ALGEBRAIC MULTIGRID FOR MODERATE ORDER FINITE ELEMENTS

ARTEM NAPOV† AND YVAN NOTAY‡

Abstract. We investigate the use of algebraic multigrid (AMG) methods for the solution of large sparse linear systems arising from the discretization of scalar elliptic partial differential equations with Lagrangian finite elements of order at most 4. The resulting system matrices do not have the M-matrix property that is required by standard analyses of classical AMG and aggregation-based AMG methods. A unified approach is presented that allows us to extend these analyses. It uses an intermediate M-matrix and highlights the role of the spectral equivalence constant that relates this matrix to the original system matrix. This constant is shown to be bounded independently of the problem size and jumps in the coefficients of the partial differential equations, provided that jumps are located at elements’ boundaries. For two-dimensional problems, it is further shown to be uniformly bounded if the angles in the triangulation also satisfy a uniform bound. This analysis validates the application of the AMG methods to the considered problems. On the other hand, because the intermediate M-matrix can be computed automatically, an alternative strategy is to define the AMG preconditioners from this matrix, instead of defining them from the original matrix. Numerical experiments are presented that assess both strategies using publicly available state-of-the-art implementations of classical AMG and aggregation-based AMG methods.

Key words. multigrid, algebraic multigrid, AMG, convergence analysis, preconditioning, aggregation

AMS subject classifications. 65F10, 65N55, 65F50

DOI. 10.1137/130922616

1. Introduction. Efficient solution of large sparse $n \times n$ linear systems

\[
Ax = b
\]

is critical for numerous applications in science and engineering, including discrete partial differential equations (PDEs). In this work we focus on symmetric positive definite (SPD) systems that arise from the discretization of a second order scalar elliptic PDE

\[
\begin{cases}
-\nabla (D \nabla u) = f \\
u = g_0 \\
\frac{\partial u}{\partial n} = g_1
\end{cases}
\text{ in } \Omega,
\text{ on } \Gamma_0,
\text{ on } \Gamma_1 = \partial \Omega \setminus \Gamma_0
\]

with a finite element method of moderate order. By “moderate order” we mean, typically, Lagrangian finite elements of order 2, 3, or 4. These methods are appealing because they offer greater accuracy than low order discretizations while still allowing a
large number of elements, as required when approximating complex geometries and/or applying local refinement.

For low order discretizations of this type of PDE, it is now standard to solve the resulting linear system with an algebraic multigrid (AMG) method. Options include classical AMG developed along the lines of the seminal works by Brandt, McCormick, Ruge, and Stüben [6, 31], smoothed aggregation AMG initiated by Vaněk, Mandel, and Brezina [37], and (plain) aggregation-based AMG as recently developed by the present authors [22, 28]. These techniques are attractive because from their multigrid nature they inherit the potentiality to solve linear systems in a computing time that scales linearly with the number of unknowns. On the other hand, unlike geometric multigrid methods [14, 35], they work essentially as black box algebraic solvers and can be used to solve problems arising from the discretization on unstructured grids.

Thanks to their algebraic nature, these methods can be applied to any linear system independently of the underlying discretization, and moderate order finite element discretizations are not an exception. However, the few results reported in the literature are mitigated; see [16, Table 1] for classical AMG and [33, Tables 2.1–2.2] for a method [19] based on smoothed aggregation AMG. Moreover, considering more specifically classical AMG and aggregation-based AMG, it is clear that their justification and validation is intimately connected with some properties that are specific to low order discretization matrices.

1) On the one hand, the standard theoretical analyses (as developed in [5, 31, 34] for classical AMG and in [21, 22] for aggregation-based AMG) mainly rely on the assumption that the system matrix is an M-matrix with nonnegative row-sum. Recall that an M-matrix has nonpositive offdiagonal entries, and hence M-matrices with nonnegative row-sum are also weakly diagonally dominant. Clearly, using moderate order finite elements implies the presence of both positive and negative offdiagonal entries. This also holds for linear finite elements when some angles of the triangulation are obtuse. Then, one loses not only the sign property of the offdiagonal entries but also the implied diagonal dominance, since for PDEs like (1.2) the row-sum remains zero everywhere except near boundaries. Hence, larger is the weight of the positive offdiagonal elements, weaker is the diagonal with respect to the remainder of the row.\footnote{It is worth noting that the M-matrix assumption allows one to also cover non-PDE applications like Laplacian of graphs; see [18] for developments in this direction.}

2) On the other hand, all AMG methods use a coarsening algorithm, which explores the graph of the system matrix to identify a sensible set of coarse variables. Although quite different in nature, the coarsening algorithms in [6, 31, 34] and [22] have in common that they have been designed with matrix graphs corresponding to some typical low connectivity stencils in mind. However, moderate order finite elements involve significantly more complex connectivity schemes. Not only is the average number of connections per node much larger, but vertices, faces, edges, and interior nodes have different connectivity patterns and need nevertheless to be treated in a uniform way by the coarsening algorithm.

Note that these observations imply not that AMG methods have to effectively fail or behave poorly but, rather, that some (re)assessment is needed to check their potential for moderate order finite element discretizations.
That said, most if not all available studies focus on alternative approaches. One family of methods is based on the properties of the matrix when the discretization is performed in the hierarchical basis. Then, partitioning the unknowns according to the degree of the associated finite element basis function, the corresponding matrix expressed in block form possesses some interesting algebraic properties [3]. Combined with the ability to easily solve subproblems associated with linear basis functions (e.g., with an AMG method), these properties allow one to easily set up fast and scalable solution algorithms; see [11] for recent results in this direction. Now, using such schemes is of course natural when the discretization is indeed performed in the hierarchical basis. But, in this work, we focus on the opposite situation, where the discretization is performed in the usual nodal basis. Of course, the matrix that allows one to change from nodal to hierarchical basis has a relatively simple structure, but information from the discretization process is needed to build it. In [33], a strategy is proposed to construct algebraically this transformation matrix, which requires a different set of parameters for different element types; the reference only covers P2 and P3 elements in two dimensions.

Another family of methods exploits an intermediate matrix that corresponds to, or resembles, a matrix from a discretization with linear finite elements. One way to obtain this matrix is to rediscretize the PDE with first order finite elements while preserving the number and the location of the degrees of freedom [16, 30]. Assuming that the element matrices are available as well as the corresponding assembly routine, another option is to compute a sparse approximation of each element matrix and then assemble them to form the intermediate matrix [1]. In both cases, once the intermediate matrix is obtained, one can safely apply to it an AMG scheme, which is then used as a preconditioner for the original problem. An advantage of these approaches is that they work for finite elements of arbitrary order or even spectral elements; see [10] for early results. However, additional information from the discretization is needed to set up the solution scheme; i.e., we no longer have a purely algebraic solver.

Now, the just mentioned approaches were likely motivated by the will to be robust with respect to refinement by increasing the element order ($p$-refinement). By contrast, we deliberately restrict ourselves to moderate order finite elements. Then, the challenge is more, given a finite element order, to maintain the robustness with respect to the refinement by decreasing the mesh size ($h$-refinement). Moreover, moderate order finite elements are frequently used in industrial codes, a context in which it is often mandatory to have a purely algebraic solver.

Here, following an approach that traces back to [2, 4] (see also [38, pp. 121–122]), we also use an intermediate matrix $\mathcal{M}(A)$, which, however, is computed automatically from the system matrix $A$ by discarding the positive offdiagonal entries and adding these entries to the diagonal in order to preserve the row-sum. It is known that $\mathcal{M}(A)$ is an M-matrix if $A$ is SPD and moreover that

$$\alpha \ v^T \mathcal{M}(A) \ v \leq v^T \ A \ v \leq v^T \ \mathcal{M}(A) \ v \quad \forall \ v \in \mathbb{R}^n$$

(1.3)

holds for some positive number $\alpha$. If $\alpha$ is not too small, $\mathcal{M}(A)$ is said to be spectrally equivalent to $A$. For any SPD matrix (preconditioner) $B$, the double inequality (1.3) implies

$$\kappa (B^{-1} \ A) \leq \alpha^{-1} \ \kappa (B^{-1} \mathcal{M}(A)),$$

(1.4)

where $\kappa(\cdot) = \lambda_{\max}(\cdot)/\lambda_{\min}(\cdot)$ is the spectral condition number, i.e., the ratio of extremal eigenvalues, which governs the convergence of the conjugate gradient (CG)
method. When $M(A)$ is spectrally equivalent to $A$, it is therefore sensible to use for $A$ a preconditioner that is defined from $M(A)$. This fact originally motivated the use of $M(A)$ with SOR methods and incomplete factorizations [2, 4], including for systems from finite element discretizations [32].

It is known from [32] that in the finite element context $\alpha$ is indeed independent of the number of elements and of their size and also of jumps in the PDE coefficients. Here we further analyze this constant for moderate order finite element discretizations of (1.2) in two dimensions. We establish that it depends only mildly on elements’ shape and is uniformly bounded if the angles of the triangulation are not too small. This analysis, which is the first main theoretical contribution in this paper, also covers linear finite elements with obtuse angles and provides us with quantitative estimates of spectral equivalence constants.

Now, we exploit the M-matrix and spectral equivalence properties in two ways. First, motivated by (1.4) we consider defining an AMG preconditioner based on the intermediate matrix $M(A)$ and subsequently using this preconditioner to solve the original linear system.

But we also consider the direct application of classical and aggregation-based AMG to the original system matrix $A$. Here, $M(A)$ is neither needed nor used at a practical level, but the spectral equivalence relations (1.3) help us to extend the available theoretical analyses and thus assess the potential of the approach.

For classical AMG, this yields a result which, although slightly more general, is not much different from that obtained with the concept of essentially positive-type matrices used in [5, 31, 34]. Hence we mainly build a bridge between this concept and the spectral equivalence with $M(A)$. This enables the direct use of the aforementioned results on the constant $\alpha$. Besides, we are not aware of any reference discussing in detail whether moderate order finite element discretizations of (1.2) lead to essentially positive-type matrices.

For aggregation-based AMG, we extend the analysis in [22] and show that the aggregation strategy proposed there can be directly applied to the original system matrix, with a penalty factor on the condition number estimate that does not exceed $\alpha^{-1}$. This is our second main theoretical contribution.

For both methods and both strategies (direct application or application to $M(A)$), we thus overcome the potential difficulties stated in (1) above: either the AMG scheme is applied to an M-matrix or the properly extended supporting theories cover any matrix satisfying (1.3) for reasonable $\alpha$. However, the impact of more involved and irregular sparsity patterns can only be assessed via numerical experiments, which are also needed to compare the different approaches. Observe here that none of these approaches requires the modification of an existing code (as long as it allows us to separate the preconditioner’s definition from the solution process, which is relatively standard). Since implementing an AMG method is difficult and time-consuming, maximal insight is then obtained by running available software without modification. We thus conduct the comparison using the Boomer AMG [15] implementation of classical AMG (which is part of the HYPRE package [17]) and the AGMG [23] implementation of aggregation-based AMG. Note that “without modification” may mean “as block box solver” but also “without modification of the source code,” leaving space to explore the tuning options that have been made available to the expert users. In our experiments, we consider Boomer AMG from both these viewpoints, whereas AGMG, which has basically no option, is tested only with the default setting.

The outline of the paper is as follows. In section 2, we present the general properties of $M(A)$ and analyze the constant $\alpha$ in (1.3). In section 3, we recall some
basic facts about AMG methods and state the algebraic results required by the analyses considered thereafter. In section 4 we consider more specifically classical AMG, whereas in section 5 we develop our analysis of aggregation-based AMG. Numerical results are reported in section 6, and concluding remarks are given in section 7.

2. M-matrix approximation to general SPD matrices.

2.1. Definition and general properties. For any $n \times n$ matrix $A = (a_{ij})$ we define $\mathcal{M}(A)$ to be a unique $n \times n$ matrix satisfying

\begin{align}
(\mathcal{M}(A))_{ij} &= \min(a_{ij}, 0) \quad \text{for } i \neq j, \\
\mathcal{M}(A) \mathbf{1} &= A \mathbf{1},
\end{align}

where $\mathbf{1} = (1 \ldots 1)^T$ is the vector of all ones. Some useful properties of $\mathcal{M}(\cdot)$ are gathered in the following lemma.

**Lemma 2.1.** Let $A = (a_{ij})$ be a $n \times n$ symmetric matrix, and let $\mathcal{M}(A)$ be defined by (2.1), (2.2).

(i) If $A$ is SPD, then the matrix $\mathcal{M}(A)$ is an M-matrix and (1.3) holds for some $\alpha > 0$.

(ii) If $P$ is a $n \times m$ Boolean matrix (i.e., matrix with entries 0 or 1) with at most one nonzero entry per row, then

\begin{align}
\mathbf{v}^T P^T \mathcal{M}(A) P \mathbf{v} &\geq \mathbf{v}^T \mathcal{M}(P^T A P) \mathbf{v} \quad \forall \mathbf{v} \in \mathbb{R}^n.
\end{align}

(iii) If $B$ is a symmetric $n \times n$ matrix and $\beta, \gamma \geq 0$ are nonnegative numbers, then

\begin{align}
\beta \mathbf{v}^T \mathcal{M}(A) \mathbf{v} + \gamma \mathbf{v}^T \mathcal{M}(B) \mathbf{v} &\geq \mathbf{v}^T \mathcal{M}(\beta A + \gamma B) \mathbf{v} \quad \forall \mathbf{v} \in \mathbb{R}^n.
\end{align}

**Proof.** We begin noting that $\mathcal{M}(A) - A$ is weakly diagonally dominant, as may be concluded from the definition of $\mathcal{M}(A)$. This implies the right inequality (1.3) and, if $A$ is SPD, further shows that $\mathcal{M}(A)$ is SPD as well and hence an M-matrix as it has nonpositive offdiagonal entries. The existence of $\alpha > 0$ satisfying the left inequality (1.3) follows from the positive definiteness of $A$. Regarding (ii), we may assume, for ease of presentation and without loss of generality, that the zero rows of $P$ are ordered first. Hence

\begin{align}
P \mathbf{1}_m &= \begin{pmatrix} 0_{n_0} \\ 1_{n_1} \end{pmatrix},
\end{align}

where $n_0$ and $n_1$ denote, respectively, the number of zero and nonzero rows in $P$. Consider then

\begin{align}
P^T \mathcal{M}(A) P \mathbf{1}_m - \mathcal{M}(P^T A P) \mathbf{1}_m = P^T (\mathcal{M}(A) - A) P \mathbf{1}_m = P^T (\mathcal{M}(A) - A) \begin{pmatrix} 0_{n_0} \\ 1_{n_1} \end{pmatrix}.
\end{align}

The weak diagonal dominance of $\mathcal{M}(A) - A$ implies that the last $n_1$ entries of

\begin{align}
\mathbf{v} = (\mathcal{M}(A) - A) \begin{pmatrix} 0_{n_0} \\ 1_{n_1} \end{pmatrix}
\end{align}

are nonnegative, whereas our assumptions on $P$ ensure that any entry in $P^T \mathbf{v}$ is the sum of some of these last $n_1$ entries. Hence $P^T \mathcal{M}(A) P - \mathcal{M}(P^T A P)$ has nonnegative
row-sum. At the same time this latter matrix has nonpositive off-diagonal entries, since, for all \( i \neq j \) (noting that \( p_{k,i} p_{\ell,j} \neq 0 \) and \( i \neq j \) imply \( k \neq \ell \) because otherwise \( P \) would have multiple nonzero entries in a row)

\[
(P^T M(A) P)_{ij} = \sum_{k, \ell} p_{k,i} p_{\ell,j} \min(a_{k,\ell}, 0) \leq \min \left( \sum_{k, \ell} p_{k,i} p_{\ell,j} a_{k,\ell}, 0 \right) = (M(P^T A P))_{ij}.
\]

\( P^T M(A) P - M(P^T A P) \) is therefore weakly diagonally dominant, proving (2.3).

To prove the last statement, we apply the second statement to \( \tilde{A} = (\beta A \gamma B) \), \( \tilde{P} = \begin{pmatrix} I & I \end{pmatrix} \); the conclusions follow because \( M(\beta A) = \beta M(A) \), \( M(\gamma B) = \gamma M(B) \) for \( \beta, \gamma \geq 0 \).

We now discuss the constant \( \alpha \) in (1.3). When \( A \) results from the finite element discretization of an elliptic PDE, it corresponds to the assembly of local element matrices, and \( \alpha \) is bounded below by the smallest \( \alpha_e \) associated with these element matrices. Hence \( \alpha \) is bounded independently of the problem size and also of jumps in the PDE coefficients as long as these are located at elements’ boundaries. This fact was proved in [32] and is also mentioned in [38, p. 122]. It can also be straightforwardly recovered from item (ii) of the above lemma. Indeed, the assembly process can be formally represented by

\[
A = E^T A_E E,
\]

where \( A_E = \text{blockdiag}(A_e) \) is the block diagonal matrix whose diagonal blocks are the element matrices, and where \( E \) is a matrix with exactly one nonzero entry per row, which is further equal to 1; that is, \( E \) can play the role of \( P \) in Lemma 2.1. If for each diagonal block \( A_e \) of \( A_E \) there exists a positive constant \( \alpha_e \) such that

\[
(2.5) \quad v_e^T A_e v_e \geq \alpha_e v_e^T M(A_e) v_e \quad \forall v_e \in \mathbb{R}^{n_e},
\]

then, setting

\[
(2.6) \quad \alpha = \min_e \alpha_e,
\]

one has \( v^T A_E v \geq \alpha v^T M(A_E) v \) for all \( v \in \mathbb{R}^n \), and further

\[
v^T A v = v^T E^T A_E E v \geq \alpha v^T E^T M(A_E) E v \geq \alpha v^T M(E^T A_E E) v = \alpha v^T M(A) v.
\]

Moreover, these arguments are easily extended to the case where Dirichlet boundary conditions are imposed by elimination (this is not covered by the proof in [32]): one simply needs to zero out some rows of \( E \), which remains consistent with the conditions of Lemma 2.1.
2.2. Analysis of element matrices. We now take a closer look at the spectral equivalence constant $\alpha_e$ for element matrices. More specifically, we consider $\alpha_e$ to be the best constant satisfying (2.5) with $A_e$ being an element matrix required by the finite element discretization of (1.2). We focus in this section on two-dimensional (2D) problems ($\Omega \subset \mathbb{R}^2$) discretized on a triangular mesh and assume that the PDE coefficient $D$ is piecewise constant with possible discontinuities located at elements’ boundaries. Although the extension of the analysis below to three-dimensional (3D) problems is not investigated, $\alpha_e$ can be computed numerically and the values obtained for regular 3D meshes are reported at the end of section 5.3 (see Table 1).

In the considered setting, entries in the element matrix $A_e$ corresponding to a mesh triangle $T_e$ are given by

$$(A_e)_{ij} = D_e \int_{T_e} (\nabla \phi_i)^T (\nabla \phi_j),$$

where $\phi_i, i = 1, \ldots, n_e$, are the restrictions of the shape functions to $T_e$ and $D_e$ is the value of the PDE coefficient on $T_e$. Further, the finite element data on $T_e$ can be determined by applying an affine transformation

$$F_e : \hat{T} \mapsto T_e : x \mapsto x_0 + J_e x$$

from a reference triangle $\hat{T}$. That is, each vertex of $\hat{T}$ is mapped on a vertex of $T_e$ and, moreover, shape functions on $T_e$ satisfy $\phi_i = \hat{\phi}_i \circ F_e^{-1}, i = 1, \ldots, n_e$, where $\hat{\phi}_i$ are the corresponding reference shape functions on $\hat{T}$. The change of variable theorem then implies

$$(A_e)_{ij} = D_e \det(J_e) \int_{\hat{T}} (J_e^{-T} \nabla \hat{\phi}_i)^T (J_e^{-T} \nabla \hat{\phi}_j).$$

The following observations further restrict the subset of relevant transformations in (2.7). First, as may be concluded from (2.8), the expression of the element matrix $A_e$ is invariant under translations and rotations; hence, it is enough to consider affine transformations to a triangle $T_e$ with one vertex being $(0,0)$ and with one edge along the $x$ axis (see Figure 1). Second, $A_e$ is also invariant under a uniform scaling: if all the edges of $T_e$ are multiplied by the same factor, $J_e$ is then multiplied by this factor and $\det(J_e)$ is divided by the square of the factor, leaving $A_e$ unchanged. As a result, it is enough to consider the affine transformation depicted in Figure 1. This transformation depends on two parameters: the angle $\theta$ of the vertex in $(0,0)$ and the translation in $x$ direction.

Fig. 1. Transformation of the reference triangle $\hat{T}$ into an arbitrary shape triangle $T_e$ with edge of length 1.
length $d$ of the adjacent edge that is not aligned with the $x$ axis. The corresponding Jacobian matrix is given by

\[(2.9)\quad J_e = \begin{pmatrix} 1 & d \cos \theta \\ d \sin \theta \end{pmatrix}.\]

With Lemma 2.2 below we provide two bounds on $\alpha_e = \alpha_e(d, \theta)$ while making no assumption on the reference shape functions $\phi_i, i = 1, \ldots, n_e$.  

**Lemma 2.2.** Let $A_e(d, \theta)$ be an element matrix defined by (2.8) with Jacobian matrix $J_e$ given by (2.9) for some $\theta$ between 0 and $\pi$. Let $\alpha_e(d, \theta)$ be the best constant such that

\[(2.10)\quad v^T A_e(d, \theta) v \geq \alpha_e(d, \theta) v^T M(A_e(d, \theta)) v \quad \forall v \in \mathbb{R}^n.\]

If $\pi/3 \leq \theta \leq \pi/2$, then

\[(2.11)\quad \alpha_e(d, \theta) \geq \min(\alpha_e(d, \pi/2), \alpha_e(d, \pi/3)).\]

If $d \leq 1$ and the triangle $T_e(d, \theta)$ is not obtuse, then

\[(2.12)\quad \alpha_e(d, \theta) \geq \min(\alpha_e(d, \pi/2), \alpha_e(d, \arccos(d))).\]

**Proof.** The proof of the first statement relies on the following relation between the element matrices for different angles:

\[(2.13)\quad A_e(d, \theta) = \frac{1 - 2 \cos \theta}{\sin \theta} A_e(d, \pi/2) + \sqrt{3} \frac{\cos \theta}{\sin \theta} A_e(d, \pi/3).\]

It is shown by setting

\[H(d, \theta) := \det(J_e)J_e^{-1}J_e^{-T} = \frac{1}{\sin \theta} \begin{pmatrix} d & -\cos \theta \\ -\cos \theta & d^{-1} \end{pmatrix}\]

and noting that the same relation holds for $H(d, \theta)$:

\[H(d, \theta) = \frac{1 - 2 \cos \theta}{\sin \theta} H(d, \pi/2) + \sqrt{3} \frac{\cos \theta}{\sin \theta} H(d, \pi/3).\]

The equality (2.13) then stems from (2.8). On the other hand, the assumption $\pi/3 \leq \theta \leq \pi/2$ implies that the weights $\beta, \gamma$ in the linear combination (2.13) are nonnegative. Hence (2.13) together with Lemma 2.1(iii) implies

\[\beta v^T M(A_e(d, \pi/2)) v + \gamma v^T M(A_e(d, \pi/3)) v \geq v^T M(A_e(d, \theta)) v \quad \forall v \in \mathbb{R}^n,\]

and therefore, setting $\alpha_{\min} = \min(\alpha_e(d, \pi/2), \alpha_e(d, \pi/3))$, there holds, for all $v \in \mathbb{R}^n$,

\[v^T A_e(d, \theta) v = \beta v^T A_e(d, \pi/2) v + \gamma v^T A_e(d, \pi/3) v \geq \beta \alpha_e(d, \pi/2) v^T M(A_e(d, \pi/2)) v + \gamma \alpha_e(d, \pi/3) v^T M(A_e(d, \pi/3)) v \geq \alpha_{\min} (\beta v^T M(A_e(d, \pi/2)) v + \gamma v^T M(A_e(d, \pi/3)) v) \geq \alpha_{\min} v^T M(A_e(d, \theta)) v,\]

where the first inequality stems from the definition (2.10) of $\alpha_e$.  


Now, setting $\theta_d = \arccos(d)$, the proof of the second statement follows the same lines and stems from the following equality:

$$A_e(d, \theta) = \frac{d - \cos \theta}{d \sin \theta} A_e(d, \pi/2) + \frac{\sin \theta_d \cos \theta}{d \sin \theta} A_e(d, \theta_d).$$

Note that if $T_e(d, \theta)$ is not obtuse, then $\theta \leq \pi/2$ and $d \geq \cos \theta$; hence $\beta, \gamma \geq 0$.  

Assume that the triangle $T_e(d, \theta)$ is not obtuse. Then the bound (2.12) holds if $\theta \geq \pi/3$, whereas the bound (2.12) holds if $d \leq 1$. These assumptions are not restrictive in the common case where the shape functions yield an element matrix $A_e(d, \theta)$ that only depends (up to a symmetric permutation) on the triangle shape and not on a particular affine mapping. Indeed, we may then choose $\theta$ as the largest angle in the triangle, thus naturally satisfying $\theta \geq \pi/3$, whereas one can select the largest of adjacent edges to be the edge aligned with the $x$ axis, entailing that $d \leq 1$.

In Figure 2, left, we depict the values of $\alpha_e(d, \pi/2)$ and the maximum of $\alpha_e(d, \pi/3)$ from (2.11) and $\alpha_e(d, \arccos(d))$ from (2.12) as a function of $d$ (with $d \leq 1$) for the Lagrangian basis functions of orders 2, 3, and 4; the minimum of both curves gives a lower bound on $\alpha_e(d, \theta)$ for nonobtuse triangles in the situation just described (triangle orientation such that $\theta \geq \pi/3$ and $d \leq 1$). For order 2 elements both curves are bounded below by 0.5, independently of the actual value of $d$; this is consistent with the known result from [4]. For elements of orders 3 and 4 both curves approach zero only when $d$ vanishes. Figure 2, right, further shows that this happens only when the smallest angle $\angle \min$ of the triangle $T_e(d, \theta)$ also vanishes. In other words, if the smallest angle in the triangulation is uniformly bounded below, $d$ and therefore $\alpha_e$ are also uniformly bounded below.

We now discuss more specifically obtuse triangles, focusing on the first order Lagrangian elements for the sake of simplicity. Indeed, while these linear elements produce M-matrices when no angle in the triangulation is larger than $\pi/2$, this property is lost as soon as there are some obtuse triangles. In Lemma 2.3 below, we provide through (2.14) an explicit expression of the corresponding spectral equivalence constant $\alpha_e(d, \theta)$. This expression is illustrated in Figure 3, left. From this figure it is clear that $\alpha_e(d, \theta)$ vanishes only if either $d \to 0$ or $\theta \to \pi$. With Figure 3, right, one sees that this happens only when the smallest angle $\angle \min$ of the triangle $T_e(d, \theta)$ tends
Fig. 3. On the left, the values of $\alpha_{e}(d, \theta)$ for first order Lagrangian element with $0 \leq d \leq 1$ and $\pi/2 \leq \theta \leq \pi$, as given by (2.14); on the right, the smallest angle in the triangle $T_{e}(d, \theta)$ as a function of $d$ and $\theta$.

to zero; that is, here again, a uniform bound on this latter entails a uniform bound on $\alpha_{e}$.

**Lemma 2.3.** Let $A_{e}(d, \theta)$ be an element matrix defined by (2.8) with $\hat{\phi}_{i}$, $i = 1, \ldots, 3$, being the first order Lagrangian basis functions on a reference triangle $\hat{T}$ and with Jacobian matrix $J_{e}$ corresponding to (2.9) for some $\theta$ between $\pi/2$ and $\pi$. Let $\alpha_{e}(d, \theta)$ be the best constant such that

$$v^{T}A_{e}(d, \theta)v \geq \alpha_{e}(d, \theta)v^{T}M(A_{e}(d, \theta))v \quad \forall v \in \mathbb{R}^{n}.$$ 

Then

$$\alpha_{e}(d, \theta) = \frac{1 - \cos^{2} \theta}{1 + \cos^{2} \theta + (d + d^{-1})|\cos \theta|}. \quad (2.14)$$

**Proof.** For simplicity we do not write the dependence $(d, \theta)$ on the triangle parameters. Setting $\sigma = d + d^{-1} - 2\cos \theta$ one obtains by direct computation

$$A_{e} = \frac{1}{2\sin \theta} \begin{pmatrix} \sigma & -d + \cos \theta & -d^{-1} + \cos \theta \\ -d + \cos \theta & d & -\cos \theta \\ -d^{-1} + \cos \theta & -\cos \theta & d^{-1} \end{pmatrix}$$

and, since $\cos \theta \leq 0$, $M(A_{e}) = A_{e} + \Delta_{e}$ with

$$\Delta_{e} = \frac{1}{2\sin \theta} \begin{pmatrix} 0 & -\cos \theta & \cos \theta \\ \cos \theta & -\cos \theta \end{pmatrix}.$$ 

Now, $\alpha_{e}$ must be such that $A_{e} - \alpha_{e}M(A_{e}) = (1 - \alpha_{e})A_{e} - \alpha_{e}\Delta_{e}$ is nonnegative definite. Since $\alpha_{e} \leq 1$ and $\sigma > 0$, this in turn requires the $2 \times 2$ Schur complement of $(1 - \alpha_{e})A_{e} - \alpha_{e}\Delta_{e}$, that is,

$$\frac{1}{2\sin \theta} \begin{pmatrix} 1 - \alpha_{e} & \frac{1 - \cos^{2} \theta}{\sigma} + \alpha_{e}\cos \theta \\ \frac{1 - \cos^{2} \theta}{\sigma} + \alpha_{e}\cos \theta & -1 \end{pmatrix},$$

to be nonnegative definite. This requirement boils down to

$$\alpha_{e} \leq \frac{1 - \cos^{2} \theta}{1 - \cos^{2} \theta - \sigma \cos \theta},$$
and injecting $\sigma = d + d^{-1} - 2\cos \theta$ into this expression together with $|\cos \theta| = -\cos \theta$ completes the proof.

3. Algebraic two-grid and multigrid schemes. Here we briefly survey the basic principles shared by most AMG methods for SPD systems, while recalling the foundation of their analyses.

We start with the description of a two-grid method which represents the building block of multigrid methods. Two-grid stationary iteration is a combination of three simpler iterations: presmoothing, coarse-grid correction, and postsmoothing. This means that the iteration matrix of a two-grid method (i.e., the matrix that multiplies the difference between the exact and the approximate solution vectors) is also a product of three simpler iteration matrices, namely,

$$T = (I - M^{-T}A)(I - P A^{-1}_c P^T A)(I - M^{-1}A).$$

In this expression, the smoother $M$ determines the pre- and postsmoothing iterations, whereas letting $n_c$ be the number of coarse unknowns, the $n \times n_c$ prolongation matrix $P$ and the $n_c \times n_c$ coarse grid matrix $A_c$ are required for the coarse grid correction. The prolongation matrix allows one to represent on the fine grid a vector defined on the coarse grid, whereas the transpose $P^T$ plays the opposite role, mapping a fine grid vector to the coarse grid; the matrix $A_c$ represents the fine grid problem on a coarse grid and, in the AMG setting, it is commonly defined by the Galerkin formula $A_c = P^T A P$.

For a two-grid method to be convergent one needs to make sure that the combination of pre- and postsmoothing iterations, namely,

$$I - \tilde{M}^{-1}A = (I - M^{-1}A)(I - M^{-T}A),$$

is itself a converging iteration; equivalently, one needs to check that the matrix

$$\tilde{M} = M^T (M + M^T - A)^{-1} M$$

is SPD. (See, e.g., [25, Lemma A.1] for the proof of this and some equivalent conditions.) This requirement is not restrictive if

- $M$ is SPD, since then $\tilde{M}$ is SPD if and only if $\lambda_{\text{max}}(M^{-1}A) < 2$; that is, if and only if a stationary iteration with $M$ is convergent;
- Gauss–Seidel smoothing is used, since then $M$ is the lower triangular part of $A$ and, because of the symmetry, $M^T$ is its upper triangular part; hence, $M + M^T - A = \text{diag}(A)$ and therefore $\tilde{M} = M^T \text{diag}(A)^{-1} M$ is always SPD.

In this work we restrict ourselves to these two families of smoothers, which in fact cover most practical uses of AMG methods.

Now, using AMG methods as a standalone solver with stationary iterations is rarely the most effective option. For SPD problems, CG method offers an efficient and inexpensive acceleration, hence it is better to use the two-grid scheme as a preconditioner for this method. The two-grid preconditioner $B_{\text{TG}}$ is defined from the iteration matrix (3.1) via the relation $I - B_{\text{TG}}^{-1}A = T$; equivalently,

$$B_{\text{TG}}^{-1} = M^{-T}(M + M^T - A) M^{-1} + (I - M^{-T}A)P A^{-1}_c P^T (I - A M^{-1}).$$

For a consistent use of the CG method, the preconditioner $B_{\text{TG}}$ has to be positive definite; again, this property holds if the smoothing iteration converges or, equivalently, if $\tilde{M}$ is SPD.
That said, the convergence depends on the interplay between the smoothing iteration and the coarse grid correction. In other words, the vectors that are not efficiently damped by the smoothing iteration should be taken care of by the coarse grid correction. The distinctive feature of AMG schemes is that the smoothing iteration is fixed to be a simple iterative method, such as Jacobi or Gauss–Seidel, and the interplay is therefore ensured by a proper choice of the coarse grid correction, that is, by a proper choice of the prolongation $P$.

Classical AMG algorithms determine $P$ essentially in two steps (see, e.g., [6, 31, 34] for more details). First, they select a subset of coarse variables in a way that favors a regular covering of the matrix graph, while ensuring that each noncoarse variable is strongly connected to at least one coarse variable. Next, interpolation weights are computed based on the matrix entries, in order to allow interpolation of noncoarse variables from the coarse ones. What connections are considered strong depends on the strong/weak coupling threshold; higher values of this threshold lead to sparser prolongation matrix $P$ and, since $A_c = P^T A P$, to a sparser coarse grid matrix, yielding a cheaper preconditioner.

On the other hand, aggregation-based AMG methods define $P$ on the basis of a partitioning of the variables set in aggregates, each aggregate corresponding to a coarse variable and vice versa. The associated prolongation $P$ is a Boolean matrix with at most one nonzero per row; see section 5.1 for details. It is thus very sparse and the corresponding coarse grid matrix has low complexity. On the negative side, it usually requires more iterations to converge than classical AMG.

Now, both types of AMG methods construct the prolongation $P$ so as to ensure small condition number $\kappa(B_{TG}^{-1} A)$. In fact, one has $\lambda_{\max}(B_{TG}^{-1} A) = 1$ (see, e.g., [25, equation (39)]), so that

$$\kappa(B_{TG}^{-1} A) = \left(\lambda_{\min}(B_{TG}^{-1} A)\right)^{-1} = \left(1 - \rho(T)\right)^{-1},$$

where $\rho(T)$ is the spectral radius of the iteration matrix (3.1). Hence minimizing the two-grid condition number (3.4) is actually equivalent to minimizing the convergence rate of two-grid stationary iterations.

Practical multigrid algorithms result from a recursive application of a two-grid method. Indeed, two-grid schemes remain costly because they rely on an exact solution of a coarse-grid system; this is highlighted by the term $A_c^{-1}$ in (3.1) or (3.3). Instead, multigrid methods solve the coarse grid system approximately, applying few iterations of a two-grid scheme which now involves an even coarser grid. This approach is followed recursively until the coarse grid matrix is small enough to be factorized at negligible cost.

Classical AMG algorithms typically use the V-cycle; that is, $A_c^{-1}$ in (3.3) is replaced by one application of the preconditioner at the corresponding coarse level. On the other hand, aggregation-based AMG methods typically use the K-cycle [26, 28, 29], although the more complex AMLI-cycle was considered in [22], mainly for theoretical reasons. With the K-cycle, the system to be solved with $A_c$ (to obtain the action of $A_c^{-1}$ in (3.3)) is approximately solved with two inner CG iterations at each level, using the two-grid preconditioner at the corresponding coarse level.

In practice the choice of the cycle is motivated by the desire to preserve two-grid convergence without significantly deteriorating the preconditioner’s cost. Classical

---

2Because this makes the preconditioner slightly variable from one step to the next, the flexible variant of CG [24] is used for both inner and outer iterations.
AMG preconditioners approximate the coarse grid matrix fairly well but often at a high cost. Hence it is unnecessary to use a more complex algorithm than the V-cycle, and doing so could further make the cost issue more critical. On the other hand, aggregation-based AMG preconditioners tend to be cheaper but approximate the matrix less closely. It is then mandatory to use the K-cycle to obtain a convergence that remains independent of the number of levels.

Now, a truly multilevel analysis is often out of reach for AMG methods (see [22, 36] for notable exceptions). But two-grid analysis is relatively standard for both classical and aggregation-based AMG, at least for the class of M-matrices. It amounts to proving upper bounds on $\kappa(B_{TG}^{-1}A)$ with $B_{TG}$ as in (3.3), and most available results can be derived from the following identity [12, 38]:

$$\kappa(B_{TG}^{-1}A) = \max_v \frac{v^T \tilde{M} \left(I - P(P^T \tilde{M} P)^{-1} P^T \tilde{M}ight) v}{v^T A v}$$

(even though results in, e.g., [5, 31, 34] were historically proved otherwise). In general, $\tilde{M}$ is too complicated to allow a direct analysis of this expression. However, if a simpler SPD matrix $\mathcal{M}$ is known such that

$$\sigma_{\mathcal{M}} = \lambda_{\min}(\tilde{M}^{-1} \mathcal{M})$$

is not too small, one can use instead the upper bound (see [38, Corollary 3.20] or [27, Corollary 2.2])

$$\kappa(B_{TG}^{-1}A) \leq \sigma_{\mathcal{M}}^{-1} \max_v \frac{v^T \mathcal{M} \left(I - P(P^T \mathcal{M} P)^{-1} P^T \mathcal{M}ight) v}{v^T A v}.$$

When $\mathcal{M} = \text{diag}(A)$, $\sigma_{\mathcal{M}}$ is denoted $\sigma$ and is better known as the constant appearing in the smoothing property. (See [31, 34] for the equivalence between (3.6) and the classical formulation of the smoothing property.) It is well known that the Gauss–Seidel smoother satisfies it with a reasonably large constant $\sigma$ [5, 31, 34].

In the following two sections we revisit the analysis of both classical AMG (section 4) and aggregation-based AMG (section 5) and explain why these methods have nice two-grid convergence when applied to matrices satisfying the spectral equivalence relation (1.3) with reasonable $\alpha$.

4. Analysis of classical AMG.

4.1. General framework. As already stated, classical AMG methods build $P$ in two steps. First, a subset of the variables is chosen to form the coarse variables. Let $n_c$ be the number of coarse variables and assume for ease of presentation that these variables are ordered last, i.e., that they correspond to the unknowns with indices $n - n_c + 1, \ldots, n$. The second step amounts to building the prolongation of the form

$$P = \begin{pmatrix} J_{FC} \\ I_{n_c} \end{pmatrix},$$

where $J_{FC}$ corresponds to the interpolation weights of noncoarse variables from the coarse ones.

Regarding the auxiliary matrix $\mathcal{M}$, it is chosen to be

$$\mathcal{M} = \text{diag}(A) = \begin{pmatrix} D_{FF} \\ D_{CC} \end{pmatrix}.$$
The following useful inequality is then known to hold for any $J_{FC}$ and any $v^T = (v^T_F, v^T_C)^T$:

\begin{equation}
(4.2) \quad v^T \overline{M} (I - P (P^T \overline{M} P)^{-1} P^T \overline{M}) v \leq (v^T_F - J_{FC} v^T_C)^T D_{FF} (v^T_F - J_{FC} v^T_C).
\end{equation}

(See [27, Corollary 2.2] with $X = Y = \overline{M}$ and $Q$ such that $Q v = (0^T, v^T_C)^T$.)

\section{4.2. M-matrices with nonnegative row-sum.}

We now state the essential of the standard two-grid analysis as developed in [5, 31, 34]. For this, we do not need to enter the details of the coarse variable selection and prolongation weight computation; we only need to know that these latter yield prolongation (4.1) such that for all $v = (v^T_F, v^T_C)^T$

\begin{equation}
(4.3) \quad (v^T_F - J_{FC} v^T_C)^T D_{FF} (v^T_F - J_{FC} v^T_C) \leq \tau v^T A v
\end{equation}

holds for a reasonably small constant $\tau$ (see [31, Theorem 5.2] or [34, Theorem A.4.3]). Hence, taking (3.7) and (4.2) into account, they ensure that

\begin{equation}
(4.4) \quad \kappa (B^{-1}_{TG} A) \leq \frac{\tau}{\sigma}.
\end{equation}

where $\sigma$ is the “smoothing property” constant (3.6) with respect to $\overline{M} = \text{diag}(A)$. We do not give the exact definition of $\tau$ because classical AMG analysis is intended to be qualitative and $\tau/\sigma$ in general strongly overestimates the true condition number.

\section{4.3. General SPD matrices with nonnegative row-sum.}

For this class of matrices, the basic AMG strategy amounts to defining $P$ ignoring positive offdiagonal entries, except that they are lumped to the diagonal; that is, it defines $P$ based on $\mathcal{M}(A)$ rather than $A$. Hence, instead of (4.3) one has now

\begin{equation}
(4.5) \quad (v^T_F - J_{FC} v^T_C)^T D^{(M)}_{FF} (v^T_F - J_{FC} v^T_C) \leq \tau v^T \mathcal{M}(A) v
\end{equation}

where $D^{(M)}_{FF}$ is the upper left block of $\overline{M}^{(M)} = \text{diag}(\mathcal{M}(A))$. Observing that the entries in $\overline{M}^{(M)}$ are at least as large as in $\overline{M}$, one then straightforwardly obtains

\begin{equation}
(4.6) \quad (v^T_F - J_{FC} v^T_C)^T D_{FF} (v^T_F - J_{FC} v^T_C) \leq \tau v^T \mathcal{M}(A) v \leq \frac{\tau}{\sigma} v^T A v,
\end{equation}

where the last inequality stems from (1.3). Hence, taking (3.7) and (4.2) into account, this yields

\begin{equation}
(4.7) \quad \kappa (B^{-1}_{TG} A) \leq \frac{\tau}{\alpha \sigma}.
\end{equation}

In [5, 34], a similar analysis is developed based on the concept of essentially positive-type matrices. Recast in our notation, it amounts to defining the matrix $A_0$ as the matrix with the same offdiagonal entries as $A$ but zero row-sum. Then, $A$ is said to be of essentially positive type if, for all $v \in \mathbb{R}^n$,

\begin{equation}
(4.8) \quad v^T A_0 v \geq c v^T \mathcal{M}(A_0) v
\end{equation}
holds for a reasonably large constant $c$, and, using (4.5), it is further shown that 
\[ \kappa \left( B_{TG}^{-1} A \right) \leq \tau/(c \sigma). \]
This is in fact slightly worse than (4.6) because $\alpha \geq c$, as seen by letting $\Delta$ be the (nonnegative) diagonal matrix such that $A = A_0 + \Delta$: one has $\mathcal{M}(A) = \mathcal{M}(A_0) + \Delta$ and hence (4.7) implies that (1.3) holds with $\alpha \geq c$. The improvement is, however, marginal in the context of the PDE (1.2) since the row-sum, and hence the corresponding diagonal entries in $\Delta$, are zero everywhere except possibly near boundaries, implying $c \approx \alpha$. Yet our analysis builds a bridge between the “essentially positive type concept” and approaches based on $\mathcal{M}(A)$ while allowing a straightforward use of the results in section 2.

4.4. Recursive use. As written in section 3, multigrid schemes are based on the recursive use of a two-grid method, hence it is relevant to check whether the assumptions used in the analysis also hold at coarser levels. In Theorem 5.6 of [31], it is shown that when $A$ is an M-matrix, then $A_c = P^T A P$ is often an M-matrix as well, although this is not always true, depending on the interpolation scheme. It is less clear how the constant $\alpha$ in (1.3) can evolve when going to coarser levels. In practice, however (as will be seen in section 6), with classical AMG schemes, the main issue associated with the recursive use is the complexity and not the quality of the two-grid approximation.

5. Analysis of aggregation-based AMG.

5.1. General framework. In this section we consider aggregation-based AMG methods as developed in [21, 22, 26, 28]. They determine the prolongation $P$ through the agglomeration of the unknowns into $n_c$ nonempty disjoint sets $G_k, k = 1, \ldots, n_c$, called aggregates. To each aggregate $G_k$ is associated one unknown at the next coarse level in the hierarchy. In addition, some unknowns also can be kept outside the coarsening process, and the corresponding (possibly empty) set is noted $G_0$; that is, $G_0$ gathers the unknowns that are not associated to any coarse unknown. Thus $G_k, k = 0, 1, \ldots, n_c$, is a partitioning of $[1, n]$, which determines $P$ via

\[
(P)_{ij} = \begin{cases} 
1 & \text{if } i \in G_j, \\
0 & \text{otherwise}
\end{cases}
\]

for $i = 1, \ldots, n$ and $j = 1, \ldots, n_c$. Hence a row of $P$ is zero if and only if the corresponding unknown is in $G_0$, whereas the other rows have exactly one nonzero entry. As made clearer below, the role of $G_0$ is to gather nodes that need not be represented on the coarse grid because the corresponding error components are sufficiently damped thanks to the sole action of the smoother.

In [21], a general bound is obtained from (3.7) for any positive definite matrix $\overline{M}$ that is block diagonal with respect to the partitioning in aggregates, that is, of the form

\[
\overline{M} = \begin{pmatrix}
M_{G_0} & & \\
& M_{G_1} & \\
& & \ddots \\
& & & M_{G_{n_c}}
\end{pmatrix},
\]

where, for ease of presentation, we assume that the unknowns are ordered consistently with respect to the partitioning in aggregates: the unknowns in $G_1$ have larger indices than those in $G_0$ and smaller indices than those in $G_2$, etc.
Another auxiliary block diagonal matrix required by the analysis is a symmetric nonnegative definite matrix

$$A_b = \begin{pmatrix}
A_{G_0} & & \\
& A_{G_1} & \\
& & \ddots \\
& & & A_{G_{nc}}
\end{pmatrix},$$

which must satisfy the following relation with the system matrix $A$:

$$v^T A v \geq \xi v^T A_b v \quad \forall v \in \mathbb{R}^n,$$

where $\xi$ is a positive constant. (Formally, the theory in [21] is stated only for $\xi = 1$, but the extension is straightforward.)

With these matrices in hand, one can associate to each aggregate a quality measure $\mu_k$, defined as

$$\mu_0 = \begin{cases}
0 & \text{if } G_0 \text{ is empty}, \\
\max_v \frac{v^T M_{G_0} v}{v^T A_{G_0} v} & \text{otherwise},
\end{cases}$$

and, for $k = 1 \ldots, n_c$,

$$\mu_k = \begin{cases}
0 & \text{if } |G_k| = 1, \\
\sup_{v \in \mathcal{N}(A_{G_k})} \frac{v^T M_{G_k} (I - 1_{G_k} (1_{G_k} M_{G_k} 1_{G_k})^{-1} 1_{G_k} M_{G_k}) v}{v^T A_{G_k} v} & \text{otherwise},
\end{cases}$$

where $|G_k|$ is the $k$th aggregate size and $1_{G_k} = (1,1,\ldots,1)^T$ is the vector of all ones of size $|G_k|$. One then has (combining [21, Theorem 3.2] with (3.7))

$$\kappa \left( B_{TC}^{-1} A \right) \leq \max_{k=0, \ldots, n_c} \frac{\mu_k}{\xi \sigma_M}.$$

5.2. M-matrices with nonnegative row-sum. When $A = (a_{ij})$ is an M-matrix with nonnegative row sum, it is possible to construct both auxiliary matrices $M$ and $A_b$ in a systematic way. First, one extracts the block diagonal part $A_d$ of $A$, except for what concerns the unknowns in $G_0$, for which $A_d$ gathers only the diagonal element; that is,

$$(A_d)_{ij} = \begin{cases}
a_{ij} & \text{if either } i = j \\
0 & \text{or both } i, j \text{ belong to the same aggregate } G_k, k = 1, \ldots, n_c,
\end{cases}$$

Next one computes

$$t = (A_d - A) 1$$
and sets

\begin{align}
\mathcal{M} &= A_d + \text{diag}(t_i), \\
A_b &= A_d - \text{diag}(t_i),
\end{align}

Such $A_b$ has the same row-sum as $A$, which is nonnegative by assumption. Hence $A_b$ is weakly diagonally dominant and therefore nonnegative definite as required, whereas $A - A_b$ has zero row sum and is therefore also weakly diagonally dominant, ensuring that (5.4) holds with $\xi = 1$. The upper bound (5.7) then reduces to

\begin{equation}
\kappa \left( B_{TG}^{-1} A \right) \leq \frac{\max_{k=0, \ldots, n_c} \mu_k}{\sigma_{\mathcal{M}}},
\end{equation}

where $\sigma_{\mathcal{M}}$ depends on the smoother. The numerical examples below indicate that it is only slightly smaller than 1 for the Gauss–Seidel smoother.

It is worth observing that each diagonal block of $\mathcal{M}$ and $A_b$ defined from (5.8), (5.9), (5.10), (5.11) can be computed using only the entries that are “local” to the aggregate, together with the global row-sum of the matrix that can be computed once for all. In that sense, $\mu_k$ in (5.6) is a local measure of the aggregate’s quality, which, nevertheless, via (5.12), gives an indication of the global behavior of the method. On the other hand, the blocks in $\mathcal{M}$ and $A_b$ corresponding to nodes in $G_0$ are diagonal, and (5.5) amounts to

\begin{equation}
\mu_0 = \begin{cases} 
0 & \text{if } G_0 \text{ is empty,} \\
\max_{i \in G_0} \frac{a_{ii} + \sum_{j \neq i} |a_{ij}|}{a_{ii} - \sum_{j \neq i} |a_{ij}|} & \text{otherwise.}
\end{cases}
\end{equation}

Thus one can put safely in $G_0$ the unknowns corresponding to rows in $A$ that are strongly dominated by the diagonal element. Intuitively, the corresponding components of the error are sufficiently damped with the sole action of the smoother, and hence these unknowns do not need to be represented on the coarse grid.

**Quality aware aggregation.** In [22], the two-grid analysis just sketched is at the basis of an aggregation algorithm that builds the aggregates in such a way that the quality indicators $\mu_k$, $k = 0, \ldots, n_c$, are as small as possible and always below an a priori chosen threshold $\overline{\mu}$. In view of the results recalled above, nothing more is needed to guarantee that the two-grid condition number is not larger than $\kappa/\sigma_{\mathcal{M}}$.

### 5.3. General SPD matrices with nonnegative row-sum

Consider now that $A$ is just SPD with nonnegative row-sum; i.e., $A$ has both positive and negative offdiagonal entries. It is nevertheless still possible to define $\mathcal{M}$ and $A_b$ using exactly the same rules (5.8), (5.9), (5.10), (5.11).

One seemingly serious shortcoming is that $A_b$ is here not necessarily nonnegative definite. However, this is not a real issue if, as in the strategy explained below, $\mu_k$ is used to assess aggregates' quality and, in particular, to reject aggregates that are not “good enough,” using the nonnegative definiteness of the corresponding block in $A_b$ as preliminary acceptance criterion. In that way, $A_b$ will always be nonnegative definite, not because this is guaranteed for any partitioning in aggregates, but because only those satisfying this requirement are declared admissible for further assessment.

With this precaution the bound (5.7) applies, but is useful only if $\xi$ in (5.4) does not become too small. In the following theorem, we show that it cannot be smaller than the constant $\alpha$ in (1.3). The second statement will be used in section 5.4.
Theorem 5.1. Let $A = (a_{ij})$ be a symmetric matrix. Let $A_b$ be the matrix defined by (5.8), (5.9), (5.11). Let $M(A)$ be the matrix defined by (2.1), (2.2), and let $\alpha$ be a positive number such that (1.3) holds. One then has

$$v^T A v \geq \alpha v^T A_b v \quad \forall v \in \mathbb{R}^n.$$  \hfill (5.14)

Moreover, if $P$ is an $n \times n_c$ matrix of the form (5.1), $A_c = P^T A P$ is such that

$$\alpha v^T M(A_c) v \leq v^T A_c v \quad \forall v \in \mathbb{R}^n.$$  \hfill (5.15)

Proof. First, it follows from $(A - A_b)1 = 0$ that $M(A - A_b)$ is symmetric non-negative definite. Then, (5.14) is implied by (1.3) together with

$$v^T M(A) v = v^T (M(A_b) + M(A - A_b)) v \geq v^T M(A_b) v \geq v^T A_b v,$$

where the last inequality stems from Lemma 2.1(i), whereas the first equality holds by virtue of (5.8) and (5.11), which imply that every nonzero offdiagonal entry of $A$ is present in either $A_b$ or $A - A_b$, but never in both. On the other hand, we have

$$v^T P^T A P v \geq \alpha v^T P^T M(A) P v \geq \alpha v^T M(P^T A P) v = \alpha v^T M(A_c) v,$$

the first inequality following from (1.3) and the second one from Lemma 2.1(ii).

Quality aware aggregation. The idea is here to use without any modification the aggregation algorithm from [22] which was outlined in the previous subsection. This is consistent with the above discussion because this algorithm effectively keeps the aggregates’ quality below $\bar{\kappa}$, with $\mu_k$ defined with (5.5), (5.6) and $A_b$, $M$ constructed according to (5.8), (5.9), (5.10), (5.11). Moreover, the way the check $\mu_k \leq \bar{\kappa}$ is implemented naturally rejects any aggregate for which the corresponding diagonal block in $A_b$ would not be nonnegative definite. As a result, the upper bound (5.7) applies and further reduces to

$$\kappa(B_{TG}^{-1} A) \leq \frac{\bar{\kappa}}{\xi \sigma M}.$$  \hfill (5.16)

Parameters entering the above upper bound are given in Table 1 (with $\kappa$ standing for $\kappa(B_{TG}^{-1} A)$) for the two-grid method implemented in the AGMG software package [23]; it uses Gauss–Seidel smoothing and the just presented aggregation strategy with default threshold $\bar{\kappa} = 8$. The problem under consideration is the model Poisson problem corresponding to the PDE (1.2) with $D = 1$ on the unit square (2D) or cube (3D), using Neumann boundary conditions everywhere except on the top boundary where Dirichlet boundary conditions are imposed by eliminations. P1, P2, P3, and P4 stands for, respectively, first, second, third, and fourth order Lagrangian finite elements on a regular grid of right triangles (2D) or right angle tetrahedra (3D).

Observe that the parameter $\xi$ remains far from its lower bound $\alpha \approx \alpha_e$, where $\alpha_e$ is the constant computed at elements’ level. Moreover, increasing the finite element order, the condition number $\kappa(B_{TG}^{-1} A)$ stays roughly proportional to $\xi^{-1}$. Yet, $\kappa(B_{TG}^{-1} A)$ never actually reaches even the threshold $\bar{\kappa} = 8$.

We present in Table 2 the relevant parameters for the alternative strategy mentioned in section 1 where the preconditioner is instead computed from $M(A)$; in this
Table 1
Two-grid parameters for the aggregation-based AMG method directly applied to the system matrix $A$; $\kappa$ stands for $\kappa(B_{TC}^{-1}A)$.

<table>
<thead>
<tr>
<th>FE</th>
<th>$\alpha$</th>
<th>$\xi$</th>
<th>$\sigma^{-1}_{TM}$</th>
<th>$\kappa$</th>
<th>$\xi$</th>
<th>$\sigma^{-1}_{TM}$</th>
<th>$\kappa$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Poisson 2D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>n = $1.3 \times 10^3$</td>
<td>n = $5.3 \times 10^3$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P1</td>
<td>1.00</td>
<td>1.00</td>
<td>1.20</td>
<td>2.1</td>
<td>1.00</td>
<td>1.20</td>
<td>2.1</td>
</tr>
<tr>
<td>P2</td>
<td>0.50</td>
<td>0.70</td>
<td>1.52</td>
<td>2.6</td>
<td>0.70</td>
<td>1.52</td>
<td>2.6</td>
</tr>
<tr>
<td>P3</td>
<td>0.31</td>
<td>0.50</td>
<td>1.74</td>
<td>3.3</td>
<td>0.49</td>
<td>1.74</td>
<td>3.3</td>
</tr>
<tr>
<td>P4</td>
<td>0.11</td>
<td>0.32</td>
<td>2.21</td>
<td>6.4</td>
<td>0.32</td>
<td>2.21</td>
<td>7.3</td>
</tr>
<tr>
<td>Poisson 3D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>n = $2.0 \times 10^3$</td>
<td>n = $1.5 \times 10^4$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P1</td>
<td>1.00</td>
<td>1.00</td>
<td>1.13</td>
<td>1.9</td>
<td>1.00</td>
<td>1.13</td>
<td>2.0</td>
</tr>
<tr>
<td>P2</td>
<td>0.25</td>
<td>0.61</td>
<td>1.54</td>
<td>2.2</td>
<td>0.61</td>
<td>1.54</td>
<td>2.3</td>
</tr>
<tr>
<td>P3</td>
<td>0.09</td>
<td>0.41</td>
<td>2.32</td>
<td>3.0</td>
<td>0.41</td>
<td>2.32</td>
<td>3.3</td>
</tr>
<tr>
<td>P4</td>
<td>0.04</td>
<td>0.30</td>
<td>3.07</td>
<td>4.2</td>
<td>0.28</td>
<td>3.07</td>
<td>4.6</td>
</tr>
</tbody>
</table>

Table 2
Two-grid parameters for the aggregation-based AMG method, when applied to $M(A)$ to define the preconditioner for the system matrix $A$; $\kappa^{(M)}$ stands for $\kappa(B_{TC}^{-1}M(A))$ and $\kappa$ for $\kappa(B_{TC}^{-1}A)$.

<table>
<thead>
<tr>
<th>FE</th>
<th>$\alpha$</th>
<th>$\sigma^{-1}_{TM}$</th>
<th>$\kappa^{(M)}$</th>
<th>$\kappa$</th>
<th>$\sigma^{-1}_{TM}$</th>
<th>$\kappa^{(M)}$</th>
<th>$\kappa$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Poisson 2D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>n = $1.3 \times 10^3$</td>
<td>n = $5.3 \times 10^3$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P1</td>
<td>1.0</td>
<td>1.20</td>
<td>2.1</td>
<td>2.1</td>
<td>1.20</td>
<td>2.1</td>
<td>2.1</td>
</tr>
<tr>
<td>P2</td>
<td>2.0</td>
<td>1.20</td>
<td>2.4</td>
<td>2.7</td>
<td>1.20</td>
<td>2.4</td>
<td>2.7</td>
</tr>
<tr>
<td>P3</td>
<td>3.3</td>
<td>1.14</td>
<td>2.8</td>
<td>4.4</td>
<td>1.14</td>
<td>2.9</td>
<td>4.5</td>
</tr>
<tr>
<td>P4</td>
<td>9.4</td>
<td>1.06</td>
<td>1.8</td>
<td>11.3</td>
<td>1.07</td>
<td>1.8</td>
<td>11.4</td>
</tr>
<tr>
<td>Poisson 3D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>n = $2.0 \times 10^3$</td>
<td>n = $1.5 \times 10^4$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P1</td>
<td>1.0</td>
<td>1.13</td>
<td>1.9</td>
<td>1.9</td>
<td>1.13</td>
<td>2.0</td>
<td>2.0</td>
</tr>
<tr>
<td>P2</td>
<td>3.9</td>
<td>0.97</td>
<td>1.8</td>
<td>4.3</td>
<td>0.97</td>
<td>1.8</td>
<td>4.3</td>
</tr>
<tr>
<td>P3</td>
<td>11.0</td>
<td>0.99</td>
<td>1.3</td>
<td>10.6</td>
<td>0.99</td>
<td>1.3</td>
<td>10.6</td>
</tr>
<tr>
<td>P4</td>
<td>26.9</td>
<td>0.92</td>
<td>1.2</td>
<td>26.5</td>
<td>0.92</td>
<td>1.2</td>
<td>26.5</td>
</tr>
</tbody>
</table>

table $\kappa^{(M)}$ stands for $\kappa(B_{TC}^{-1}M(A))$ and $\kappa$ for $\kappa(B_{TC}^{-1}A)$. One sees that the former quantity benefits from the fact that the preconditioning strategy is applied to an M-matrix and hence does not vary much—or even improves—when increasing the finite element order. However, the actual condition number behaves essentially like indicated in (4.1), i.e., is roughly proportional to $\alpha^{-1}$.

Comparing the results in both tables we note that constructing the preconditioner based on $A$ leads to a better condition number than when the intermediate matrix $M(A)$ is used. This is mainly explained by the fact that $\xi$ decreases significantly slower than $\alpha$ with the finite element order. More numerical results are reported in section 5.

5.4. Recursive use. It is well known that if $A$ is an M-matrix with nonnegative row-sum, then the successive coarse grid matrices obtained by aggregation always share these properties; see, e.g., [28, Lemma 3.1] for a proof. With inequality (5.15)
of Theorem 5.1, one further sees that, more generally, the constant $\alpha$ in (1.3) at some level is a lower bound on $\alpha$ at the next coarser level; that is, the initial (fine grid) constant $\alpha$ provides a lower bound on $\xi$ at every level, showing that our analysis and the related aggregation strategy are consistent with a recursive application of the latter to define a hierarchy of coarse grids.

6. Numerical results. We did not present numerical two-grid results for classical AMG, but it is clear (and will be also illustrated below) that this method provides preconditioners that actually approximate fairly well the matrix they are computed from, in both M-matrix and non-M-matrix cases. Potential issues are rather associated with the computational cost: for the method to be efficient, the number of selected coarse variables should be a small enough fraction of the overall number of variables to allow an effective reduction of the grid size from one level to the next; it is also crucial to have the interpolation $J_{FC}$ in (4.1) sparse enough so that the number of nonzero entries in $A_c = P^T A P$ decreases proportionally to its size.

The aggregation-based method from [22, 28] may similarly become inefficient if the aggregation strategy fails to form aggregates large enough. The implementation considered here targets producing aggregates of size 4, to obtain at each level a reduction of the number of unknowns by a factor of 4. But the “quality control” mentioned in section 5 implies that “bad” aggregates are rejected, and this may impact the computational cost of the method if this occurs too often.

Figure 4 shows the size ($n$) and the number of nonzero entries ($\#nz$) of coarse grid matrices at different levels of the multigrid hierarchy for the classical AMG method as implemented in Boomer AMG (Bo-AMG) [15], and the aggregation-based AMG method as implemented in AGMG [23]; default options were used in both cases, except that, for Boomer AMG, the strong/weak coupling threshold was set to 0.5 for the 3D problem as recommended in the user guide, whereas AGMG was told that the matrix is symmetric. The matrix stems from the discretization with third order Lagrangian finite elements of the model Poisson problem specified at the end of section 5.3.

Regarding the decrease in size, both methods behave satisfactorily; the aggregation-based method provides faster reduction but, as mentioned in section 3,
Fig. 5. Coarsening at different levels for the model Poisson 2D problem discretized with third order Lagrangian (P3) elements; initial system matrix connectivity graph (top left); first, second, and third level aggregates superposed with the corresponding connectivity graph (top right, bottom left and right, respectively); light gray (green) boxes correspond to “regular” aggregates, whereas dark (red) boxes correspond to nodes in $G_0$, which are therefore no longer represented on the subsequent grids.

This is at the expense of the quality of the preconditioner, with the consequence that the more involved K-cycle has to be used instead of the V-cycle. Hence, the aggregation method requires faster reduction to achieve similar computational cost per iteration step. However, the situation is more critical for Boomer AMG regarding the evolution of the number of nonzero entries, especially in the 3D example: one has to reach the fifth level to obtain a number of nonzero entries significantly less than in the fine grid matrix. On the contrary, with AGMG, the reduction of the number of nonzero entries is even faster than that of the size. To better understand this behavior, we depict in Figure 5 how the aggregation takes place for a smaller version of this problem. One sees that the connectivity pattern becomes simpler as the aggregation proceeds: only connections with the nearest neighbors subsist once the aggregates are large enough to contain all nodes belonging to a same element in the initial triangulation.
We now consider the performances in terms of number of iterations and computing time on the following more realistic test problems.

Jump 2D/3D is the discretization of the PDE (1.2) on the unit square (2D) or cube (3D), with Neumann boundary conditions everywhere, except on the top boundary where Dirichlet boundary conditions are imposed by eliminations. The diffusion coefficient $D$ is equal to 1 in the lower half of the domain and to $10^3$ in the upper half. A regular grid is used with right triangles (2D) or right angle tetrahedra (3D).

Lshape is the discretization of the PDE (1.2) with $D = 1$ and Dirichlet boundary conditions on an L-shaped domain $\Omega = [-1,1]^2 \setminus [0,1] \times [-1,0]$. The grid is unstructured and locally refined near the reentering corner, in such a way the simplex size is progressively decreased to be in its neighborhood about $10^4$ time smaller than near the other corners.

Sphere is the discretization of the PDE (1.2) on the unit cube with Dirichlet boundary conditions; $D = 1$ everywhere except within a small sphere of radius $1/25$ at the center of the domain, where $D = 10^3$. The grid is unstructured and locally refined near the surface of the sphere, where simplices are about 10 times smaller than in the major part of the domain.

The above test suite includes 2D and 3D problems on both structured and unstructured grids. Each problem is discretized using first (P1), second (P2), third (P3), and fourth (P4) order Lagrangian finite elements. The size of the resulting matrices and the corresponding number of nonzero entries are as indicated in Table 3; the problems discretized on regular grids have two different sizes.

Results for both strategies considered in this work are presented in Tables 4 and 5, where we consider, respectively, the direct application of AMG methods to the system matrix $A$ and the use of the AMG preconditioner constructed from $\mathcal{M}(A)$. $\mathcal{T}_{\text{su}}$ stands for the setup time, i.e., the time needed to compute the preconditioner by constructing the hierarchy of coarse grids. $\mathcal{T}_{\text{sol}}$ is the solution time, i.e., the time to solve the linear system; in all cases, CG acceleration was used with the stopping criterion being a $10^{-6}$ reduction in the residual norm. $\mathcal{T}_{\text{tot}}$ is the total time, i.e., $\mathcal{T}_{\text{tot}} = \mathcal{T}_{\text{su}} + \mathcal{T}_{\text{sol}}$. All these times are elapsed wall clock times reported in seconds per $10^6$ unknowns, as obtained running the software codes on a computing node with two Intel XEON L5420 processors at 2.50GHz and 16Gb RAM memory.

These results were obtained running Boomer AMG [15] (from the HYPRE package [17], version 2.7.0, written in C) and AGMG [23] (version 3.2.0, written in Fortran 90) software codes in black box fashion with the default setting, except

---

Table 3

<table>
<thead>
<tr>
<th>Problem</th>
<th>P1</th>
<th>P2</th>
<th>P3</th>
<th>P4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>n10^6</td>
<td>n</td>
<td>n10^6</td>
<td>n</td>
</tr>
<tr>
<td>Jump 2D</td>
<td>1.3-5.5</td>
<td>5.0</td>
<td>1.3-5.5</td>
<td>6.0</td>
</tr>
<tr>
<td></td>
<td>7.1</td>
<td>7.0</td>
<td>7.1</td>
<td>11.5</td>
</tr>
<tr>
<td>Jump 3D</td>
<td>0.9-3.0</td>
<td>6.9</td>
<td>0.9-3.0</td>
<td>22.9</td>
</tr>
<tr>
<td>Sphere</td>
<td>1.2</td>
<td>14.7</td>
<td>1.2</td>
<td>28.4</td>
</tr>
</tbody>
</table>

---

According to results reported in [33], the presence of such discontinuity may raise convergence problems for a method [19] based on smoothed aggregation AMG.
### Table 4

Results for the AMG methods applied directly to the system matrix $A$; for Jump 2D/3D, the first (second) line of results for each finite element order corresponds to the smallest (largest) of the tested sizes; reported times are elapsed times in seconds per $10^6$ unknowns.

<table>
<thead>
<tr>
<th></th>
<th>Boomer AMG</th>
<th></th>
<th>AGMG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>S/W = 0.25</td>
<td>S/W = 0.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>It $T_{su}$ $T_{sol}$ $T_{tot}$</td>
<td>It $T_{su}$ $T_{sol}$ $T_{tot}$</td>
<td></td>
</tr>
</tbody>
</table>

#### Jump 2D

<table>
<thead>
<tr>
<th>Order</th>
<th>$P_1$</th>
<th>$P_2$</th>
<th>$P_3$</th>
<th>$P_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5</td>
<td>4</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>It</td>
<td>5.2</td>
<td>3.3</td>
<td>7.3</td>
<td>12.5</td>
</tr>
<tr>
<td>$T_{su}$</td>
<td>2.6</td>
<td>2.3</td>
<td>5.5</td>
<td>21.0</td>
</tr>
<tr>
<td>$T_{sol}$</td>
<td>5.8</td>
<td>6.0</td>
<td>6.1</td>
<td>33.5</td>
</tr>
<tr>
<td>$T_{tot}$</td>
<td>8.2</td>
<td>8.4</td>
<td>8.3</td>
<td>44.0</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>11</td>
<td>11</td>
<td>18</td>
</tr>
<tr>
<td>It</td>
<td>0.8</td>
<td>0.8</td>
<td>0.8</td>
<td>0.8</td>
</tr>
<tr>
<td>$T_{su}$</td>
<td>2.3</td>
<td>2.6</td>
<td>2.6</td>
<td>2.5</td>
</tr>
<tr>
<td>$T_{sol}$</td>
<td>3.1</td>
<td>3.4</td>
<td>3.4</td>
<td>13.9</td>
</tr>
<tr>
<td>$T_{tot}$</td>
<td>3.1</td>
<td>3.4</td>
<td>3.4</td>
<td>13.9</td>
</tr>
</tbody>
</table>

#### Lshape

<table>
<thead>
<tr>
<th>Order</th>
<th>$P_1$</th>
<th>$P_2$</th>
<th>$P_3$</th>
<th>$P_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4</td>
<td>7</td>
<td>7</td>
<td>14</td>
</tr>
<tr>
<td>It</td>
<td>4.3</td>
<td>5.3</td>
<td>6.4</td>
<td>14.8</td>
</tr>
<tr>
<td>$T_{su}$</td>
<td>2.9</td>
<td>6.3</td>
<td>8.4</td>
<td>20.8</td>
</tr>
<tr>
<td>$T_{sol}$</td>
<td>7.2</td>
<td>11.6</td>
<td>14.8</td>
<td>35.6</td>
</tr>
<tr>
<td>$T_{tot}$</td>
<td>9.9</td>
<td>14.8</td>
<td>30.4</td>
<td>44.0</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>9</td>
<td>12</td>
<td>13</td>
</tr>
<tr>
<td>It</td>
<td>0.9</td>
<td>0.9</td>
<td>0.9</td>
<td>1.9</td>
</tr>
<tr>
<td>$T_{su}$</td>
<td>2.1</td>
<td>4.9</td>
<td>7.3</td>
<td>10.0</td>
</tr>
<tr>
<td>$T_{sol}$</td>
<td>2.9</td>
<td>6.2</td>
<td>9.2</td>
<td>12.5</td>
</tr>
<tr>
<td>$T_{tot}$</td>
<td>2.9</td>
<td>6.2</td>
<td>9.2</td>
<td>12.5</td>
</tr>
</tbody>
</table>

#### Jump 3D

<table>
<thead>
<tr>
<th>Order</th>
<th>$P_1$</th>
<th>$P_2$</th>
<th>$P_3$</th>
<th>$P_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>6</td>
<td>6</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>It</td>
<td>16.9</td>
<td>17.6</td>
<td>28.2</td>
<td>57.9</td>
</tr>
<tr>
<td>$T_{su}$</td>
<td>6.1</td>
<td>6.2</td>
<td>6.3</td>
<td>6.4</td>
</tr>
<tr>
<td>$T_{sol}$</td>
<td>21.0</td>
<td>23.9</td>
<td>63.3</td>
<td>85.7</td>
</tr>
<tr>
<td>$T_{tot}$</td>
<td>13.2</td>
<td>13.3</td>
<td>46.1</td>
<td>84.5</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>10</td>
<td>14</td>
<td>15</td>
</tr>
<tr>
<td>It</td>
<td>1.2</td>
<td>1.7</td>
<td>3.6</td>
<td>2.4</td>
</tr>
<tr>
<td>$T_{su}$</td>
<td>2.5</td>
<td>2.6</td>
<td>9.2</td>
<td>10.0</td>
</tr>
<tr>
<td>$T_{sol}$</td>
<td>3.7</td>
<td>4.3</td>
<td>12.8</td>
<td>12.5</td>
</tr>
<tr>
<td>$T_{tot}$</td>
<td>3.7</td>
<td>4.3</td>
<td>12.8</td>
<td>12.5</td>
</tr>
</tbody>
</table>

#### Sphere

<table>
<thead>
<tr>
<th>Order</th>
<th>$P_1$</th>
<th>$P_2$</th>
<th>$P_3$</th>
<th>$P_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>6</td>
<td>7</td>
<td>13</td>
<td>9</td>
</tr>
<tr>
<td>It</td>
<td>40.9</td>
<td>61.0</td>
<td>57.9</td>
<td>81.3</td>
</tr>
<tr>
<td>$T_{su}$</td>
<td>19.7</td>
<td>38.5</td>
<td>50.8</td>
<td>78.0</td>
</tr>
<tr>
<td>$T_{sol}$</td>
<td>60.5</td>
<td>99.5</td>
<td>150.0</td>
<td>159.3</td>
</tr>
<tr>
<td>$T_{tot}$</td>
<td>40.5</td>
<td>42.8</td>
<td>91.2</td>
<td>93.0</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>10</td>
<td>14</td>
<td>12</td>
</tr>
<tr>
<td>It</td>
<td>3.4</td>
<td>5.0</td>
<td>6.8</td>
<td>7.3</td>
</tr>
<tr>
<td>$T_{su}$</td>
<td>4.8</td>
<td>9.5</td>
<td>25.2</td>
<td>20.8</td>
</tr>
<tr>
<td>$T_{sol}$</td>
<td>8.2</td>
<td>14.5</td>
<td>32.1</td>
<td>27.9</td>
</tr>
</tbody>
</table>

*Failure* indicates insufficient memory. That AGMG was told that the matrix is symmetric, whereas we used Boomer AMG with a symmetric Gauss–Seidel smoothing scheme as defined in this work (i.e., one iteration of forward Gauss–Seidel for presmoothing and one of backward Gauss–Seidel for postsmoothing), overwriting the default which produces a nonsymmetric preconditioner. We further report the results for two values of the strong/weak coupling threshold: 0.25, which is the default, and 0.5 which is recommended for 3D problems.

The reported values correspond to specific runs and 10% variations in the elapsed time has been observed from one run to another. This explains why the results for linear elements ($P_1$) for Jump 2D/3D problems are not identical in both tables; regarding Lshape and Sphere problems, the system matrix has positive off-diagonal entries even with $P_1$, making larger variations possible. The reported failures of Boomer AMG are due to insufficient memory; they occur with Jump 3D ($P_4$) and

---

*Using the default does not change significantly the performances when applying the method directly to $A$ but often implies misconvergence when applying it to $M(A)$.
Table 5
Results for the AMG methods when applied to $\mathcal{M}(A)$ to define the preconditioner for the system matrix $A$. For Jump 2D/3D, the first (second) line of results for each finite element order corresponds to the smallest (largest) of the tested sizes. Reported times are elapsed times in seconds per $10^6$ unknowns.

<table>
<thead>
<tr>
<th></th>
<th>Boomer AMG</th>
<th>AGMG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>It</td>
<td>$\mathcal{T}_{su}$</td>
</tr>
<tr>
<td><strong>Jump 2D</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P1</td>
<td>5</td>
<td>3.2</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>3.3</td>
</tr>
<tr>
<td>P2</td>
<td>6</td>
<td>3.2</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>3.4</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>6.1</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>6.2</td>
</tr>
<tr>
<td>P4</td>
<td>19</td>
<td>8.3</td>
</tr>
<tr>
<td></td>
<td>19</td>
<td>8.6</td>
</tr>
<tr>
<td><strong>Lshape</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P1</td>
<td>5</td>
<td>4.3</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>5.3</td>
</tr>
<tr>
<td>P2</td>
<td>11</td>
<td>6.8</td>
</tr>
<tr>
<td>P3</td>
<td>12</td>
<td>8.3</td>
</tr>
<tr>
<td>P4</td>
<td>19</td>
<td>9.9</td>
</tr>
<tr>
<td><strong>Jump 3D</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P1</td>
<td>6</td>
<td>14.9</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>17.5</td>
</tr>
<tr>
<td>P2</td>
<td>12</td>
<td>15.1</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>15.3</td>
</tr>
<tr>
<td>P3</td>
<td>20</td>
<td>73.7</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>82.5</td>
</tr>
<tr>
<td>P4</td>
<td>30</td>
<td>148.0</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>182.5</td>
</tr>
<tr>
<td><strong>Sphere</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P1</td>
<td>8</td>
<td>36.4</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>15.8</td>
</tr>
<tr>
<td>P2</td>
<td>13</td>
<td>47.8</td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>15.8</td>
</tr>
<tr>
<td>P3</td>
<td>25</td>
<td>74.7</td>
</tr>
<tr>
<td></td>
<td>25</td>
<td>178.5</td>
</tr>
</tbody>
</table>

Sphere (P3) problems, as the corresponding matrices have the largest number of nonzero entries ($\#nz \approx 2 \times 10^8$) in the test suite (see Table 3).

That said, leaving aside memory issues, the following comments apply to both AGMG and Boomer AMG used with the strong/weak threshold correctly selected according to the problem's dimensions. First, all strategies give robust results. For regular grids, the time per $10^6$ unknowns is in each case about constant from one size to the next, indicating that scalability in time is practically achieved. Moreover, in Table 4, the numbers of iterations only slightly grow when increasing the elements' order, and, in fact, the times would be nearly constant if reported per nonzero entry instead of per unknown (compare with the growth of $\#nz/n$ in Table 3). In Table 5, the numbers of iterations grow faster, in fact, roughly like $\alpha^{-1/2}$, as expected from (1.3) and convergence estimates of the CG method (see Table 1 or 2 for $\alpha$ values). This is, however, to a large extent compensated by the fact that the higher the element order, the sparser the $M(A)$ compared to $A$, making the preconditioner computed from $M(A)$ relatively cheaper per iteration step. Hence times in Table 5...
Table 6

Total solution time per $10^6$ unknowns (and iteration count) spent by Boomer AMG to solve the Sphere (P4) problem as a function of various coarse variable selection strategies and interpolation schemes; asterisks mark the default choice.

<table>
<thead>
<tr>
<th></th>
<th>clas*</th>
<th>mp</th>
<th>std</th>
<th>ext</th>
<th>ext+i</th>
<th>ext+i-cc</th>
</tr>
</thead>
<tbody>
<tr>
<td>RS*</td>
<td>93.2</td>
<td>88.7</td>
<td>104.6</td>
<td>115.7</td>
<td>110.1</td>
<td>98.0</td>
</tr>
<tr>
<td>RS1</td>
<td>132.7</td>
<td>64.7</td>
<td>79.7</td>
<td>90.4</td>
<td>95.2</td>
<td>84.3</td>
</tr>
<tr>
<td>CLJP</td>
<td>89.8</td>
<td>89.6</td>
<td>108.1</td>
<td>115.0</td>
<td>113.8</td>
<td>94.0</td>
</tr>
<tr>
<td>PIMS</td>
<td>152.7</td>
<td>65.3</td>
<td>77.4</td>
<td>89.9</td>
<td>97.7</td>
<td>89.1</td>
</tr>
<tr>
<td>CGC</td>
<td>62.8</td>
<td>59.6</td>
<td>70.0</td>
<td>78.7</td>
<td>78.1</td>
<td>66.4</td>
</tr>
</tbody>
</table>

are often comparable to times in Table 4, although slightly larger in general. Since, however, setup times are accordingly always smaller with the strategy based on $\mathcal{M}(A)$, this latter may be the best option when only a modest reduction of the error is needed.

Another observation that stems from the results in Tables 4 and 5 is that AGMG appears faster and less memory consuming (at least, less prone to memory issues) than Boomer AMG when both codes are used in a black box fashion. Whereas such usage seems natural for AGMG, Boomer AMG provides, besides the default setting, numerous options that implement important research efforts which have been made over the past decade to improve the robustness and efficiency of classical AMG schemes, in particular to address complexity issues.

We therefore also investigated the tuning of Boomer AMG for the problems at hand, focusing on options that determine the two steps of the coarsening process: coarse variables selection and interpolation. These options are indeed the most critical ones regarding the complexity. Considering the coarse variables selection, we first observe that some of the proposed options yield different results only when Boomer AMG is used in parallel, so that for our sequential tests the possible strategies boil down to classical two passes Ruge–Stüben (RS) \cite{15,31}—the default choice; one pass Ruge–Stüben (RS1), the standard algorithm in \cite{34}; Cleary–Luby–Jones–Plasman (CLJP) \cite{7}; maximal independent set (PMIS), a form of aggressive coarsening \cite{9}; and coarse grid classification (CGC) \cite{13}. Moreover, each of the aforementioned coarse variable selection schemes can be combined with aggressive coarsening \cite{34} on a few finest levels. For the interpolation, the following schemes are available, all described in \cite{8}: classical (clas, the default choice), multipass (mp), standard (st), extended (ext), extended+i (e+i), and extended+i for nodes with no common C neighbor (e+i-cc) interpolations.

Table 6 lists the total solution time and number of iterations needed to solve the Sphere (3D) problem with P4 elements, using Boomer AMG with the above mentioned options, the strong/weak coupling threshold being set to 0.5. We selected the Sphere problem because it combines all potential sources of difficulties: P4 irregular connectivity pattern in three dimensions with discontinuities and local refinement. Hence the best variants for this problem are also likely the most robust.

Clearly, the best results are obtained with the multipass interpolation scheme. We then fix the interpolation to multipass and further test the aggressive coarsening.

\footnote{Some of these options may also be combined with a separation of weights or adapted to hyperbolic PDEs; we do not display the results because, for the problems at hand, such modifications always lead to larger solution times.}
option. The results are reported in Table 7. Two levels of aggressive coarsening are the most appropriate option in all cases, and it is interesting to check the impact of this option on the complexity. Figure 6 gives the size and number of nonzero entries of the coarse grid matrices at the first six coarse levels for Boomer AMG with the best tuned parameters (CLJP coarse variable selection with two levels of aggressive coarsening and multipass interpolation). The figure also allows comparison with “default” Boomer AMG and AGMG. One sees that as long as aggressive coarsening takes place, Boomer AMG produces coarse grid matrices of complexity close to that obtained with AGMG. We checked that a similar effect is obtained with RS and CGC coarsening schemes whose performance are close to that of CLJP (see Table 7). Hence aggressive coarsening is indeed the right tool to address complexity issues. From this viewpoint, it would even be desirable to use this option on more levels, but, as shown in Table 7, the negative impact on the number of iterations makes such an option not cost-effective.

This latter observation suggests we further include some tests using, instead of the default V-cycle, the W-cycle for which the number of iterations is likely less sensitive to aggressive coarsening. Indeed, with the W-cycle, the systems with \( A_c \) in (3.3) are approximately solved with two stationary iterations (instead of one with the V-cycle). On the other hand, the impact of the extra stationary iteration on the preconditioner’s cost is limited if the coarsening is fast enough, which means, regarding

---

### Table 7

Total solution time per 10^6 unknowns (and iteration count) spent by Boomer AMG with multipass interpolation to solve the Sphere (P4) problem as a function of various coarse variable selection strategies and the number of coarse levels that use aggressive coarsening; asterisks marks the default choice and shaded cells highlight the best coarse variable selection schemes.

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>RS</td>
<td>88.7 (13)</td>
<td>63.3 (15)</td>
<td>56.2 (16)</td>
<td>60.1 (18)</td>
<td>60.5 (19)</td>
<td>62.3 (20)</td>
</tr>
<tr>
<td>RS1</td>
<td>64.7 (20)</td>
<td>65.7 (22)</td>
<td>64.8 (23)</td>
<td>71.4 (26)</td>
<td>79.6 (29)</td>
<td>77.9 (28)</td>
</tr>
<tr>
<td>CLJP</td>
<td>89.6 (13)</td>
<td>62.8 (15)</td>
<td>55.3 (16)</td>
<td>55.3 (17)</td>
<td>59.8 (19)</td>
<td>64.0 (21)</td>
</tr>
<tr>
<td>PIMS</td>
<td>65.3 (21)</td>
<td>68.5 (23)</td>
<td>70.0 (25)</td>
<td>75.5 (28)</td>
<td>81.7 (30)</td>
<td>81.2 (31)</td>
</tr>
<tr>
<td>CGC</td>
<td>59.6 (13)</td>
<td>57.1 (14)</td>
<td>56.6 (16)</td>
<td>55.6 (17)</td>
<td>59.3 (19)</td>
<td>64.8 (21)</td>
</tr>
</tbody>
</table>

---

**Fig. 6.** Size and number of nonzero entries of coarse grid matrices as a function of the level index for the Sphere problem with fourth order (P4) Lagrangian elements; level 0 corresponds to the fine grid matrix and the bottom line gives the slope corresponding to a constant reduction by a factor of 4 from each level to the next.
Results for the Boomer AMG (applied directly to the system matrix $A$) used with multipass interpolation and the best coarse variable selection schemes from Table 7. The strong/weak coupling threshold is equal 0.5 in all cases, V-cycle results are for two steps of aggressive coarsening, and W-cycle results are for aggressive coarsening at all levels. For Jump 2D/3D, the first (second) line of results for each finite element order corresponds to the smallest (largest) of the tested sizes; reported times are elapsed times in seconds per $10^6$ unknowns.

<table>
<thead>
<tr>
<th></th>
<th>V CLJP</th>
<th>W</th>
<th>V CGC</th>
<th>W</th>
<th>V RS</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>#It</td>
<td>#It</td>
<td>#It</td>
<td>#It</td>
<td>#It</td>
<td>#It</td>
<td>#It</td>
</tr>
<tr>
<td>$\tau_{tot}$</td>
<td>$\tau_{tot}$</td>
<td>$\tau_{tot}$</td>
<td>$\tau_{tot}$</td>
<td>$\tau_{tot}$</td>
<td>$\tau_{tot}$</td>
<td>$\tau_{tot}$</td>
</tr>
<tr>
<td>Jump 2D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P1</td>
<td>17</td>
<td>11</td>
<td>9.4</td>
<td>16</td>
<td>7.7</td>
<td>11</td>
</tr>
<tr>
<td>P2</td>
<td>18</td>
<td>12</td>
<td>10.3</td>
<td>17</td>
<td>9.0</td>
<td>12</td>
</tr>
<tr>
<td>P3</td>
<td>19</td>
<td>12</td>
<td>10.4</td>
<td>18</td>
<td>9.4</td>
<td>12</td>
</tr>
<tr>
<td>P4</td>
<td>22</td>
<td>18</td>
<td>2.2</td>
<td>19</td>
<td>16.4</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lshape</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P1</td>
<td>10</td>
<td>9.1</td>
<td>8.2</td>
<td>10</td>
<td>7.8</td>
<td>7</td>
</tr>
<tr>
<td>P2</td>
<td>18</td>
<td>15.4</td>
<td>12.4</td>
<td>18</td>
<td>13.8</td>
<td>12</td>
</tr>
<tr>
<td>P3</td>
<td>18</td>
<td>20.1</td>
<td>17.2</td>
<td>18</td>
<td>20.2</td>
<td>13</td>
</tr>
<tr>
<td>P4</td>
<td>21</td>
<td>24.7</td>
<td>21.8</td>
<td>21</td>
<td>25.0</td>
<td>16</td>
</tr>
<tr>
<td>Jump 3D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P1</td>
<td>14</td>
<td>17.6</td>
<td>16.8</td>
<td>14</td>
<td>11.0</td>
<td>10</td>
</tr>
<tr>
<td>P2</td>
<td>18</td>
<td>18.9</td>
<td>16.9</td>
<td>15</td>
<td>12.1</td>
<td>10</td>
</tr>
<tr>
<td>P3</td>
<td>14</td>
<td>22.7</td>
<td>21.1</td>
<td>14</td>
<td>21.5</td>
<td>12</td>
</tr>
<tr>
<td>P4</td>
<td>15</td>
<td>24.3</td>
<td>22.2</td>
<td>15</td>
<td>23.2</td>
<td>12</td>
</tr>
<tr>
<td>Sphere</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P1</td>
<td>17</td>
<td>59.4</td>
<td>57.7</td>
<td>16</td>
<td>59.3</td>
<td>13</td>
</tr>
<tr>
<td>P2</td>
<td>15</td>
<td>58.3</td>
<td>57.6</td>
<td>16</td>
<td>60.5</td>
<td>13</td>
</tr>
<tr>
<td>P3</td>
<td>16</td>
<td>55.3</td>
<td>51.5</td>
<td>16</td>
<td>55.6</td>
<td>13</td>
</tr>
</tbody>
</table>

Boomer AMG, that using this option is most sensible if aggressive coarsening is applied at all levels.

We then report in Table 8, for the whole set of test problems, the results obtained with the three best option sets from Table 7 (with thus two steps of aggressive coarsening), as well as with the corresponding W-cycle variants (with aggressive coarsening at all levels). Comparing with the second column of Table 4, the tuning brings a significant speedup (an improvement by a factor of up to 2) for all 3D problems, whereas the comparison is less conclusive for 2D examples. However, Boomer AMG remains significantly slower than AGMG. Eventually, we summarize the numerical results in Table 9, where, for completeness, we also report the results obtained with a truncated version of AGMG that uses the W-cycle instead of the K-cycle. The fact that using the K-cycle significantly improves aggregation-based multigrid methods is well known and its observation traces back to [20].
### Table 9

Results for the AMG methods applied directly to the system matrix $A$. “Aggregation” refers to aggregation-based AMG, implemented either as in the AGMG software (i.e., as in Table 4, columns 9–13) or as in a truncated version that uses the W-cycle instead of the K-cycle. Boomer AMG results are given for both the default version (i.e., as in Table 4, columns 2–5) or with tuned options, selecting the variant that overall performs best (i.e., CGC with the W-cycle, as in Table 8, columns 8–9). $C_A$ is the operator complexity, that is, the sum of the number of nonzero entries at all levels (including the initial fine grid level) relative to the number of nonzero entries at the fine grid level. The aggregation variant with W-cycle has same operator complexity as AGMG, and also the same setup time, for which we refer to Table 4, column 11; the setup time for default Boomer AMG is also given in Table 4 (column 3); in all cases, the solution time can be recovered by subtracting the setup time from the total time.

<table>
<thead>
<tr>
<th></th>
<th>Aggregation</th>
<th>Boomer AMG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$C_A$</td>
<td>It</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Jump 2D</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P1</td>
<td>1.3</td>
<td>10</td>
</tr>
<tr>
<td>P2</td>
<td>1.3</td>
<td>9</td>
</tr>
<tr>
<td>P3</td>
<td>1.4</td>
<td>12</td>
</tr>
<tr>
<td>P4</td>
<td>1.4</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>1.3</td>
<td>17</td>
</tr>
<tr>
<td><strong>Lshape</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P1</td>
<td>1.4</td>
<td>7</td>
</tr>
<tr>
<td>P2</td>
<td>1.3</td>
<td>13</td>
</tr>
<tr>
<td>P3</td>
<td>1.3</td>
<td>14</td>
</tr>
<tr>
<td>P4</td>
<td>1.2</td>
<td>15</td>
</tr>
<tr>
<td><strong>Jump 3D</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P1</td>
<td>1.3</td>
<td>10</td>
</tr>
<tr>
<td>P2</td>
<td>1.4</td>
<td>14</td>
</tr>
<tr>
<td>P3</td>
<td>1.4</td>
<td>14</td>
</tr>
<tr>
<td>P4</td>
<td>1.5</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td>1.5</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td>1.3</td>
<td>13</td>
</tr>
<tr>
<td><strong>Sphere</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P1</td>
<td>1.7</td>
<td>8</td>
</tr>
<tr>
<td>P2</td>
<td>1.7</td>
<td>11</td>
</tr>
<tr>
<td>P3</td>
<td>1.4</td>
<td>11</td>
</tr>
<tr>
<td>P4</td>
<td>1.4</td>
<td>12</td>
</tr>
</tbody>
</table>

### 7. Conclusions

We have explored two strategies that apply without modification existing classical or aggregation-based AMG methods for solving linear systems that arise from moderate order finite element discretizations. One strategy amounts to building an AMG preconditioner directly from the system matrix $A$, and the other one consists in building it from an intermediate matrix $M(A)$ obtained by discarding positive offdiagonal entries while lumping them to the diagonal. Both approaches can be theoretically justified from the analysis of the constant $\alpha$ in the spectral equivalence relations (1.3). Numerical experiments with the Boomer AMG and AGMG software packages suggest that both strategies are indeed robust, with a slight advantage to
the former. These experiments also allowed us to compare the performance of both solvers for the considered class of problems.

REFERENCES


