 49].

*Remark 3.1.* Note that the co-chain projector  $\Pi_i^H$  is a left inverse of the prolongation operator  $P_i$  that satisfies two important properties: (1) it can be constructed locally (agglomerated entity by agglomerated entity) and (2) satisfies the commutative property (3.3). Thus,  $\Pi_i^H$  differs from the restriction  $P_i^T$ , which may not satisfy (3.4) or (3.3). As such,  $\Pi_i^H$  is a projection operator from fine to coarse finite element function spaces, while  $P_i^T$  can be viewed as acting on functionals (i.e., mapping dual spaces). That is,  $P_i^T$  provides restrictions suitable for Galerkin projections of linear or bilinear forms as elaborated in Section 3.2.1. Specifically,  $P_i^T : [\mathcal{V}^h(D_i)]' \rightarrow [\mathcal{V}^H(D_i)]'$ , where  $\ell^H = P_i^T \ell^h \in [\mathcal{V}^H(D_i)]'$ , for  $\ell^h \in [\mathcal{V}^h(D_i)]'$ , acts like  $\ell^H(v^H) = \ell^h(P_i v^H)$ , for all  $v^H \in \mathcal{V}^H(D_i)$ . That is, as the name suggests,  $P_i^T$  restricts the action of  $\ell^h$  from  $\mathcal{V}^h(D_i)$  to  $\text{Range}(P_i) = \mathcal{V}^H(D_i)$ .

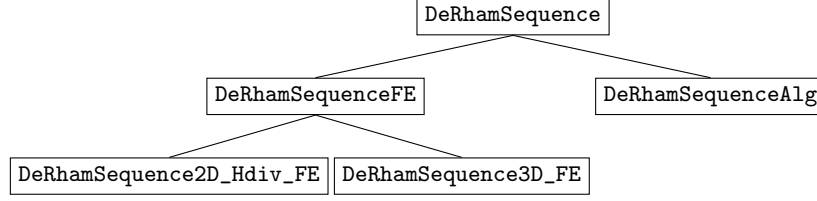


FIGURE 3.4. An illustration of the hierarchy of ParELAG classes for de Rham sequences.

It is worth highlighting that co-chain projection operators do not merely constitute a theoretical tool, but they are needed for the construction of the coarse level auxiliary space preconditioners described in Section 4.2. Moreover, they are needed in the implementation of multilevel Monte Carlo methodologies [57, 56] and efficient multilevel nonlinear solvers like FAS (full approximation scheme) [47].

Next, consider the coarse exterior derivatives. The commutativity property (3.3) implies that the coarse derivatives  $D_i^H$  can be formally defined in terms of the prolongation  $P_i$ , cochain projector  $\Pi_{i+1}^H$ , and finer level exterior derivative  $D_i^h$  as

$$D_i^H = \Pi_{i+1}^H \circ D_i^h \circ P_i,$$

as can be easily shown by (right-)multiplying (3.3) by  $P_i$  and using (3.4).

In ParELAG, the coarse exterior derivatives  $D_i^H$  are constructed via independent local agglomerate-by-agglomerate procedures starting from lower-dimensional entities (agglomerated faces for  $D_3^H$ , agglomerated edges for  $D_2^H$ , and agglomerated vertices for  $D_1^H$ ) to agglomerated elements  $T$ . The global exterior derivative operator  $D_i^H$  is then constructed by an overwriting assembly of local-on- $T$  coarse exterior derivatives  $D_{T,i}^H$  in a similar manner to the assembly of the finite element level operators discussed in Section 2.2. Note that, differently from matrices arising from discretizations of variational forms, at each level of the hierarchy the restrictions of the exterior derivative to a single entity or to the union of connected entities (agglomerate) are submatrices of  $D_i^H$ .

**3.2.3. ParELAG's classes for AMGe hierarchies of de Rham sequences.** ParELAG implements the construction and coarsening of de Rham sequences via the small hierarchy of classes depicted in Figure 3.4. The base class is `DeRhamSequence`, which contains the main toolset necessary for constructing and working with de Rham sequences, including the procedures for building coarse spaces outlined in this paper. Two subclasses provide specialized methods for de Rham sequences on the finest (geometric) level (`DeRhamSequenceFE`), which is produced employing MFEM, and *algebraic* levels (`DeRhamSequenceAlg`), which are coarse levels produced by ParELAG that are not associated with a given mesh (i.e., that are not *geometric*). The class `DeRhamSequenceFE` is further specialized to address special cases like the dimensionality of the domain.

**3.3. Coarse bases and the extension procedure.** The abstract construction of coarse basis functions, which is applicable on all levels, is outlined now. The target traces and the extension process are discussed. A detailed presentation of a closely related procedure for coarse space construction can be found in [51]. After a short introduction, the procedure is described, followed by a schematic summary in Section 3.3.4.

The coarse mesh  $\mathcal{T}^H$  with all its entities (elements or agglomerates, facets, edges, and vertices) and its topology is available. Note that coarse basis functions are obtained in terms of the fine ones using an extension procedure involving the solution of local finite element problems, which translates into inverting local matrices. This uses information on the association between fine dofs and coarse entities, which is easily derived from the association between fine dofs and the fine entities that constitute each coarse entity. In the terminology of ParELAG, this is called *dof agglomeration* (or *aggregation*), implemented in the `DofAgglomeration` class.

As indicated in Section 3.2, the extension can be viewed in the local context of an agglomerate  $T \in \mathcal{T}^H$  and its associated lower-dimensional coarse entities. Note that some basis functions are supported on multiple agglomerates; see Figure 3.3. Nevertheless, they are constructed by independent local agglomerate-by-agglomerate processes. The procedures executed on a single agglomerate  $T$  produce the respective shape functions. Shape functions that form a particular basis function are conforming, i.e., they agree on fine-scale dofs shared between agglomerates. In the formation of the prolongation matrices, the final basis functions are obtained by joining together all associated shape functions. The shape functions on  $T$  alone comprise the local-on- $T$  prolongation matrices,  $P_{T,i}$ ; see Section 3.2. Other basis functions are entirely supported in a single agglomerate  $T$ . As is customary, they are called *bubble functions*; see [18]. More specifically, for  $i = 1, \dots, 3$ , a  $\mathcal{V}^h(D_{T,i}; T)$  bubble function is fully supported in  $T$ , and it is globally a function in  $H(D_i; \Omega)$ , i.e., has a vanishing  $\gamma_i$ -trace (see Section 2.2) on  $\partial T$ . Here,  $D_{X,i}$  denotes the restriction of the derivative operator  $D_i$  ( $i = 1, \dots, 4$ ) to an entity  $X \in \{V, E, F, T\}$ , where  $V$ ,  $E$ ,  $F$ , and  $T$  represent an agglomerated vertex, edge, facet, and element, respectively. Figure 3.6 illustrates coarse basis functions on agglomerates in two dimensions.

**3.3.1. Approximation property targets.** The first step is to select so called *targets*. The results in this paper are obtained using global polynomial targets due to their simplicity and inherent approximation properties following from standard polynomial approximation theory. Such polynomial targets are utilized to produce target traces on coarse entities (elements or agglomerates, facets, edges, vertices) and allow the construction of high-order (practically of arbitrary order) conforming and stable coarse spaces. The resulting target traces, obtained essentially by restricting the approximation property targets to the respective coarse entities, are then extended via the procedure in Section 3.3.3, providing the final coarse basis functions and coarse spaces with the desired order and approximation properties.

Global polynomial targets are set once in ParELAG on the finest level via simple call to the `SetUpScalingTargets()` member function of the `DeRhamSequenceFE` class. The procedure in ParELAG for building the targets is quite simple. On the finest level, the finite element interpolants (via  $\Pi_i^h$ ) of monomials up to a prescribed order constitute the targets. On coarse levels, the targets are transferred as needed via projection, i.e., by applying the operators  $\Pi_i^H$ ,  $i = 1, \dots, 4$ . Note that it is admissible to utilize polynomial targets of order higher than the order of the finest-level finite elements. In such a case, coarse basis functions have a high-order complexion but are represented piecewise by lower-order polynomials. Alternatively, local targets on coarse entities can be obtained by, e.g., solving local eigenvalue problems, which also provide approximation properties (cf. [41, 19]). In any case, for the approach outlined here, appropriate respective target traces on coarse entities (elements or agglomerates, facets, edges, vertices) are obtained and available as needed, represented in terms of respective  $\mathcal{V}^h(D_i)$  dofs,  $i = 1, \dots, 4$ .

**3.3.2. PV traces and coarse trace spaces.** The lowest-dimensional traces for  $\mathcal{V}^H(D_i)$ ,  $i = 1, \dots, 4$ , from which extensions are initiated, are defined on agglomerated vertices, edges, facets, and elements, respectively. On these entities, the so called *PV traces* (coming from [59]) are locally defined agglomerated entity by agglomerated entity. These traces alone provide, once extended following Section 3.3.3, the lowest-order stable coarse spaces on  $\mathcal{T}^H$ .

Specifically, the PV traces  $\phi_{PV,1}^V \in \mathcal{V}^H(D_{V,1}; V)$ ,  $\phi_{PV,2}^E \in \mathcal{V}^H(D_{E,2}; E)$ ,  $\phi_{PV,3}^F \in \mathcal{V}^H(D_{F,3}; F)$ , and  $\phi_{PV,4}^T \in \mathcal{V}^H(D_{T,4}; T)$  are defined to have unit integral on the corresponding agglomerated entity,  $V$ ,  $E$ ,  $F$ , and  $T$ . That is, the representation of the PV traces in terms of finer level dofs are given by

$$(3.5) \quad \begin{aligned} \phi_{PV,1}^V &= \phi_{PV,1}^v = 1, & v &\equiv V(v \text{ is the fine vertex forming } V) \\ \phi_{PV,2}^E \cdot \underline{\tau}_E &= \sum_{e \in E} (\varphi^E)_e \phi_{PV,2}^e \cdot \underline{\tau}_e, & \text{satisfying } \int_E \phi_{PV,2}^E \cdot \underline{\tau}_E \, dl &= 1, \\ \phi_{PV,3}^F \cdot \underline{n}_F &= \sum_{f \in F} (\varphi^F)_f \phi_{PV,3}^f \cdot \underline{n}_f, & \text{satisfying } \int_F \phi_{PV,3}^F \cdot \underline{n}_F \, d\sigma &= 1, \\ \phi_{PV,4}^T &= \sum_{\tau \in T} \phi_{PV,4}^\tau, & \text{satisfying } \int_T \phi_{PV,4}^T \, d\Omega &= 1, \end{aligned}$$

where  $\underline{\tau}_E$  and  $\underline{\tau}_e$  are the corresponding tangent vectors to the coarse edge  $E$  and its constituting fine edges,  $e$ ,  $\underline{n}_F$  and  $\underline{n}_f$  are the corresponding normal vectors to the coarse facet  $F$  and its constituting fine facets,  $f$ , and the already available fine-scale PV traces  $\phi_{PV,1}^v$ ,  $\phi_{PV,2}^e$ ,  $\phi_{PV,3}^f$ , and  $\phi_{PV,4}^\tau$  are utilized. Above, the quantities  $(\varphi^E)_e$  and  $(\varphi^F)_f$  denote the reciprocal orientation of the fine edge  $e$  or fine facet  $f$  and coarse edge  $E$  or coarse facet  $F$ ; see Section 3.1.

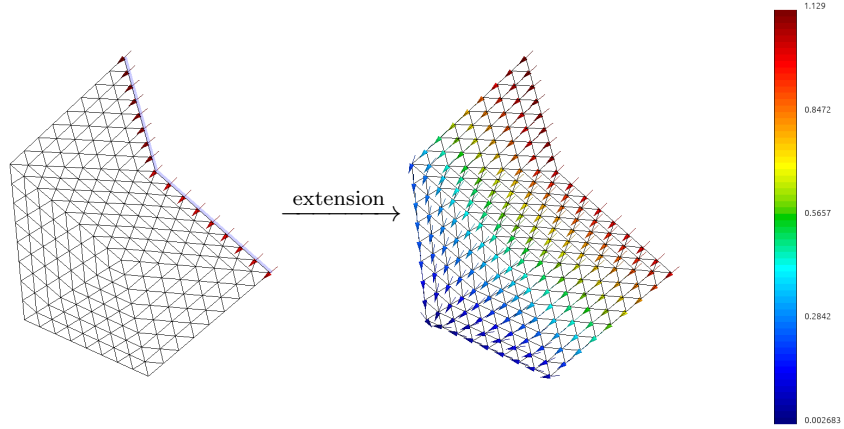
Next, the approximation targets are added to the coarse space after removing any linear dependence. Specifically, coarse basis functions orthogonal (with respect to the  $L^2$  inner product on those entities) to the PV traces and among each other are generated using SVD and local mass matrices formulated on the respective agglomerated entities. Doing so, all coarse basis functions, apart from those stemming the PV traces, have a zero mean and the corresponding trace-space mass matrices produced via RAP on all algebraic levels are diagonal.

Note that this concludes the construction of the space  $\mathcal{V}^H(D_4) = \bigoplus_{T \in \mathcal{T}_H} \mathcal{V}^H(D_{T,4}; T)$  and no extension procedure is needed for this space.

**3.3.3. Extension process.** The extension procedure moves from right to left in the de Rham sequence (3.1) and it considers two types of extensions: boundary extensions from lower to higher dimensional entities and cross-space extensions to ensure the exactness of the coarse sequence. A detailed review of the extension process is provided here and an expository illustration is shown in Figure 3.5. For further analysis, including the feasibility of the extension problems, the demonstration of the exactness (3.2) and commutativity (3.3) properties, see [51].

*Extension from the lowest-dimensional traces.* The first extension is from the lowest-dimensional, for the respective space, agglomerated entity to a one-dimension-higher agglomerated entity; see Figures 3.5a and 3.5b for an illustration. That is, for  $i = 1, \dots, 3$ , the extensions are respectively vertex to edge, edge to facet, and facet to element. The discussion here is associated with the method `hFacetExtension()` of the class `DeRhamSequence`, called within `DeRhamSequence::Coarsen()`.

Let  $L$  be a lowest-dimensional entity,  $K$  a one-dimension-higher entity such that  $L \subset \partial K$ , and  $\mu \in \mathcal{V}^h(D_{L,i}; L)$  be a given target trace on  $L$ . Denote with  $D_{K,i}$  the restriction of the differential operator  $D_i$  to the entity  $K$ , and with  $D_{K,i}^*$  its adjoint. The discretized version of  $D_{K,i}$  is obtained by extracting the corresponding submatrix



(A) Extension of the PV trace on the agglomerated facet

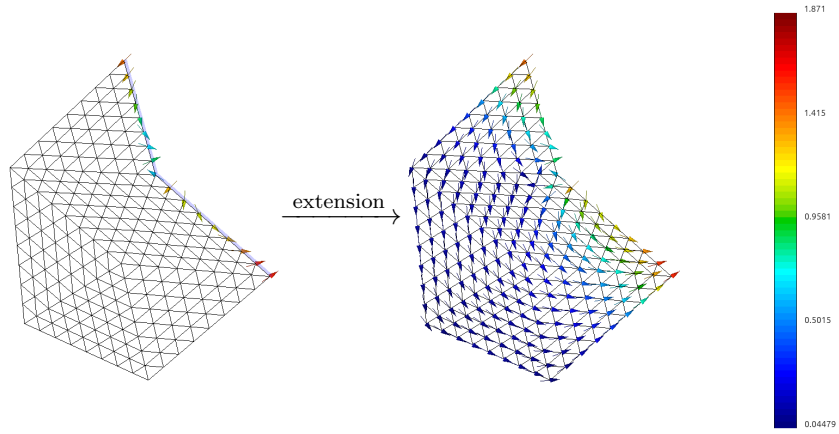
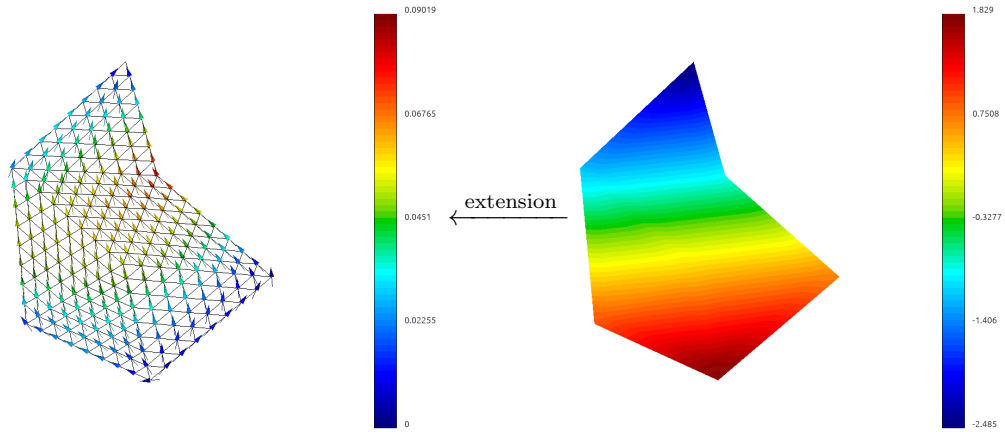
(B) Extension of a trace, on the agglomerated facet,  $L^2$ -orthogonal to the PV trace(C) Cross-space extension from an  $L^2$  basis function to an  $H(\text{div})$  bubble function

FIGURE 3.5. A two-dimensional illustration of local extension procedures producing shape functions in  $H(\text{div})$  on a sample agglomerated element, involving respective traces on an agglomerated facet (marked with a light shade) and bubble functions in the agglomerate.

from  $D_i^h$ , while the discretized adjoint is obtained by matrix transposition. Also, introduce the trace operator  $\gamma_{K,i} : \mathcal{V}^h(D_{K,i}; K) \rightarrow \mathcal{V}^h(D_{\partial K,i}; \partial K)$ , which is defined in Section 2.2 for an arbitrary domain  $\Omega$ . Namely, for  $i = 1, \dots, 3$ ,  $L$  is respectively an agglomerated vertex, edge, and facet, while  $\gamma_{K,i}$  on  $L$  is respectively a point value on the vertex, tangential flow on the edge, and normal flux on the facet. Using the above notation, the extension of  $\mu$  to  $K$ ,  $\phi_e \in \mathcal{V}^h(D_{K,i}; K)$ , is obtained by solving the local PDE formally expressed as:

$$(3.6) \quad \begin{cases} \phi_e + D_{K,i}^* \psi & = 0 & \text{in } K, \\ D_{K,i} \phi_e & = \lambda \phi_{\text{PV},i+1}^K & \text{in } K, \\ (\psi, \phi_{\text{PV},i+1}^K)_{L^2(K)} & = 0 & \text{in } \mathbb{R}, \\ \gamma_{K,i} \phi_e & = \mu & \text{on } L, \\ \gamma_{K,i} \phi_e & = 0 & \text{on } \partial K \setminus L, \end{cases}$$

where  $\phi_{\text{PV},i+1}^K$  is the PV trace in  $\mathcal{V}^h(D_{K,i+1}; K)$  associated with  $K$ ,  $\psi$  is a local-on- $K$  function in  $\mathcal{V}^h(D_{K,i+1}; K)$ , and  $\lambda$  is the scalar Lagrangian multiplier associated with the orthogonality constraint  $(\psi, \phi_{\text{PV},i+1}^K)_{L^2(K)} = 0$ . The orthogonality constraint guarantees the solvability of the above problem by ensuring that  $\psi$  has zero mean on  $K$ . The value of  $\lambda = (\mu, \phi_{\text{PV},i}^L)_{L^2(L)}$  is determined by use of the Stokes' theorem thanks to (3.5).

Next,  $\mathcal{V}^h(D_{K,i}; K)$  bubble functions on  $K$  are obtained by a cross-space extension from  $\mathcal{V}^h(D_{K,i+1}; K)$  to  $\mathcal{V}^h(D_{K,i}; K)$ ; see Figure 3.5c for an illustration. This is necessary to preserve the exactness of the coarse sequence; see (3.2). For each target trace  $\phi_{\perp,i+1}^K \in \mathcal{V}^h(D_{K,i+1}; K)$  such that  $(\phi_{\perp,i+1}^K, \phi_{\text{PV},i+1}^K)_{L^2(K)} = 0$ , the corresponding bubble function,  $\phi_b \in \mathcal{V}^h(D_{K,i}; K)$ , is obtained by solving

$$(3.7) \quad \begin{cases} \phi_b + D_{K,i}^* \psi & = 0 & \text{in } K, \\ D_{K,i} \phi_b & = \phi_{\perp,i+1}^K + c \phi_{\text{PV},i+1}^K & \text{in } K, \\ (\psi, \phi_{\text{PV},i+1}^K)_{L^2(K)} & = 0 & \text{in } \mathbb{R}, \\ \gamma_{K,i} \phi_b & = 0 & \text{on } \partial K, \end{cases}$$

where  $\psi \in \mathcal{V}^h(D_{K,i+1}; K)$  is also zero-mean,  $c = 0$ , and the constraint serves to stabilize the system.

Finally, to ensure the approximation properties, additional  $D_{K,i}$ -free bubble functions<sup>3</sup> in  $\mathcal{V}^h(D_{K,i}; K)$  are produced by projecting the given target traces in  $\mathcal{V}^h(D_{K,i}; K)$  associated with  $K$  onto the respective space of  $D_{K,i}$ -free bubble functions and filtering out any linear dependence.

Upon completion of all the extensions in this steps, shape functions, co-chain projectors and exterior derivative operators for the spaces  $\mathcal{V}^H(D_{E,1}; E)$ ,  $\mathcal{V}^H(D_{F,2}; F)$ ,  $\mathcal{V}^H(D_{T,3}; T)$  are defined for all coarse edges  $E$ , facets  $F$ , and elements  $T$  of the coarse mesh. In particular, the construction of  $\mathcal{V}^H(D_3) = \bigoplus_{T \in \mathcal{T}_H} \mathcal{V}^H(D_{T,3}; T)$  is now complete.

*Further extensions to higher-dimensional agglomerated entities.* For the case of  $i = 2$ , one more extension step (facets  $\rightarrow$  elements) is necessary, while two such steps (edges  $\rightarrow$  facets  $\rightarrow$  elements) are required for  $i = 1$ . Each such step has the form presented below. The discussion here, for each extension step, is associated with the method `hRidgePeakExtension()` of the class `DeRhamSequence`, called within `DeRhamSequence::Coarsen()`.

<sup>3</sup>That is, functions  $\phi \in \mathcal{V}^h(D_{K,i}; K)$  such that  $D_{K,i}\phi = 0$  and  $\gamma_{K,i}\phi = 0$ .



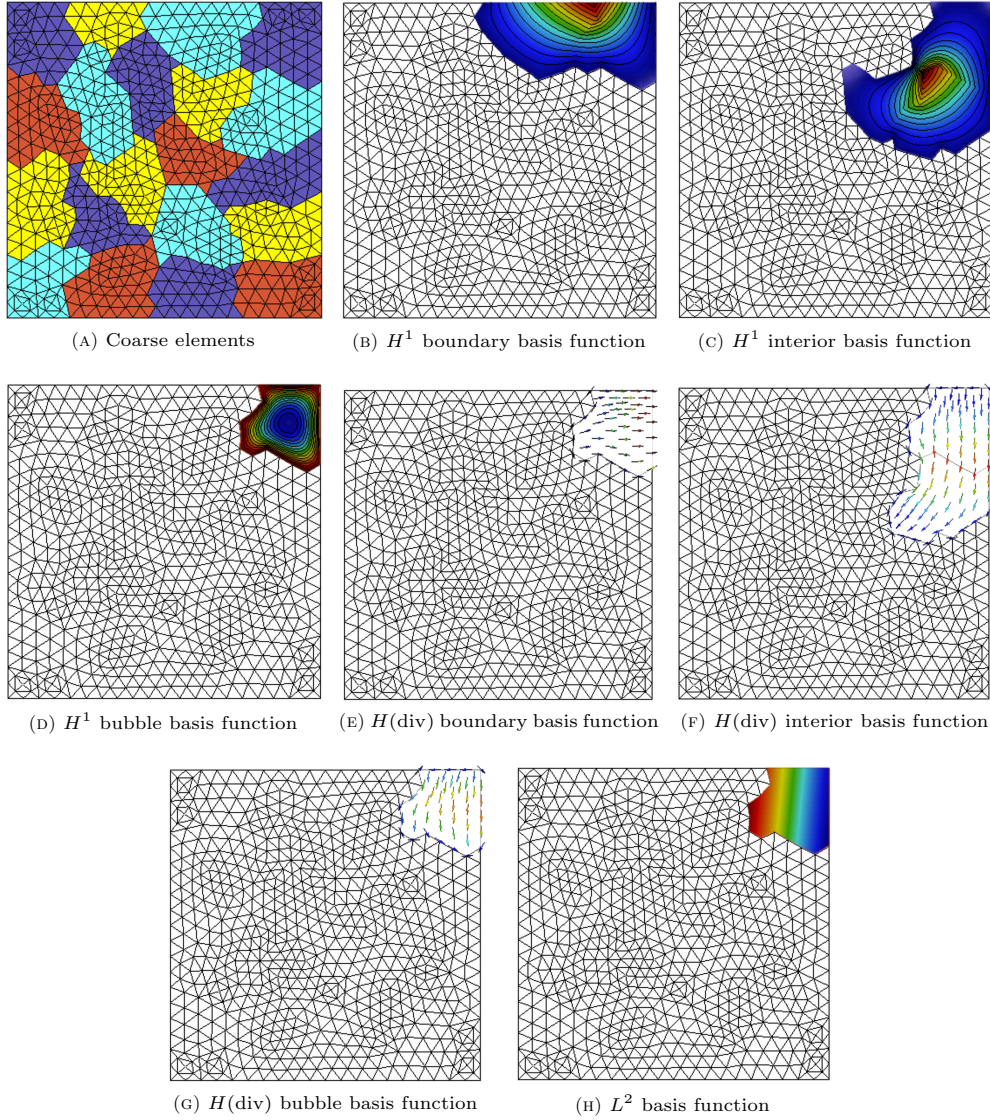


FIGURE 3.6. A two-dimensional illustration of coarse basis functions. Their supports match exactly the respective coarse elements.

Let  $N$  be a lower-dimensional agglomerated entity (but not a lowest-dimensional one),  $M$  a one-dimension-higher agglomerated entity such that  $N \subset \partial M$ , and  $\eta \in \mathcal{V}^h(D_{N,i}; N)$  a trace on  $N$  produced by a previous (lower-dimensional) extension. Let  $s^\eta \in \mathcal{V}^h(D_{M,i+1}; M)$  be the extension of  $D_{M,i}\eta$  from  $N$  to  $M$ . Note the  $s^\eta$  is known since the spaces  $\mathcal{V}^H(D_{N,i}; N)$  and  $\mathcal{V}^H(D_{M,i+1}; M)$  were already constructed during the previous extension step. Then, the respective extension of  $\eta$  to  $M$ ,  $\phi_E$ , in  $\mathcal{V}^h(D_{M,i}; M)$  is obtained by solving the following formal local PDE:

$$(3.8) \quad \begin{cases} \phi_E + D_{M,i}^* \chi & = 0 & \text{in } M, \\ D_{M,i} \phi_E - D_{M,i+1}^* D_{M,i+1} \chi & = s^\eta & \text{in } M, \\ \gamma_{M,i} \phi_E & = \eta & \text{on } N, \\ \gamma_{M,i} \phi_E & = 0 & \text{on } \partial M \setminus N, \end{cases}$$

where  $\chi$  is a local-on- $M$  function in  $\mathcal{V}^h(D_{M,i+1}; M)$ . Above, the stabilization term  $D_{M,i+1}^* D_{M,i+1} \chi$  is added to guarantee the uniqueness of  $\chi$ .

To ensure the exactness (3.2),  $\mathcal{V}^h(D_{M,i}; M)$  bubble functions on  $M$  must be included. These bubble functions,  $\phi_B$ , are such that  $D_{M,i} \phi_B = \phi_{0,i+1}^M$ , where  $\phi_{0,i+1}^M \in \text{Ker}(D_{M,i+1}^H) \subset \mathcal{V}^H(D_{M,i+1}; M)$  but expressed in practice in fine-scale  $\mathcal{V}^h(D_{M,i+1}; M)$ -dofs. Note that a basis of  $\text{Ker}(D_{M,i+1}^H)$  was already constructed in the previous extension step and it is associated with respective  $D_{M,i+1}$ -free bubble functions and target traces with a zero mean (i.e., orthogonal to the respective PV targets) in  $\mathcal{V}^H(D_{M,i+1}; M)$ . Then, for each such  $\phi_{0,i+1}^M$ , the corresponding bubble function,  $\phi_B$ , is obtained by solving

$$(3.9) \quad \begin{cases} \phi_B + D_{M,i}^* \chi &= 0 & \text{in } M, \\ D_{M,i} \phi_B - D_{M,i+1}^* D_{M,i+1} \chi &= \phi_{0,i+1}^M & \text{in } M, \\ \gamma_{M,i} \phi_B &= 0 & \text{on } \partial M. \end{cases}$$

Note that the stabilization term  $D_{M,i+1}^* D_{M,i+1} \chi$  is zero for this particular choice of right-hand side.

In the end, the given target traces in  $\mathcal{V}^h(D_i)$  associated with  $M$  are projected on the space of  $D_{M,i}$ -free bubble functions on  $M$  in  $\mathcal{V}^h(D_{M,i}; M)$  and added towards the basis (possibly awaiting further extension) for  $\mathcal{V}^H(D_{M,i}; M)$ , after filtering out any linear dependence.

After one sweep of the above procedure, the space  $\mathcal{V}^H(D_2) = \bigoplus_{T \in \mathcal{T}_H} \mathcal{V}^H(D_{T,2}; T)$  is finalized, while an additional sweep is needed for constructing  $\mathcal{V}^H(D_1) = \bigoplus_{T \in \mathcal{T}_H} \mathcal{V}^H(D_{T,1}; T)$  and thus completing the coarse sequence. Finally, note that approximation targets are not included in  $\mathcal{V}^H(D_1)$ , since any such bubble function would vanish everywhere.

**3.3.4. Summary.** The construction of the coarse shape functions for all spaces in the sequence is summarized in the diagram below, where  $V$ ,  $E$ ,  $F$ , and  $T$  represent an agglomerated vertex, edge, facet, and element, respectively, and  $D_{X,i}$  the restriction of the exterior derivative  $D_i$  ( $i = 1, \dots, 4$ ) to an entity  $X \in \{V, E, F, T\}$ .

$$\begin{array}{c} \mathcal{V}^H(D_{V,1}; V) \blacktriangleleft \\ \downarrow (3.6) \\ \mathcal{V}^H(D_{E,1}; E) \xleftarrow{(3.7)} \mathcal{V}^H(D_{E,2}; E) \circ \blacktriangleleft \\ \downarrow (3.8) \qquad \qquad \downarrow (3.6) \\ \mathcal{V}^H(D_{F,1}; F) \xleftarrow{(3.9)} \mathcal{V}^H(D_{F,2}; F) \circ \xleftarrow{(3.7)} \mathcal{V}^H(D_{F,3}; F) \circ \blacktriangleleft \\ \downarrow (3.8) \qquad \qquad \downarrow (3.8) \qquad \qquad \downarrow (3.6) \\ \mathcal{V}^H(D_{T,1}; T) \xleftarrow{(3.9)} \mathcal{V}^H(D_{T,2}; T) \circ \xleftarrow{(3.9)} \mathcal{V}^H(D_{T,3}; T) \circ \xleftarrow{(3.7)} \mathcal{V}^H(D_{T,4}; T) \circ \blacktriangleleft \end{array}$$

The symbols  $\circ$  and  $\blacktriangleleft$  denote, respectively, the insertion—after filtering out linear dependencies—of the approximation property targets (Section 3.3.1) and the construction of the PV traces (Section 3.3.2). Vertical arrows denote boundary extensions from lower-dimensional entities to higher-dimensional ones, while horizontal arrows denote cross-space extensions necessary to ensure the exactness property (3.2). The label next to each arrow represents the local PDE that is solved to compute the extension (cf. Section 3.3.3).

Given a finer level hierarchy, the construction of the coarse level sequence starts from the trace spaces on the main diagonal of the diagram and moves towards the

bottom left corner of the diagram using the boundary and cross-space extension operators. Particularly, for any  $i \in [1, 4]$  and any sensible coarse entity  $X$ , coarse basis functions (or traces thereof) on  $X$  are obtained, either from given targets or via local computational procedures like the extensions above, as vectors in  $\mathcal{V}^h(D_{X,i}; X)$  on the respective finer dofs, thus building the coarse  $\mathcal{V}^H(D_{X,i}; X)$  by producing its basis and the corresponding coarse  $\mathcal{V}^H(D_{X,i}; X)$ -dofs associated with  $X$ .

*Remark 3.2.* In ParELAG, the extension operators in (3.6), (3.7), (3.8), (3.9) can be attuned to the particular problem to be solved. For example, the coefficients  $\alpha$  and  $\beta$  in (2.2) are incorporated in the appropriate local PDEs. Thus, the extension procedure can be informed about the particular problem of interest and the obtained coarse bases become problem-dependent.

#### 4. SMOOTHERS, COARSE SOLVERS, AND THE MULTIGRID

---

**Algorithm 4.1** A procedure implementing a single multilevel V-cycle. It computes the action of a multilevel preconditioner  $B_{\text{ML}}^{-1}$ , i.e.,  $\mathbf{x}_{\text{ML}} = \mathbf{x}_0 + B_{\text{ML}}^{-1}(\mathbf{b} - A\mathbf{x}_0)$ , applied for solving the linear system  $A\mathbf{x} = \mathbf{b}$ , formulated on the finest level. The hierarchy of system matrices,  $\{A_k\}_{k=1}^{\ell-1}$ , where  $A_1 = A$  denotes the fine-level system matrix, is precomputed before the invocation of the V-cycle. In particular, the coarser-level system matrices  $A_k = (P_k^{k-1})^T A_{k-1} P_k^{k-1}$  for  $k = 2, \dots, \ell - 1$  are obtained via a Galerkin RAP procedure.

---

**PROCEDURE:**  $\mathbf{x}_{\text{ML}} \leftarrow \text{ML}(\{A_k\}_{k=1}^{\ell-1}, \mathbf{b}, \mathbf{x}_0, \{M_k\}_{k=1}^{\ell-1}, \{P_{k+1}^k\}_{k=1}^{\ell-1}, B_\ell^{-1}, l)$

**INPUT:** A hierarchy of linear-system matrices  $\{A_k\}_{k=1}^{\ell-1}$ , a right-hand side vector  $\mathbf{b}$ , a current iterate  $\mathbf{x}_0$ , a hierarchy of relaxation  $\{M_k\}_{k=1}^{\ell-1}$  and prolongation  $\{P_{k+1}^k\}_{k=1}^{\ell-1}$  operators, a solver or preconditioner  $B_\ell^{-1}$  on the coarsest level, and a current level  $l$ .

**OUTPUT:** A new multigrid iterate  $\mathbf{x}_{\text{ML}} \leftarrow \mathbf{x}$ .

**STEPS:**

Initialize:  $\mathbf{x} \leftarrow \mathbf{x}_0$ .

Pre-relax:  $\mathbf{x} \leftarrow \mathbf{x} + M_l^{-1}(\mathbf{b} - A_l \mathbf{x})$ .

Correct (evoke  $B_\ell^{-1}$  or recur):

**if**  $l = \ell - 1$  (i.e., coarsest level reached) **then**

$\mathbf{e}_c \leftarrow B_\ell^{-1}(P_{l+1}^l)^T(\mathbf{b} - A_l \mathbf{x})$ ;

**else**

$\mathbf{e}_c \leftarrow \text{ML}(\{A_k\}_{k=1}^{\ell-1}, (P_{l+1}^l)^T(\mathbf{b} - A_l \mathbf{x}), \mathbf{0}, \{M_k\}_{k=1}^{\ell-1}, \{P_{k+1}^k\}_{k=1}^{\ell-1}, B_\ell^{-1}, l + 1)$ ;

**end if**

$\mathbf{x} \leftarrow \mathbf{x} + P_{l+1}^l \mathbf{e}_c$ .

Post-relax:  $\mathbf{x} \leftarrow \mathbf{x} + M_l^{-T}(\mathbf{b} - A_l \mathbf{x})$ .

---

This section is devoted to describing a scalable multigrid preconditioner for the finite element matrices stemming from discretizations of the bilinear forms in (2.2) attuned to the AMGe hierarchies of de Rham sequences as constructed by ParELAG (see Section 3).

Multigrid preconditioners, implemented via multilevel cycles such as the well-known V-cycle in Algorithm 4.1 (see [64]), have three main components: a hierarchy of spaces given in the form of a hierarchy of prolongator/restriction operators, a *relaxation* (or *smoothing*) procedure, and a solver or preconditioner for the coarsest problem.

In Algorithm 4.1,  $\ell$  denotes the number of levels in the hierarchy, and  $l$  the current level in the hierarchy. As usual in algebraic multigrid literature, the finest level (where

the matrix  $A$  and right-hand side  $\mathbf{b}$  are defined) is denoted by  $l = 1$ . The smoothing procedure and prolongator operator at level  $l$  in the hierarchy are denoted by  $M_l$  and  $P_{l+1}^l$ , respectively, while the coarse grid solver is denoted by  $B_\ell^{-1}$ . Externally, the procedure is invoked with  $l = 1$  and it calls itself recursively for an increasing value of  $l$ , until the coarsest level  $l = \ell$  is reached. Note that while in principle Algorithm 4.1 can be used standalone iteratively to obtain a stationary (or fixed-point) iterative method, the main interest here is to apply the preconditioner  $B_{\text{ML}}^{-1}$  within a preconditioned conjugate gradient (PCG) method. In that case, Algorithm 4.1 is invoked at each PCG iteration with  $\mathbf{x}_0 = \mathbf{0}$  and  $\mathbf{b}$  denoting the residual at the current iteration.

In what follows,  $A_{D_i}$  denotes the finite element matrix stemming from a discretization of the bilinear form  $a_{D_i}(u, v) = (\alpha_i D_i u, D_i v)_0 + (\beta_i u, v)_0$  for  $u, v \in H(D_i)$  and  $\alpha_i > 0$ ,  $\beta_i \geq 0$ . The cases  $i = 2, 3$  corresponds, respectively, to the  $H(\text{curl})$  and  $H(\text{div})$  forms in (2.2) considered here. By use of the ParELAG prologation operators  $\{P_{k+1}^k\}_{k=1}^{l-1}$ , the Galerkin projections of  $A_{D_i}$  at a generic coarse level  $l + 1$  are defined as  $A_{D_i}^{l+1} = (P_{l+1}^l)^T A_{D_i}^l P_{l+1}^l$ , where  $A_{D_i}^1 = A_{D_i} = A_{D_i}^h$  corresponds to the finest level (i.e., the finite element level).

The constructions of hybrid (“combined,” a.k.a. “Hiptmair”) smoothers  $M_l$  for  $A_{D_i}^l$  and coarse AMS/ADS solvers  $B_\ell^{-1}$  for  $A_{D_i}^\ell$  are respectively described in Sections 4.1 and 4.2. For the coarse solvers, in particular, ParELAG constructs the transition and projection operators on the AMGe coarse de Rham sequence, which mimic those of the fine-grid de Rham sequence constructed in MFEM, for use within the AMS and ADS solvers in the HYPRE library [1].

**4.1. Relaxation via hybrid smoothers.** A general level-independent smoothing procedure based on [34] (see also [64, Appendix F]) is outlined here, providing the relaxation processes  $\{M_k\}$  in Algorithm 4.1 for all levels.

For each matrix  $A_{D_i}^l$ , let  $M_{D_i}^l$  denote the corresponding (point) smoother, e.g., a Jacobi-type, Gauss-Seidel-type, or their block or  $\ell^1$ -scaled variants smoothers. It is well known [11] that these type of smoothers do not lead to optimal (mesh independent) multigrid preconditioners for  $H(\text{curl})$  and  $H(\text{div})$  forms, due to the large near-null-space of these operators. Hybrid smoothers need to “reach” in the reverse direction of the de Rham sequence to perform an additional smoothing step on the near-null-space components.

Specifically, hybrid smoothers leverage certain decompositions of the spaces and the exactness of the de Rham sequence (3.2), similar to the methods in Section 4.2 below. For  $i = 2, 3$ , at a generic level  $l$  of the hierarchy, the stable decomposition  $\mathcal{V}^l(D_i) = \text{Ker}(D_i^l) \oplus [\text{Ker}(D_i^l)]^\perp$  allows to define efficient smoothers for each component separately. Smoothing the  $[\text{Ker}(D_i^l)]^\perp$  component is addressed by  $M_{D_i}^l$ . Smoothing the component in  $\text{Ker}(D_i^l)$  requires the construction of a point smoother  $M_{D_{i-1}}^l$  for the *auxiliary* matrix  $(D_{i-1}^l)^T A_{D_i}^l D_{i-1}^l$ , where the transition operator  $D_{i-1}^l$  is an exterior derivative at level  $l$  constructed in ParELAG.

Then, for  $i = 2, 3$ , the smoothing step  $\mathbf{x}_1 = \mathbf{x}_0 + (\mathbb{M}_{D_i}^l)^{-1}(\mathbf{b} - A_{D_i}^l \mathbf{x}_0)$  with the combined smoother  $(\mathbb{M}_{D_i}^l)$  reads

$$\begin{aligned} (4.1) \quad \mathbf{x}_{\frac{1}{2}} &= \mathbf{x}_0 + (M_{D_i}^l)^{-1}(\mathbf{b} - A_{D_i}^l \mathbf{x}_0), \\ \mathbf{x}_1 &= \mathbf{x}_{\frac{1}{2}} + D_{i-1}^l (M_{D_{i-1}}^l)^{-1} (D_{i-1}^l)^T (\mathbf{b} - A_{D_i}^l \mathbf{x}_{\frac{1}{2}}). \end{aligned}$$

That is, the *error propagation* operator satisfies

$$I - (\mathbb{M}_{D_i}^l)^{-1} A_{D_i}^l = \left[ I - D_{i-1}^l (M_{D_{i-1}}^l)^{-1} (D_{i-1}^l)^T A_{D_i}^l \right] \left[ I - (M_{D_i}^l)^{-1} A_{D_i}^l \right].$$

Notice that, since  $D_i^l D_{i-1}^l = 0$  (see (3.2)), recursively utilizing  $\mathbb{M}_{D_{i-1}}^l$  in place of  $M_{D_{i-1}}^l$  in (4.1) changes nothing. Therefore, there is no need to “reach” further than one step backwards into the de Rham sequence. To compute an iteration with  $(\mathbb{M}_{D_i}^l)^{-T}$ , reverse the order of the steps in (4.1), while respectively using  $(M_{D_i}^l)^{-T}$  and  $(M_{D_{i-1}}^l)^{-T}$  in place of the ones in (4.1).

In the numerical results presented in Section 5, the so called  $\ell^1$ -scaled symmetric block Gauss-Seidel smoother [13], as implemented in HYPRE, is used for all  $M_{D_i}^l$ . For more details on the analysis of the hybrid approach in a multigrid setting, which counts on the exactness property (3.2), see [64, Appendix F].

**4.2. Coarse solvers using AMS and ADS.** Similarly to Section 4.1, special (a.k.a. regular) decompositions (cf. [37]) of the finite element spaces of interest are used to break the problem of obtaining a holistic preconditioner into the preconditioning of each component of the decomposition. This provides an *auxiliary space preconditioner* that reduces the problem to preconditioning a few  $H^1$ -type forms, which can be efficiently addressed by AMG, and smoothing. As a part of HYPRE, BoomerAMG is used in this case. In this work, AMS and ADS, possibly wrapped in PCG and performing multiple iterations up to a given tolerance, are to be used as coarse solvers  $B_\ell^{-1}$  in Algorithm 4.1.

To understand how to generalize AMS and ADS to the coarse problems generated by ParELAG, a brief overview of these methods applied to the finite element discretization level is presented below. To this aim, the finite element space  $\underline{\mathcal{V}}^h(D_1) = \underline{\mathcal{V}}^h(\text{grad}) = [\mathcal{V}^h(\text{grad})]^3$  of vectorial  $H^1$ -conforming functions is introduced. Next, for  $i = 2, 3$  the interpolation operators  $\hat{\Pi}_i^h : \underline{\mathcal{V}}^h(D_1) \rightarrow \mathcal{V}^h(D_i)$  are constructed by means of an overwriting assembly procedure collecting the local (element-by-element) versions of these operators.

Using the exterior derivative  $D_{i-1}^h$  and the newly introduced interpolator  $\hat{\Pi}_i^h$ , an arbitrary function  $\mathbf{v}^h \in \mathcal{V}^h(D_i)$  ( $i = 2, 3$ ) admits the stable decomposition [37]

$$\mathbf{v}^h = \tilde{\mathbf{v}}^h + \hat{\Pi}_i^h \mathbf{r}^h + D_{i-1}^h \mathbf{z}^h \quad \text{for } \mathbf{v}^h \in \mathcal{V}^h(D_i), i = 2, 3,$$

where  $\tilde{\mathbf{v}}^h \in \mathcal{V}^h(D_i)$ ,  $\mathbf{r}^h \in \underline{\mathcal{V}}^h(\text{grad})$ , and  $\mathbf{z}^h \in \mathcal{V}^h(D_{i-1})$ . Based on the above decomposition, (an additive version of<sup>4</sup>) the auxiliary space preconditioner has the form [45, 46]

$$(4.2) \quad (B_{D_i}^h)^{-1} = (M_{D_i}^h)^{-1} + \hat{\Pi}_i^h (B_{H^1}^h)^{-1} (\hat{\Pi}_i^h)^T + D_{i-1}^h (B_{D_{i-1}}^h)^{-1} (D_{i-1}^h)^T,$$

where  $M_{D_i}^h$  is a smoother for  $A_{D_i}^h$  and  $B_{H^1}^h$  is an AMG preconditioner (e.g., BoomerAMG) of the vector  $H^1$ -type matrix  $A_{H^1}^h = (\hat{\Pi}_i^h)^T A_{D_i}^h \hat{\Pi}_i^h$ , and  $B_{D_{i-1}}^h$  is a multilevel preconditioner for  $A_{D_{i-1}}^h = (D_{i-1}^h)^T A_{D_i}^h D_{i-1}^h$ . Note that, for  $i = 2$ , the matrix  $A_{D_{i-1}}^h$  is equivalent to the discretization of an  $H^1$ -conforming form, and thus can be preconditioned using AMG. For  $i = 3$ ,  $A_{D_{i-1}}^h$  is equivalent to the discretization of a singular  $H(\text{curl})$ -conforming form, thus requiring the use of another auxiliary space preconditioner (4.2) with  $i = 2$ .

<sup>4</sup>Several different variations, including multiplicative and ones that treat  $\underline{\mathcal{V}}^h(\text{grad})$  in a scalar component-wise fashion, are implemented in HYPRE [1].

The auxiliary space preconditioner (4.2) can be seamlessly generalized to the preconditioning the coarse matrices  $A_{D_i}^\ell$  ( $i = 2, 3$ ), generated using the ParELAG hierarchy of nested de Rham sequences, by replacing the finite element exterior derivative  $D_{i-1}^h$  and interpolation operator  $\hat{\Pi}_i^h$  with their coarse level counterparts. In particular, the coarse level interpolation operator  $\hat{\Pi}_i^\ell : \underline{\mathcal{V}}^\ell(\text{grad}) \rightarrow \mathcal{V}^\ell(D_i)$  is constructed in ParELAG by means of a RAP procedure from the finer level interpolator. That is, denoting the vectorial counterpart of the ParELAG prolongator  $(P_1)_{l+1}^l : \mathcal{V}^{l+1}(D_1) \rightarrow \mathcal{V}^l(D_1)$  by  $\hat{P}_{l+1}^l : \underline{\mathcal{V}}^{l+1}(D_1) \rightarrow \underline{\mathcal{V}}^l(D_1)$ ,  $\hat{\Pi}_i^\ell$  is obtained by applying  $\ell - 1$  times the recursion

$$\hat{\Pi}_i^{l+1} = (\Pi_i)_{l+1}^l \hat{\Pi}_i^l \hat{P}_{l+1}^l, \quad l = 1, \dots, \ell - 1,$$

where  $\hat{\Pi}_i^1 = \hat{\Pi}_i^h$  is the interpolation operator at the finite element discretization level (provided by MFEM) and  $(\Pi_i)_{l+1}^l$  denotes the respective ParELAG-generated co-chain projector from level  $l$  onto level  $l + 1$ .

**4.3. Overview of composite solvers in ParELAG.** The ParELAG library provides access to a variety of solvers for sparse linear systems, including for block systems. Some of them are implemented within ParELAG itself, like the hybrid smoothers of Section 4.1 and the V-cycle of Algorithm 4.1 for AMGe, while others make use of external libraries, such as HYPRE [1], MFEM [3], SUPERLU\_DIST [52], and STRUMPACK [6]. Furthermore, solvers can be combined into a composite solver for the linear system of interest.

ParELAG achieves this by first generating a *solver (or preconditioner) library* (an object of class `SolverLibrary`) from an XML configuration file with a very intuitive syntax. Such a file declares all the solvers and preconditioners needed by the application, together with their specific parameters and how they are combined. A solver is declared by assigning a name and a list of parameters for a particular method internally provided by ParELAG. For example, one can declare a solver in the XML file that represents a multigrid method like the one in Algorithm 4.1 and appoint other solvers from the solver library to act as smoothers and coarse solvers, which have their own sets of parameters and may, in turn, internally employ other solvers or preconditioners from the solver library. The static member function `CreateLibrary()` is defined to instantiate a `SolverLibrary` object from the provided XML configuration file.

To instantiate a solver, users first interrogate the `SolverLibrary` by calling its member function `GetSolverFactory()`, which returns a *solver factory* (an object of class `SolverFactory`) for the desired solver. The solver (i.e., an object of class `Solver`<sup>5</sup>) is then instantiated by calling the member function `BuildSolver()` of `SolverFactory`. For solving a linear system, users then call the `Mult()` member, as they would do with any linear solver implemented in MFEM. Such a paradigm of solver structuring may sound familiar to a reader that has been exposed to some of the popular available solver libraries.

Examples of XML configuration files for solving the  $H(\text{curl})$  and  $H(\text{div})$  problems presented in Section 5 are included in the ParELAG mini application of MFEM. Additional examples to configuring solver factories for AMS, ADS, Krylov space methods, hybrid smoothers, AMGe cycles, block preconditioners, and other methods within HYPRE and MFEM, can be found in the ParELAG library.

<sup>5</sup>`Solver` is a virtual class defined in MFEM.

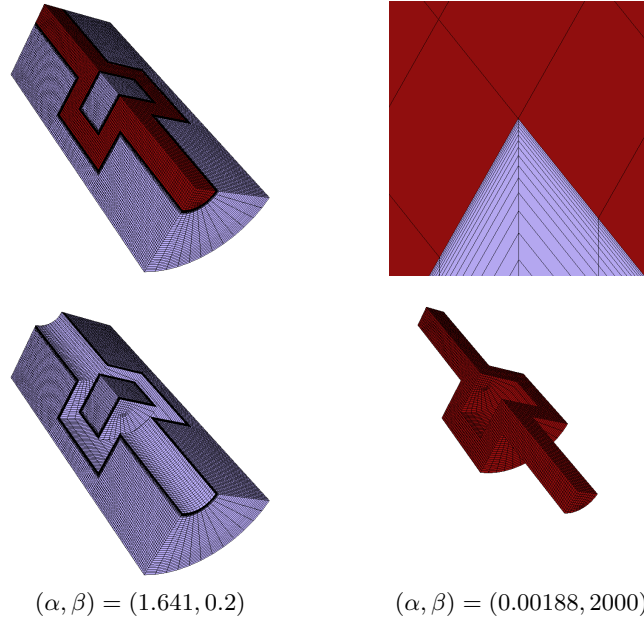


FIGURE 5.1. Domain, initial or starting mesh (including a close-up of the graded mesh in the upper right) of 116640 hexahedral elements, and piecewise constant coefficients for the numerical examples.

## 5. NUMERICAL EXAMPLES

This section contains numerical results employing ParELAG in the context of the discussed AMGe multigrid solvers for (2.2). The results are produced using the ParELAG mini application [4] in MFEM.

**5.1. On the benchmark problem.** The benchmark considered here is inspired by the so-called “crooked pipe” problem (see [32, 29, 46]). This involves solving  $H(\text{curl})$  and  $H(\text{div})$  forms with scalar discontinuous coefficients with large jumps on a graded mesh with highly stretched (anisotropic) elements. For demonstration, the methodology is applied for solving linear systems coming from discretizations of formulations using the bilinear forms in (2.2), a constant right-hand side, and homogeneous essential boundary conditions for simplicity. The computational domain comprised of two different materials and the (coarsest) finite element mesh are depicted in Figure 5.1. The coefficients are set as  $(\alpha, \beta) = (1.641, 0.2)$  in the outer material (depicted with the lighter color) and  $(\alpha, \beta) = (0.00188, 2000)$  in the inner core (depicted with the darker color).

The systems coming from (2.2) are solved using inexact PCG (see, e.g., [30]) preconditioned by a single AMGe V-cycle; see Section 4. The iterative process is stopped when the relative size of the residual, measured by the preconditioner-induced norm, is reduced by six order of magnitude (i.e.,  $10^{-6}$  relative tolerance).

The V-cycles use AMGe hierarchies, as described above, and hybrid smoothers for pre and post-relaxation; see Section 4.1. An application of the hybrid smoother invokes two sweeps of  $\ell^1$ -scaled symmetric block Gauss-Seidel for each *primary* and *auxiliary* smoothings within the hybrid approach. A fixed number of five iterations of



PCG preconditioned by AMS or ADS, respectively, serves as a solver on the coarsest level<sup>6</sup>; see Section 4.2.

In the tests, the mesh in Figure 5.1 is uniformly refined multiple times to obtain a fine-grid problem, which is consequently solved in parallel by the methods discussed in this paper. *Weak scaling* experiments are considered here. That is, as the mesh is refined the number of processors is also increased so that the number of elements per processor is maintained constant.

**5.2. Results.** Computational results on solving systems induced by (2.2) in parallel for these generally challenging problems are presented here, employing lowest and next to the lowest order finite elements and uniformly refining the initial mesh in Figure 5.1. The tests utilize the Quartz cluster at Lawrence Livermore National Laboratory. It is equipped on each node with 128 GB of memory and two 18-core Intel Xeon E5-2695 v4 CPUs at 2.1 GHz, resulting in 36 computational cores per node, and the total number of computational nodes (cores) is 2,988 (107,568). The peak single CPU memory bandwidth is 77 GB/s and the Cornelis Networks Omni-Path provides the inter-node connection.

The number of PCG iterations ( $it_e$ ), the number of dofs, and the number of elements (elems) on the finest level are reported, as well as the number of uniform mesh refinements (refs) employed to obtain the fine mesh, the total number of levels (denoted by  $\ell$ ) in the AMGe hierarchy, and the number of utilized processors<sup>7</sup> (procs). Also, the *grid complexity* (GC) is reported, which is the total number of dofs in the hierarchy, respectively of nested subspaces of  $H(\text{curl})$  or  $H(\text{div})$ , over the number of finest dofs. The *operator complexity* (OC) is defined similarly but using the number of nonzeros in the sparsity patterns of the matrices in the AMGe hierarchy instead.

For comparison, results obtained with PCG preconditioned by HYPRE AMS and ADS acting on the finite element level are also presented. These includes the number of iterations ( $it_a$ ) and wall-clock timings. Also, for comparison and completeness, two test cases are demonstrated: one where the number of AMGe levels is kept fixed (equal to 3) as the fine mesh is refined, essentially also refining the coarsest level, and another where as the fine mesh is refined the number of levels is increased so that the coarsest level is constant and coinciding with the initial mesh in Figure 5.1.

Recall that the construction of the whole coarse de Rham sequences includes the element agglomeration times, the local extension procedures, and building other necessary constructs. As expected, we observe that the construction time almost does not grow when refining the mesh and increasing the number of processors to maintain constant number of elements per processor, especially in the case of constant number of levels, since the majority of the time spent is on the local extension procedures, which scale perfectly (they are embarrassingly parallel), as they involve no communication. Therefore, the weak scaling of the construction of the whole coarse de Rham sequences on all levels is almost perfect, i.e., it takes almost constant time.

**5.2.1. Results for lowest order elements.** Problem information and solvers iterations are shown in Table 5.1 for both  $H(\text{curl})$  and  $H(\text{div})$ . Timing plots, using wall-clock times as reported by the code of the miniapp, are shown in Figure 5.2, including wall-clock times for the entire program executions. In the legend of Figure 5.2,

<sup>6</sup>The particular choice is largely motivated by the objective to demonstrate the flexibility of the ParELAG SolverLibrary and its ability to combine a variety of solvers and smoothers. Using a single or a few applications of the respective AMS or ADS, without PCG, is also a valid option here.

<sup>7</sup>Strictly speaking, this is the number of individual independent computational units, i.e., cores.



refs	procs	elems	$\ell$	$H(\text{curl})$			$H(\text{div})$		
				dofs	it <sub>e</sub>	it <sub>a</sub>	dofs	it <sub>e</sub>	it <sub>a</sub>
2	72	7,464,960	3	22,772,484	131	93	22,583,232	36	31
3	576	59,719,680	3	180,667,656	198	118	179,912,448	55	44
4	4,608	477,757,440	3	1,439,303,184	231	148	1,436,285,952	86	60
3	576	59,719,680	4	180,667,656	169	118	179,912,448	54	44
4	4,608	477,757,440	5	1,439,303,184	190	148	1,436,285,952	77	60

TABLE 5.1. Weak scaling results with lowest order finite elements for both  $H(\text{curl})$  and  $H(\text{div})$  problems, as provided by (2.2) and Figure 5.1. Here,  $\ell$  denotes the number of levels in the AMGe hierarchy, it<sub>e</sub> – the number of PCG iteration using the proposed AMGe preconditioner, and it<sub>a</sub> – the number of PCG iteration using the auxiliary space (AMS or ADS) preconditioners from the HYPRE library. In all cases, elems / procs = 103,680, while the GC and OC of the AMGe hierarchy are approximately constant and equal to 1.14 for all refinement levels.

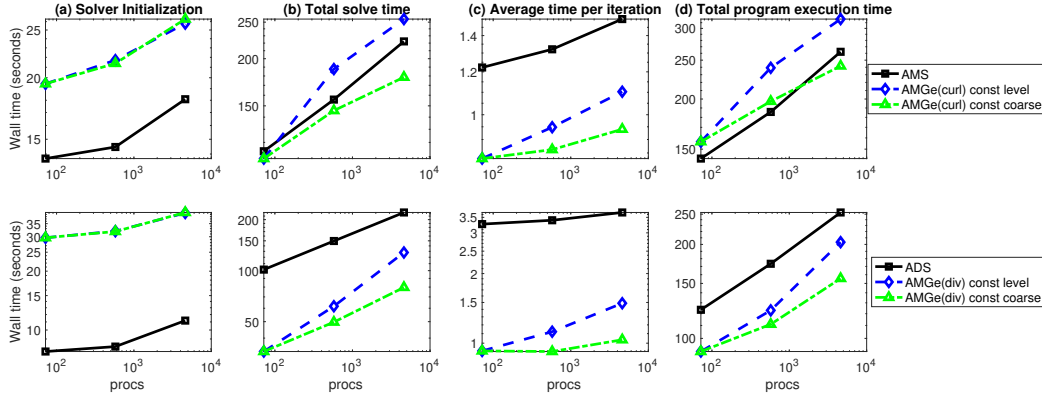


FIGURE 5.2. Solver weak scaling with lowest order finite elements, where elems / procs = 103,680:  $H(\text{curl})$ -problem (top row) and  $H(\text{div})$ -problem (bottom row).

AMGe(curl) and AMGe(div) denote the AMGe solvers for the  $H(\text{curl})$  and  $H(\text{div})$  problems, respectively. The cases of constant number of levels and a constant size of the coarsest problem are indicated with *const levels* and *const coarse*. Finally, AMS and ADS represent the solvers from the HYPRE library.

The AMGe approach delivers good and competitive performance in comparison with the state of the art represented by AMS and ADS. Moreover, due to the higher set-up cost but shorter solve times, ParELAG has a competitive advantage compared the AMS and ADS when the same hierarchy can be reused in solving multiple linear systems with different right-hand sides. Observe that Figure 5.2 indicates that the more standard case of utilizing AMGe, of increasing the number of AMGe levels, demonstrates better scalability. To further study and exploit this scalability potential in practice for extremely large problems in the setting of extreme parallelism, parallel redistribution and load balancing are needed on coarse levels obtained via AMGe to allow sufficient coarsening when large number of processors are utilized. This is a subject of an ongoing work.

refs	procs	elems	$\ell$	$H(\text{curl})$			$H(\text{div})$		
				dofs	it <sub>e</sub>	it <sub>a</sub>	dofs	it <sub>e</sub>	it <sub>a</sub>
2	720	7,464,960	3	180,667,656	191	166	179,912,448	60	57
3	5,760	59,719,680	3	1,439,303,184	269	223	1,436,285,952	100	78
3	5,760	59,719,680	4	1,439,303,184	285	223	1,436,285,952	84	78

TABLE 5.2. Weak scaling results with next to the lowest order finite elements for both  $H(\text{curl})$  and  $H(\text{div})$  problems, as provided by (2.2) and Figure 5.1. Here,  $\ell$  denotes the number of levels in the AMGe hierarchy, it<sub>e</sub> – the number of PCG iteration using the proposed AMGe preconditioner, and it<sub>a</sub> – the number of PCG iteration using the auxiliary space (AMS or ADS) preconditioners from the HYPRE library. In all cases, elems / procs = 10,368, while the GC and OC of the AMGe hierarchy are approximately constant and equal to 1.09 for all refinement levels.

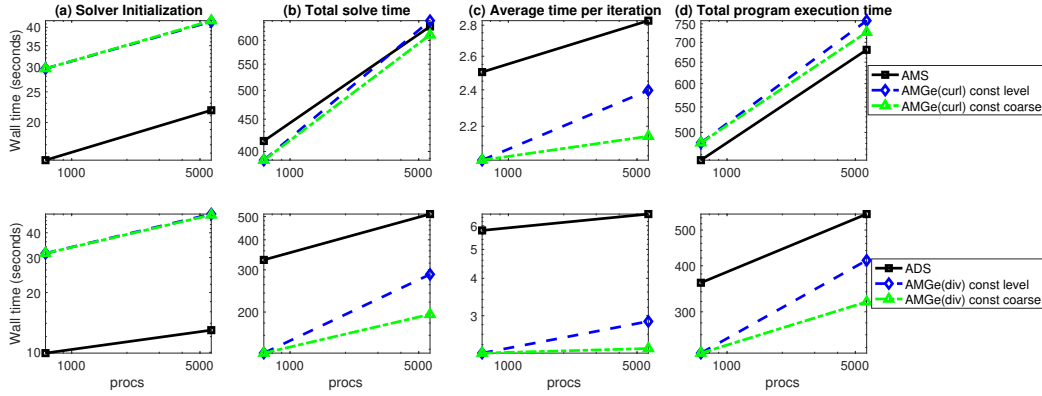


FIGURE 5.3. Solver weak scaling with next to the lowest order finite elements, where elems / procs = 10,368:  $H(\text{curl})$ -problem (top row) and  $H(\text{div})$ -problem (bottom row).

5.2.2. *Results for next to the lowest order elements.* For completeness, results using next to the lowest order finite elements are presented in Table 5.2 and Fig. 5.3, following the paradigm of the previous subsection. Again, the AMGe methodology performs well and is comparable to the state of the art.

*Remark 5.1.* Note that the finest MFEM-generated level and the coarse ParELAG-generated levels in the miniapp by default interpret *next to the lowest order* slightly differently, even if similar, especially when employing hexahedral elements. Consider for example the  $L^2$ -conforming finite elements spaces of piecewise polynomial functions. Being informed about the geometry of the elements and their tensor-product structure, MFEM produces bilinear elements with 8 dofs and basis functions per element. In contrast, ParELAG operates in a generic geometry-agnostic way and by default the finite element order determines the order of the targets. Thus, by default it produces targets and spaces that provide piecewise linear interpolation independently of the shape of the element, resulting in 4 dofs and basis functions per element. Generally, this behavior can be easily altered by changing the way targets are selected, but the concentration here is on the default behavior.

## 6. CONCLUSIONS AND FUTURE WORK

In this paper, we have introduced an AMGe approach for  $H(\text{curl})$  and  $H(\text{div})$  formulations. It involves the construction of coarse de Rham sequences on agglomerated meshes, the use of hybrid (Hiptmair) smoothers, and state-of-the-art auxiliary space multigrid solvers, HYPRE AMS and ADS, for the coarsest level. The methods are described in detail using the exterior calculus framework, which allows for a unified (independent of the number of space dimensions) presentation of the local problems that need be solved to construct the hierarchy of de Rham sequences. A key characteristic of the AMGe technique implemented in ParELAG is that the de Rham sequences at each level of the hierarchy possess the same properties and structures that are defined at the finite element levels, including boundary attributes, exterior derivative operators, co-chain projectors, and interpolation operators between spaces. The paper also provide an overview of the ParELAG implementation of the above methods. The numerical results presented here demonstrate the good performance and weak scaling properties of ParELAG, comparable to the state-of-the-art  $H(\text{curl})$  and  $H(\text{div})$  solvers in HYPRE. A potential limitation of the current implementation is that ParELAG does not, yet, admit agglomerated elements that are shared or redistributed between processors. The development of such functionality is a currently ongoing work, which would also allow us to exploit the improved scaling of deep AMGe cycles. Future work also includes the implementation of AMGe variants of the hybridization and static condensation techniques in [25, 43, 42] for solving the coarsest  $H(\text{div})$  problems.

## ACKNOWLEDGEMENTS

The authors would like to acknowledge all the other developers and contributors to the ParELAG library, including Andrew Barker, Thomas Benson, Ilya Lashuk, Chak Shing Lee, and Sara Osborn (contributors listed in alphabetical order).

## REFERENCES

- [1] HYPRE: Scalable Linear Solvers and Multigrid Methods. <http://computing.llnl.gov/projects/hypre-scalable-linear-solvers-multigrid-methods>.
- [2] METIS: Graph Partitioning and Fill-reducing Matrix Ordering. <http://glaros.dtc.umn.edu/gkhome/views/metis>.
- [3] MFEM: Modular Finite Element Methods Library. <http://mfem.org>. doi:10.11578/dc.20171025.1248.
- [4] ParELAG mini applications in MFEM. <http://github.com/mfem/mfem/tree/master/miniapps/parelag>.
- [5] ParELAG: Parallel Element Agglomeration Algebraic Multigrid Upscaling and Solvers. <http://github.com/LLNL/parelag>.
- [6] STRUMPACK: STRUctured Matrix PACKage. <http://portal.nersc.gov/project/sparse/strumpack>.
- [7] J H Adler and P S Vassilevski. Improving Conservation for First-Order System Least-Squares Finite-Element Methods. In Oleg P Iliev, Svetozar D Margenov, Peter D Minev, Panayot S Vassilevski, and Ludmil T Zikatanov, editors, *Numer. Solut. Partial Differ. Equations Theory, Algorithms, Their Appl.*, pages 1–19, 2013. doi:10.1007/978-1-4614-7172-1\_1.
- [8] JH Adler and Panayot S Vassilevski. Error analysis for constrained first-order system least-squares finite-element methods. *SIAM Journal on Scientific Computing*, 36(3):A1071–A1088, 2014.
- [9] Douglas N Arnold, Richard S Falk, and Jay Gopalakrishnan. Mixed Finite Element Approximation of the Vector Laplace with Dirichlet Boundary Conditions. *Math. Model. Methods Appl. Sci.*, 22(09):1250024, 2012. doi:10.1142/S0218202512500248.
- [10] Douglas N. Arnold, Richard S. Falk, and Ragnar Winther. Preconditioning in  $\mathbf{H}(\text{div})$  and applications. *Math. Comput.*, 66(219):957–985, 1997. doi:10.1090/S0025-5718-97-00826-0.

- [11] Douglas N Arnold, Richard S Falk, and Ragnar Winther. Multigrid in  $H(\text{div})$  and  $H(\text{curl})$ . *Numer. Math.*, 85(2):197–217, 2000. doi:10.1007/PL00005386.
- [12] Douglas N. Arnold, Richard S. Falk, and Ragnar Winther. Finite element exterior calculus: from Hodge theory to numerical stability. *Bull. Am. Math. Soc.*, 47(2):281–354, 2010. doi:10.1090/S0273-0979-10-01278-4.
- [13] A Baker, R Falgout, T Kolev, and U Yang. Multigrid Smoothers for Ultraparallel Computing. *SIAM J. Sci. Comput.*, 33(5):2864–2887, 2011. doi:10.1137/100798806.
- [14] Nathan Bell and Luke N Olson. Algebraic multigrid for  $k$ -form Laplacians. *Numer. Linear Algebr. with Appl.*, 15(2-3):165–185, 2008. doi:10.1002/nla.577.
- [15] Pavel B Bochev, Christopher J Garasi, Jonathan J Hu, Allen C Robinson, and Raymond S Tuminaro. An Improved Algebraic Multigrid Method for Solving Maxwell’s Equations. *SIAM J. Sci. Comput.*, 25(2):623–642, 2003. doi:10.1137/S1064827502407706.
- [16] Pavel B Bochev, Jonathan J Hu, Allen C Robinson, and Raymond S Tuminaro. Towards robust 3D Z-pinch simulations: Discretization and fast solvers for magnetic diffusion in heterogeneous conductors. *Electron. Trans. Numer. Anal.*, 15:186–210, 2003.
- [17] Pavel B Bochev, Jonathan J Hu, Christopher M Siefert, and Raymond S Tuminaro. An Algebraic Multigrid Approach Based on a Compatible Gauge Reformulation of Maxwell’s Equations. *SIAM J. Sci. Comput.*, 31(1):557–583, 2008. doi:10.1137/070685932.
- [18] Daniele Boffi, Franco Brezzi, and Michel Fortin. *Mixed Finite Element Methods and Applications*, volume 44 of *Springer Series in Computational Mathematics*. Springer, Berlin, Heidelberg, 2013.
- [19] Marian Brezina and Panayot S Vassilevski. Smoothed Aggregation Spectral Element Agglomeration AMG: SA- $\rho$ AMGe. In Ivan Lirkov, Svetozar Margenov, and Jerzy Waśniewski, editors, *Large-Scale Sci. Comput.*, pages 3–15, Berlin, Heidelberg, 2012. Springer.
- [20] Thomas A Brunner. Forms of Approximate Radiation Transport. Technical report, SAND2002-1778, Sandia National Laboratories, 2002. doi:10.2172/800993.
- [21] Z Cai, R Lazarov, T A Manteuffel, and S F McCormick. First-Order System Least Squares for Second-Order Partial Differential Equations: Part I. *SIAM J. Numer. Anal.*, 31(6):1785–1799, 1994. doi:10.1137/0731091.
- [22] Zhiqiang Cai, Charles Tong, Panayot S Vassilevski, and Chunbo Wang. Mixed finite element methods for incompressible flow: Stationary Stokes equations. *Numer. Methods Partial Differ. Equ.*, 26(4):957–978, 2010. doi:10.1002/num.20467.
- [23] Zhiqiang Cai, Chunbo Wang, and Shun Zhang. Mixed Finite Element Methods for Incompressible Flow: Stationary Navier-Stokes Equations. *SIAM J. Numer. Anal.*, 48(1):79–94, 2010. doi:10.1137/080718413.
- [24] T Chartier, R Falgout, V Henson, J Jones, T Manteuffel, S McCormick, J Ruge, and P Vassilevski. Spectral AMGe ( $\rho$ AMGe). *SIAM J. Sci. Comput.*, 25(1):1–26, 2003. doi:10.1137/S106482750139892X.
- [25] V Dobrev, T Kolev, C S Lee, V Tomov, and P S Vassilevski. Algebraic Hybridization and Static Condensation with Application to Scalable  $H(\text{div})$  Preconditioning. *SIAM J. Sci. Comput.*, 41(3):B425–B447, 2019. doi:10.1137/17M1132562.
- [26] Hillary R Fairbanks, Sarah Osborn, and Panayot S Vassilevski. Estimating posterior quantity of interest expectations in a multilevel scalable framework. *Numer. Linear Algebr. with Appl.*, 28(3):e2352, 2021. doi:10.1002/nla.2352.
- [27] Hillary R Fairbanks, Umberto Villa, and Panayot S Vassilevski. Multilevel Hierarchical Decomposition of Finite Element White Noise with Application to Multilevel Markov Chain Monte Carlo. *SIAM J. Sci. Comput.*, pages S293–S316, 2021. doi:10.1137/20M1349606.
- [28] Robert D Falgout and Panayot S Vassilevski. On Generalizing the Algebraic Multigrid Framework. *SIAM J. Numer. Anal.*, 42(4):1669–1693, 2004. doi:10.1137/S0036142903429742.
- [29] NA Gentile. Implicit Monte Carlo diffusion—an acceleration method for Monte Carlo time-dependent radiative transfer simulations. *Journal of Computational Physics*, 172(2):543–571, 2001.
- [30] Gene H Golub and Qiang Ye. Inexact preconditioned conjugate gradient method with inner-outer iteration. *SIAM Journal on Scientific Computing*, 21(4):1305–1320, 1999.
- [31] J Gopalakrishnan, M Neumüller, and P S Vassilevski. The Auxiliary Space Preconditioner for the de Rham Complex. *SIAM J. Numer. Anal.*, 56(6):3196–3218, 2018. doi:10.1137/17M1153376.
- [32] F Graziani and J LeBlanc. The crooked pipe test problem. *Lawrence Livermore National Laboratory Report UCRL-MI-143393*, 2000.
- [33] R Hiptmair. Multigrid method for  $H(\text{div})$  in three dimensions. *Electron. Trans. Numer. Anal.*, 6:133–152, 1997.

- [34] R Hiptmair. Multigrid Method for Maxwell's Equations. *SIAM J. Numer. Anal.*, 36(1):204–225, 1998. doi:10.1137/S0036142997326203.
- [35] R Hiptmair. Finite elements in computational electromagnetism. *Acta Numer.*, 11:237–339, 2002. doi:10.1017/S0962492902000041.
- [36] R Hiptmair, G Widmer, and J Zou. Auxiliary space preconditioning in  $H_0(\text{curl}; \Omega)$ . *Numer. Math.*, 103(3):435–459, 2006. doi:10.1007/s00211-006-0683-0.
- [37] R Hiptmair and J Xu. Nodal Auxiliary Space Preconditioning in  $\mathbf{H}(\text{curl})$  and  $\mathbf{H}(\text{div})$  Spaces. *SIAM J. Numer. Anal.*, 45(6):2483–2509, 2007. doi:10.1137/060660588.
- [38] Ralf Hiptmair and Andrea Toselli. Overlapping and Multilevel Schwarz Methods for Vector Valued Elliptic Problems in Three Dimensions. In Petter Bjørstad and Mitchell Luskin, editors, *Parallel Solut. Partial Differ. Equations*, pages 181–208, 2000. doi:10.1007/978-1-4612-1176-1\_8.
- [39] J Jones and B Lee. A Multigrid Method for Variable Coefficient Maxwell's Equations. *SIAM J. Sci. Comput.*, 27(5):1689–1708, 2006. doi:10.1137/040608283.
- [40] Jim E Jones and Panayot S Vassilevski. AMGe Based on Element Agglomeration. *SIAM J. Sci. Comput.*, 23(1):109–133, 2001. doi:10.1137/S1064827599361047.
- [41] D Z Kalchev, C S Lee, U Villa, Y Efendiev, and P S Vassilevski. Upscaling of Mixed Finite Element Discretization Problems by the Spectral AMGe Method. *SIAM J. Sci. Comput.*, 38(5):A2912–A2933, 2016. doi:10.1137/15M1036683.
- [42] Delyan Z Kalchev and Panayot Vassilevski. A Condensed Constrained Nonconforming Mortar-Based Approach for Preconditioning Finite Element Discretization Problems. *SIAM J. Sci. Comput.*, 42(5):A3136–A3156, 2020. doi:10.1137/19M1305690.
- [43] Delyan Z Kalchev and Panayot S Vassilevski. Auxiliary Space Preconditioning of Finite Element Equations Using a Nonconforming Interior Penalty Reformulation and Static Condensation. *SIAM J. Sci. Comput.*, 42(3):A1741–A1764, 2020. doi:10.1137/19M1286815.
- [44] Tzanio V Kolev, Joseph E Pasciak, and Panayot S Vassilevski.  $\mathbf{H}(\text{curl})$  auxiliary mesh preconditioning. *Numer. Linear Algebr. with Appl.*, 15(5):455–471, 2008. doi:10.1002/nla.534.
- [45] Tzanio V Kolev and Panayot S Vassilevski. Parallel Auxiliary Space AMG for  $H(\text{curl})$  Problems. *J. Comput. Math.*, 27(5):604–623, 2009. doi:10.4208/jcm.2009.27.5.013.
- [46] Tzanio V Kolev and Panayot S Vassilevski. Parallel Auxiliary Space AMG Solver for  $H(\text{div})$  Problems. *SIAM J. Sci. Comput.*, 34(6):A3079–A3098, 2012. doi:10.1137/110859361.
- [47] Max la Cour Christensen, Panayot S Vassilevski, and Umberto Villa. Nonlinear multigrid solvers exploiting AMGe coarse spaces with approximation properties. *J. Comput. Appl. Math.*, 340:691–708, 2018. doi:10.1016/j.cam.2017.10.029.
- [48] Max la Cour Christensen, Umberto Villa, Allan P Engsig-Karup, and Panayot S Vassilevski. Numerical Multilevel Upscaling for Incompressible Flow in Reservoir Simulation: An Element-Based Algebraic Multigrid (AMGe) Approach. *SIAM J. Sci. Comput.*, 39(1):B102–B137, 2017. doi:10.1137/140988991.
- [49] I V Lashuk and P S Vassilevski. Element agglomeration coarse Raviart-Thomas spaces with improved approximation properties. *Numer. Linear Algebr. with Appl.*, 19(2):414–426, 2012. doi:10.1002/nla.1819.
- [50] Ilya Lashuk and Panayot S Vassilevski. On some versions of the element agglomeration AMGe method. *Numer. Linear Algebr. with Appl.*, 15(7):595–620, 2008. doi:10.1002/nla.585.
- [51] Ilya V Lashuk and Panayot S Vassilevski. The Construction of the Coarse de Rham Complexes with Improved Approximation Properties. *Comput. Methods Appl. Math.*, 14(2):257–303, 2014. doi:10.1515/cmam-2014-0004.
- [52] Xiaoye S Li and James W Demmel. SuperLU\_DIST: A Scalable Distributed-Memory Sparse Direct Solver for Unsymmetric Linear Systems. *ACM Trans. Math. Softw.*, 29(2):110–140, jun 2003. doi:10.1145/779359.779361.
- [53] Ping Lin. A Sequential Regularization Method for Time-Dependent Incompressible Navier-Stokes Equations. *SIAM J. Numer. Anal.*, 34(3):1051–1071, 1997. doi:10.1137/S0036142994270521.
- [54] Peter Monk. *Finite Element Methods for Maxwell's Equations*. Numerical Mathematics and Scientific Computation. Clarendon Press, Oxford, 2003.
- [55] Duk-Soon Oh, Olof B Widlund, Stefano Zampini, and Clark R Dohrmann. BDDC Algorithms with deluxe scaling and adaptive selection of primal constraints for Raviart-Thomas vector fields. *Math. Comput.*, 87(310):659–692, 2018. doi:10.1090/mcom/3254.
- [56] Sarah Osborn, Panayot S Vassilevski, and Umberto Villa. A Multilevel, Hierarchical Sampling Technique for Spatially Correlated Random Fields. *SIAM J. Sci. Comput.*, 39(5):S543–S562, 2017. doi:10.1137/16M1082688.

- [57] Sarah Osborn, Patrick Zulian, Thomas Benson, Umberto Villa, Rolf Krause, and Panayot S Vassilevski. Scalable hierarchical PDE sampler for generating spatially correlated random fields using nonmatching meshes. *Numer. Linear Algebr. with Appl.*, 25(3):e2146, 2018. doi:10.1002/nla.2146.
- [58] J E Pasciak and J Zhao. Overlapping Schwarz methods in  $\mathbf{H}(\text{curl})$  on polyhedral domains. *J. Numer. Math.*, 10(3):221–234, 2002. doi:10.1515/JNMA.2002.221.
- [59] Joseph E Pasciak and Panayot S Vassilevski. Exact de Rham Sequences of Spaces Defined on Macro-Elements in Two and Three Spatial Dimensions. *SIAM J. Sci. Comput.*, 30(5):2427–2446, 2008. doi:10.1137/070698178.
- [60] A I Pehlivanov, G F Carey, and P S Vassilevski. Least-squares mixed finite element methods for non-selfadjoint elliptic problems: I. Error estimates. *Numer. Math.*, 72(4):501–522, 1996. doi:10.1007/s002110050179.
- [61] S Reitzinger and J Schöberl. An algebraic multigrid method for finite element discretizations with edge elements. *Numer. Linear Algebr. with Appl.*, 9(3):223–238, 2002. doi:10.1002/nla.271.
- [62] R N Rieben, D A White, B K Wallin, and J M Solberg. An arbitrary Lagrangian-Eulerian discretization of MHD on 3D unstructured grids. *J. Comput. Phys.*, 226(1):534–570, 2007. doi:10.1016/j.jcp.2007.04.031.
- [63] Panayot S. Vassilevski. Sparse matrix element topology with application to amg and preconditioning. *Numer. Lin. Alg. Appl.*, 9:429–444, 2002.
- [64] Panayot S Vassilevski. *Multilevel Block Factorization Preconditioners: Matrix-based Analysis and Algorithms for Solving Finite Element Equations*. Springer, New York, 2008. doi:10.1007/978-0-387-71564-3.
- [65] Panayot S Vassilevski. Coarse Spaces by Algebraic Multigrid: Multigrid Convergence and Upscaling Error Estimates. *Adv. Adapt. Data Anal.*, 03(01n02):229–249, 2011. doi:10.1142/S1793536911000830.
- [66] Panayot S Vassilevski and Umberto Villa. A Block-Diagonal Algebraic Multigrid Preconditioner for the Brinkman Problem. *SIAM J. Sci. Comput.*, 35(5):S3–S17, 2013. doi:10.1137/120882846.
- [67] Panayot S. Vassilevski and Junping Wang. Multilevel iterative methods for mixed finite element discretizations of elliptic problems. *Numer. Math.*, 63(1):503–520, 1992. doi:10.1007/BF01385872.
- [68] Jinchao Xu, Long Chen, and Ricardo H Nochetto. Optimal multilevel methods for  $H(\text{grad})$ ,  $H(\text{curl})$ , and  $H(\text{div})$  systems on graded and unstructured grids. In Ronald DeVore and Angela Kunoth, editors, *Multiscale, Nonlinear Adapt. Approx.*, pages 599–659, Berlin, Heidelberg, 2009. Springer. doi:10.1007/978-3-642-03413-8\_14.
- [69] Stefano Zampini. PCBDDC: A Class of Robust Dual-Primal Methods in PETSc. *SIAM J. Sci. Comput.*, 38(5):S282–S306, 2016. doi:10.1137/15M1025785.
- [70] Stefano Zampini and David E Keyes. On the Robustness and Prospects of Adaptive BDDC Methods for Finite Element Discretizations of Elliptic PDEs with High-Contrast Coefficients. In *Proc. Platf. Adv. Sci. Comput. Conf.*, New York, 2016. Association for Computing Machinery. doi:10.1145/2929908.2929919.

CENTER FOR APPLIED SCIENTIFIC COMPUTING, LAWRENCE LIVERMORE NATIONAL LABORATORY, P.O. BOX 808, L-561, LIVERMORE, CA 94551, USA.

Email address: kalchev1@llnl.gov

DEPARTMENT OF MATHEMATICS AND STATISTICS, PORTLAND STATE UNIVERSITY, PORTLAND, OR 97207, USA, AND CENTER FOR APPLIED SCIENTIFIC COMPUTING, LAWRENCE LIVERMORE NATIONAL LABORATORY, P.O. BOX 808, L-561, LIVERMORE, CA 94551, USA.

Email address: panayot@pdx.edu, vassilevski1@llnl.gov

ODEN INSTITUTE FOR COMPUTATIONAL ENGINEERING & SCIENCES, THE UNIVERSITY OF TEXAS AT AUSTIN, AUSTIN, TX 78712, USA.

Email address: uvilla@oden.utexas.edu