SIAM J. Sci. Comput.
Vol. 42, No. 2, pp. A1147–A1173

ALGEBRAIC MULTIGRID SCHEMES FOR HIGH-ORDER NODAL DISCONTINUOUS GALERKIN METHODS[*]

PAOLA F. ANTONIETTI[†] AND LAURA MELAS[†]

Abstract. We present algebraic multigrid (AMG) methods for the efficient solution of the linear system of equations stemming from high-order discontinuous Galerkin (DG) discretizations of second-order elliptic problems. For DG methods, standard multigrid approaches cannot be employed because of redundancy of the degrees of freedom associated to the same grid point. We present new aggregation procedures and test them in extensive two-dimensional numerical experiments to demonstrate that the proposed AMG method is uniformly convergent with respect to all of the discretization parameters, namely the mesh-size and the polynomial approximation degree.

Key words. algebraic multigrid, high-order, discontinuous Galerkin

AMS subject classifications. 65N30, 65N55

DOI. 10.1137/18M1204383

1. Introduction.

High-order discontinuous Galerkin (DG) methods are widely employed for the numerical solution of partial differential equations (PDEs) because of their flexibility in dealing with nonconforming grids and elementwise varying approximation orders; see, e.g., [44, 30, 23] for an overview on DG methods.

In this work we focus on multigrid methods and present a new algebraic multigrid (AMG) iterative scheme for the efficient solution of the linear system of equations stemming from high-order DG finite element approximations of second-order elliptic differential equations. Since the pioneering work of Gopalakrishnan and Kanschat [26], multigrid methods for DG finite element discretizations of PDEs have been intensively studied.

The first developments of geometric multigrid methods for low-order, i.e., linear, DG methods can be found in [26, 50, 17, 24, 16, 15, 14, 10, 9]. Multigrid techniques coupling geometric and p-multigrid approaches have also been studied; cf. [27, 28, 35]. Recently, new hp-multigrid schemes for high-order DG discretizations have been proposed and analyzed; cf. [6, 3, 5, 4]. Algebraic multigrid techniques for matrices stemming from low-order DG finite element discretizations of elliptic equations can be found in [43, 11, 46]. The first scalable AMG method for high-order DG discretizations of the Poisson operator was developed by Olson and Schroder [38]. It assumes the access to mesh points in order to perform the first step of coarsening and therefore employs geometric information. To the best of our knowledge, purely AMG methods for high-order DG discretizations have not been addressed so far. Indeed, the work by Prill, Lukáčová-Medviďová, and Hartmann [43] requires knowledge of the grid in order to build all of the aggregates, Olson and Schroder [38] assume access to the mesh information for the first coarsening step, and the method of Bastian, Blatt, and Scheichl [11] requires that the natural embedding operator be provided. More

[*] Submitted to the journal's Methods and Algorithms for Scientific Computing section August 1, 2018; accepted for publication (in revised form) January 13, 2020; published electronically April 13, 2020.
https://doi.org/10.1137/18M1204383

Funding: This work was supported by SIR Starting grant RBSI14VT0S funded by the Italian Ministry of Education, Universities and Research (MIUR).

[†] MOX, Dipartimento di Matematica, Politecnico di Milano, Milan, 20133, Italy (paola.antonietti@polimi.it, laura.melas@polimi.it).

precisely, the AMG method proposed by Olson and Schroder [38] is quasi–purely AMG because it employs the geometric assumptions only for the first aggregation step.

In this paper we present a new AMG method for the efficient solution of the linear systems of equations stemming from high-order DG approximations of second-order elliptic problems. We modify the first step of coarsening of the AMG method of Olson and Schroder [38] within an algebraic framework proposing a block-aggregation scheme applied to the finest level. For the coarse levels, we employ the classical aggregation of Vaněk, Mandel, and Brezina [51] following the guideline given in [38]. With these steps our algorithm is fully algebraic because it employs only the entries of the matrix. We demonstrate that for the proposed AMG iterative scheme, convergence is achieved independently of both of the discretization parameters, namely the mesh-size and the polynomial approximation degree, making the method well suited for both low- and high-order DG approximations.

The remainder of the paper is organized as follows. In section 2 we introduce the model problem and its DG discretization. In section 3 we propose our AMG method based on smoothed aggregation and extend it to high-order discontinuous discretizations. In section 4 we present extensive numerical experiments to investigate the efficiency and robustness of our method. In section 5 we give a summary of the achieved results and draw some conclusions.

2. Model problem and its discontinuous Galerkin discretization.

In this section we present the model problem and its DG discretization. Throughout the paper, we use the standard notation for Sobolev spaces; cf. [1]. Let $\Omega \subset \mathbb{R}^2$ be a bounded polygonal domain, and let $\mathbf{n}$ be the unit outward normal vector to the boundary $\partial\Omega$. For a given function $f \in L^2(\Omega)$ and a given $g \in H^{\frac{1}{2}}(\partial\Omega)$, we consider the following weak formulation of the Poisson problem subject to essential boundary conditions: find $u \in V = \{v \in H^1(\Omega)\colon\ u = g \text{ on } \partial\Omega\}$ such that

$$\int_\Omega \nabla u \cdot \nabla v \, d\Omega = \int_\Omega f v \, d\Omega \quad \forall v \in H^1_0(\Omega). \tag{2.1}$$

Now we describe the numerical solution of (2.1) based on employing the discontinuous finite element method. We begin by constructing a conforming mesh $\mathcal{T}_h$ of the domain $\Omega \subset \mathbb{R}^2$ made of nonoverlapping shape-regular triangles/quads of diameter h_K, and we set $h = \max_K h_K$. We assume that each K is the affine map of a fixed master element $\widehat{K}$, i.e., $K = F_K(\widehat{K})$, where $\widehat{K}$ is either the unit reference simplex $\widehat{T} = \{(x, y) : x, y \geq 0,\ x + y \leq 1\}$ or the reference square $\widehat{Q} = (-1, 1)^2$. Let $\mathcal{E}_I$ be the set of interior edges of the mesh $\mathcal{T}_h$, let $\mathcal{E}_B$ be the set of boundary edges, and let $\mathcal{E}=\mathcal{E}_I \cup \mathcal{E}_B$ be the set of all edges. Let $e \in \mathcal{E}_I$ be shared by two neighboring elements $K^\pm$. For (regular enough) scalar and vector-valued functions v and $\boldsymbol{\tau}$, respectively, we define the averages and jumps as

$$\{\!\{v\}\!\} = \frac{1}{2}(v^+ + v^-), \qquad [\![v]\!] = v^+\mathbf{n}^+ + v^-\mathbf{n}^-,$$

$$\{\!\{\boldsymbol{\tau}\}\!\} = \frac{1}{2}(\boldsymbol{\tau}^+ + \boldsymbol{\tau}^-), \qquad [\![\boldsymbol{\tau}]\!] = \boldsymbol{\tau}^+ \cdot \mathbf{n}^+ + \boldsymbol{\tau}^- \cdot \mathbf{n}^-,$$

where $\mathbf{n}^\pm$ is the unit normal vector to e pointing outward to $K^\pm$, and $v^\pm$ and $\boldsymbol{\tau}^\pm$ are the traces of the functions v and $\boldsymbol{\tau}$ on $K^\pm$; cf. [8]. If $e \in \mathcal{E}_B$ belongs to the boundary $\partial\Omega$, we extend these definitions as follows: $\{\!\{v\}\!\} = v$, $[\![v]\!] = v\mathbf{n}$, $\{\!\{\boldsymbol{\tau}\}\!\} = \boldsymbol{\tau}$, and $[\![\boldsymbol{\tau}]\!] = \boldsymbol{\tau} \cdot \mathbf{n}$ (cf. [8]). Let V_{hp} be a family of finite dimensional spaces defined as

$$V_{hp} = \{v \in L^2(\Omega) : v \circ F_K \in \mathbb{M}_p(\widehat{K}) \quad \forall K \in \mathcal{T}_h\},$$

where $\mathbb{M}_p(\widehat{K})$ is either the space $\mathbb{P}_p(\cdot)$ of polynomials of degree less than or equal to $p \geq 1$ on $\widehat{K}$ if $\widehat{K} \equiv \widehat{T}$ is the reference triangle, or the space $\mathbb{Q}_p(\cdot)$ of all tensor-product polynomials on $\widehat{K}$ of degree p in each coordinate direction if $\widehat{K} \equiv \widehat{Q}$ is the reference square. The space V_{hp} is equipped with the norm

$$\|v\|_{DG}^2 = \sum_{K \in \mathcal{T}_h} \|\nabla v\|_{L^2(K)}^2 + \sum_{e \in \mathcal{E}} \|\sqrt{\gamma_e}[\![v]\!]\|_{L^2(e)}^2,$$

where, for a given penalty parameter $\sigma^e > 0$, γ_e is defined edgewise as $\gamma_e = \sigma^e p^2/|e|$, with $|e|$ being the length of the edge e.

Next we define the bilinear form $\mathcal{A} : V_{hp} \times V_{hp} \to \mathbb{R}$ as

$$\begin{aligned}\mathcal{A}(u,v) = &\sum_{K \in \mathcal{T}_h} \int_K \nabla u \cdot \nabla v \, d\Omega - \sum_{e \in \mathcal{E}} \int_e \{\!\{\nabla u\}\!\} \cdot [\![v]\!] \, d\gamma \\ &- \sum_{e \in \mathcal{E}} \int_e \{\!\{\nabla v\}\!\} \cdot [\![u]\!] \, d\gamma + \sum_{e \in \mathcal{E}} \gamma_e \int_e [\![u]\!] \cdot [\![v]\!] \, d\gamma\end{aligned}$$

and the functional $F : V_{hp} \to \mathbb{R}$ as

$$F(v) = \int_\Omega f v \, d\Omega - \sum_{e \in \mathcal{E}_B} \int_e \nabla v \cdot \mathbf{n}_e \, g \, d\gamma + \sum_{e \in \mathcal{E}_B} \gamma_e \int_e v g \, d\gamma.$$

The DG discrete problem reads as follows: find $u_h \in V_{hp}$ such that

$$\mathcal{A}(u_h, v_h) = F(v_h) \quad \forall v_h \in V_{hp}, \tag{2.2}$$

which is known as the symmetric interior penalty (SIP) method [54, 7]. The following result ensures the well-posedness of (2.2); cf. [54, 7, 8] and see, e.g., [42, 31, 47, 25] for *hp*-version error estimates.

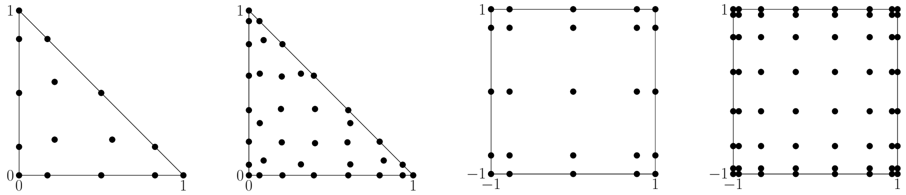
Proposition 2.1. *If $\sigma^e > \sigma_{min}$, the solution of* (2.2) *exists and is unique.*

Let $\{\phi_j\}_{j=1}^{N_h}$ be a basis for the discrete space V_{hp}, i.e., $V_{hp} = span\{\phi_j\}_{j=1}^{N_h}$; then (2.2) is equivalent to the following linear system of equations:

$$A\mathbf{u} = \mathbf{f}, \tag{2.3}$$

where $\mathbf{u} = [u_1, \ldots, u_{N_h}]^T \in \mathbb{R}^{N_h}$ is the vector containing the unknown coefficients of the expansion of the discrete solution u_h in the chosen basis. The stiffness matrix A in (2.3) is symmetric and positive definite, provided that σ^e is large enough.

Next we describe the choice of the shape functions employed to span the discontinuous finite element space. We point out that since in the DG framework the shape functions are supported on a single mesh element, we trivially fit the smoothed aggregation AMG framework of Vaněk, Mandel, and Brezina [51], which requires that the support of each basis function be bounded. To span the discrete space V_{hp} we require that a nodal basis be employed; i.e., $v_h \in V_{hp}$ is characterized by the values it takes at the points $P_i = (x_i, y_i)$, with $i = 1, \ldots, N_h$, and consequently the shape functions associated to the finite element space V_{hp} are defined as the Lagrangian functions associated to these interpolation nodes with support on a single element.



(a) $p = 4$, $\widehat{K} = \widehat{T}$ (b) $p = 7$, $\widehat{K} = \widehat{T}$ (c) $p = 4$, $\widehat{K} = \widehat{Q}$ (d) $p = 7$, $\widehat{K} = \widehat{Q}$

FIG. 1. *Fekete* (a, b) *and GLL* (c, d) *points on the reference triangle/square for different choices of* p.

Remark 2.2. We point out that our AMG method is developed under the assumption that a nodal basis associated to suitable (stable) interpolation points is employed to span the discrete space. The extension to modal basis, e.g., tensor product of Legendre polynomials, requires a completely different algorithm to identify connections and build the interpolation operator; such an extension is under development and will be the subject of future research.

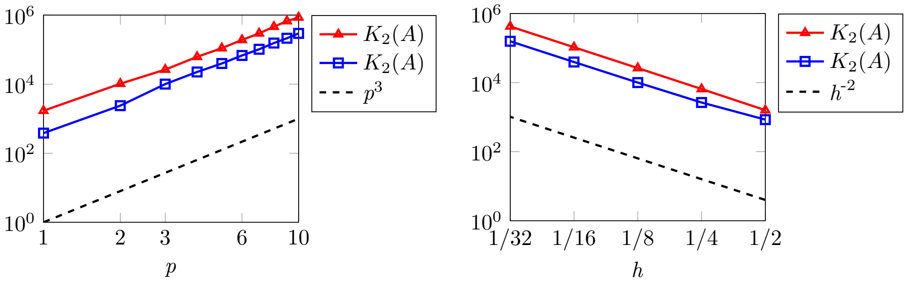
To specify the interpolation points, on the reference triangle $\widehat{T}$ we define therein the Fekete points [48, 19]; on the reference square $\widehat{Q}$ we consider the Gauss–Legendre–Lobatto (GLL) points [22]. Then, for any $K \in \mathcal{T}_h$, with K being a shape-regular triangle or quad, those points are mapped through the linear map $F_K : \widehat{K} \to K$, $K \in \mathcal{T}_h$, $\widehat{K} = \widehat{T}, \widehat{Q}$. We point out that the choice of Fekete and GLL interpolation points is dictated by the fact that we are interested in high-order approximations, where it is known that equidistributed interpolation points lead to numerical instabilities. In Figure 1 we show the Fekete/GLL points on the reference triangle/square for $p = 4, 7$.

Remark 2.3. We point out that other choices of interpolation points for the reference triangle could be employed. Among them we mention the warped interpolation points of Warburton and Hesthaven; cf. [29, 53]. We tested our AMG method also employing the warped interpolation points [29, 53], and from the numerical results it seems to be robust for this choice; for brevity these results are omitted.

Remark 2.4 (condition number of A). We are interested in considering the condition number $K_2(A)$ of the matrix A stemming from the DG approximation of problem (2.2). We observe that, with the above choice of interpolation points, the condition number $K_2(A)$ of A, defined as the ratio of its extreme eigenvalues, seems to behave as $K_2(A) = \mathcal{O}(p^3/h^2)$ if the underlying grid is made of triangles and as $K_2(A) = \mathcal{O}(p^3/h^2)$ if the underlying grid is made of quads, as shown in Figure 2.

These results seem to show that, at least on triangular meshes in two dimensions, a set of Lagrangian basis functions associated with Fekete points seems to lead to an improvement of the condition number as a function of p. Indeed, in [2] it is proved that, whenever a modal basis based on Legendre polynomials is employed, the condition number of the resulting stiffness matrix behaves as $K_2(A) = \mathcal{O}(p^4/h^2)$. On the other hand, the results obtained for quadrilateral meshes in two dimensions are in agreement with the literature on spectral-element methods; cf. [12].

We point out that in the conforming setting for triangular grids, i.e., continuous triangular spectral-element methods, Pasquetti and Rapetti [40, 41] observed that the condition number is of order p^4/h^2 whenever Fekete points are employed, whereas Toselli and Widlund [49] and Bernardi and Maday [12] proved a behavior of order



(a)

(b)

FIG. 2. *Condition number $K_2(A)$ as a function of p (left) and h (right), unstructured triangular grids* ─▲─*, and quadrilateral meshes* ─□─*.* $\sigma^e = 10 \; \forall e \in \mathcal{E}$.

p^3/h^2 if the interpolation points are obtained based on mapping, through the Dubiner map, the classical Gauss–Legendre points defined on the reference square onto the reference triangle.

The issue of proving sharp bounds on the condition number of A, stemming from DG methods when the discrete space is spanned based on employing Lagrangian functions associated to Fekete points, is under investigation and will be the subject of future research.

3. Smoothed-block aggregation algebraic multigrid.

In this section, we introduce the main ingredients for the AMG algorithm. We assume to have a sequence of successively coarser matrices $A_k \in \mathbb{R}^{N_k \times N_k}$, $k = 1, \dots, K$, with the convention that $A_1 = A$, $N_k > N_{k+1}$, and

$$A_{k+1} = I_k^{k+1} A_k I_{k+1}^k \quad k = 1, \dots, K-1.$$

Here, $I_k^{k+1} : \mathbb{R}^{N_k} \to \mathbb{R}^{N_{k+1}}$ is a linear operator to be properly defined and $I_{k+1}^k = (I_k^{k+1})^T$; see, for example, [34, 21, 56].

Given a suitable smoother (e.g., damped Jacobi, symmetrized Gauss–Seidel), one iteration of the algebraic μ-cycle scheme, $\mu = 1, 2$, is shown in Algorithm 3.1; cf. also [20, 56]. More precisely, given $\mathbf{u}_k^l$, $\mathbf{u}_k^{l+1} = \text{AMG-}\mu\text{Cycle}(\dots, \mathbf{u}_k^l, \dots)$ returns the $(l+1)$th iteration to solve $A_k \mathbf{u}_k = \mathbf{f}_k$. If we have a μ-cycle scheme with $\mu = 1$, we refer to it as a V-cycle, whereas for $\mu = 2$ we call the method a W-cycle. In particular, we denote by V(ν_1,ν_2)-cycle and W(ν_1,ν_2)-cycle the two methods above with ν_1 presmoothing and ν_2 postsmoothing iterations [21]. Algebraic multigrid can be used as a stand-alone solver or as a preconditioner to accelerate the convergence of Krylov-based iterative schemes, such as the conjugate gradient method.

Our method is based on smoothed aggregation [51] and extends the results of Olson and Schroder [38] to high-order DG methods. The main challenge in the DG setting is the redundancy of the degrees of freedom (DOFs) associated to the same grid points. A similar issue occurs, for example, for systems of PDEs, where there exists multiple unknowns at the same grid point. This difficulty can be solved with strategies known as “point” or “block” approaches [45, 51]; these techniques are based on local aggregation of variables associated to the same grid point. Here, we extend this idea to deal with the multiple unknowns associated to the same grid point, which are typical of DG methods. This extension is based on the use of local aggregation

Algorithm 3.1. One iteration of AMG-μCycle to solve $A_k\mathbf{u}_k = \mathbf{f}_k$.

```
function $\mathbf{u}_k^{l+1}$ = AMG-$\mu$Cycle($\nu_1$,$\nu_2$,$A_k$,$\mathbf{u}_k^l$,$\mathbf{f}_k$,$I_k^{k+1}$,$I_{k+1}^k$)
    if $k = K$ then
        $\mathbf{u}_K = A_K^{-1}\mathbf{f}_K$                                        ▷ Coarsest level
        return $\mathbf{u}_K$
    else
        Relax $\nu_1$ times on $A_k\mathbf{u}_k = \mathbf{f}_k$ with initial guess $\mathbf{u}_k^0$            ▷ Presmoothing
        $\mathbf{f}_{k+1} = I_k^{k+1}(\mathbf{f}_k - A_k\mathbf{u}_k^{\nu_1})$                     ▷ Restriction of the residual
        $\mathbf{e}_{k+1}^0 = \mathbf{0}_{k+1}$
        for $\lambda = 1 : \mu$ do
            $\mathbf{e}_{k+1}^{\lambda}$ = AMG-$\mu$Cycle($\nu_1$,$\nu_2$,$A_{k+1}$,$\mathbf{e}_{k+1}^{\lambda-1}$,$\mathbf{f}_{k+1}$,$I_{k+1}^{k+2}$,$I_{k+2}^{k+1}$)
        end for
        $\mathbf{u}_k^{\nu_1+1} = \mathbf{u}_k^{\nu_1} + I_{k+1}^k\mathbf{e}_{k+1}^{\mu}$                 ▷ Interpolation and correction
        Relax $\nu_2$ times on $A_k\mathbf{u}_k = \mathbf{f}_k$ with initial guess $\mathbf{u}_k^{\nu_1+1}$      ▷ Postsmoothing
        return $\mathbf{u}_k^{\nu_1+\nu_2+1}$
    end if
end function
```

for the first-level coarsening; then for all other levels, different aggregation schemes can be used.

In the next sections, we detail the main steps at the basis of our AMG solvers.

3.1. Algebraic block-aggregation algorithm.

In the literature, we can find different algorithms of the aggregation techniques for AMG methods applied to problems with DG discretizations that exploit the idea of local aggregation, but all are based on the knowledge of geometric or topological information; cf. [32, 43, 38].

Here we propose a new purely algebraic block-aggregation coarsening strategy based on block-aggregation. Our approach differs from the ones proposed in the works by Prill, Lukáčová-Medvid̆ová, and Hartmann [43] and Olson and Schroder [38] because it selects coinciding nodes typical of DG discretizations by employing only the entries of the stiffness matrix, whereas in [43, 38] this step is taken based on employing knowledge of the mesh information. On the other hand, the classical smoothed aggregation proposed by Vaněk, Mandel, and Brezina [51] is not able to deal with the redundancy of DOFs associated to the same grid point; therefore we modify their aggregation technique in order to take into account this characteristic of DG methods, and we employ our block-aggregation approach for the first level of coarsening. The algorithm that we present is built through analysis of the matrix entries associated with each DOF, as described in the following.

Given the matrix $A_k \in \mathbb{R}^{N_k \times N_k}$, its entries a_{ij}, $i, j = 1, \ldots, N_k$, and its set of unknowns $\mathcal{V} = \{1, \ldots, N_k\}$, namely the DOFs of the problem, we split the set of points into a disjoint covering such that $\mathcal{V} = \bigcup_{j=1}^{N_{k+1}} \mathcal{V}_j$, $N_{k+1} \leq N_k$, and $\mathcal{V}_l \cap \mathcal{V}_j = \emptyset$ for $l \neq j$. In particular, the algorithm aims at providing suitable disjoint sets such that each contains the multiple variables associated to the same physical grid point; cf. Algorithm 3.2.

Algorithm 3.2 is made up of three steps: startup singleton or aggregation, enlargement of the decomposition sets, and cancellation of the empty sets. First, for each $i \in \mathcal{V}$, the function find_strongest_connection(i) chooses the node $I \in \mathcal{V}$ to which the unknown i has the strongest connection; cf. subsection 3.3 below. If the strongest connection between i and I is negative, i.e., $a_{iI} < 0$, then the nodes i and I

Algorithm 3.2. Algebraic block-aggregation algorithm.

```
$h = 0$
for all $i \in \mathcal{V}$ do
    $I$=find_strongest_connection($i$)
    if $a_{iI} \geq 0$ then
        if $\forall h : \mathcal{V}_h \cap \{i\} = \emptyset$ then                              ▷ Startup singleton
            $h = h + 1$, $\mathcal{V}_h = \{i\}$
        end if
    else
        if $\forall h : \mathcal{V}_h \cap \{i, I\} = \emptyset$ then                         ▷ Startup aggregation
            $h = h + 1$, $\mathcal{V}_h = \{i, I\}$
        else                                             ▷ Enlarging the decomposition sets
            if $\exists \tilde{h} : \mathcal{V}_{\tilde{h}} \cap \{i\} \neq \emptyset$ & $\forall h : \mathcal{V}_h \cap \{I\} = \emptyset$ then
                $\mathcal{V}_{\tilde{h}} = \mathcal{V}_{\tilde{h}} \cup \{I\}$
            else if $\exists \tilde{h} : \mathcal{V}_{\tilde{h}} \cap \{I\} \neq \emptyset$ & $\forall h : \mathcal{V}_h \cap \{i\} = \emptyset$ then
                $\mathcal{V}_{\tilde{h}} = \mathcal{V}_{\tilde{h}} \cup \{i\}$
            else if $\exists \tilde{h}_1 : \mathcal{V}_{\tilde{h}_1} \cap \{i\} \neq \emptyset$ & $\exists \tilde{h}_2 : \mathcal{V}_{\tilde{h}_2} \cap \{I\} \neq \emptyset$ & $\tilde{h}_1 \neq \tilde{h}_2$ then
                $\mathcal{V}_{\tilde{h}_1} = \mathcal{V}_{\tilde{h}_1} \cup \mathcal{V}_{\tilde{h}_2}$, $\mathcal{V}_{\tilde{h}_2} = \emptyset$
            end if
        end if
    end if
end for
$j = 0$                                                    ▷ Deleting the empty sets
for all $h$ do
    if $\mathcal{V}_h \neq \emptyset$ then
        $j = j + 1$, $\mathcal{V}_j = \mathcal{V}_h$
    end if
end for
```

are grouped together (startup aggregation); otherwise, the node i is processed alone (startup singleton). Once the startup phase is concluded, the algorithm proceeds with the enlargement of the decomposition sets, which is based on joining sets with at least one node in common. Finally, empty sets are deleted from the disjoint covering. Algorithm 3.2 is based on the function find_strongest_connection that is detailed in the following. We introduce a symmetric, positive strength function $s(i, j)$ that quantifies the "amount" of connection between nodes i and j. Hereafter, we focus on strength functions $s(i, j) \geq 1$ such that small values of $s(i, j)$ indicate "strong" connections, whereas large values of $s(i, j)$ indicate "weak" connections. With this framework, we assume that the strongest connected points to i are given by

$$\mathcal{S}_i = \{j : s(i, j) \leq \theta\}, \tag{3.1}$$

where $\theta \geq 1$ is a given user-defined threshold. Next, by fixing θ, the function find_strongest_connection(i) returns a point in the set $\mathcal{S}_i$.

3.2. Interpolation operator $\boldsymbol{I_k^{k+1}} : \mathbb{R}^{\boldsymbol{N_k}} \to \mathbb{R}^{\boldsymbol{N_{k+1}}}$. Given the disjoint partition $\mathcal{V} = \bigcup_{j=1}^{N_{k+1}} \mathcal{V}_j$, $N_{k+1} \leq N_k$ given by Algorithm 3.2, it is natural to construct the interpolation operator in a manner similar to that for the smoothed aggregation AMG by Vaněk, Mandel, and Brezina [51].

In particular, we modify the scheme of Vaněk, Mandel, and Brezina [51] in an energy-minimization framework as follows. We define algebraically smooth error modes to be grid functions with a small Rayleigh quotient (cf. [36]) and therefore equivalent to the near null-space or low energy modes. Hence, a tentative interpolation operator is constructed in such a way that it preserves the near null-space mode vector $\mathbf{w}_k \in \mathbb{R}^{N_k}$; cf. [33, 52, 55, 18, 13, 39]. More precisely, the vector $\mathbf{w}_k$ is the numerical solution of $A_k\mathbf{w}_k = \mathbf{0}_k$ obtained after η smoothing steps with initial guess $\mathbf{w}_k^0 = \mathbf{1}_k$.

We first set

$$[\tilde{I}_{k+1}^k]_{ij} = \begin{cases} w_i, & i \in \mathcal{V}_j, \\ 0 & \text{otherwise}, \end{cases} \qquad i = 1, \dots, N_k,\ j = 1, \dots, N_{k+1},$$

where w_i is the ith component of vector $\mathbf{w}_k$, and apply the Gram–Schmidt orthonormalization algorithm to each column of $\tilde{I}_{k+1}^k$ to improve conditioning. Then, the interpolation operator is defined by a classical damped Jacobi smoothing step, i.e.,

$$I_{k+1}^k = (I_k - \omega D_k^{-1}A_k)\tilde{I}_{k+1}^k,$$

where $\omega = 2/3$, D_k is the diagonal of A_k, and I_k is the identity matrix.

Remark 3.1. Other approaches can be employed to construct the interpolation matrix. For example, since our problem is symmetric and positive definite, we can employ the Krylov-based framework (cf. [39, 38]), where we substitute the simple damped Jacobi smoothing step with a fixed number of iterations of the conjugate gradient method.

3.3. Evolution measure. In this section we recall the evolution measure proposed by Olson, Shroder, and Tuminaro [37], which combines the local knowledge of both the algebraic smooth error and the behavior of the interpolation. In the DG framework this measure is necessary to define the strongest connections in Algorithm 3.2 and in the aggregation scheme; cf. [51, 38]. This choice is motivated by the fact that most of the strength measures proposed in the literature so far are not well suited to taking into account the connections between the DOFs typical of DG methods, as outlined in [38].

In order to take into account the algebraic smooth error, we define $\mathbf{z}_k \in \mathbb{R}^{N_k}$ as

$$\mathbf{z}_k = (I_k - \omega_k D_k^{-1}A_k)^m \mathbf{e}_k(i),$$

where $\mathbf{e}_k(i) \in \mathbb{R}^{N_k}$ is the unit vector centered at $i \in \mathcal{V}$, $\omega_k = 1/\rho(D_k^{-1}A_k)$, and m is an integer that has to be properly chosen.

Then we have to consider the local knowledge of the interpolation. Assume that the interpolation operator is defined as in subsection 3.2. Given a point $i \in \mathcal{V}$ we would like to be able to measure the ability of each column of the tentative interpolation operator $\tilde{I}_{k+1}^k$ to interpolate $\mathbf{z}_k$ for all points j in the algebraic neighborhood of i, i.e., $j \in N_i$, where $N_i = \{j : a_{ij} \neq 0\}$. Therefore this quantity is measured only for points $j \in N_i$, in particular with exact interpolation enforced at point i.

We define the evolution measure as

$$e(i,j) = \left| 1 - \frac{w_j z_i}{w_i z_j} \right| \qquad i, j = 1, \dots, N_k,$$

where w_j and z_j are the jth components of vectors $\mathbf{w}_k$, defined in subsection 3.2, and $\mathbf{z}_k$, respectively. Since our problem is symmetric, we define the symmetrized version of the evolution measure as

$$e_S(i,j) = e(i,j) + e(j,i).$$

Finally, the symmetric evolution strength function is defined as

$$s(i,j) = \frac{e_S(i,j)}{\min_{k\neq i} e_S(i,k)}. \tag{3.2}$$

The symmetric evolution measure defined above is employed to identify the connections in our algorithm; cf. section 4 below. Our algorithm makes use of the following two steps that are the block-aggregation and classical aggregation, respectively. On the finest level, we employ Algorithm 3.2 with the choice of evolution strength function (3.2) and $\theta = 1$ in (3.1). On the coarser levels, we use the aggregation scheme of [51], still with evolution strength function (3.2) but with $\theta \in [2,4]$; cf. (3.1).

This choice is guided by the following properties that hold in the DG framework: we employ our block-aggregation for the finest level because it is suited to aggregate the multiple DOFs associated to each grid point; on the other hand, we use the classical aggregation for the coarser levels because it builds larger agglomerates, and this is better in order to have fewer unknowns associated to these levels.

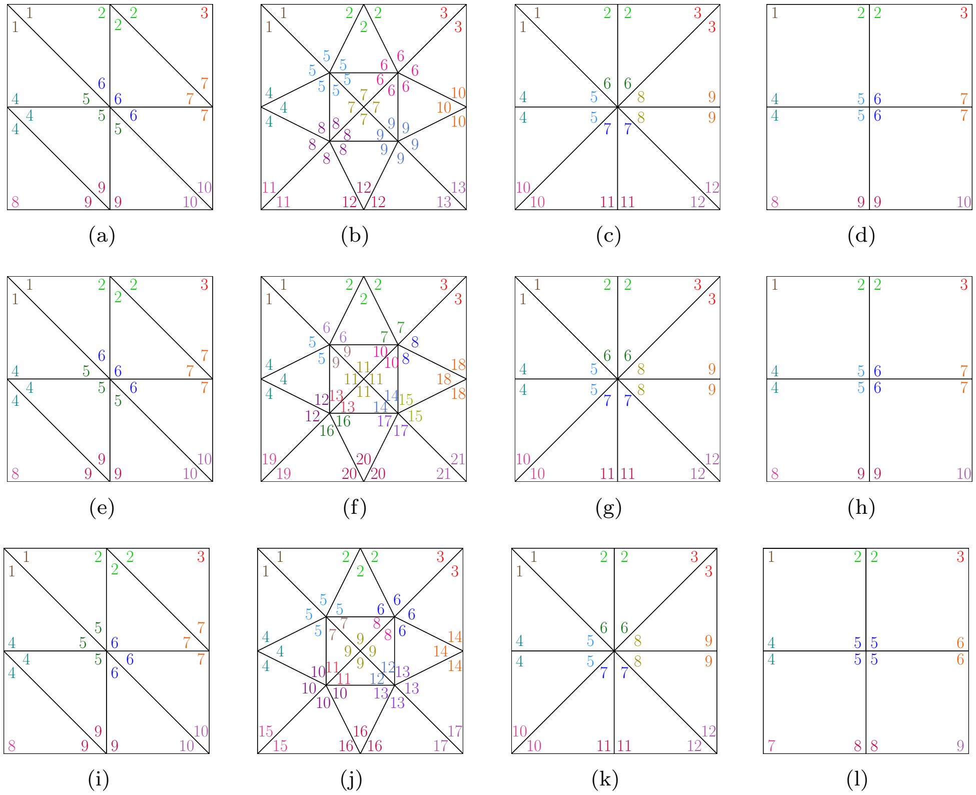
In Figure 3 we show some examples of block-aggregation for matrices stemming from linear DG discretizations on structured/unstructured triangular and Cartesian meshes and with penalty parameters $\sigma^e = 5, 10, 20, 30$; cf. section 2. For $\sigma^e = 30$, we obtain the same aggregations as those for $\sigma^e = 20$. For simplicity, these results are omitted. Moreover, when we compute the evolution measure, we fix $\mathbf{w}_k = \mathbf{1}_k$. Each aggregate set is represented by a distinct number.

We notice that, as expected, our block-aggregation algorithm seems to be fairly insensitive to the value of the penalty parameter.

4. Numerical experiments.

In this section we test the robustness and the efficiency of our AMG method in solving the linear system of equations stemming from high-order discontinuous finite element discretizations of problem (2.1). We consider a sequence of structured/unstructured triangular and Cartesian meshes with granularity $h = 2^{-l}$, $l = 1, \ldots, 5$, and let the polynomial approximation degree p vary from 1 to 10. Moreover we also take into consideration $l = 6, 7$ when $p = 1$. For each h and p, we obtain a linear system of equations that we solve with our smoothed block-aggregation AMG; cf. section 3. At the first step of coarsening, i.e., $k = 1$, we use the block-aggregation algorithm (cf. Algorithm 3.2) with the strongest evolution connection defined in (3.2) and with $\theta = 1$; cf. (3.1). For the coarser levels, i.e., $k = 2, \ldots, K$, we use the classical aggregation of Vaněk, Mandel, and Brezina [51], still with the evolution strength function in (3.2) and $\theta = 2$ in (3.1); cf. also [38]. In our numerical experiments, we compute the evolution measure with $m = 4$ and $m = 2$ when we solve the problem on triangular and quadrilateral grids, respectively; cf. subsection 3.3 and [37, 38].

For any multigrid level k, the associated interpolation operator is the one proposed in subsection 3.2 with $\eta = 0$ and $\eta = p$ smoothing iterations of classical Gauss–Seidel, where p is the polynomial degree, when the problem is discretized on triangular and quadrilateral meshes, respectively. For the smoothing interpolation, we compare both the Jacobi iteration and the Krylov-based framework with two and four iterations


(a) (b) (c) (d)

(e) (f) (g) (h)

(i) (j) (k) (l)

FIG. 3. *Examples of block-aggregation for different meshes with* $p = 1$, $h \cong 1/2$, *and* $\sigma^e = 5$ *(top),* $\sigma^e = 10$ *(middle), and* $\sigma^e = 20$ *(bottom).*

of the CG method whenever triangular and quadrilateral grids are employed, respectively; cf. Remark 3.1. We remark that for moderate values of p, one iteration of CG is enough. In our numerical results, we denote by J-smoother/CG-smoother the smoothed block-aggregation AMG with Jacobi/CG smoothing interpolation steps, respectively; cf. Remark 3.1.

In our numerical tests, we test the W(ν_1,ν_2)-cycle with the classical Gauss–Seidel relaxation as a stand-alone AMG solver. Moreover, we also consider a preconditioned conjugate gradient (PCG) method with a preconditioner given by the W(ν_1,ν_2) iteration with a symmetric Gauss–Seidel smoother. We refer to the PCG with W(ν_1,ν_2)-cycle preconditioner as PCG W(ν_1,ν_2)-cycle. We point out that in each step of the proposed AMG, we employ Jacobi, Gauss–Seidel, and CG as pointwise smoothers.

Let N be the iteration counts needed to reduce the initial relative residual below a tolerance $tol = 10^{-8}$; we compute the convergence factor ρ defined by

$$\rho = \exp\left(\frac{1}{N}\log\frac{||\mathbf{r}_N||}{||\mathbf{r}_0||}\right),$$

where $\mathbf{r}_N$ and $\mathbf{r}_0$ are the final and initial residuals, respectively.

In subsections 4.1 and 4.2 we report the results when employing the W-cycle algorithm as an iterative scheme and as a preconditioner for the CG method, respectively. We first present results obtained with the W-cycle iteration; in subsection 4.3

we report a comparison between the V- and W-cycles in terms of convergence factor and computational cost. All of the proposed solver components are summarized in Figure 4.

Remark 4.1. In the case of triangular grids, we can employ different sets of interpolation points as DOFs to span the discrete space, such as Fekete [48, 19], Warburton [53], and Hesthaven [29] points. Here, for the sake of brevity, we present the results obtained based on employing Fekete nodes. We tested our AMG algorithm also by using the Warburton [53] and Hesthaven [29] nodes, and our schemes displayed equivalent performances; for the sake of brevity, these results are omitted.

Remark 4.2. Algorithm 3.2 has been tested based on employing different ordering of the DOFs. For example, on triangular meshes we order DOFs in the following two ways:

1. First, we number the DOFs associated with the vertices, then the DOFs associated with the internal edges, and finally the interior DOFs.
2. First, we order the DOFs associated with the edges counterclockwise and then the interior DOFs.

On quadrilateral grids, we order DOFs based on employing the lexicographic order. For all of the cases, our AMG algorithms exhibit the same performance and get consistent results, suggesting that Algorithm 3.2 seems to be robust with respect to DOFs ordering. For brevity this comparison is omitted.

4.1. W-cycle algorithm as iterative scheme. In this section we present some numerical results to investigate the performance of the W-cycle AMG algorithm as an iterative scheme. In Figures 5 and 6 we compare the W-cycle AMG with J- and CG-smoothers in terms of p- and h-scalability, respectively, when employing $\nu_1 = \nu_2 = \nu = 1, 3$ pre- and postsmoothing iterations. We note that if we employ the AMG method with the CG- rather than the J-smoother, we obtain better results in terms of both convergence factor and scalability.

We remark that in our tests, the W-cycle AMG with the J-smoother does not converge for $p = 4, \dots, 10$ and $p = 10$ in the case of Cartesian and triangular grids, respectively; therefore in Figure 6 these results are omitted.

Concerning the h- and p-scalability, we observe that the J-smoother AMG method seems to be scalable only if the number of smoothing steps is sufficiently large. Whenever we employ the J-smoother AMG with smaller values of smoothing steps (e.g., $\nu = 1, 2, 3$), we still observe hp-weak-scalability for $p = 1, \dots, p_{max}$, with $p_{max} = 7$ on triangular and $p_{max} = 3$ on quadrilateral meshes. On the other hand, for both triangular and quadrilateral grids, the CG-smoother AMG method is hp-quasi-scalable for $\nu = 1, 2$ and hp-scalable for $\nu = 3$ for all considered h and p. The difference between h- and p-scalability when varying $\nu = 1, 2, 3$ for the AMG method with the CG-smoother is small, so it is worth considering the method with $\nu = 1$ because it has lower computational cost. For brevity, in the following we focus only on the results obtained on triangular meshes.

In Table 1 we report the computed convergence factors for the J- and CG-smoother AMG methods on both structured and unstructured triangular grids when varying the number of smoothing iterations $\nu_1 = \nu_2 = \nu = 1, 2, 3$. From the results reported in Table 1 we can conclude that, as expected, the AMG algorithms perform better for a larger number of smoothing iterations.

In Table 2 we report the results obtained for the J- and CG-smoother AMG methods when varying the number of coarsening levels $K = 2, \dots, 10$ and solving the problem discretized on structured and unstructured grids, respectively. From

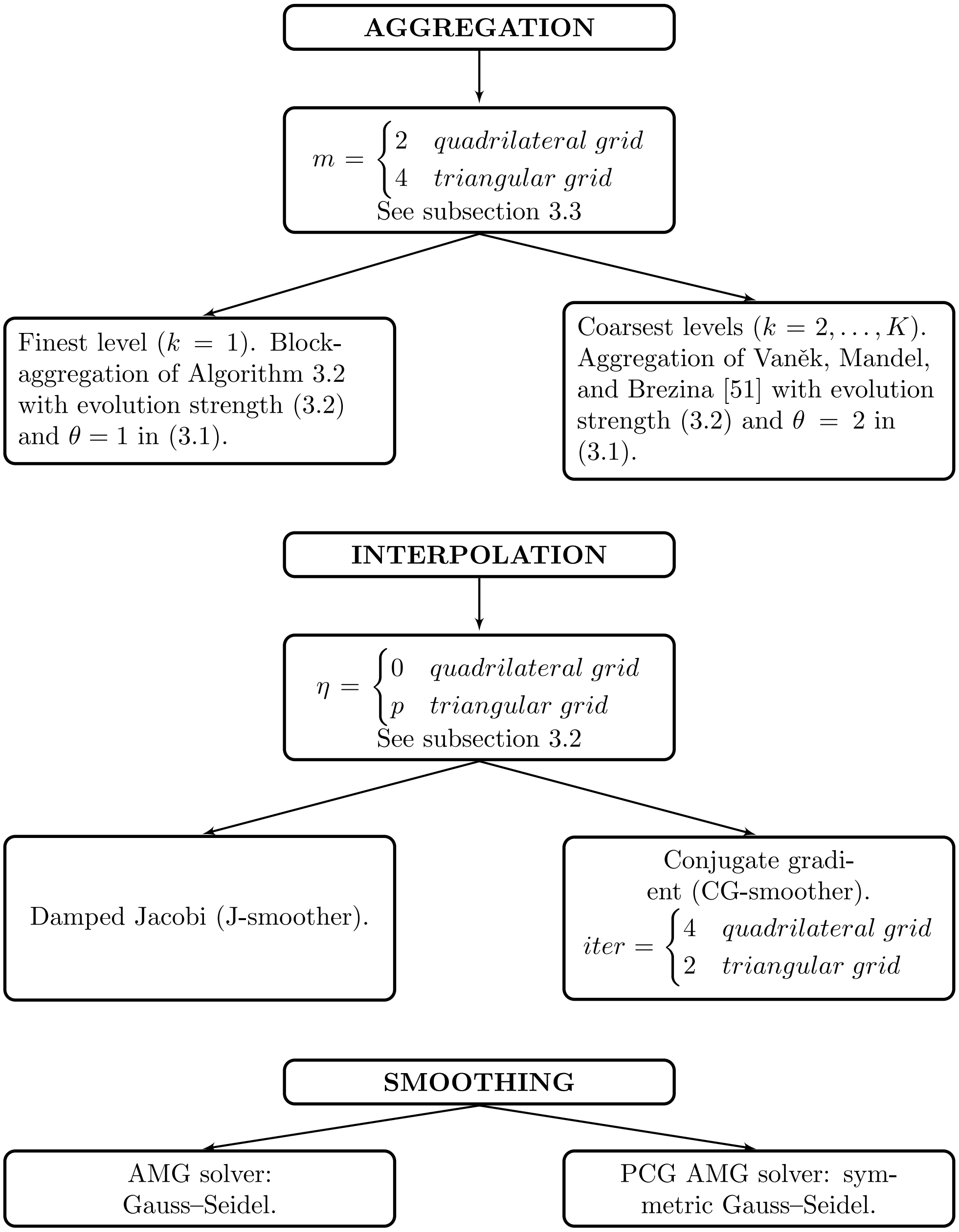
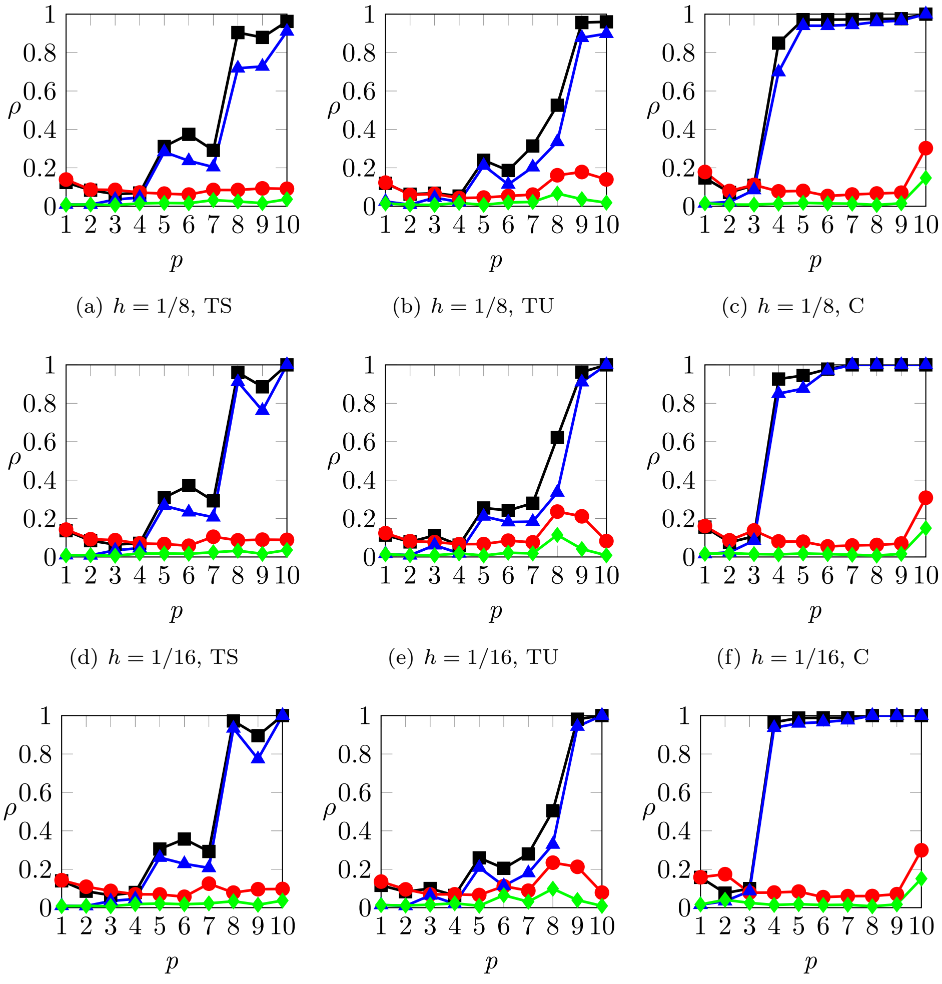

AGGREGATION
$m = \begin{cases} 2 & \textit{quadrilateral grid} \\ 4 & \textit{triangular grid} \end{cases}$
See subsection 3.3
Finest level ($k = 1$). Block-aggregation of Algorithm 3.2 with evolution strength (3.2) and $\theta = 1$ in (3.1).
Coarsest levels ($k = 2, \dots, K$). Aggregation of Vaněk, Mandel, and Brezina [51] with evolution strength (3.2) and $\theta = 2$ in (3.1).
INTERPOLATION
$\eta = \begin{cases} 0 & \textit{quadrilateral grid} \\ p & \textit{triangular grid} \end{cases}$
See subsection 3.2
Damped Jacobi (J-smoother).
Conjugate gradient (CG-smoother).
$\textit{iter} = \begin{cases} 4 & \textit{quadrilateral grid} \\ 2 & \textit{triangular grid} \end{cases}$
SMOOTHING
AMG solver: Gauss–Seidel.
PCG AMG solver: symmetric Gauss–Seidel.

Fig. 4. *Aggregation, interpolation, and smoothing steps in our AMG algorithm.*

the results reported in Table 2, it seems that all of the proposed methods converge uniformly with respect to the number of levels K. As already observed, we can summarize the following considerations:

- the J-smoother AMG method seems to be scalable with respect to both the discretization parameters h and p and the number of multigrid levels, provided that the number ν of smoothing steps is chosen large enough ($\nu \gg 3$). It seems to be hp-weak-scalable for $p = 1, \dots, 7$ for smaller values of ν;



(a) $h = 1/8$, TS

(b) $h = 1/8$, TU

(c) $h = 1/8$, C

(d) $h = 1/16$, TS

(e) $h = 1/16$, TU

(f) $h = 1/16$, C

(g) $h = 1/32$, TS

(h) $h = 1/32$, TU

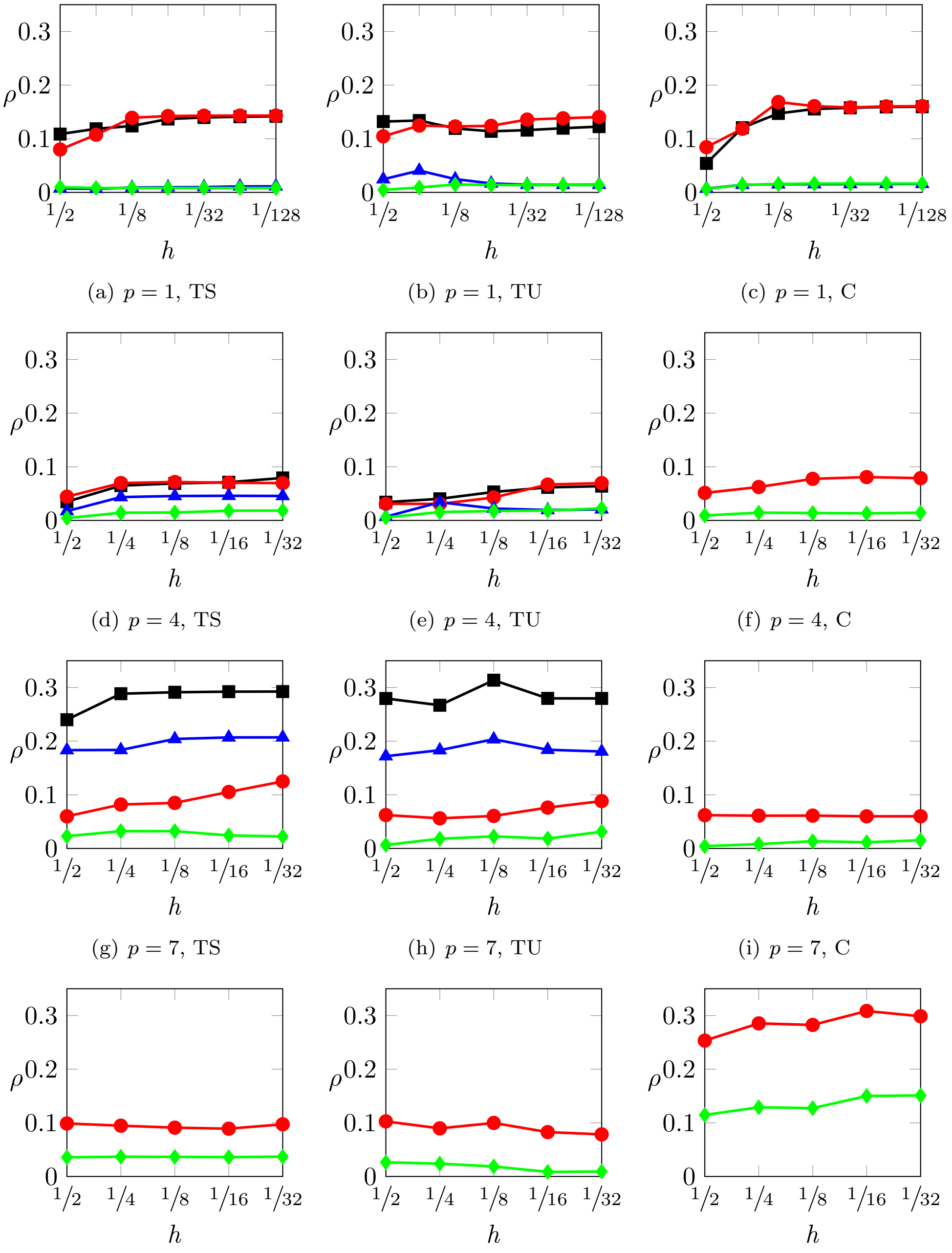
(i) $h = 1/32$, C

FIG. 5. *Convergence factor of the W-cycle algorithm as a function of* p *for different values of* $h = 1/8, 1/16, 1/32$ *on structured triangular (TS, left column), unstructured triangular (TU, center column), and Cartesian (C, right column) meshes: J-smoother,* $\nu = 1$ *(—■—); J-smoother,* $\nu = 3$ *(—▲—); CG-smoother,* $\nu = 1$ *(—●—); CG-smoother,* $\nu = 3$ *(—◆—).*

- the CG-smoother AMG method seems to be scalable with respect to both the discretization parameters h and p and the number of levels, provided that the number of smoothing steps is large enough (in our computations, $\nu = 3$);
- the J-smoother AMG method, even if it seems to be only hp-weak-scalable, features lower computational cost compared to the CG-smoother AMG method.

4.2. W-cycle algorithm as preconditioner for the preconditioned conjugate gradient method. In this section we repeat the numerical tests presented in subsection 4.1, and we present some numerical results to test the efficiency of the W-cycle algorithm as a preconditioner for the PCG method.

In Figures 7 and 8 we report the computed convergence factors based on employing the J- and CG-smoother AMG methods as preconditioners for the PCG method in



(a) $p = 1$, TS

(b) $p = 1$, TU

(c) $p = 1$, C

(d) $p = 4$, TS

(e) $p = 4$, TU

(f) $p = 4$, C

(g) $p = 7$, TS

(h) $p = 7$, TU

(i) $p = 7$, C

(j) $p = 10$, TS

(k) $p = 10$, TU

(l) $p = 10$, C

FIG. 6. *Convergence factor of the W-cycle algorithm as a function of h for different values of $p = 1, 4, 7, 10$ on structured triangular (TS, left column), unstructured triangular (TU, center column), and Cartesian (C, right column) meshes: J-smoother, $\nu = 1$ (—■—); J-smoother, $\nu = 3$ (—▲—); CG-smoother, $\nu = 1$ (—●—); CG-smoother, $\nu = 3$ (—◆—).*

terms of p- and h-scalability, respectively, when employing $\nu_1 = \nu_2 = \nu = 1, 3$ pre- and postsmoothing iterations. We remark that in our tests the J-smoother PCG method

TABLE 1
Convergence factor of the W-cycle algorithm as a function of ν, $K = 4$.

Structured triangular grids				
	ν	$p = 1$	$p = 4$	$p = 7$
		$h = 1/128$	$h = 1/32$	$h = 1/16$
J-smoother	1	0.1418	0.0792	0.2923
	2	0.0282	0.0587	0.2342
	3	0.0113	0.0456	0.2070
CG-smoother	1	0.1432	0.0696	0.1053
	2	0.0228	0.0245	0.0346
	3	0.0082	0.0224	0.0243
Unstructured triangular grids				
	ν	$p = 1$	$p = 4$	$p = 7$
		$h = 1/128$	$h = 1/32$	$h = 1/16$
J-smoother	1	0.1226	0.0639	0.2798
	2	0.0257	0.0225	0.2096
	3	0.0143	0.0206	0.1840
CG-smoother	1	0.1405	0.0796	0.0762
	2	0.0277	0.0249	0.0217
	3	0.0136	0.0226	0.0184

TABLE 2
Convergence factor of the $W(1,1)$-cycle algorithm as a function of the number of levels K.

Structured triangular grids				
	K	$p = 1$	$p = 4$	$p = 7$
		$h = 1/128$	$h = 1/32$	$h = 1/16$
J-smoother	2	0.1418	0.0718	0.2853
	3	0.1418	0.0792	0.2919
	5	0.1418	0.0792	0.2924
	7	0.1418	0.0792	0.2924
	10	0.1418	0.0792	0.2924
CG-smoother	2	0.1432	0.0695	0.1053
	3	0.1432	0.0695	0.1053
	5	0.1432	0.0696	0.1053
	7	0.1432	0.0696	0.1053
	10	0.1432	0.0696	0.1053
Unstructured triangular grids				
	K	$p = 1$	$p = 4$	$p = 7$
		$h = 1/128$	$h = 1/32$	$h = 1/16$
J-smoother	2	0.1225	0.0639	0.2784
	3	0.1226	0.0639	0.2798
	5	0.1226	0.0639	0.2798
	7	0.1226	0.0639	0.2798
	10	0.1226	0.0639	0.2798
CG-smoother	2	0.1405	0.0796	0.0719
	3	0.1405	0.0796	0.0762
	5	0.1405	0.0796	0.0762
	7	0.1405	0.0796	0.0762
	10	0.1405	0.0796	0.0762

does not converge for $p = 4, \dots, 10$ and $p = 10$ in the case of Cartesian and triangular grids, respectively; therefore in Figure 8 these results are omitted. For the sake of brevity, in the following we report only the results obtained with triangular grids.

In Table 3 we report the convergence factor for the J- and CG-smoother PCG methods on both structured and unstructured triangular grids, respectively, when varying the number of smoothing iterations $\nu_1 = \nu_2 = \nu = 1, 2, 3$.

Table 3

Convergence factor of the PCG W-cycle algorithm as a function of ν, $K = 4$ and in comparison with CG.

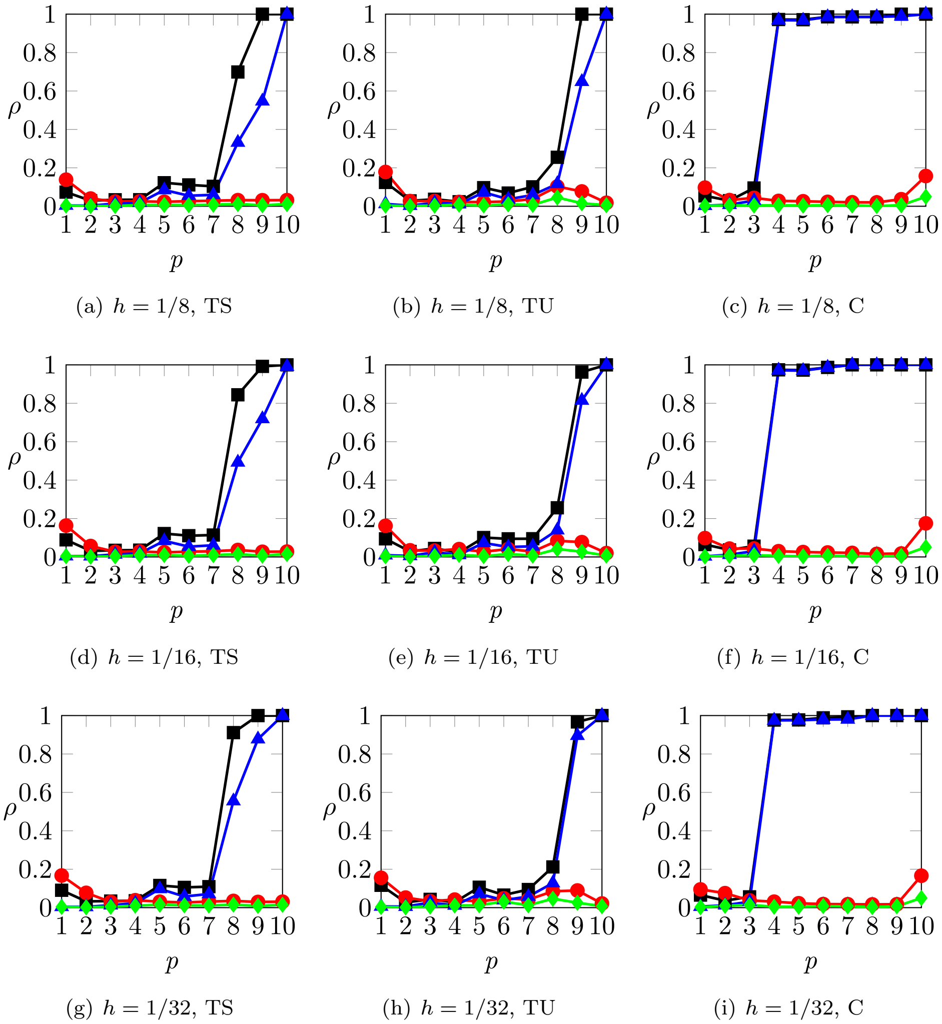
Structured triangular grids				
	ν	$p = 1$ $h = 1/128$	$p = 4$ $h = 1/32$	$p = 7$ $h = 1/16$
J-smoother	1	0.0918	0.0359	0.1141
	2	0.0110	0.0260	0.0773
	3	0.0049	0.0231	0.0607
CG-smoother	1	0.1694	0.0388	0.0298
	2	0.0128	0.0127	0.0126
	3	0.0040	0.0102	0.0098
CG		0.9862	–	–
Unstructured triangular grids				
	ν	$p = 1$ $h = 1/128$	$p = 4$ $h = 1/32$	$p = 7$ $h = 1/16$
J-smoother	1	0.1227	0.0283	0.0949
	2	0.0132	0.0137	0.0626
	3	0.0071	0.0115	0.0554
CG-smoother	1	0.1640	0.0419	0.0280
	2	0.0092	0.0143	0.0114
	3	0.0036	0.0114	0.0071
CG		0.9814	0.9836	0.9841

In Table 4 we report the results obtained for the J- and CG-smoother PCG methods when varying the number of coarsening levels $K = 2, \dots, 10$ and solving the problem discretized on structured and unstructured triangular grids, respectively. For the sake of comparison, in Tables 3 and 4 we also report the computed convergence factor when we employ the unpreconditioned CG method. In particular we specify that the "–" notation indicates that the CG method does not satisfy the stopping criteria within 1500 iterations.

As before, we observe that the J-smoother seems to be scalable with respect to all of the discretization parameters and the number of multigrid levels, provided that the number of smoothing iterations is sufficiently large and the CG-smoother AMG seems to be scalable for smaller values of ν (in our computations, $\nu = 3$). In addition we note that when we employ the two algorithms as preconditioners for the CG method, we obtain better values for the convergence factor.

4.3. Comparison between V-cycle and W-cycle algorithms. In this section we compare the performance of V- and W-cycle algorithms based on employing both J-smoother and CG-smoother AMG methods with $\nu = 1$ pre- and postsmoothing iterations. In this set of experiments the V-cycle and W-cycle AMG algorithms are employed as both an iterative scheme and a preconditioner for the CG to solve the linear system (2.3). We test the V- and W-cycle iterations in the case of structured/unstructured triangular and Cartesian grids, and we obtain similar performance. For the sake of brevity, we report only the results for unstructured triangular meshes.

In Figures 9 and 10 we compare the V- and W-cycle AMG methods with J- and CG-smoothers in terms of p- and h-scalability, respectively, when employed as a stand-alone iterative scheme and as a preconditioner for the CG method with $\nu_1 = \nu_2 = \nu = 1$ pre- and postsmoothing iterations. In both cases we note that if we employ the V- and W-cycle AMG methods with the CG- rather than the J-smoother we obtain better results in terms of both convergence factor and scalability. In particular we



(a) $h = 1/8$, TS

(b) $h = 1/8$, TU

(c) $h = 1/8$, C

(d) $h = 1/16$, TS

(e) $h = 1/16$, TU

(f) $h = 1/16$, C

(g) $h = 1/32$, TS

(h) $h = 1/32$, TU

(i) $h = 1/32$, C

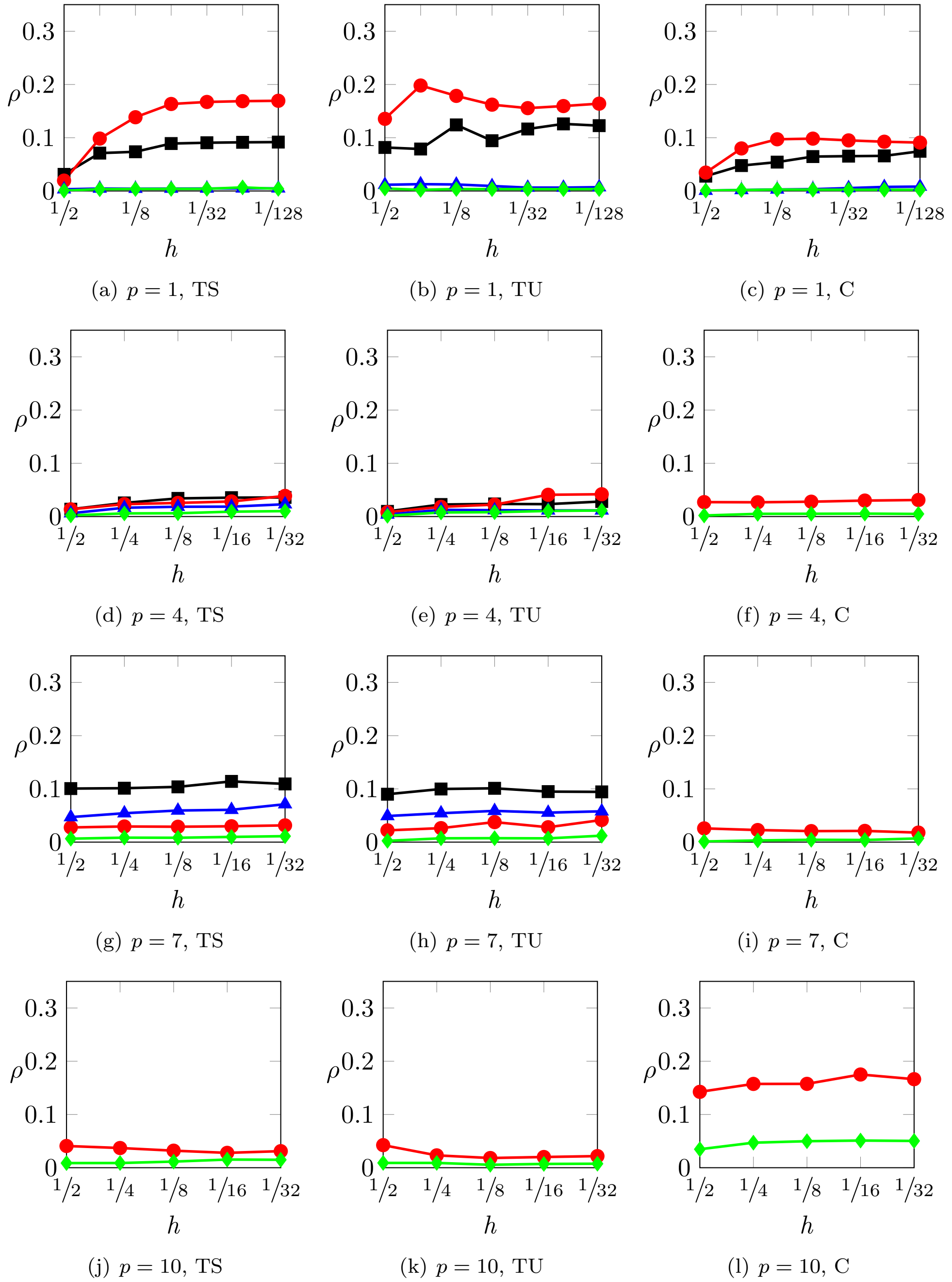
FIG. 7. *Convergence factor of the PCG W-cycle algorithm as a function of p for different values of $h = 1/8, 1/16, 1/32$ on structured triangular (TS, left column), unstructured triangular (TU, center column), and Cartesian (C, right column) meshes: J-smoother, $\nu = 1$ (—■—); J-smoother, $\nu = 3$ (—▲—); CG-smoother, $\nu = 1$ (—●—); CG-smoother, $\nu = 3$ (—◆—).*

observe a similar performance for V- and W-cycles, but for large values of p, the behavior of the W-cycle is more robust.

We remark that in our tests the J-smoother AMG does not converge for $p = 10$ in the case of triangular grids; therefore in Figure 10 we report only the results for the CG-smoother AMG when $p = 10$.

In Figure 11 we compare our CG-smoother AMG with the classical smoothed aggregation scheme by Vaněk, Mandel, and Brezina [51]. We can conclude that our scheme seems to be both h- and p-scalable, whereas the standard smoothed aggregation seems to be only h-scalable.

In Figure 12 we compare the V-cycle AMG algorithm employed as an iterative scheme when we use the damped Jacobi (J), symmetric successive overrelaxation



(a) $p = 1$, TS

(b) $p = 1$, TU

(c) $p = 1$, C

(d) $p = 4$, TS

(e) $p = 4$, TU

(f) $p = 4$, C

(g) $p = 7$, TS

(h) $p = 7$, TU

(i) $p = 7$, C

(j) $p = 10$, TS

(k) $p = 10$, TU

(l) $p = 10$, C

FIG. 8. *Convergence factor of the PCG W-cycle algorithm as a function of h for different values of $p = 1, 4, 7, 10$ on structured triangular (TS, left column), unstructured triangular (TU, center column), and Cartesian (C, right column) meshes: J-smoother, $\nu = 1$ (—■—); J-smoother, $\nu = 3$ (—▲—); CG-smoother, $\nu = 1$ (—●—); CG-smoother, $\nu = 3$ (—◆—).*

(SSOR), and CG methods for the smoothing interpolation step. Similar performances have been obtained in the case of the W-cycle AMG iteration; for the sake of brevity

Table 4

Convergence factor of the PCG $W(1,1)$-cycle algorithm as a function of the number of levels K and in comparison with CG.

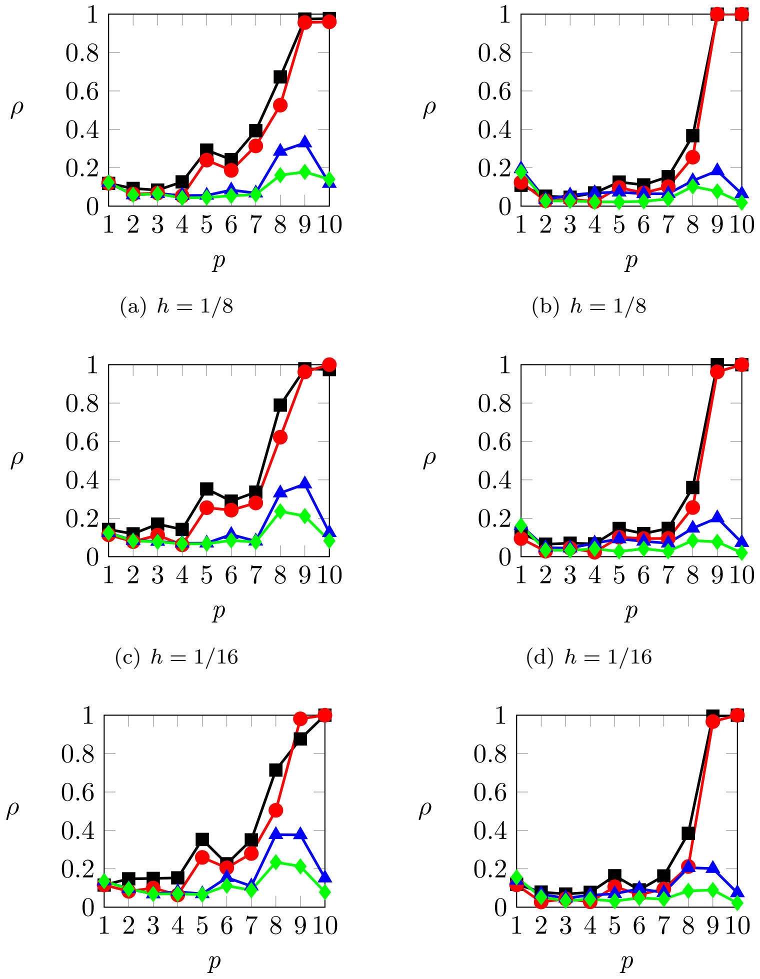
Structured triangular grids				
	K	$p=1$ $h=1/128$	$p=4$ $h=1/32$	$p=7$ $h=1/16$
J-smoother	2	0.0916	0.0358	0.0953
	3	0.0918	0.0359	0.1127
	5	0.0918	0.0359	0.1146
	7	0.0918	0.0359	0.1146
	10	0.0918	0.0359	0.1146
CG-smoother	2	0.1693	0.0385	0.0298
	3	0.1694	0.0388	0.0298
	5	0.1694	0.0388	0.0298
	7	0.1694	0.0388	0.0298
	10	0.1694	0.0388	0.0298
CG		0.9862	–	–

Unstructured triangular grids				
	K	$p=1$ $h=1/128$	$p=4$ $h=1/32$	$p=7$ $h=1/16$
J-smoother	2	0.1227	0.0215	0.0930
	3	0.1228	0.0283	0.0949
	5	0.1227	0.0283	0.0949
	7	0.1227	0.0283	0.0949
	10	0.1227	0.0283	0.0949
CG-smoother	2	0.1640	0.0419	0.0277
	3	0.1640	0.0419	0.0280
	5	0.1640	0.0419	0.0280
	7	0.1640	0.0419	0.0280
	10	0.1640	0.0419	0.0280
CG		0.9814	0.9836	0.9841

these results are omitted. We observe a worsening of the convergence factor and scalability as we increase the number of Jacobi iterations, whereas we note that the SSOR smoother seems to be efficient only in the case where the number of coarse levels is kept small. In contrast, the CG-smoother seems to be competitive both in terms of *hp*-scalability and overall efficiency. For the sake of brevity, in the previous sections we reported only the results obtained with the J-smoother, with one iteration, and those obtained with the CG-smoother.

In Figure 13 we investigate the AMG setup cost (measured in terms of CPU time) to perform the evaluation of the evolution measure (see subsection 3.3), the aggregation operation (see section 3 and subsection 3.1), the interpolation step (see subsection 3.2), and the sum of all runtimes. To make a fair comparison we measured the relative CPU time normalized with respect to the maximum CPU time (obtained with the total runtime of the test with $p = 1$, $h = 1/128$). For the sake of brevity, we show only the results when the CG-smoother is employed in the smoothing interpolation step. We observe that the most expensive operation is the computation of the strength measure. Next, we investigate the memory requirements of our AMG algorithm.

In Figure 14 we report the dimension and the sparsity pattern (number of nonzero entries in the matrix) of the fine and coarse matrices (left, center) and show a comparison of the computational costs between the V- and W-cycles in terms of runtime per iteration as a function of the number of coarse levels (right). We focus the discussion on the computational cost by considering only the CG-smoother AMG, since from



(a) $h = 1/8$

(b) $h = 1/8$

(c) $h = 1/16$

(d) $h = 1/16$

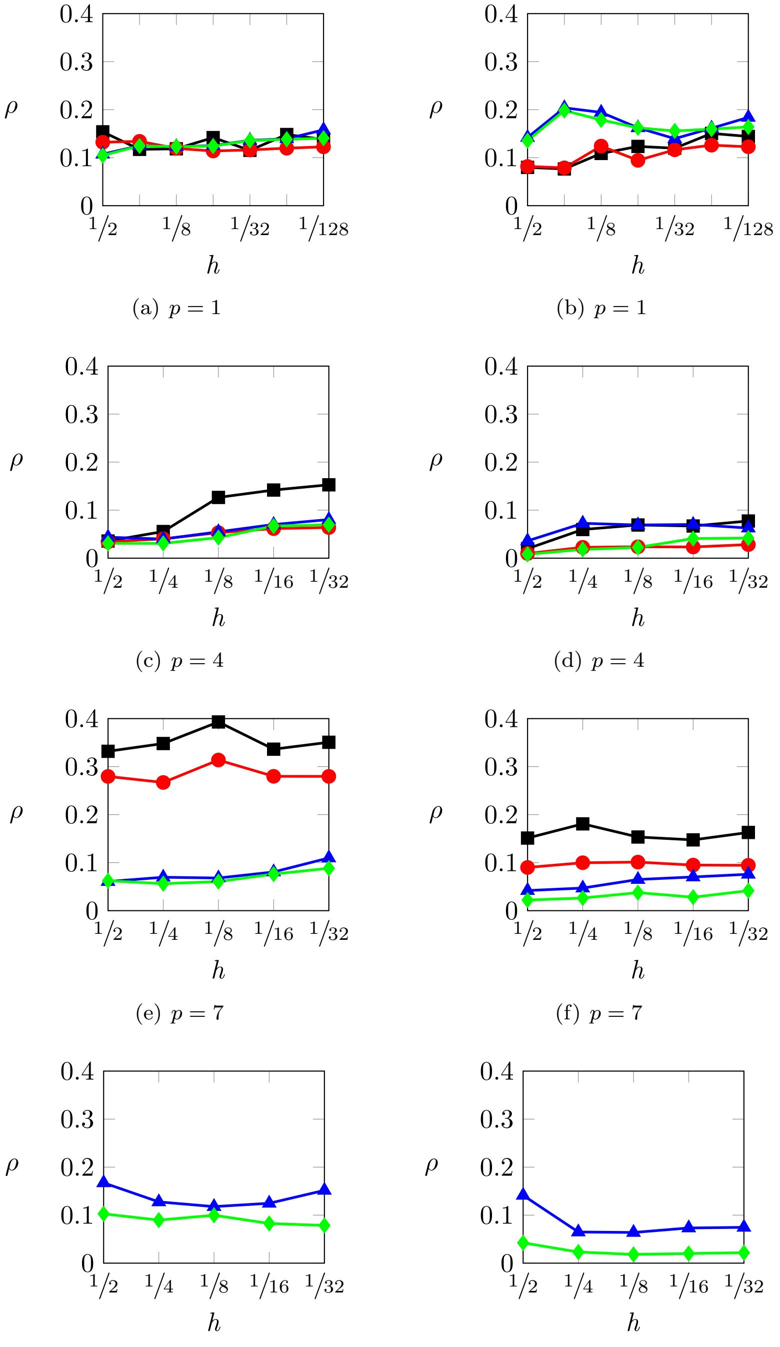
(e) $h = 1/32$

(f) $h = 1/32$

FIG. 9. *Convergence factor of the V- and W-cycle algorithms with $\nu = 1$ as a function of p for different values of $h = 1/8, 1/16, 1/32$ as an iterative scheme (left column) and as a preconditioner for the CG method (right column): V-cycle, J-smoother (—■—); V-cycle, CG-smoother (—▲—); W-cycle, J-smoother (—●—); W-cycle, CG-smoother (—◆—).*

the results reported in subsections 4.1 and 4.2, it seems to exhibit better behavior in terms of scalability. For the sake of brevity, we show only the results for the AMG method employed as an iterative scheme; a similar performance was observed when the AMG method was employed as a preconditioner for the CG method.

Concerning the overall efficiency, we can conclude that the W-cycle algorithm seems to be more robust than the V-cycle—at least for high values of p—but it is more expensive in terms of overall runtime.



(a) $p = 1$

(b) $p = 1$

(c) $p = 4$

(d) $p = 4$

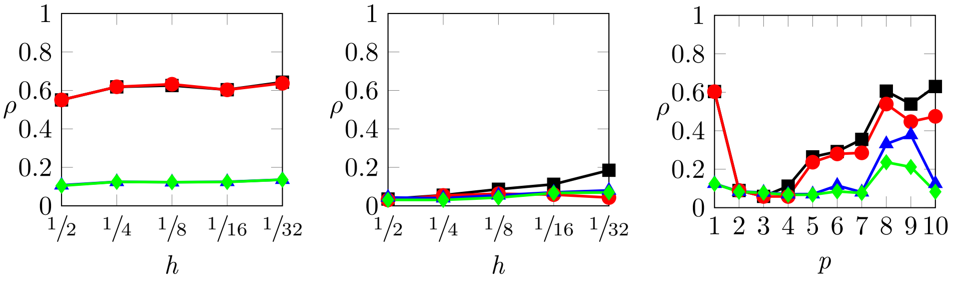
(e) $p = 7$

(f) $p = 7$

(g) $p = 10$

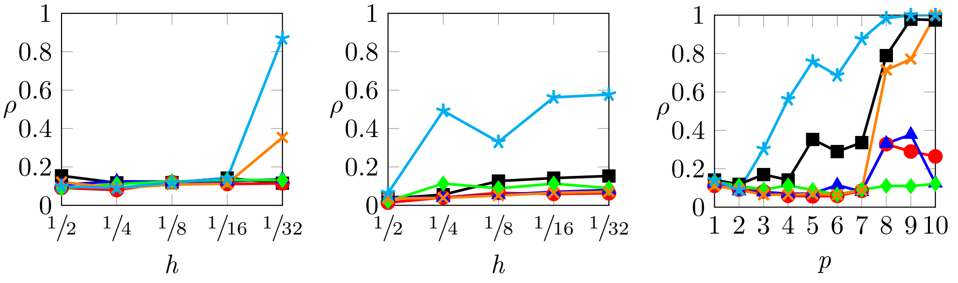
(h) $p = 10$

Fig. 10. *Convergence factor of the V- and W-cycle algorithms with $\nu = 1$ as a function of h for different values of $p = 1, 4, 7, 10$ as an iterative scheme (left column) and as a preconditioner for the CG method (right column): V-cycle, J-smoother (■); V-cycle, CG-smoother (▲); W-cycle, J-smoother (●); W-cycle, CG-smoother (◆).*



(a) $p = 1$ (b) $p = 4$ (c) $h = 1/16$

FIG. 11. *Convergence factor of the V- and W-cycle algorithms as an iterative scheme with $\nu = 1$ as a function of h (left, center) and p (right): V-cycle, SA (—■—); V-cycle, CG-smoother (—▲—); W-cycle, SA (—●—); W-cycle, CG-smoother (—◆—).*



(a) $p = 1$ (b) $p = 4$ (c) $h = 1/16$

FIG. 12. *Convergence factor of the V-cycle algorithm as an iterative scheme with $\nu = 1$ as a function of h (left, center) and p (right): J-smoother, iter = 1 (—■—); J-smoother, iter = 3 (—×—); J-smoother, iter = 8 (—✳—); CG-smoother, iter = 2 (—▲—); SSOR-smoother, iter = 1 (—●—); SSOR-smoother, iter = 3 (—◆—).*

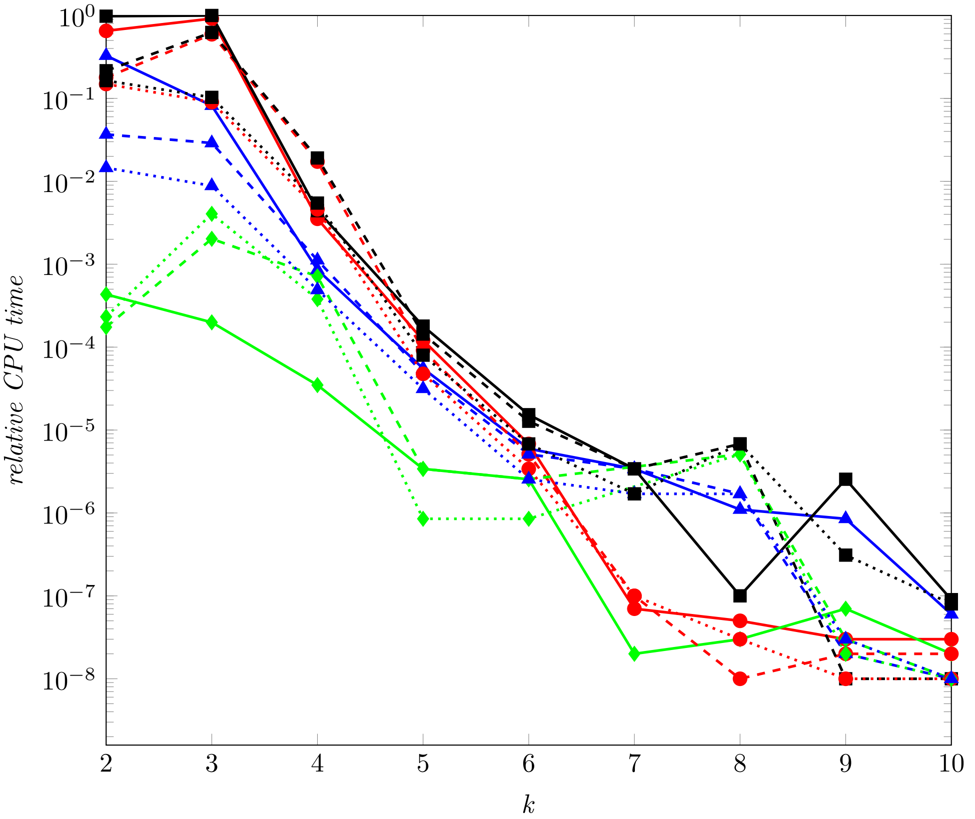
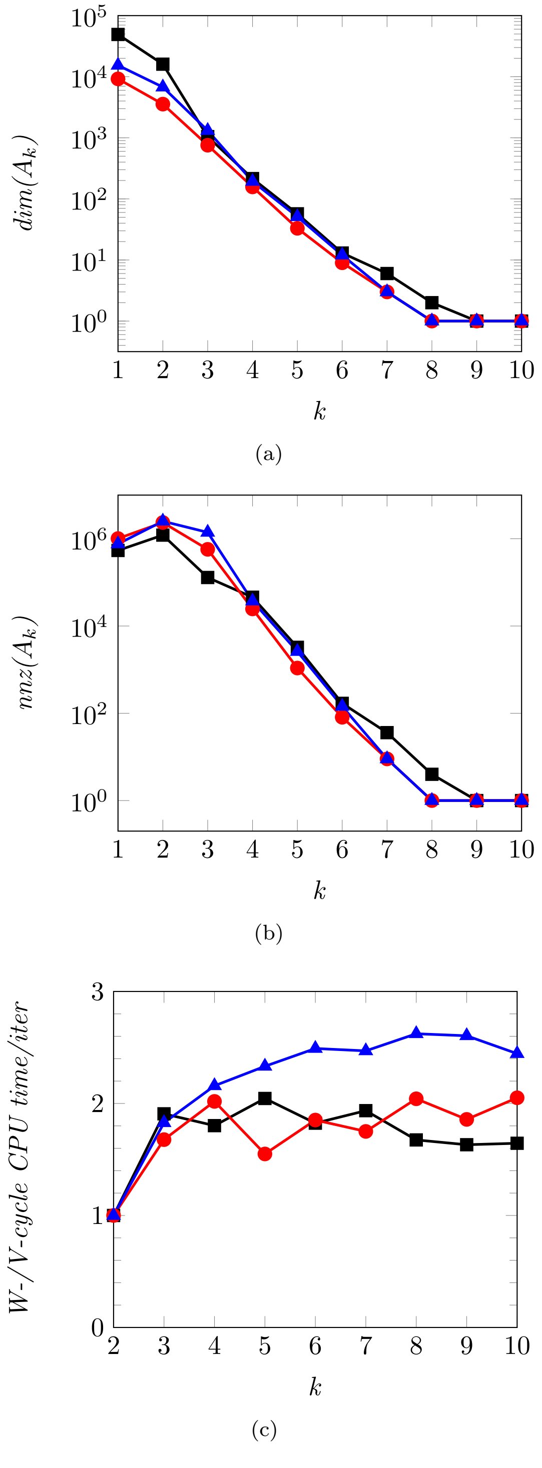


Fig. 13. *Relative CPU time (normalized with respect to the maximum CPU time) of the setup phase of the AMG algorithms as a function of the number of coarse levels* $k = 2, \ldots, K$*, CG-smoother.* $p = 1$, $h = 1/128$*: total (—■—), evolution (—●—), aggregation (—▲—), and interpolation (—◆—) costs.* $p = 4$, $h = 1/32$*: total (- ■ -), evolution (- ● -), aggregation (- ▲ -), and interpolation (- ◆ -) costs.* $p = 7$, $h = 1/16$*: total (··■··), evolution (··●··), aggregation (··▲··), and interpolation (··◆··) costs.*



(a)

(b)

(c)

FIG. 14. *Dimension (top), number of nonzero entries of the matrices A_k (middle), and ratio of the CPU time per iteration of the V- and W-cycle algorithms with $\nu = 1$ (bottom) as a function of the number of coarser levels $k = 1, \dots, K$, CG-smoother: $p = 1$, $h = 1/128$ (—■—); $p = 4$, $h = 1/32$ (—▲—); $p = 7$, $h = 1/16$ (—●—).*

5. Conclusions.

We have presented a new algebraic multigrid (AMG) method for solving linear systems of equations stemming from high-order discontinuous Galerkin (DG) finite element discretizations of second-order elliptic problems.

We have extended the standard AMG approach of Vaněk, Mandel, and Brezina [51] by proposing a new algebraic block-aggregation scheme that suitably handles the redundancy of the degrees of freedom (DOFs) associated to the same grid point. In addition, we have employed a different definition of the strength function of connections and an adaptive smoothed aggregation method, following the ideas of Olson and Schroder [38]. In particular we modified the first step of geometric coarsening within an algebraic framework, leading our schemes to being purely AMG methods for high-order DG discretizations. A set of numerical experiments carried out on both triangular and quadrilateral grids suggests that the proposed AMG methods are scalable with respect to the mesh-size h, the polynomial degree p, and the number of multigrid levels.

Possible future developments include testing of 3D problems. In addition we will expand the proposed methods to a variable diffusion coefficient; cf. [46]. We could also deepen new AMG methods whenever modal shape functions are employed to span the discrete DG space. Finally, concerning the computational aspects of our method, we point out that for higher values of p, the construction of the coarse matrices and interpolation operators becomes more expensive, and therefore we should develop a new implementation based on massively parallel strategies.

REFERENCES

[1] R. A. Adams, *Sobolev Spaces*, Pure Appl. Math. 65, Academic Press, New York, 1975.

[2] P. F. Antonietti and P. Houston, *A class of domain decomposition preconditioners for hp-discontinuous Galerkin finite element methods*, J. Sci. Comput., 46 (2011), pp. 124–149.

[3] P. F. Antonietti, P. Houston, X. Hu, M. Sarti, and M. Verani, *Multigrid algorithms for hp-version interior penalty discontinuous Galerkin methods on polygonal and polyhedral meshes*, Calcolo, 54 (2017), pp. 1169–1198.

[4] P. F. Antonietti, P. Houston, G. Pennesi, and E. Süli, *An agglomeration-based massively parallel non-overlapping additive Schwarz preconditioner for high-order discontinuous Galerkin methods on polytopic grids*, Math. Comp., to appear; published online February 18, 2020, https://doi.org/10.1090/mcom/3510.

[5] P. F. Antonietti and G. Pennesi, *V-cycle multigrid algorithms for discontinuous Galerkin methods on non-nested polytopic meshes*, J. Sci. Comput., 78 (2019), pp. 625–652.

[6] P. F. Antonietti, M. Sarti, and M. Verani, *Multigrid algorithms for hp-discontinuous Galerkin discretizations of elliptic problems*, SIAM J. Numer. Anal., 53 (2015), pp. 598–618, https://doi.org/10.1137/130947015.

[7] D. N. Arnold, *An interior penalty finite element method with discontinuous elements*, SIAM J. Numer. Anal., 19 (1982), pp. 742–760, https://doi.org/10.1137/0719052.

[8] D. N. Arnold, F. Brezzi, B. Cockburn, and L. D. Marini, *Unified analysis of discontinuous Galerkin methods for elliptic problems*, SIAM J. Numer. Anal., 39 (2002), pp. 1749–1779, https://doi.org/10.1137/S0036142901384162.

[9] F. Bassi, L. Botti, and A. Colombo, *Agglomeration-based physical frame DG discretizations: An attempt to be mesh free*, Math. Models Methods Appl. Sci., 24 (2014), pp. 1495–1539.

[10] F. Bassi, L. Botti, A. Colombo, D. A. Di Pietro, and P. Tesini, *On the flexibility of agglomeration based physical space discontinuous Galerkin discretizations*, J. Comput. Phys., 231 (2012), pp. 45–65.

[11] P. Bastian, M. Blatt, and R. Scheichl, *Algebraic multigrid for discontinuous Galerkin discretizations of heterogeneous elliptic problems*, Numer. Linear Algebra Appl., 19 (2012), pp. 367–388.

[12] C. Bernardi and Y. Maday, *Approximations spectrales de problèmes aux limites elliptiques*, Math. Appl. (Berlin) 10, Springer-Verlag, Paris, 1992.

[13] J. Brannick, M. Brezina, S. MacLachlan, T. Manteuffel, S. McCormick, and J. Ruge, *An energy-based AMG coarsening strategy*, Numer. Linear Algebra Appl., 13 (2006), pp. 133–148.

[14] S. C. Brenner, J. Cui, T. Gudi, and L.-Y. Sung, *Multigrid algorithms for symmetric discontinuous Galerkin methods on graded meshes*, Numer. Math., 119 (2011), pp. 21–47.

[15] S. C. Brenner, J. Cui, and L.-Y. Sung, *Multigrid methods for the symmetric interior penalty method on graded meshes*, Numer. Linear Algebra Appl., 16 (2009), pp. 481–501.

[16] S. C. Brenner and L. Owens, *A W-cycle algorithm for a weakly over-penalized interior penalty method*, Comput. Methods Appl. Mech. Engrg., 196 (2007), pp. 3823–3832.

[17] S. C. Brenner and J. Zhao, *Convergence of multigrid algorithms for interior penalty methods*, Appl. Numer. Anal. Comput. Math., 2 (2005), pp. 3–18.

[18] M. Brezina, R. Falgout, S. MacLachlan, T. Manteuffel, S. McCormick, and J. Ruge, *Adaptive smoothed aggregation* (αSA) *multigrid*, SIAM Rev., 47 (2005), pp. 317–346, https://doi.org/10.1137/050626272.

[19] M. Briani, A. Sommariva, and M. Vianello, *Computing Fekete and Lebesgue points: Simplex, square, disk*, J. Comput. Appl. Math., 236 (2012), pp. 2477–2486.

[20] W. Briggs and S. McCormick, *Introduction*, in Multigrid Methods, Frontiers Appl. Math. 3, S. F. McCormick, ed., SIAM, Philadelphia, 1987, pp. 1–30, https://doi.org/10.1137/1.9781611971057.ch1.

[21] W. L. Briggs, V. E. Henson, and S. F. McCormick, *A Multigrid Tutorial*, 2nd ed., SIAM, Philadelphia, 2000, https://doi.org/10.1137/1.9780898719505.

[22] C. Canuto, M. Y. Hussaini, A. Quarteroni, and Th. A. Zang, *Spectral Methods: Fundamentals in Single Domains*, Sci. Comput., Springer-Verlag, Berlin, 2006.

[23] D. A. Di Pietro and A. Ern, *Mathematical Aspects of Discontinuous Galerkin Methods*, Math. Appl. (Berlin) 69, Springer, Heidelberg, 2012.

[24] V. A. Dobrev, R. D. Lazarov, P. S. Vassilevski, and L. T. Zikatanov, *Two-level preconditioning of discontinuous Galerkin approximations of second-order elliptic equations*, Numer. Linear Algebra Appl., 13 (2006), pp. 753–770.

[25] E. H. Georgoulis and E. Süli, *Optimal error estimates for the hp-version interior penalty discontinuous Galerkin finite element method*, IMA J. Numer. Anal., 25 (2005), pp. 205–220.

[26] J. Gopalakrishnan and G. Kanschat, *A multilevel discontinuous Galerkin method*, Numer. Math., 95 (2003), pp. 527–550.

[27] B. T. Helenbrook and H. L. Atkins, *Application of p-multigrid to discontinuous Galerkin formulations of the Poisson equation*, AIAA J., 44 (2006), pp. 566–575.

[28] B. T. Helenbrook and H. L. Atkins, *Solving discontinuous Galerkin formulations of Poisson's equation using geometric and p multigrid*, AIAA J., 46 (2008), pp. 894–902.

[29] J. S. Hesthaven, *From electrostatics to almost optimal nodal sets for polynomial interpolation in a simplex*, SIAM J. Numer. Anal., 35 (1998), pp. 655–676, https://doi.org/10.1137/S003614299630587X.

[30] J. S. Hesthaven and T. Warburton, *Nodal Discontinuous Galerkin Methods: Algorithms, Analysis, and Applications*, Texts Appl. Math. 54, Springer-Verlag, New York, 2008.

[31] P. Houston, C. Schwab, and E. Süli, *Discontinuous hp-finite element methods for advection-diffusion-reaction problems*, SIAM J. Numer. Anal., 39 (2002), pp. 2133–2163, https://doi.org/10.1137/S0036142900374111.

[32] J. E. Jones and P. S. Vassilevski, *AMGe based on element agglomeration*, SIAM J. Sci. Comput., 23 (2001), pp. 109–133, https://doi.org/10.1137/S1064827599361047.

[33] J. Mandel, M. Brezina, and P. Vaněk, *Energy optimization of algebraic multigrid bases*, Computing, 62 (1999), pp. 205–228.

[34] J. Mandel, S. McCormick, and R. Bank, *Variational multigrid theory*, in Multigrid Methods, Frontiers Appl. Math. 3, S. F. McCormick, ed., SIAM, Philadelphia, 1987, pp. 131–177, https://doi.org/10.1137/1.9781611971057.ch5.

[35] B. S. Mascarenhas, B. T. Helenbrook, and H. L. Atkins, *Coupling p-multigrid to geometric multigrid for discontinuous Galerkin formulations of the convection-diffusion equation*, J. Comput. Phys., 229 (2010), pp. 3664–3674.

[36] S. F. McCormick and J. W. Ruge, *Multigrid methods for variational problems*, SIAM J. Numer. Anal., 19 (1982), pp. 924–929, https://doi.org/10.1137/0719067.

[37] L. N. Olson, J. Schroder, and R. S. Tuminaro, *A new perspective on strength measures in algebraic multigrid*, Numer. Linear Algebra Appl., 17 (2010), pp. 713–733.

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

[39] L. N. Olson, J. B. Schroder, and R. S. Tuminaro, *A general interpolation strategy for algebraic multigrid using energy minimization*, SIAM J. Sci. Comput., 33 (2011), pp. 966–991, https://doi.org/10.1137/100803031.

[40] R. Pasquetti and F. Rapetti, *Spectral element methods on triangles and quadrilaterals: Comparisons and applications*, J. Comput. Phys., 198 (2004), pp. 349–362.

[41] R. Pasquetti and F. Rapetti, *Spectral element methods on unstructured meshes: Comparisons and recent advances*, J. Sci. Comput., 27 (2006), pp. 377–387.

[42] I. Perugia and D. Schötzau, *An hp-analysis of the local discontinuous Galerkin method for diffusion problems*, J. Sci. Comput., 17 (2002), pp. 561–571.

[43] F. Prill, M. Lukáčová-Medviďová, and R. Hartmann, *Smoothed aggregation multigrid for the discontinuous Galerkin method*, SIAM J. Sci. Comput., 31 (2009), pp. 3503–3528, https://doi.org/10.1137/080728457.

[44] B. Rivière, *Discontinuous Galerkin Methods for Solving Elliptic and Parabolic Equations: Theory and Implementation*, Frontiers Appl. Math. 35, SIAM, Philadelphia, 2008, https://doi.org/10.1137/1.9780898717440.

[45] J. W. Ruge and K. Stüben, *Algebraic multigrid*, in Multigrid Methods, Frontiers Appl. Math. 3, S. F. McCormick, ed., SIAM, Philadelphia, 1987, pp. 73–130, https://doi.org/10.1137/1.9781611971057.ch4.

[46] J. B. Schroder, *Smoothed aggregation solvers for anisotropic diffusion*, Numer. Linear Algebra Appl., 19 (2012), pp. 296–312.

[47] B. Stamm and T. P. Wihler, *hp-optimal discontinuous Galerkin methods for linear elliptic problems*, Math. Comp., 79 (2010), pp. 2117–2133.

[48] M. A. Taylor, B. A. Wingate, and R. E. Vincent, *An algorithm for computing Fekete points in the triangle*, SIAM J. Numer. Anal., 38 (2000), pp. 1707–1720, https://doi.org/10.1137/S0036142998337247.

[49] A. Toselli and O. Widlund, *Domain Decomposition Methods: Algorithms and Theory*, Springer Ser. Comput. Math. 34, Springer-Verlag, Berlin, 2005.

[50] M. H. van Raalte and P. W. Hemker, *Two-level multigrid analysis for the convection-diffusion equation discretized by a discontinuous Galerkin method*, Numer. Linear Algebra Appl., 12 (2005), pp. 563–584.

[51] P. Vaněk, J. Mandel, and M. Brezina, *Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems*, Computing, 56 (1996), pp. 179–196.

[52] W. L. Wan, T. F. Chan, and B. Smith, *An energy-minimizing interpolation for robust multigrid methods*, SIAM J. Sci. Comput., 21 (2000), pp. 1632–1649, https://doi.org/10.1137/S1064827598334277.

[53] T. Warburton, *An explicit construction of interpolation nodes on the simplex*, J. Engrg. Math., 56 (2006), pp. 247–262.

[54] M. F. Wheeler, *An elliptic collocation-finite element method with interior penalties*, SIAM J. Numer. Anal., 15 (1978), pp. 152–161, https://doi.org/10.1137/0715010.

[55] J. Xu and L. Zikatanov, *On an energy minimizing basis for algebraic multigrid methods*, Comput. Vis. Sci., 7 (2004), pp. 121–127.

[56] J. Xu and L. Zikatanov, *Algebraic multigrid methods*, Acta Numer., 26 (2017), pp. 591–721.