SIAM J. SCI. COMPUT.
Vol. 34, No. 2, pp. A1079–A1109

AN ALGEBRAIC MULTIGRID METHOD WITH GUARANTEED CONVERGENCE RATE*

ARTEM NAPOV† AND YVAN NOTAY‡

Abstract. We consider the iterative solution of large sparse symmetric positive definite linear systems. We present an algebraic multigrid method which has a guaranteed convergence rate for the class of nonsingular symmetric M-matrices with nonnegative row sum. The coarsening is based on the aggregation of the unknowns. A key ingredient is an algorithm that builds the aggregates while ensuring that the corresponding two-grid convergence rate is bounded by a user-defined parameter. For a sensible choice of this parameter, it is shown that the recursive use of the two-grid procedure yields a convergence independent of the number of levels, provided that one uses a proper AMLI-cycle. On the other hand, the computational cost per iteration step is of optimal order if the mean aggregate size is large enough. This cannot be guaranteed in all cases but is analytically shown to hold for the model Poisson problem. For more general problems, a wide range of experiments suggests that there are no complexity issues and further demonstrates the robustness of the method. The experiments are performed on systems obtained from low order finite difference or finite element discretizations of second order elliptic partial differential equations (PDEs). The set includes two- and three-dimensional problems, with both structured and unstructured grids, some of them with local refinement and/or reentering corner, and possible jumps or anisotropies in the PDE coefficients.

Key words. multigrid, algebraic multigrid, iterative methods, preconditioner, convergence analysis, aggregation

AMS subject classifications. 65F10, 65N12, 65N55, 65F50

DOI. 10.1137/100818509

1. Introduction.

Efficient solution of large sparse symmetric positive definite (SPD) $n \times n$ linear systems

$$A\mathbf{x} = \mathbf{b} \tag{1.1}$$

is critical for many of today's applications in science and engineering. For systems arising from the discretization of elliptic partial differential equations (PDEs), algebraic multigrid (AMG) methods are known to be particularly suitable and robust [5, 6, 7, 8, 16, 25, 26, 29, 28]. These algorithms combine the effects of a *smoother* and of a *coarse grid correction*. The smoother is generally based on a simple iterative scheme such as the Gauss–Seidel method. The coarse grid correction consists in computing an approximate solution to the residual equation on a coarser grid which has fewer unknowns. The approach is applied recursively until the coarse system is small enough to make negligible the cost of an exact solution. Unlike geometric multigrid methods [14, 27], which require information from the discretization of the PDE, AMG

*Submitted to the journal's Methods and Algorithms for Scientific Computing section December 17, 2010; accepted for publication (in revised form) January 11, 2012; published electronically April 10, 2012.
http://www.siam.org/journals/sisc/34-2/81850.html

†Computational Research Division, Lawrence Berkeley National Laboratory (M.S. 50A-1148), 1 Cyclotron Rd., Berkeley, CA 94720 (anapov@lbl.gov). This author's research was supported by the Belgian FNRS ("Aspirant"). The work on the revised version of the manuscript was supported by Director, Office of Science, Office of Advanced Scientific Computing Research of the U.S. Department of Energy under contract DE-AC02-05CH11231.

‡Service de Métrologie Nucléaire, Université Libre de Bruxelles (C.P. 165/84), 50, Av. F.D. Roosevelt, B-1050 Brussels, Belgium (ynotay@ulb.ac.be, homepages.ulb.ac.be/∼ynotay). This author's research was supported by the Belgian FNRS ("Directeur de recherches").

methods build the needed hierarchy of coarse systems exclusively from the system matrix, without additional input. This provides them with enough flexibility to solve problems arising from the discretization of PDEs on unstructured grids.

In this work, we consider more particularly aggregation-based multigrid schemes, in which the hierarchy of coarse systems is obtained from a mere aggregation of the unknowns. This approach is sometimes referred to as "plain" or "unsmoothed" aggregation to distinguish it from "smoothed aggregation AMG" initiated in [29].[1] Plain aggregation AMG has some appealing features such as cheap setup stage and modest memory requirements. However, it is somehow nonstandard because it does not mimic any well-established geometric multigrid method. As a result, although the seminal papers [4, 9] are not recent, the method has attracted little attention as it was originally unclear how to achieve optimal convergence rate; see, e.g., [26, pp. 522–524] (here and in what follows, the rate of convergence is "optimal" if it is bounded independently of the system size).

However, a number of recent works show that the approach can be both theoretically well founded and practically efficient [12, 17, 19, 20, 22], provided that the aggregates are formed in an appropriate way, and that one uses an enhanced multigrid cycle, that is, the K-cycle [24], in which the iterative solution of the residual equation at each level is accelerated with a Krylov subspace method.

In particular, in [20], we analyze the basic two-grid method (with exact solution of the coarse system) for a class of matrices which includes symmetric M-matrices with nonnegative row sum (that is, weakly diagonally dominant matrices with nonpositive off-diagonal entries). We show that the convergence rate can be bounded assessing for each aggregate a local quantity which in some sense measures its quality, the bound being determined by the worst aggregate's quality. Moreover, this bound seems able to accurately predict the actual convergence, at least when applied to PDEs discretized on regular grids with regular (geometric-based) aggregation patterns.

However, these results suffer from two important limitations. First, they are only for a model two-grid scheme. In practice, the method has to be recursively used in a multigrid cycle, and no guarantee is given that the convergence rate will then remain optimal, i.e., will not deteriorate with the number of levels. Second, the analysis can in principle be applied to any "algebraic" aggregation algorithm that automatically builds aggregates inspecting connections in the system matrix. However, as a general rule, such algorithms tend to always produce a limited number of badly shaped aggregates, and, since the bound is determined by the worst aggregate's quality, even a few of these can significantly impact the resulting bound. The latter is also difficult to predict in the case of PDEs discretized on unstructured grids. Hence, although we have proper tools to assess aggregation-based two-grid methods, what can actually be proved for a truly algebraic method remains unclear.

Here we overcome these limitations mainly by introducing a new aggregation algorithm based on the explicit control of aggregate quality. It tends to optimize the latter while imposing some minimal requirements; i.e., the algorithm has as main input parameter the upper bound on the two-grid condition number that is required to hold. Recall that the condition number κ is the ratio of the extremal eigenvalues of the preconditioned matrix.[2] Note that not only is the aggregation algorithm new, but the approach itself seems to correspond to a new paradigm. It follows the current trend

[1] This latter approach has its own convergence theory [28, 31, 8], which, however, has little in common with the one developed in this paper; see section 2 for further comments.

[2] For this type of methods, it is related to the convergence factor ρ via the relation $\rho = 1 - 1/\kappa$.

with AMG schemes, which is to define the ingredients of the method by algorithms which attempt to optimize quantities that are relevant with respect to some theoretical analysis; see, e.g., [6, 7, 10, 15, 18, 32, 33, 34, 35]. However, to the best of our knowledge, we present here the first attempt to go one step further by imposing an explicit control on the two-grid convergence rate.

Of course, such an approach potentially induces an increase of the algorithmic complexity. In the multigrid context, mastering the complexity means ensuring that the cost of each iteration step does not grow more than linearly with the number of the unknowns. Taking into account that the two-grid method has to be used recursively, this further means ensuring that the number of unknowns decreases sufficiently fast from one level to the next. With aggregation-based methods, the factor by which the number of unknowns is decreased is actually the mean aggregate size (i.e., the mean number of unknowns inside an aggregate). Whereas with heuristic aggregation algorithms it is relatively easy to control this mean aggregate size, the present approach introduces more uncertainty: one may form aggregates of a given target size, but there is no a priori guarantee that they will satisfy the quality criterion that allows us to control the condition number.

Now, we prove below that our aggregation algorithm, when applied to the five-point discretization of the Poisson problem, allows us to guarantee that the condition number of the matrix preconditioned by the two-grid method is below 11.5 while reducing the number of unknowns by a factor of at least 8. This is still a model problem analysis, but is somehow nonstandard because it holds for a truly algebraic method applied in a black box fashion. Moreover, it does not hold only at the fine grid level: we show that one can recursively apply the coarsening algorithm and yet at each stage guarantee the same combination of condition number and coarsening speed. Further, numerical experiments performed on a wide set of discrete PDEs suggest that there are no complexity issues: putting the same threshold of 11.5 on the condition number, the algorithm succeeds in producing aggregates of mean size 8 or close to 8 also for, e.g., three-dimensional problems and/or problems discretized on unstructured grids.

All in all, there are two advantages in explicitly controlling the convergence rate (or the condition number) instead of the complexity. First, in a typical situation, a "complexity oriented" algorithm will form a few badly shaped aggregates. As already mentioned, this may have a dramatic impact on the convergence analysis. If there are only a few such bad aggregates, their influence on the actual convergence may or not be significant; in general we just do not know. However, one should be careful as the analysis in [20] reveals that our bound may remain sharp even in the presence of a relatively small number of "bad" aggregates. On the other hand, an algorithm that explicitly controls the condition number will refuse to form these bad aggregates and stay instead with a few aggregates of smaller size or even some unaggregated nodes. Compared with the previous situation, here we know that if there is only a small number of such aggregates, then the impact on the efficiency of the method will be minor, since it would only affect the mean aggregate size in a unessential way.

A second advantage of the approach is that it fits well with the use of the so-called AMLI-cycle [2, 3, 30, 31]. With this cycle, the iterative solution of the coarse system at each level is accelerated with a semi-iterative method based on Chebyshev polynomials. As the main advantage, it allows us to build optimal multigrid schemes using only two-grid analysis. Hence, using the AMLI-cycle, the explicit control of the convergence rate mentioned above is not restricted to a model two-grid scheme: it

carries over to the multilevel variant. In this paper, we exploit this feature to obtain a bound on the convergence rate that remains fixed independently of the number of levels.

The remainder of this paper is organized as follows. We first outline in section 2 our main result, namely the bound on a convergence rate that is guaranteed by our AMG method. In section 3 we recall needed two-grid analysis results from [20], particularizing them to the context of this paper. The AMLI-cycle is described in section 4. In section 5 we present our aggregation algorithm and explain why and how it allows us to guarantee the bound announced in section 2. Numerical results are reported in section 6.

Notation. For any ordered set Γ, $|\Gamma|$ is its size and $\Gamma(i)$ its ith element. $[1,k] = \{1,2,\ldots,k\}$ stands for the set of the first k integers. For any matrix B, $\mathcal{N}(B)$ is its null space. For any square matrix C, $\rho(C)$ is its spectral radius (that is, its largest eigenvalue in modulus). For any SPD matrix D, $\kappa(D)$ is its condition number (that is, the quotient of its largest and smallest eigenvalues). I stands for identity matrix.

2. Outline of the main result. Building an optimal method with the AMLI-cycle is possible if there are a known upper bound $\bar{\kappa}_{\mathrm{TG}}$ on the condition number of the two-grid method at every level and an integer γ (related to the degree of Chebyshev polynomials) such that

$$\sqrt{\bar{\kappa}_{\mathrm{TG}}} \;<\; \gamma \;<\; \tau, \tag{2.1}$$

where τ stands for the mean factor by which the matrix size is reduced from one level to the next. Both inequalities should be interpreted in a restrictive way: for an efficient method, $\sqrt{\bar{\kappa}_{\mathrm{TG}}}$ should be substantially away from γ and the latter itself substantially away from τ. The left inequality guarantees that the condition number remains bounded even for infinitely many levels, and the right one ensures an optimal computational complexity. That said, the left inequality is the most difficult to deal with because, recall, it has to be checked by an *upper bound* on the two-grid condition number, which has to hold at every level of the hierarchy and, further, has to be known explicitly because it enters the definition of the parameters of the AMLI-cycle. Therefore, by introducing a new approach based on the explicit control of the two-grid condition number, we facilitate the use of the AMLI-cycle, which here is, seemingly for the first time, combined with a truly AMG method.

Observe that, with $\gamma = 4$, the conditions (2.1) fit well with the numbers $\bar{\kappa}_{\mathrm{TG}} = 11.5$ and $\tau = 8$ cited in the introduction. This is on purpose: when assessing our aggregation algorithm below, we target parameter choices allowing a sensible application of the AMLI-cycle. Using the latter with $\gamma = 4$ and $\bar{\kappa}_{\mathrm{TG}} = 11.5$, the induced multilevel preconditioner B satisfies

$$\kappa\left(B^{-1}A\right) \;\leq\; 27.06, \tag{2.2}$$

independently of the number of levels; that is, the explicit control of the two-grid condition number obtained thanks to our aggregation algorithms carries over to the multigrid scheme thanks to the use of the AMLI-cycle. On the other hand, with $\tau \approx 8$, the algorithmic complexity of one application of this preconditioner is only about twice that of the smoothing iterations at the fine grid level.

Regarding assumptions, the upper bound (2.2) requires only the system matrix A to be a symmetric M-matrix with nonnegative row sum. We consider that this property holds in the reminder of this paper. It is in particular satisfied with low

order finite difference or finite element discretizations of scalar second order elliptic PDEs, with, in the case of finite elements, some restriction on the elements' shape. For this class of applications, we thus obtain the same uniform bound independently of the mesh or problem size and of the structure or lack of structure of the discretization grid, regardless of whether the problem is two-dimensional (2D) or three-dimensional (3D), and independently of problem peculiarities such as jumps or anisotropy in the PDE coefficients, reentering corners, or lack of full elliptic regularity.

The bound (2.2) provides the first complete convergence analysis of an AMG method based on plain aggregation. Moreover, it also seems to compare favorably with available theoretical analyses of other AMG schemes. Classical AMG methods along the lines of the seminal papers [5, 25] also have a supporting theory based on the M-matrix assumption; see [26] for a nice summary. However, this theory is only for the two-grid case, and it is unclear whether it can produce such uniform estimates. For instance, it should in principle apply to the method implemented in the old but classical code that was used in [22] for numerical comparison and which failed to converge for some anisotropic problems, even though the matrix was an M-matrix. It is more difficult to develop a comparison with analyses of methods based on smoothed aggregation, as initiated in [28] and further improved in [31] and [8]. Indeed just the statement of the assumptions in [8] would require a significant amount of space; hence we refer the reader to the aforementioned works and just point out that these analyses are also for the multilevel case, while the resulting bound is hard to compare with (2.2) since it depends on the number of levels and involves an unknown constant C. Regarding assumptions, it is probably correct to state that the theory in [28, 31, 8] allows us to cover several discretizations of PDEs that do not result in an M-matrix, whereas some M-matrix cases such as finite difference discretizations are excluded. Note also that the discussion of these assumptions involves geometric considerations such as the mesh size h at the fine grid level and the diameter of the aggregates; hence it is unclear how it could apply to problems with strong local refinement.

Eventually, as already stated above, except for the model Poisson problem, the bound (2.2) should not be seen as a proof of optimality, since there remains some uncertainty on the algorithmic complexity. However, as commented in the first paragraph of section 7 in [8], the difficulty of *proving* that an AMG method has bounded algorithmic complexity is not peculiar to our approach and remains a challenge for truly black box methods that do not exploit any geometric information. Hence, although limited to a model problem, our complexity analysis also seems to represent some advance with respect to the state of the art.

3. Two-grid analysis. We first introduce some notation (related to our two-grid setting). We consider symmetric two-grid schemes, using SPD smoother M with one pre- and one post-smoothing step. We also assume that the coarse grid matrix is of Galerkin type; that is, given an $n \times n_c$ prolongation matrix P, the coarse grid matrix is $A_c = P^T A\, P$. The corresponding iteration matrix is then

$$T \;=\; (I - M^{-1}A)(I - P\, A_c^{-1} P^T A)(I - M^{-1}A),$$

which also implicitly defines the two-grid preconditioner B_{TG} via the relation

$$I - B_{\mathrm{TG}}^{-1} A \;=\; T.$$

Equivalently, one has

$$B_{\mathrm{TG}}^{-1} = M^{-1}(2\, M - A)M^{-1} + (I - M^{-1}A)P\, A_c^{-1} P^T (I - A\, M^{-1})\,. \tag{3.1}$$

Now, with the coarsening by aggregation, the prolongation is obtained from the agglomeration of the unknowns into n_c nonempty disjoint sets G_k, $k = 1, \dots, n_c$, called *aggregates*. To each aggregate G_k is associated one unknown at the next coarse level in the hierarchy. In addition, following [20], some unknowns can also 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 with any coarse unknown. As a result, G_0 together with G_k, $k = 1, \dots, n_c$, defines a partitioning of the index set $[1, n]$ which uniquely determines the prolongation P: for $i = 1, \dots, n$ and $j = 1, \dots, n_c$,

$$(P)_{ij} = \begin{cases} 1 & \text{if } i \in G_j, \\ 0 & \text{otherwise}. \end{cases} \tag{3.2}$$

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 by the smoother alone.

Note also that the entries in the coarse grid matrix $A_c = P^T A P$, where $A = (a_{ij})$, can be obtained from a simple summation process:

$$(A_c)_{kl} = \sum_{i \in G_k} \sum_{j \in G_l} a_{ij}, \qquad k, l = 1, \dots, n_c. \tag{3.3}$$

It follows from this relation that if A is an M-matrix with nonnegative row sum, then A_c inherits these properties; see Theorem 3.6 in [17] for an explicit proof.[3] This observation is important: it means that if a matrix A satisfies the basic assumptions of Theorem 3.2 below, then all successive coarse grid matrices will satisfy them as well; i.e., the theorem can be applied at any level in the hierarchy.

Our analysis is first based on the well-known result from [13, 31], which, when the smoother M is SPD and such that $\mathbf{v}M\mathbf{v} \geq \mathbf{v}A\mathbf{v}$ for all $\mathbf{v}$, states that

$$\mathbf{v}^T A \mathbf{v} \;\leq\; \mathbf{v}^T B_{\mathrm{TG}} \mathbf{v} \;\leq\; \mu\, \mathbf{v}^T A \mathbf{v} \quad \forall \mathbf{v} \in \mathbb{R}^n \tag{3.4}$$

with

$$\mu \;=\; \max_{\mathbf{v} \neq 0} \frac{\mathbf{v}^T M \left(I - P (P^T M P)^{-1} P^T M \right) \mathbf{v}}{\mathbf{v}^T A \mathbf{v}}. \tag{3.5}$$

This is further combined with Theorem 3.2 in [20], which yields an upper bound on μ for prolongations P based on aggregation. For the sake of readability, we recall this latter result in the following lemma.

LEMMA 3.1. *Let A and M be $n \times n$ SPD matrices. Let G_k, $k = 0, \dots, n_c$, be some partitioning of $[1, n]$, and define P by (3.2). Assume that M is block diagonal with respect to the partitioning G_k; i.e., $(M)_{ij} = 0$ if i, j do not belong to the same subset G_k.*

Further, let A_b, A_r be nonnegative definite symmetric matrices such that $A = A_b + A_r$ and A_b is also block diagonal with respect to the partitioning G_k. For $k =$

[3] Strictly speaking, it is assumed in [17] that the prolongation matrix has exactly one nonzero per row, whereas we allow some zero rows; the extension of the proof is, however, straightforward since the "exactly one nonzero" property is used only for further results related to irreducibility, which are not needed here.

$0, \ldots, n_c$, *let* A_{G_k} *be the diagonal block of* A_b *corresponding to indices in* G_k*; that is, with an appropriate reordering of unknowns,* $A_b = \text{blockdiag}(A_{G_0}, A_{G_1}, \ldots, A_{G_{n_c}})$. *Similarly, let* M_{G_k} *denote the diagonal block of* M *corresponding to indices in* G_k, *and, hence,* $M = \text{blockdiag}(M_{G_0}, M_{G_1}, \ldots, M_{G_{n_c}})$. *There holds*

$$\max_{\mathbf{v}\neq 0} \frac{\mathbf{v}^T M \left(I - P(P^T M P)^{-1} P^T M\right) \mathbf{v}}{\mathbf{v}^T A \mathbf{v}} \leq \max_{k=0,\ldots,n_c} \mu^{(k)}, \tag{3.6}$$

where

$$\mu^{(0)} = \begin{cases} 0 & \text{if } G_0 \text{ is empty,} \\ \max_{\mathbf{v}} \frac{\mathbf{v}^T M_{G_0} \mathbf{v}}{\mathbf{v}^T A_{G_0} \mathbf{v}} & \text{otherwise,} \end{cases}$$

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

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

with $\mathbf{1}_{G_k} = (1, 1, \ldots, 1)^T$ *being a vector of size* $|G_k|$.

To apply this lemma, one first needs a proper splitting of the matrix into two nonnegative definite matrices. When A is an M-matrix with nonnegative row sum, this splitting is easy to obtain: regarding the off-diagonal entries, A_b gathers those connecting unknowns inside every aggregate, and A_r keeps the others; on the other hand, the diagonals are such that A_r has zero row sum and therefore A_b acquires the same (nonnegative) row sum as A. In other words, A_b is obtained from A by discarding and lumping to the diagonal the entries that are in the block off-diagonal part with respect to the partitioning in aggregates. This lumping ensures that both A_b and A_r are weakly diagonally dominant and hence nonnegative definite.

Then, as follows from (3.5) and (3.6), the measure μ involved in (3.4) can be controlled if the parameter $\mu^{(k)}$ associated with each aggregate is efficiently bounded. And this parameter depends only on the corresponding diagonal blocks in A_b and M; i.e., on quantities "local" to the aggregate: the "local" matrix entries and the row sum at "local" nodes. Observe that the lumping process mentioned above tends to make these diagonal blocks A_{G_k} ill-conditioned, or even singular when the row sum of A is zero at every node of the aggregate. But, for regular aggregates ($k > 0$), it does not mean that $\mu^{(k)}$ is unbounded since the matrix $M_{G_k}(I - \mathbf{1}_{G_k}(\mathbf{1}_{G_k}^T M_{G_k} \mathbf{1}_{G_k})^{-1} \mathbf{1}_{G_k}^T M_{G_k})$ is also, by construction, singular with the constant vector in its kernel. This is not true for $\mu^{(0)}$, because nodes in G_0 are kept outside the aggregation and are actually not represented anymore on the coarse grid. Therefore, smoothing iterations alone should be sufficient to efficiently damp the error at these nodes, which, in this analysis, is reflected by the requirement that the corresponding block A_{G_0} be well-conditioned. In practice, as will be seen below, this requirement can be checked via a strong diagonal dominance criterion.

Now, this works for smoothers that are block diagonal with respect to the partitioning in aggregates. A particular example is of course the damped Jacobi smoother. As discussed below, better results can, however, be obtained with more general smoothers. We therefore introduce the class of smoothers $M = (m_{ij})$, with off-diagonal entries obtained from that of $A = (a_{ij})$ by keeping those inside a given

symmetric sparsity pattern and discarding the others:

$$m_{ij} = \begin{cases} a_{ij} & \text{if } i \neq j \text{ and } (i,j) \in \mathrm{sp}(M), \\ a_{ii} + \displaystyle\sum_{\substack{s \neq i \\ (i,s) \notin \mathrm{sp}(M)}} |a_{is}| & \text{if } i = j, \\ 0 & \text{otherwise,} \end{cases} \tag{3.7}$$

where $\mathrm{sp}(M)$ stands for the chosen sparsity pattern; the discarded off-diagonal entries are further added in absolute value to the diagonal. This ensures the weak diagonal dominance of $M - A$, which is therefore nonnegative definite as required for (3.4).

Note that the smoother defined by (3.7) is not necessarily block diagonal. This depends upon the chosen sparsity pattern. We can allow this because our analysis is actually slightly more general than the mere combination of (3.4) with Lemma 3.1. Indeed, as shown in the proof of Theorem 3.2 below (based on previous results in [20]; see also [31]), (3.4) also holds with μ defined as in (3.5) in which the smoother M is replaced by any matrix $\overline{M}$ such that $\mathbf{v}M\mathbf{v} \leq \mathbf{v}\overline{M}\mathbf{v}$ for all $\mathbf{v}$. Then, to properly apply Lemma 3.1, only this $\overline{M}$ should be block diagonal. Given a smoother M defined by (3.7), a relevant $\overline{M} = (\overline{m}_{ij})$ is obtained using the same rule while restricting the sparsity pattern to the block diagonal part:

$$\mathrm{sp}(\overline{M}) \;=\; \{(i,j) \in \mathrm{sp}(M) \mid i,j \,\text{belong to the same aggregate}\}.$$

Such $\overline{M}$ has the required block diagonal structure, whereas, according to the rule (3.7), the further discarded entries from the off-diagonal blocks are added in absolute value to the diagonal, ensuring that $\overline{M} - M$ is weakly diagonal dominant and hence nonnegative definite. It is worth noting that, since the bound depends upon $\overline{M}$, it is thus not influenced by the off-diagonal block part of the sparsity pattern. This observation is taken into account below when we discuss some relevant choices for $\mathrm{sp}(M)$.

Now, particularizing the analysis sketched above to smoothers defined via (3.7) yields the following theorem. Note that the matrices A_G and M_G in this theorem represent the diagonal blocks of $A_b = \mathrm{blockdiag}(A_{G_0}, A_{G_1}, \ldots, A_{G_{n_c}})$ and $\overline{M} = \mathrm{blockdiag}(M_{G_0}, M_{G_1}, \ldots, M_{G_{n_c}})$.

THEOREM 3.2. *Let $A = (a_{ij})$ be an $n \times n$ nonsingular symmetric M-matrix with nonnegative row sum. Let G_k, $k = 0, \ldots, n_c$, be a partitioning of $[1,n]$, let $\mathrm{sp}(M)$ be some symmetric sparsity pattern, and define P, $M = (m_{ij})$, and B_{TG} by* (3.2), (3.7), *and* (3.1), *respectively.*

For any subset G of $[1,n]$, let $A|_G$ and $M|_G$ be the submatrices of A and M, respectively, corresponding to indices in G. Moreover, let Σ_G, Δ_G be $|G| \times |G|$ diagonal matrices with

$$(\Sigma_G)_{ii} = \sum_{j \notin G} |a_{G(i)j}| \quad \textit{and} \quad (\Delta_G)_{ii} = a_{G(i)G(i)} + \sum_{\substack{j \in G \\ j \neq G(i) \ \textit{and} \ (G(i),j) \notin \mathrm{sp}(M)}} |a_{G(i)j}|,$$

where $G(i)$ is the ith element in G. Set $A_G = A|_G - \Sigma_G$ and let M_G be such that $\mathrm{offdiag}(M_G) = \mathrm{offdiag}(M|_G)$ *and* $\mathrm{diag}(M_G) = \Delta_G + \Sigma_G$. *Define, if $|G| > 1$,*

$$\mu(G) \;=\; \sup_{\mathbf{v} \notin \mathcal{N}(A_G)} \frac{\mathbf{v}^T M_G (I - \mathbf{1}_G (\mathbf{1}_G^T M_G \mathbf{1}_G)^{-1} \mathbf{1}_G^T M_G)\mathbf{v}}{\mathbf{v}^T A_G \mathbf{v}}, \tag{3.8}$$

where $\mathbf{1}_G = (1, 1, \ldots, 1)^T$ is a vector of size $|G|$.

If for a given $\bar{\kappa}_{\rm TG} \geq 1$, there holds

(i)

$$a_{ii} \geq \frac{\bar{\kappa}_{\rm TG}+1}{\bar{\kappa}_{\rm TG}-1}\left(\sum_{j=1,j\neq i}^{n}|a_{ij}|\right) \quad \forall i \in G_0;\ \textit{and} \tag{3.9}$$

(ii) *for $k=1,\ldots,n_c$, either $|G_k|=1$ or*

$$\mu(G_k) \leq \bar{\kappa}_{\rm TG}, \tag{3.10}$$

then

$$\mathbf{v}^T A\mathbf{v} \leq \mathbf{v}^T B_{\rm TG}\mathbf{v} \leq \bar{\kappa}_{\rm TG}\,\mathbf{v}^T A\mathbf{v} \quad \forall \mathbf{v}\in\mathbb{R}^n\,. \tag{3.11}$$

Proof. We first show that (3.4) holds with

$$\mu = \max_{\mathbf{v}\neq 0}\frac{\mathbf{v}^T\overline{M}\left(I-P(P^T\overline{M}\,P)^{-1}P^T\overline{M}\right)\mathbf{v}}{\mathbf{v}^T A\mathbf{v}} \tag{3.12}$$

for any matrix $\overline{M}$ such that $\mathbf{v}M\mathbf{v}\leq\mathbf{v}\overline{M}\mathbf{v}$.

The left inequality is actually a standard result (e.g., [31, section 3.2]); we give a short proof for the sake of completeness: since A is SPD, $A^{1/2}PA_c^{-1}P^TA^{1/2}$ is an orthogonal projector; hence, $\mathbf{v}^TPA_c^{-1}P^T\mathbf{v}\leq\mathbf{v}^TA^{-1}\mathbf{v}$ for all $\mathbf{v}$ and, using this inequality in (3.1), one has for all $\mathbf{v}\in\mathbb{R}^n$

$$\mathbf{v}^TB_{\rm TG}^{-1}\mathbf{v}\leq\mathbf{v}^TM^{-1}(2\,M-A)M^{-1}\mathbf{v}+\mathbf{v}^T(I-M^{-1}A)\,A^{-1}\,(I-A\,M^{-1})\mathbf{v}=\mathbf{v}^TA^{-1}\mathbf{v}.$$

To prove the right inequality, we note that

$$\lambda_{\min}(B_{\rm TG}^{-1}A)=1-\lambda_{\max}(I-B_{\rm TG}^{-1}A)=1-\rho(I-B_{\rm TG}^{-1}A)\,,$$

where the second equality follows from the left inequality just proved. Hence we need to prove $\rho(I-B_{\rm TG}^{-1}A)\leq 1-\mu^{-1}$. To this end, one may apply Theorem 3.1 of [20]. In this theorem, X denotes the equivalent global smoother. Hence, for $B_{\rm TG}$ defined by (3.1), one has $X=(2\,M^{-1}-M^{-1}AM^{-1})^{-1}$. Since $\overline{M}-M$ is weakly diagonally dominant, one then has, for all $\mathbf{v}\in\mathbb{R}^n$ (using also $\mathbf{v}^TM^{-1/2}AM^{-1/2}\mathbf{v}\leq 1$ which follows from the nonnegative definiteness of $M-A$),

$$\mathbf{v}^T\overline{M}^{-1}\mathbf{v}\leq\mathbf{v}^TM^{-1}\mathbf{v}\leq\mathbf{v}^T(2\,M^{-1}-M^{-1}AM^{-1})\mathbf{v}=\mathbf{v}^TX^{-1}\mathbf{v};$$

that is,

$$\max_{\mathbf{v}}\frac{\mathbf{v}^TX\mathbf{v}}{\mathbf{v}^T\overline{M}\mathbf{v}}\leq 1.$$

Then, noting that the matrix D referenced in Theorem 3.1 of [20] can be freely chosen, and setting $D=\overline{M}=\text{blockdiag}(M_{G_0},M_{G_1},\ldots,M_{G_{n_c}})$, the relations (7) and (8) in this theorem together with $\lambda_{\max}(X^{-1}A)\leq\lambda_{\max}(M^{-1}A)\leq 1$ imply the required result; that is, $\rho(I-B_{\rm TG}^{-1}A)\leq 1-\mu^{-1}$ and therefore (3.4) holds with μ as in (3.12).

Clearly, the required result (3.11) follows if $\mu\leq\bar{\kappa}_{\rm TG}$. We then consider Lemma 3.1 with the splitting $A=A_b+A_r$, $A_b=\text{blockdiag}(A_{G_0},A_{G_1},\ldots,A_{G_{n_c}})$, which in fact

corresponds to the construction procedure described just after the lemma. One sees that the condition (3.10) indeed entails $\mu \leq \bar{\kappa}_{\rm TG}$ if, in addition,

$$\max_{\mathbf{v}\in\mathbb{R}^{|G_0|}} \frac{\mathbf{v}^T M_{G_0}\mathbf{v}}{\mathbf{v}^T A_{G_0}\mathbf{v}} \leq \bar{\kappa}_{\rm TG}.$$

Now, $\mathbf{v}^T M_{G_0}\mathbf{v} \leq \mathbf{v}^T M^{(0)}_{G_0}\mathbf{v}$, where $M^{(0)}_{G_0} = \mathrm{diag}(a_{G_0(i)G_0(i)} + \sum_{j\neq G_0(i)} |a_{G_0(i)j}|)$ (one may check that $M^{(0)}_{G_0} - M_{G_0}$ is weakly diagonally dominant). The result then follows from

$$\max_{\mathbf{v}\in\mathbb{R}^{|G_0|}} \frac{\mathbf{v}^T M^{(0)}_{G_0}\mathbf{v}}{\mathbf{v}^T A_{G_0}\mathbf{v}} \leq \max_{i\in G_0} \frac{a_{ii} + \sum_{j=1,j\neq i}^n |a_{ij}|}{a_{ii} - \sum_{j=1,j\neq i}^n |a_{ij}|} \leq \bar{\kappa}_{\rm TG}\,,$$

where the first inequality holds since $A_{G_0} - \mathrm{diag}(a_{G_0(i)G_0(i)} - \sum_{j=1,j\neq G_0(i)}^n |a_{G_0(i)j}|)$ is nonnegative definite (it has negative off-diagonal entries and zero row sum), whereas the second inequality stems from (3.9). ∎

We now discuss how to best select the sparsity pattern of the smoother. A first possibility consists in discarding all off-diagonal entries, yielding a diagonal smoother $M = \mathrm{diag}(\sum_{j=1}^n |a_{ij}|)$. Since A is diagonally dominant, we then have $m_{ii} \leq 2a_{ii}$, with equality when the corresponding row of A has zero sum. Hence such a smoother does not differ much from the classical damped Jacobi smother $M = \omega_{Jac}^{-1}\mathrm{diag}(a_{ii})$ with damping factor $\omega_{Jac} = 0.5$.

Here we advocate further choices, which yield better "quality" estimates $\mu(G_k)$ for the aggregates and therefore make the constraint (3.10) less restrictive. As noted above, the bound is not improved if more entries connecting *different* aggregates are added to the sparsity pattern. However, consider that, starting from a given smoother, say $M^{(1)}$, one creates another smoother $M^{(2)}$ by including in the sparsity pattern some additional connections *internal* to an aggregate G_k. One sees from (3.7) that the related matrices $M^{(1)}_{G_k}$ and $M^{(2)}_{G_k}$ will be such that $M^{(1)}_{G_k} - M^{(2)}_{G_k}$ is weakly diagonally dominant and, hence, nonnegative definite. It can then be inferred from [20, Theorem 3.1] that $\mu(G_k)$ is at least as small for $M^{(2)}$ as it is for $M^{(1)}$. That is, putting in the sparsity pattern of M more off-diagonal entries internal to the aggregates never deteriorates the quality estimates $\mu(G_k)$, and, in fact, as numerical computation reveals, most often improves them.

Moreover, optimizing the sparsity pattern by including for every aggregate all its internal connections has only a moderate impact on the computational cost as long as the aggregates are relatively small or have small bandwidth. From now on, we therefore consider that the smoother is one of the two described below; except in numerical experiments, it does not matter which one is actually used since, according to the discussion above, they lead to identical convergence estimates.

Block diagonal smoother. $(i,j) \in \mathrm{Sp}(M)$ if and only if i and j belong to the same aggregate G_k, $1 \leq k \leq n_c$.

Band smoother. This smoother assumes that the matrix has been explicitly reordered according to the partitioning in aggregates; that is, the unknowns in the same set G_k, $k = 0, 1, \ldots, n_c$, occupy successive positions in the index set. It then follows that the above block diagonal smoother has an explicit block diagonal structure, and it may be interesting to extend the sparsity pattern so that it comprises the whole band that includes all these diagonal blocks (except, possibly, the one corresponding to G_0); that is, $(i,j) \in \mathrm{Sp}(M)$ if and only if $|i-j| \leq \delta$, where $\delta = \max_{1\leq k\leq n_c} \mathrm{bandwidth}(A_{G_k})$. This allows us to use a fast band solver such as the one available in LAPACK [1].

4. The AMLI-cycle. The AMLI-cycle implements the recursive use of a two-level method. The exact solution of the coarse system that is needed to compute the action of a two-grid preconditioner is exchanged for the approximate solution obtained with a few steps of an iterative solution method. This iterative solution itself uses the two-grid preconditioner on that coarse level, leading to a recursive scheme. The recursion is stopped when the coarse system is sufficiently small to allow a direct solution method at negligible cost. Note that this description applies as well to the other cycles, such as the W-cycle [14, 27]. What makes the AMLI-cycle different is the use, for the solution of each coarse system, of a semi-iterative method based on shifted Chebyshev polynomials. The number of iterations γ is also typically larger than the two iterations associated with the W-cycle.

This polynomial acceleration allows us to obtain a stabilized condition number (i.e., a condition number that is bounded independently of the number of levels) under rather weak conditions; see below. There is a price to pay, however. At each level, the parameters of the polynomial acceleration are to be defined according to explicit bounds on the eigenvalues of the matrix preconditioned by the two-grid method. Moreover, the overall efficiency of the scheme depends on the sharpness of these bounds.

The AMLI-cycle was originally developed in [2, 3, 30] for multilevel preconditioners based on recursive 2×2 block incomplete factorizations, and was further adapted to multigrid methods in [31]. We do not bring additional contributions to that topic and therefore present only a short summary here, focusing on facts and practical aspects, whereas justifications are omitted and theoretical properties are stated without proof. We essentially follow the presentation in section 5.6 of [31], to which we refer the reader for more details.

First we introduce some additional notation. We assume that the multigrid preconditioner involves L levels, level 1 being the finest (on which the system (1.1) is solved) and level L being the coarsest. A_ℓ, M_ℓ, and P_ℓ are, respectively, the matrix, the smoother, and the prolongation at level ℓ. Hence we have $A_1 = A$ and, because we assume that the successive coarse grid matrices are of Galerkin type, $A_{\ell+1} = P_\ell^T A_\ell P_\ell$, $\ell = 1, \ldots, L-1$.

Regarding assumptions, the system matrix has to be SPD and the smoother such that

$$\rho\left(I - M_\ell^{-1} A_\ell\right) < 1, \quad \ell = 1, \ldots, L-1. \tag{4.1}$$

Furthermore, as stated above, one should know constants α_ℓ, $\ell = 1, \ldots, L-1$, such that, for all $\mathbf{v}$ of appropriate size,

$$\mathbf{v}^T A_\ell \mathbf{v} \;\le\; \mathbf{v}^T B_{\mathrm{TG},\ell} \mathbf{v} \;\le\; \alpha_\ell \; \mathbf{v}^T A_\ell \mathbf{v}, \tag{4.2}$$

where $B_{TG,\ell}$ is a two-grid preconditioner at level ℓ; that is, the matrix defined in (3.1) with $A = A_\ell$, $M = M_\ell$, $P = P_\ell$, and $A_c = A_{\ell+1}$ on the right-hand side.

Now, as discussed in the preceding section, the smoothers considered in this work are SPD and such that $M_\ell - A_\ell$ is nonnegative definite. Hence the eigenvalues of $M_\ell^{-1} A_\ell$ belong to the interval $(0, 1)$, and the condition (4.1) always holds. On the other hand, we intend to use Theorem 3.2 to guarantee (4.2) with a uniform bound $\bar{\kappa}_{\mathrm{TG}}$ on α_ℓ. We therefore continue the presentation, setting $\alpha_\ell = \bar{\kappa}_{\mathrm{TG}}$ for all ℓ.

In Algorithm 4.1 we make clear how the AMLI-cycle can be used in a practical implementation. As stated above, except for $\ell = L - 1$, the coarse system is solved with γ iterations that use the preconditioner on the next coarse level (see step 4).

Algorithm 4.1. Multigrid preconditioner at level ℓ: $\mathbf{z} \leftarrow B_\ell^{-1}\mathbf{r}$.

(1) Relax with smoother M_ℓ: $\mathbf{z} \leftarrow M_\ell^{-1}\mathbf{r}$
(2) Compute residual: $\mathbf{r} \leftarrow \mathbf{r} - A_\ell\mathbf{z}$
(3) Restrict residual: $\mathbf{r}_c \leftarrow P_\ell^T\mathbf{r}$
(4) Compute (approximate) solution $\widetilde{\mathbf{e}}_c$ to $A_{\ell+1}\mathbf{e}_c = \mathbf{r}_c$
    if $(\ell = L-1)$ $\widetilde{\mathbf{e}}_c = A_L^{-1}\mathbf{r}_c$
    else: Initialize: $\widetilde{\mathbf{e}}_c \leftarrow \mathbf{0}$; $\widetilde{\mathbf{w}}_c \leftarrow \mathbf{r}_c$
        Iterate: **for** $j = 0, \ldots, \gamma-1$ **do**
            if $(j > 0)$ $\mathbf{w}_c \leftarrow A_{\ell+1}\mathbf{v}_c$
            $\mathbf{v}_c \leftarrow B_{\ell+1}^{-1}\mathbf{w}_c$
            $\widetilde{\mathbf{e}}_c \leftarrow \widetilde{\mathbf{e}}_c + \xi_\ell^{(j)}\mathbf{v}_c$
(5) Coarse grid correction: $\mathbf{z} \leftarrow \mathbf{z} + P_\ell\widetilde{\mathbf{e}}_c$
(6) Compute residual: $\mathbf{r} \leftarrow \mathbf{r} - A_\ell\mathbf{z}$
(7) Relax with smoother M_ℓ: $\mathbf{z} \leftarrow \mathbf{z} + M_\ell^{-1}\mathbf{r}$

The weights $\xi_\ell^{(j)}$, $j = 0, \ldots, \gamma-1$, are defined as follows. First, one sets $\kappa_{L-1} = \bar{\kappa}_{\mathrm{TG}}$ and recursively computes (bottom to top)

(4.3)
$$\kappa_\ell = \bar{\kappa}_{\mathrm{TG}} + \bar{\kappa}_{\mathrm{TG}} \frac{\kappa_{\ell+1}(1-\kappa_{\ell+1}^{-1})^\gamma}{\left(\sum_{j=1}^{\gamma}\left(1+\sqrt{\kappa_{\ell+1}^{-1}}\right)^{\gamma-j}\left(1-\sqrt{\kappa_{\ell+1}^{-1}}\right)^{j-1}\right)^2}, \qquad \ell = L-2, \ldots, 1.$$

Then, at each level ℓ, the weights in Algorithm 4.1 are the coefficients of the polynomial

$$p_\ell(t) = \sum_{j=0}^{\gamma-1} \xi_\ell^{(j)}\, t^j = \frac{1}{t}\frac{T_\gamma\left(\frac{1+\kappa_{\ell+1}^{-1}}{1-\kappa_{\ell+1}^{-1}}\right) - T_\gamma\left(\frac{1+\kappa_{\ell+1}^{-1}-2t}{1-\kappa_{\ell+1}^{-1}}\right)}{1+T_\gamma\left(\frac{1+\kappa_{\ell+1}^{-1}}{1-\kappa_{\ell+1}^{-1}}\right)}, \tag{4.4}$$

where T_γ is the Chebyshev polynomial of degree γ, as defined by the recursion

$$T_0(t) = 1\,, \quad T_1(t) = t, \quad \text{and, for } k > 1, \quad T_k(t) = 2t\,T_k(t) - T_{k-1}(t)\,. \tag{4.5}$$

Under the assumptions recalled above, it can be shown that the preconditioner at level ℓ as defined in Algorithm 4.1 satisfies

$$\mathbf{v}^T A_\ell \mathbf{v} \;\le\; \mathbf{v}^T B_\ell \mathbf{v} \;\le\; \kappa_\ell\, \mathbf{v}^T A_\ell \mathbf{v}\,. \tag{4.6}$$

Hence, in particular, κ_1 is an upper bound on the condition number of the multigrid preconditioner used to solve the fine grid linear system.

Further, the analysis of the recursion (4.3) reveals that $\kappa_\ell \ge \kappa_{\ell+1}$. In other words, the fine grid upper bound κ_1 increases with the depth of the recursion, that is, with the number of levels. However, κ_1 is uniformly bounded above if and only if

$$\bar{\kappa}_{\mathrm{TG}} < \gamma^2; \tag{4.7}$$

that is, under this condition, one has

$$\kappa_1 \le \kappa^* = \lim_{L\to\infty} \kappa_1 < \infty\,.$$

Thus, when sharp two-grid bounds are available, the AMLI-cycle allows us to automatically construct a multigrid method with level-independent convergence rate, selecting the number of iterations γ large enough to match the condition (4.7).

However, the computational cost has to be taken into account. Indeed, each application of the preconditioner B_1 at the first (finest) level involves $\gamma - 1$ multiplications by the coarse grid matrix A_2 and γ applications of the preconditioner B_2 at the second level, and each of the latter involves $\gamma - 1$ multiplications by the matrix A_3 and γ applications of the preconditioner B_3 at the third level, and so on. Now, to be more specific, let $nnz(A_\ell)$ be the number of nonzero entries in A_ℓ, and, further, for $\ell = 1, \ldots, L-1$, let $c_\ell^{(mv)} nnz(A_\ell)$ represent the cost of one matrix vector product with A_ℓ, and let $c_\ell^{(sm)} nnz(A_\ell)$ represent the cost of all operations in the above algorithm except step 4 (which is essentially the cost of smoothing operations at level ℓ); moreover, let $c_L nnz(A_L)$ be the cost of the exact solution at level L. The overall cost associated with the top level preconditioner is then

(4.8)
$$\mathcal{W}_1 = c_1^{(sm)} nnz(A_1) + \sum_{\ell=2}^{L-1} \gamma^{\ell-1} \left(c_\ell^{(sm)} + \frac{\gamma - 1}{\gamma} c_\ell^{(mv)} \right) nnz(A_\ell) + \gamma^{L-1} c_L nnz(A_L).$$

Now, as is made clear in section 6, the bandwidth of the smoother is bounded at every level, and, hence, so is $c_\ell^{(sm)}$. On the other hand, the idea is to use enough levels to ensure that the size of A_L is small enough to make c_L of the same order as, say, $c_1^{(sm)}$. As a consequence, there exist constants c_1, c_2 independent of problem peculiarities and such that

$$c_1 \ \leq \ c_1^{(sm)},\ c_L,\ c_\ell^{(sm)} + \tfrac{\gamma-1}{\gamma} c_\ell^{(mv)}, \quad \ell = 2, \ldots, L-1 \ \leq \ c_2.$$

Therefore, setting

$$\mathcal{C}_W = \sum_{\ell=1}^{L} \gamma^{\ell-1} \frac{nnz(A_\ell)}{nnz(A_1)},$$

there holds

$$c_1 \ nnz(A_1)\ \mathcal{C}_W \ \leq \ \mathcal{W}_1 \ \leq \ c_2 \ nnz(A_1)\ \mathcal{C}_W.$$

Hence, while γ has to be chosen sufficiently large so that (4.7) holds, it should also be sufficiently small to ensure that $\mathcal{C}_W$ is not much larger than one. We call the latter number the *weighted complexity*; it can be seen as a generalization of the so-called *operator complexity*

$$\mathcal{C}_A = \sum_{\ell=1}^{L} \frac{nnz(A_\ell)}{nnz(A_1)},$$

which is often used to assess the coarsening of AMG methods (see, e.g., [26, p. 487]). Whereas this operator complexity reflects well the memory requirements of the method, it is representative of its cost only if the so-called V-cycle is used. The weighted complexity is certainly more appropriate in this respect when using an AMLI-cycle.

Now, let τ be such that, for $\ell = 2, \ldots, L$,

$$\frac{nnz(A_\ell)}{nnz(A_{\ell-1})} < \frac{1}{\tau};$$

that is, let τ be a uniform bound on the coarsening factor (with respect to the number of nonzero entries). If

$$\gamma < \tau, \tag{4.9}$$

then $\mathcal{C}_W \leq \frac{1}{1-\gamma/\tau}$, independently of L.

Thus γ should be chosen such that both (4.7) and (4.9) hold. The first condition ensures that the number of iterations is bounded independently of L, whereas the second guarantees an optimal bound on the cost of each iteration. Obviously, these requirements are conflicting and such a γ might well not exist. In the context of this work, this can be seen as a constraint on the coarsening process: it should be designed in such a way that both (4.7) and (4.9) hold for some γ. In this respect, note also that in practice these conditions have to be interpreted in a rather restrictive sense: actually, one needs $\frac{\bar{\kappa}_{\rm TG}}{\gamma^2}$ *significantly below one* for having the bound κ^* on the multigrid condition number not much larger than $\bar{\kappa}_{\rm TG}$, and also $\frac{\gamma}{\tau}$ *significantly below one* for having a meaningful bound on $\mathcal{C}_W$.

5. Aggregation procedure.

5.1. Preliminaries. AMG methods are often designed in such a way as to reproduce the behavior of a geometric multigrid method on some model problems. Along the same lines, and before we focus on practical details of the aggregation scheme, it is instructive to determine which kind of regular aggregation pattern would allow us to satisfy the two conflicting requirements stated in the previous section.

In this view, consider the matrix associated with a 3D Cartesian grid equipped with a seven-point symmetric stencil, as obtained from the finite difference discretization of the PDE

$$-\alpha_x \frac{\partial^2 u}{\partial x^2} - \alpha_y \frac{\partial^2 u}{\partial y^2} - \alpha_z \frac{\partial^2 u}{\partial z^2} = f$$

in the interior of any region in which the (positive) coefficients $\alpha_x, \alpha_y, \alpha_z$ are constant. Note that for the discussion below neither the matrix scaling nor the direction of the strongest coefficient is important. Therefore, we assume $\alpha_x = 1 \geq \alpha_y, \alpha_z$, without loss of generality.

The aggregation scheme considered in this work is based on a few passes of a pairwise matching algorithm, which groups unknowns into pairs. Therefore, we consider model aggregates that can be obtained with two or three passes, that is, the size 4 and size 8 aggregates depicted in Figure 1, oriented in any direction. In Figure 2, we represent on the left the smallest quality $\mu(G)$ from all possible such size 4 aggregates, as a function of α_y and α_z. We do the same on the right for size 8 aggregates. The qualities $\mu(G)$ are computed according to (3.8) with respect to the block diagonal or band smoother. We consider the aggregate of the best quality for each value of α_y and α_z because of our intention to design an aggregation algorithm that automatically adapts the aggregate's shape to match a given quality requirement.

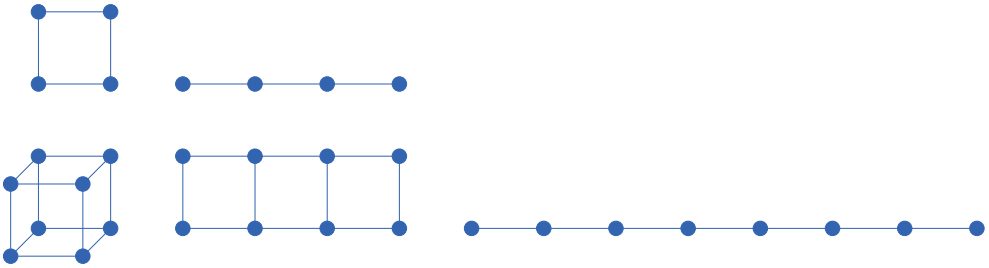

FIG. 1. *Model aggregate shapes.*

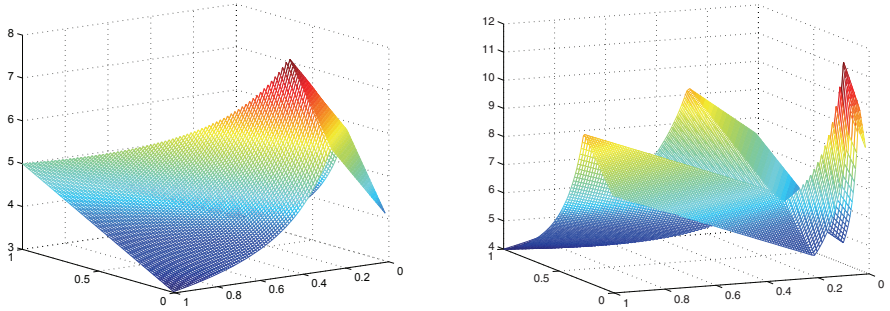

FIG. 2. *Smallest $\mu(G)$ from all possible aggregates of shape depicted in Figure* 1*, as a function of α_y and α_z. Left: size* 4 *aggregates; right: size* 8 *aggregates.*

We first discuss size 4 aggregates. The best quality $\mu(G)$ is most often between 4 and 5, but the function grows up to 7.65 in some region. It means that an aggregation procedure driven by our analysis should use a quality threshold $\bar{\kappa}_{\mathrm{TG}}$ at least as large as 7.65 to guarantee that aggregates of size at least 4 can be formed for any α_x, α_y, α_z. Now, the coarsening factor as defined in the previous section is about equal to the aggregates size. The only integer γ such that both conflicting requirements $\bar{\kappa}_{\mathrm{TG}} \approx 7.65 < \gamma^2$ and $\gamma < 4$ hold is $\gamma = 3$. However, this would not be very cost effective. Indeed, the corresponding bound on the multigrid condition number is $\kappa^* = 31.17$, whereas, even with a coarsening factor effectively around 4, one gets a relatively high complexity $\mathcal{C}_W \approx 4$.

This leads us to consider size 8 aggregates. In this case, the best quality $\mu(G)$ is most often around 5–6, with a peak value equal to 11.46 for $\alpha_x, \alpha_y \approx 0.07$. Setting $\bar{\kappa}_{\mathrm{TG}} = 11.5$, it is possible to use $\gamma = 4$. Then, the condition (4.7) holds and the limit value is $\kappa^* = 27.06$, yielding the upper bound (2.2) stated in section 2. On the other hand, if one effectively succeeds in forming size 8 aggregates almost everywhere, one will get $\mathcal{C}_W \approx 2$. This is a sensible target, especially if one takes into account that the smoothing procedure is cheap (and fast if properly implemented).

Before presenting the algorithm that implements this strategy, let us remark that the above discussion also implicitly includes 2D model problems. Indeed, it turns out that the best quality from all possible aggregates in a 2D grid coincides with that in a 3D grid in the limit case of one vanishing coefficient. Of course, as one can see from Figure 1, restricting the discussion to this limit case would allow smaller threshold $\bar{\kappa}_{\mathrm{TG}}$ and hence possibly some other choice for γ. We did not pursue this direction because we aimed at a truly algebraic procedure, which should therefore work regardless of the dimensionality of the underlying PDE.

5.2. Automatic aggregation. A naive approach when trying to build high quality aggregates of desired size would be to explore the matrix graph while testing all suitable aggregates. This is a costly strategy because of the often large number of possibilities and the need to compute the quality estimate for each of them. An alternative would be to guess a few potentially interesting aggregates of targeted size or larger and choose the most appropriate.

Here we follow the latter idea by considering an aggregation procedure based on a few passes of a pairwise algorithm, which attempts to group unknowns into pairs. This approach is inspired by the double pairwise aggregation algorithm in [22], which, although based on heuristic arguments, was found to be capable of building sensible aggregates in an inexpensive way. Further motivation comes from the fact that, when there are only two unknowns in G, the quality $\mu(G)$ as defined in (3.8) reduces to an easy-to-compute function involving only the off-diagonal entry connecting these two unknowns, their respective diagonal entries, and the sum of all off-diagonal elements in the corresponding rows. Therefore, precomputing the latter, it is inexpensive to find the best pair that contains a given unknown.

On the other hand, we observe that, whereas assessing $\mu(G)$ for larger $|G|$ may become costly, it remains relatively cheap to check that $\mu(G) \leq \bar{\kappa}_{\mathrm{TG}}$ for a given threshold. Indeed, this condition holds if and only if

$$\bar{\kappa}_{\mathrm{TG}}\, A_G - M_G \left(I - \mathbf{1}_G(\mathbf{1}_G^T M_G \mathbf{1}_G)^{-1}\mathbf{1}_G^T M_G\right)$$

is nonnegative definite, which is true if and only if the Cholesky factorization of this matrix exists (i.e., no pivot is negative). Hence, requirement (ii) of Theorem 3.2 can be checked in only $\mathcal{O}(|G|^3)$ operations. This is taken into account in our aggregation procedure, which allows us to ensure that all aggregates satisfy the needed quality requirement, and hence that the relation (3.11) holds, while avoiding an explicit computation of $\mu(G)$ for any subset G with more than two unknowns.

Now the initial pairwise aggregation as applied during the first pass is presented in Algorithm 5.1. It is largely inspired by Theorem 3.2. One first forms the set G_0 of unknowns that can be kept outside the aggregation by checking whether the condition (3.9) holds. Next one picks up one unknown at a time and searches, among its still unassigned neighbors, for the unknown yielding the pair with the best quality. Then, it is checked whether or not this quality satisfies the acceptance criterion; if not, the unknown initially picked up remains unassociated in the coarse grid. It is shown in Appendix A that the given expression for $\mu(\{i,j\})$ indeed matches the definition (3.8).

To obtain larger aggregates, we compute the auxiliary coarse grid matrix $\widetilde{A} = (\widetilde{a})_{ij}$ corresponding to this initial pairwise aggregation. Then, we apply essentially the same algorithm to this matrix to form pairs of pairs, or, in subsequent applications, pairs of aggregates from the previous pass.

However, some care is needed because both conditions (3.9) and (3.10) are to be checked with respect to the initial matrix. In particular, the set G_0 has to remain the one initially defined during the first pass. Furthermore, the estimate $\widetilde{\mu}(\{i,j\})$ used to assess the quality of the pair $\{i,j\}$ has to be adapted so as to correctly reflect the quality of the corresponding aggregate $\mu(G_i \cup G_j)$ in the original matrix. This is obtained by using the same formula as in Algorithm 5.1 but slightly changing the definition of $\widetilde{s}_i$. This change is needed to ensure that $\widetilde{\mu}(\{i,j\})$ is a *lower* bound on $\mu(G_i \cup G_j)$ (see Appendix B for a proof). It means that if $\widetilde{\mu}(\{i,j\})$ is above the threshold, the pair should be rejected anyway because $\mu(G_i \cup G_j)$ cannot be smaller

Algorithm 5.1. Initial pairwise aggregation.

input: $n \times n$ matrix $A = (a_{ij})$
threshold $\bar{\kappa}_{\mathrm{TG}}$

output: n_c and sets $G_0, \ldots, G_{n_c}$

(0) Initialize: $G_0 = \{i \,|\, a_{ii} \;\geq\; \frac{\bar{\kappa}_{\mathrm{TG}}+1}{\bar{\kappa}_{\mathrm{TG}}-1}(\sum_{k=1,k\neq i}^{n} |a_{ik}|)\}$

(1) $U \leftarrow [1,n]\backslash G_0$
$n_c \leftarrow 0$
$s_i \leftarrow -\sum_{j\neq i} a_{ij}$ for all $i \in U$

(2) Iterate: **while** $U \neq \emptyset$ **do**

(2a) Select $i \in U$

(2b) Find $j \in U\backslash\{i\}$ such that $a_{ij} \neq 0$ and

$$\mu(\{i,j\}) = \frac{-a_{ij} + \left(\frac{1}{a_{ii}+s_i+2a_{ij}} + \frac{1}{a_{jj}+s_j+2a_{ij}}\right)^{-1}}{-a_{ij} + \left(\frac{1}{a_{ii}-s_i} + \frac{1}{a_{jj}-s_j}\right)^{-1}}$$

is minimal

(2c) $n_c \leftarrow n_c + 1$

(2d) **if** $(\mu(\{i,j\}) \leq \bar{\kappa}_{\mathrm{TG}})$ $G_{n_c} = \{i,j\}$; $U \leftarrow U\backslash\{i,j\}$
else $G_{n_c} = \{i\}$; $U \leftarrow U\backslash\{i\}$

than $\bar{\kappa}_{\mathrm{TG}}$. It also means that $\widetilde{\mu}(\{i,j\}) \leq \bar{\kappa}_{\mathrm{TG}}$ can only be a preliminary acceptance criterion, which has to be supplemented by an explicit check that $\mu(G_i \cup G_j) \leq \bar{\kappa}_{\mathrm{TG}}$; however, as indicated above, this can be done by factorizing a $|G_i \cup G_j| \times |G_i \cup G_j|$ matrix. These considerations lead to Algorithm 5.2.

Eventually, we make explicit in Algorithm 5.3 how these pairwise aggregation procedures are put together. An uncommon feature is that one can perform an arbitrary number of passes of pairwise aggregation without degrading the upper bound on the condition number. In practice, the process is stopped either if the specified maximal number of passes has been reached or once the coarsening factor is above a given target.

5.3. Priority rules. So far, we have not discussed how to select the unknown in U at step (2a) of the pairwise aggregation algorithms. For a given unknown i, there also may be several neighbors j for which $\mu(\{i,j\})$ or $\widetilde{\mu}(\{i,j\})$ is minimal. If no priority rules are specified, the resulting aggregation will be sensitive to the ordering of the unknowns and/or the way in which off-diagonal entries are stored.

Now, note that whereas the regularity of the aggregation is not an objective in itself, in practice it favors the coarsening speed at the subsequent coarse levels. Hence, after having tested several priority rules inspired from those in [26, 22], we found that the best results were obtained with a simple rule which primarily aims at producing regular aggregation patterns on regular grids. Somewhat surprisingly, good performance carries over to problems on unstructured grids: in such cases results obtained with the rule given below are at least as good as those obtained with more sophisticated choices based on the dynamic update of the unknowns' degree.

The rule is as follows for the initial pairwise aggregation of the top level (fine grid) matrix. We first compute a Cuthill–McKee (CMK) permutation [11]; that is, we assign the number 1 to a node with minimal degree, and we assign the next numbers to its neighbors, ordered again by increasing degree; then, we number the

Algorithm 5.2. Further pairwise aggregation.

input: $n \times n$ matrix $A = (a_{ij})$
threshold $\bar{\kappa}_{\rm TG}$
tentative $\widetilde{n}_c$ and sets $\widetilde{G}_k$, $k = 1, \ldots, n_c$
corresponding $\widetilde{n}_c \times \widetilde{n}_c$ matrix $\widetilde{A} = (\widetilde{a}_{ij})$

output: n_c and sets $G_1, \ldots, G_{n_c}$

(1) Initialize: $U \leftarrow [1, \widetilde{n}_c]$
$n_c \leftarrow 0$
$\widetilde{s}_i \leftarrow -\sum_{k \in G_i} \sum_{j \notin G_i} a_{kj}$ for all $i \in U$

(2) Iterate: **while** $U \neq \emptyset$ **do**

(2a) Select $i \in U$

(2b) Set $T = \{j \mid \widetilde{a}_{ij} \neq 0 \text{ and } \widetilde{\mu}(\{i,j\}) \leq \bar{\kappa}_{\rm TG}\}$, where

$$\widetilde{\mu}(\{i,j\}) = \frac{-\widetilde{a}_{ij} + \left(\frac{1}{\widetilde{a}_{ii}+\widetilde{s}_i+2\widetilde{a}_{ij}} + \frac{1}{\widetilde{a}_{jj}+\widetilde{s}_j+2\widetilde{a}_{ij}} \right)^{-1}}{-\widetilde{a}_{ij} + \left(\frac{1}{\widetilde{a}_{ii}-\widetilde{s}_i} + \frac{1}{\widetilde{a}_{jj}-\widetilde{s}_j} \right)^{-1}}$$

(2c) $n_c \leftarrow n_c + 1$

(2b) **if** $(T \neq \emptyset)$
Select $j \in T$ with minimal $\widetilde{\mu}(\{i,j\})$
if $(\mu(\widetilde{G}_i \cup \widetilde{G}_j) \leq \bar{\kappa}_{\rm TG})$ $G_{n_c} = \widetilde{G}_i \cup \widetilde{G}_j$; $U \leftarrow U \backslash \{i,j\}$
else $T \leftarrow T \backslash \{j\}$; **goto** step (2b)
else $G_{n_c} = \widetilde{G}_i$; $U \leftarrow U \backslash \{i\}$

Algorithm 5.3. Multiple pairwise aggregation.

input: $n \times n$ matrix A
threshold $\bar{\kappa}_{\rm TG}$
maximal number of passes $n_{\rm pass}$
target coarsening factor τ

output: n_c and sets $G_0, \ldots, G_{n_c}$
corresponding aggregation matrix A_c

(1) First pass: Apply *Initial pairwise aggregation* algorithm to matrix A with threshold $\bar{\kappa}_{\rm TG}$;
Output: $n_c^{(1)}$, G_0, and $G_1^{(1)}, \ldots, G_{n_c}^{(1)}$;

(2) Form $A^{(1)} \leftarrow P^T A P$ with P defined via (3.2) with respect to $G_1^{(1)}, \ldots, G_{n_c}^{(1)}$

(3) Iterate: **for** $s = 2, \ldots, n_{\rm pass}$

(3a) Next passes: Apply *Further pairwise aggregation* algorithm to matrix A with threshold $\bar{\kappa}_{\rm TG}$, $\widetilde{n}_c = n_c^{(s-1)}$, $\widetilde{G}_k = G_k^{(s-1)}$, $k = 1, \ldots, n_c^{(s-1)}$, and $\widetilde{A} = A^{(s-1)}$;
Output: $n_c^{(s)}$, $G_1^{(s)}, \ldots, G_{n_c}^{(s)}$;

(3b) Form $A^{(s)} \leftarrow P^T A P$ with P defined via (3.2) with respect to $G_1^{(s)}, \ldots, G_{n_c}^{(s)}$

(3c) **if** $(nnz(A^{(s)}) \leq \frac{nnz(A)}{\tau})$ **goto** step (4)

(4) $n_c \leftarrow n_c^{(s)}$, $G_k \leftarrow G_k^{(s)}$, $k = 1, \ldots, n_c$, and $A_c \leftarrow A^{(s)}$

15	19	22	24	25
10	14	18	21	23
6	9	13	17	20
3	5	8	12	16
1	2	4	7	11

FIG. 3. *CMK ordering for a five-point stencil on a* 5×5 *grid.*

unnumbered neighbors of the node that has just been given the number 2, still by increasing degree; we next proceed with the neighbors of node 3, and so on until all nodes are numbered. For example, when applying this procedure to a rectangular grid equipped with a five-point stencil, one obtains an ordering as illustrated in Figure 3.

Note that there is still some uncertainty in this process, since there are in general several nodes with minimal degree, and hence several possible starting nodes. A given node may also have several unnumbered neighbors with same degree. However, the impact of the choices made here seems minimal. For instance, consider the example of Figure 3. The only possible starting nodes are the four corner nodes. Once the algorithm has started from, say (as illustrated), the bottom left corner node, we can assign the number 2 to either its northern or eastern neighbor—the eastern one in the case of Figure 3. This is, however, the last step where some freedom is left: all nodes processed subsequently have only one unnumbered neighbor, except those on the bottom line of the grid, but their neighbor which is itself on the bottom line has a smaller degree and therefore should be numbered first. Moreover, with other choices during the first two steps one would obtain orderings with identical structure, just rotated and/or mirrored. Hence, the remaining choices seem unimportant, and we do not specify how they should be made. Note that we tested the consistency of our approach by repeating the numerical experiments in the next section with randomized choices anywhere it was possible, obtaining for each problem nearly identical performances.

Now, once this CMK permutation has been computed, we apply the initial pairwise aggregation algorithm, always giving priority to the node with the smallest number in this CMK permutation. This rule is used in Algorithm 5.1 both at step (2a) to select the unknown i in U and at step (2b) to discriminate between neighbors j yielding the same quality (up to rounding errors) for the pair $\{i, j\}$.

For the subsequent passes (Algorithm 5.2), and also for all passes at the coarser levels, we note that the ordering in the matrix at hand is driven by the ordering in which the pairs have been formed, which, during the first pass, is itself driven by the CMK permutation. Hence we simply give priority at any stage to the node with the smallest number in the current ordering. Proceeding recursively in this way, the initial CMK permutation induces the choices throughout all levels.

We illustrate in Figure 4 the performance of the resulting aggregation algorithm with three passes at each level. This configuration is obtained, e.g., by using Algorithm 5.3 with $\bar{\kappa}_{\mathrm{TG}} = 11.5$, $\tau = 8$, and $n_{\mathrm{pass}} \geq 3$. The first three pictures show the aggregation pattern produced by these three successive passes at the top level, whereas the last picture (bottom right) gives at once the result of the multiple pairwise aggregation at the next level.

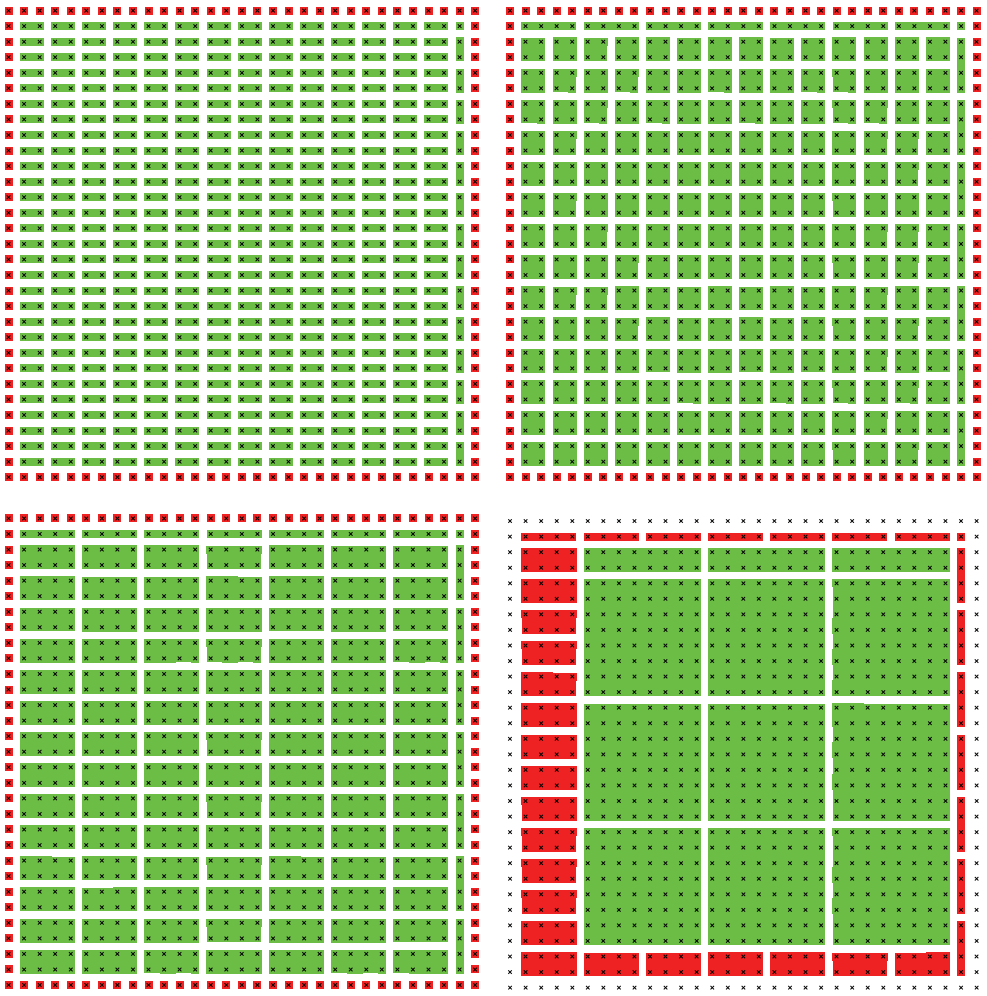

FIG. 4. *Aggregation for the model Poisson problem on the unit square with* $h = 1/32$ *; dark (red) boxes correspond to nodes in* G_0*, whereas light gray (green) boxes correspond to aggregates. Top: after one (left) and two (right) pairwise aggregations passes applied to the level* 1 *matrix. Bottom: the complete result of the multiple pairwise aggregation algorithm with three passes at level* 1 *(left) and level* 2 *(right); in the right-hand picture, nodes that are not in any box are those in* G_0 *at the previous level and hence are not represented anymore in the coarse grid matrix.*

5.4. Model problem analysis. One sees in Figure 4 that after two successive aggregation steps with three passes each, the coarse grid has a regular structure similar to the one on the fine grid. One may then wonder if this regularity depends on the problem size. In Figure 5 we display the aggregation pattern obtained for the same problem with $h = 1/64$. Clearly, the structure of the resulting coarse grid is similar to that for $h = 1/32$. This is not surprising. With our algorithm, aggregates are formed in a regular fashion starting from the corner in which the CMK ordering has been initiated. Irregularities occur only when reaching opposite boundaries. But if two grids with $n_x^{(1)} \times n_y^{(1)}$ and $n_x^{(2)} \times n_y^{(2)}$ nodes are such that $|n_x^{(1)} - n_x^{(2)}|$ and $|n_y^{(1)} - n_y^{(2)}|$ are multiples of 8, these irregularities will be treated in exactly the same way and hence the resulting matrices will have similar structure. Therefore, generally speaking,

FIG. 5. *Aggregation for the model Poisson problem on the unit square with* $h = 1/64$*: result of the multiple pairwise aggregation algorithm with three passes at level* 1 *(left) and level* 2 *(right).*

starting from a matrix corresponding to $h = 2^{-k}$ (that is, a grid with $(2^k-1)\times(2^k-1)$ nodes), one obtains a $(2^{k-3}-1)\times 2^{k-3}$ grid with five-point connectivity; this holds for any $k \geq 5$. Note that, starting from a square grid we obtain a rectangular grid with rectangle "orientation" depending on which neighbor of the starting corner node has been numbered first when computing the CMK ordering (the discussion above assumes that the CMK permutation is as in Figure 3).

Now, one may wonder if these observations can be recursively applied to anticipate the result of further coarsening steps. Here, a key argument is that when discarding the (lines and columns of) nodes that are closest to boundaries (which will be assigned to G_0 in a further aggregation process), all remaining aggregates are square, regularly aligned, and have the same size. Hence (see (3.3)), the entries in the stencil will be those of the standard five-point stencil we have started from, up to a scaling factor. Moreover, one may check that the ordering of this coarse grid matrix is a CMK ordering, similar to (and induced by) the fine grid one. Hence a recursive application of the above observations is possible, providing that they can be extended to cases where the starting grid is a $(2^k-1)\times 2^k$ grid, instead of a $(2^k-1)\times(2^k-1)$ grid as considered so far. In Figure 6, we display the aggregation pattern obtained for such a 63×64 grid. One sees that the coarse grid (and, hence, the associated coarse grid matrix) after two steps corresponds to a 7×8 grid and is in fact identical to that obtained in Figure 5 when coarsening the matrix associated to the 63×63 grid. Moreover, one may check that the ordering associated to every coarse grid node is identical in both cases. We have therefore shown the essential part of the following proposition.

PROPOSITION 5.1. *Let* $A = A_1$ *be the matrix corresponding to the five-point discretization of the Poisson equation on the unit square with uniform mesh size* $h = 2^{-k}$ *for some positive integer* k*. Consider* ℓ *successive applications of the multiple pairwise aggregation algorithm with three passes, using the priority rules described in section* 5.3*. If* ℓ *is a positive even integer such that* $k - \frac{3\ell}{2} \geq 2$*, then the resulting coarse grid matrix* A_ℓ *corresponds to a five-point stencil on a rectangular grid with*

FIG. 6. *Aggregation for the model Poisson problem on the* $1 \times (1+h)$ *rectangle with* $h = 1/64$*: result of the multiple pairwise aggregation algorithm with three passes at level* 1 *(left) and level* 2 *(right).*

$(2^{k-\frac{3\ell}{2}} - 1) \times 2^{k-\frac{3\ell}{2}}$ *nodes, and one has*

$$nnz(A_\ell) \leq 8^{-\ell}\, nnz(A). \tag{5.1}$$

Proof. The discussion above shows the main result, and it remains to prove (5.1). We have $nnz(A_1) = 5(2^k - 1)^2 - 4(2^k - 1)$, whereas

$$nnz(A_\ell) = 5(2^{k-\frac{3\ell}{2}} - 1)2^{k-\frac{3\ell}{2}} - 2(2^{k-\frac{3\ell}{2}} - 1) - 2 \cdot 2^{k-\frac{3\ell}{2}}.$$

Hence, since $k \geq \frac{3\ell}{2} + 2$, by assumption

$$\begin{aligned} nnz(A_1) - 2^{3\ell}\, nnz(A_\ell) &= 2^k(9 \cdot 2^{\frac{3\ell}{2}} - 14) + 9 - 2 \cdot 2^{3\ell} \\ &\geq 4(9 \cdot 2^{3\ell} - 14 \cdot 2^{\frac{3\ell}{2}}) + 9 - 2 \cdot 2^{3\ell} \\ &= 34 \cdot 2^{3\ell} - 56 \cdot 2^{\frac{3\ell}{2}} + 9. \end{aligned}$$

The latter (bottommost) expression is a second degree polynomial in $2^{\frac{3\ell}{2}}$, which is indeed always positive when the argument is not less than 8, as ensured by the condition ℓ positive and even, which entails $\ell \geq 2$. ∎

5.5. Illustration. We conclude this section with an illustration of our aggregation procedure on an example with unstructured mesh. More precisely, we consider the linear finite element discretization of

$$\begin{cases} -\frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial y^2} = 0 & \text{on } \Omega = [-1,1]^2 \backslash [0,1] \times [-1,0], \\ u = r^{\frac{2}{3}} \sin(\frac{2\theta}{3}) & \text{on } \partial\Omega, \end{cases} \tag{5.2}$$

where (r, θ) is the polar coordinate representation of (x, y). The discretization is performed on the mesh illustrated in Figure 7, in which the simplex size is progressively decreased near the reentering corner, in such a way that the mesh size in its neighborhood is about 10 times smaller.

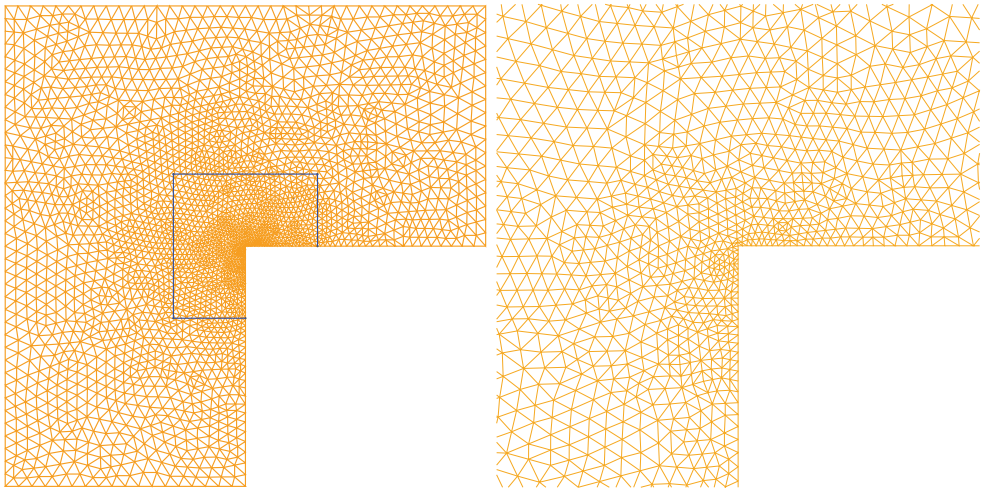

FIG. 7. *L-shaped domain with unstructured mesh refined near the reentering corner (left) and zoom on the region near the corner delimited with blue (dark) contour (right).*

The result of the application of Algorithm 5.3 to this problem is illustrated in Figure 8. Priority rules are as indicated in subsection 5.3, and the parameters are threshold $\bar{\kappa}_{\mathrm{TG}} = 11.5$, maximal number of passes $n_{\mathrm{pass}} = 5$, and target coarsening factor $\tau = 8$. In practice, four passes have been necessary at level 1, three at level 2, and only two at level 3 (thanks to the large number of nodes in G_0). Hence the coarsening factor is in all cases above the target, but this sometimes requires more than the three passes needed in the case of a regular grid. In fact, as one can see in the figure, especially at the top level, the aggregates here are far from "geometric," and, if the mean aggregate size is above 8, the actual aggregate size ranges from 1 to 16.

6. Numerical experiments. Our first set of test problems corresponds to finite element and finite difference discretizations on structured grids with uniform mesh size h in all directions; all problems are defined on a unit square (2D) or a unit cube (3D). We give only brief descriptions of these problems here and refer the reader to [22] for further details. Our test problems include

- five-point discretization of constant coefficients problem in two dimensions with coefficients $(1\,,\,\varepsilon_y)$; we consider $\varepsilon_y = 1$ (MOD2D), $\varepsilon_y = 10^{-2}$ (ANI2D$_a$), and $\varepsilon_y = 10^{-4}$ (ANI2D$_b$);
- seven-point discretization of constant coefficients problem in three dimensions with coefficients $(\varepsilon_x\,,\,\varepsilon_y\,,\,1)$; we consider ε_x, $\varepsilon_y = 1$ (MOD3D), $\varepsilon_x = 0.07$, $\varepsilon_y = 1$ (ANI3D$_a$), $\varepsilon_x = 0.07$, $\varepsilon_y = 0.25$ (ANI3D$_b$), $\varepsilon_x = 0.07$, $\varepsilon_y = 0.07$ (ANI3D$_c$), $\varepsilon_x = 0.005$, $\varepsilon_y = 1$ (ANI3D$_d$), $\varepsilon_x = 0.005, \varepsilon_y = 0.07$ (ANI3D$_e$), $\varepsilon_x = 0.005$, $\varepsilon_y = 0.005$ (ANI3D$_f$);
- five- and seven-point discretizations of piecewise-constant coefficient problem in two dimensions (JUMP2D) and in three dimensions (JUMP3D);
- nine-point bilinear finite element discretization of isotropic constant coefficient problem in two dimensions (BFE).

The remaining test problems consist of finite element discretizations on (mainly) unstructured grids.

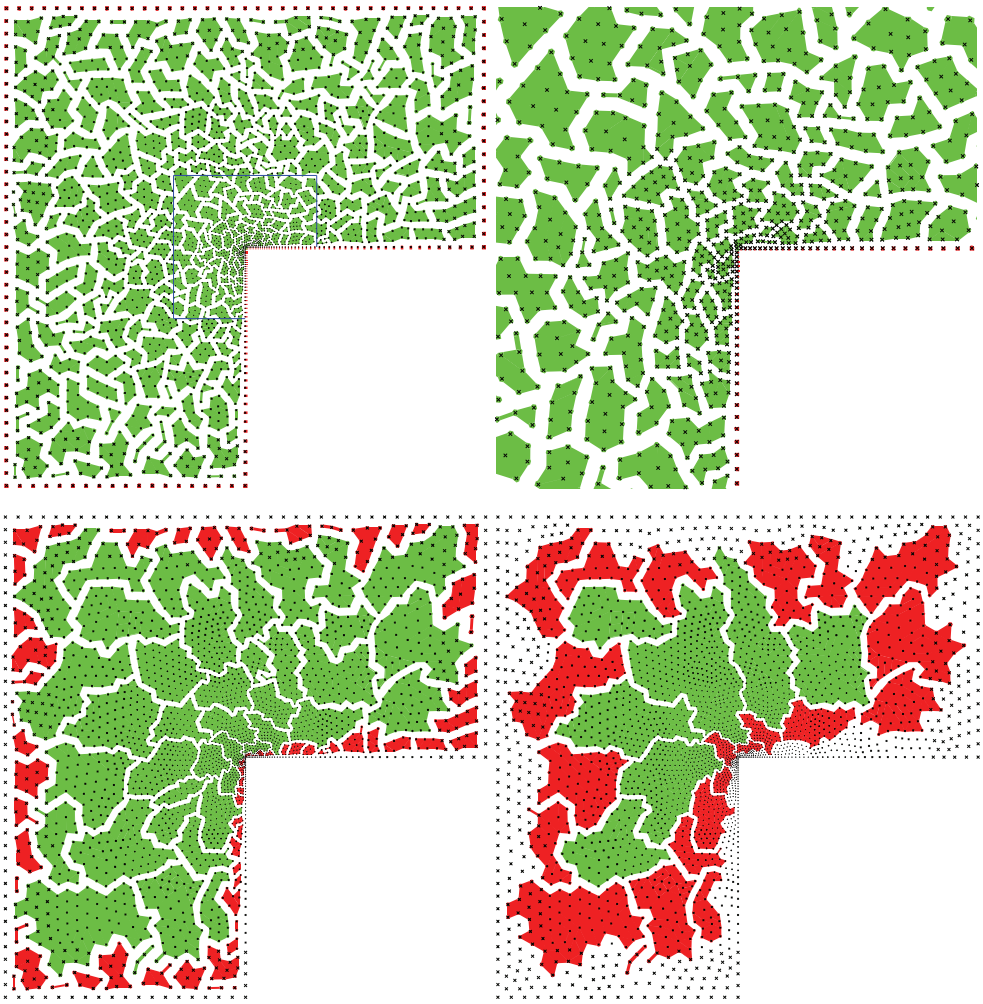

FIG. 8. *Aggregation for the problem* (5.2) *discretized on the mesh depicted in Figure* 7. *Top: after one application of the multiple pairwise aggregation algorithm (left) and zoom near the reentering corner (right). Bottom: after two (left) and three (right) applications of the multiple pairwise aggregation algorithm.*

Problem LUNFST: *Linear finite element discretization of* (5.2) *on a uniform mesh with right triangles.*

Problem LRFUST$_r$: *Linear finite element discretization of* (5.2) *on an unstructured mesh with simplex size progressively decreased near the reentering corner, in such a way that the mesh size in its neighborhood is about* $10^{\,r}$ *times smaller, with* r *going from* 0 *(*LRFUST$_0$*) to* 5 *(*LRFUST$_5$*); for* $r = 1$*, this is the mesh illustrated in Figure* 7*, further uniformly refined four times (size S1) or five times (size S2).*

Problem SPHUNF$_d$: *Linear finite element discretization of*

$$\begin{cases} -\nabla \cdot d\nabla u = 0 & \text{on } \widetilde{\Omega} = \{\mathbf{r} = (x,\ y,\ z)^T \mid \|\mathbf{r}\| < 1\}, \\ -\nabla \cdot \nabla u = 0 & \text{on } \Omega = [-2.5, 2.5]^3 \backslash \widetilde{\Omega}, \\ u = 0 & \text{for } |x| = 2.5,\ -2.5 \le y,\ z \le 2.5, \\ u = 1 & \text{elsewhere on } \partial\Omega \end{cases}$$

TABLE 1
Problem sizes.

Problem	*S1*		*S2*	
	$\frac{n}{10^6}$	$\frac{nnz(A)}{10^6}$	$\frac{n}{10^6}$	$\frac{nnz(A)}{10^6}$
LUNFST	0.20	1.0	3.1	16
LRFUST	$0.72 \to 1.8$	$5.1 \to 12$	$2.9 \to 7.1$	$20 \to 50$
SPHUNF	0.53	7.6	4.2	61
SPHRF	0.15	2.2	1.2	18

with $\nabla = \mathbf{1}_x \frac{\partial}{\partial x} + \mathbf{1}_y \frac{\partial}{\partial y} + \mathbf{1}_z \frac{\partial}{\partial z}$ *on an unstructured quasi-uniform mesh; we consider* $d = 10^{-3}$ *(*SPHUNF$_{-3}$*),* $d = 1$ *(*SPHUNF$_0$*), and* $d = 10^3$ *(*SPHUNF$_3$*).*

Problem SPHRF$_d$: *Same as above, but with five times smaller sphere* $\widetilde{\Omega}$ *having* 10 *times smaller simplices near its surface.*

Every test problem may lead to systems with variable size n. For finite difference problems on structured grids we consider three system sizes:

- *S1*, corresponding to $h^{-1} = 600$ in two dimensions ($n \approx 0.36\,10^6$) and $h^{-1} = 80$ in three dimensions ($n \approx 0.51\,10^6$);
- *S2*, corresponding to $h^{-1} = 1600$ in two dimensions ($n \approx 2.5\,10^6$) and $h^{-1} = 160$ in three dimensions ($n \approx 4.1\,10^6$);
- *S3*, corresponding to $h^{-1} = 5000$ in two dimensions ($n \approx 25\,10^6$) and $h^{-1} = 320$ in three dimensions ($n \approx 33\,10^6$).

Regarding finite element discretizations on unstructured grids, we use only two sizes: *S1* and *S2*, which roughly correspond to $\mathcal{O}(10^5)$ and $\mathcal{O}(10^6)$ unknowns, respectively; see Table 1 for more details.

Our method has been applied to all test problems in a uniform (black box) fashion. The multilevel algorithm was used as a preconditioner for the conjugate gradient method and implemented as in Algorithm 4.1, using the band smoother described in section 3. The AMLI cycle was used with $\gamma = 4$ iterations on the coarse levels and with weights $\xi_\ell^{(j)}$ being the coefficients of polynomials (4.4) based on the recursive estimate (4.3) with $\bar{\kappa}_{\mathrm{TG}} = 11.5$. For the coarsening, we uniformly use the same parameters as in section 5.5: threshold $\bar{\kappa}_{\mathrm{TG}} = 11.5$, maximal number of passes $n_{\mathrm{pass}} = 5$, and target coarsening factor $\tau = 8$. It means that one may occasionally obtain a few aggregates of size much larger than the target (up to 32). We therefore slightly modified Algorithm 5.2 to avoid a harmful impact on the smoother's bandwidth; that is, we further reject at step (2b) also those pairs whose acceptance would induce a bandwidth larger than 10 for the band smoother. This turns out to have a limited impact on the coarsening (only aggregates of size larger than 10 can be rejected) while ensuring a reasonable computational cost of the smoothing iterations. We also checked that this modification has no impact on the coarsening depicted in Figure 8; i.e., although there are some aggregates of size 10 to 16, none of them has a bandwidth larger than 10.

The conjugate gradient method was used with the zero vector as initial approximation, and iterations were stopped when the relative residual error was below 10^{-6}. Results are reported in Tables 2 and 3. All timings are elapsed times when running a Fortran 90 implementation of the method on a computing node with two Intel XEON L5420 processors at 2.50 GHz and 16 Gb RAM memory.

What is perhaps the most striking from these results is the regularity of the method performance. Take, for instance, as reference the MOD2D problem for which our analysis also covers the coarsening speed, and consider the *S3* size. The weighted

TABLE 2
Computational complexity, condition number, and number of iterations needed to reduce the residual norm by 10^{-6}.

	$\mathcal{C}_W$			$\kappa(B_1^{-1}A)$			#Iter		
Problem	*S1*	*S2*	*S3*	*S1*	*S2*	*S3*	*S1*	*S2*	*S3*
MOD2D	1.9	2.0	2.0	7.0	7.4	7.6	23	24	26
MOD3D	1.8	1.9	1.9	6.0	6.1	6.1	18	18	19
ANI2D$_a$	1.7	1.9	1.9	9.1	8.8	8.9	21	25	25
ANI2D$_b$	1.8	1.8	1.9	1.3	2.0	7.5	7	11	16
ANI3D$_a$	1.4	1.4	1.4	6.4	6.9	7.4	20	22	24
ANI3D$_b$	1.5	1.5	1.5	7.1	7.6	8.2	18	19	26
ANI3D$_c$	1.7	1.7	1.8	6.0	6.0	6.4	19	20	21
ANI3D$_d$	1.4	1.4	1.4	10.2	13.1	13.1	26	30	31
ANI3D$_e$	1.4	1.4	1.4	11.5	12.0	12.6	26	28	29
ANI3D$_f$	1.6	1.6	1.9	1.8	7.2	8.5	10	20	23
JUMP2D	1.5	1.5	1.6	10.4	11.2	12.2	27	29	32
JUMP3D	1.5	1.5	1.5	6.5	7.1	7.5	22	24	25
BFE	1.6	1.6	1.6	5.6	5.9	6.1	21	23	24
LRFUST$_0$	1.7	1.7		5.3	6.5		7	9	
LRFUST$_1$	1.7	1.7		5.0	5.6		6	7	
LRFUST$_2$	1.7	1.7		5.0	6.4		6	8	
LRFUST$_3$	1.7	1.7		5.0	6.2		6	8	
LRFUST$_4$	1.7	1.7		5.0	6.2		6	8	
LRFUST$_5$	1.7	1.6		5.1	5.3		6	6	
LUNFST	1.9	2.0		6.2	6.8		16	17	
SPHRF$_{-3}$	2.6	2.8		4.2	4.4		12	12	
SPHRF$_0$	2.6	2.8		4.2	4.4		12	12	
SPHRF$_3$	2.6	2.8		4.2	4.4		12	12	
SPHUNF$_{-3}$	2.5	2.8		4.3	4.8		12	12	
SPHUNF$_0$	2.6	2.8		4.3	4.8		12	12	
SPHUNF$_3$	2.6	2.9		4.3	4.9		12	12	

complexity is actually *smaller* for all other test problems, except 3D problems on an unstructured grid, for which it is at worst 50% larger. The resulting condition number is also at most twice that for MOD2D, and in fact remains in all cases smaller than half of the guaranteed bound (2.2). Along the same lines, the number of iterations ranges between half of and 25% more than that needed to solve the reference model problem. Eventually, the total time per unknown is in all cases within a modest factor of that of the MOD2D problem—a factor which most often is related to a larger number of nonzero entries per row in the matrix.

Note also that this stability in performance holds for both structured and unstructured grid problems, despite the fact that our aggregation algorithm mimics well a "geometric" aggregation algorithm in the structure case but may work quite differently in the unstructured one (compare Figures 4 and 8).

Eventually, we compare the method presented here with different versions of aggregation-based AMG, and also, when feasible, with the sparse direct solver available in MATLAB via the "\" command. The total solution time (including setup) is reported in Table 4 for a representative sample of our test examples. AggGar stands for the method described in the present paper, and AGMG 2.3 is the 2.3 version of the AGMG software [21], which implements the heuristic method from [22]. On the other hand AGMGar uses our new aggregation algorithm but defines the other ingredients essentially as in [22]; that is, the coarsening is obtained using Algorithm 5.3 with threshold $\bar{\kappa}_{\mathrm{TG}} = 8$, maximal number of passes $n_{\mathrm{pass}} = 2$, and target coarsening factor $\tau = 4$, and this is combined with the Gauss–Seidel smoother and the K-cycle (i.e., the coarse system is solved at each level with two conjugate gradient iterations).

TABLE 3

Set-up time, solution time, and total time reported here per million of unknowns (with $\mathcal{T}_{\text{total}} = \mathcal{T}_{\text{sol}} + \mathcal{T}_{\text{setup}}$*).*

	$\mathcal{T}_{\text{setup}}$			$\mathcal{T}_{\text{sol}}$			$\mathcal{T}_{\text{tot}}$		
Problem	*S1*	*S2*	*S3*	*S1*	*S2*	*S3*	*S1*	*S2*	*S3*
MOD2D	1.2	1.1	1.7	6.4	7.6	8.7	7.5	8.8	10.3
MOD3D	1.6	2.6	2.9	5.7	6.5	7.3	7.3	9.1	10.1
ANI2D$_a$	1.8	1.7	2.3	5.1	7.4	8.0	6.8	9.1	10.3
ANI2D$_b$	1.8	1.7	2.5	1.8	3.1	4.9	3.6	4.9	7.4
ANI3D$_a$	2.1	3.6	4.0	5.3	6.5	7.3	7.5	10.1	11.4
ANI3D$_b$	2.4	4.0	4.5	4.8	5.6	8.3	7.1	9.6	12.8
ANI3D$_c$	2.8	4.7	5.2	6.2	7.4	8.1	9.1	12.1	13.3
ANI3D$_d$	2.3	3.8	4.2	6.6	8.6	9.0	8.9	12.4	13.3
ANI3D$_e$	2.4	4.1	4.6	6.3	7.3	7.9	8.6	11.4	12.5
ANI3D$_f$	2.3	4.2	4.6	2.9	6.3	8.2	5.2	10.5	12.8
JUMP2D	1.6	1.7	2.2	8.5	10.4	11.9	10.1	12.1	14.1
JUMP3D	2.2	3.1	3.4	7.2	8.5	9.3	9.4	11.6	12.6
BFE	1.8	1.7	2.2	7.7	9.4	10.3	9.5	11.1	12.5
LRFUST$_0$	2.2	2.2		2.9	3.9		5.1	6.0	
LRFUST$_1$	2.2	2.2		2.5	3.0		4.7	5.2	
LRFUST$_2$	2.1	2.1		2.5	3.6		4.6	5.6	
LRFUST$_3$	2.2	2.1		2.6	3.6		4.7	5.7	
LRFUST$_4$	2.2	2.1		2.5	3.6		4.7	5.8	
LRFUST$_5$	2.2	2.1		2.5	2.6		4.7	4.7	
LUNFST	1.8	1.8		6.1	6.8		7.8	8.7	
SPHRF$_{-3}$	7.5	7.3		8.0	10.0		15.5	17.4	
SPHRF$_0$	6.7	7.3		8.2	10.1		14.9	17.4	
SPHRF$_3$	6.6	7.2		8.1	10.0		14.7	17.3	
SPHUNF$_{-3}$	6.7	7.4		8.6	10.4		15.3	17.9	
SPHUNF$_0$	6.7	7.4		8.7	10.4		15.4	17.8	
SPHUNF$_3$	6.7	7.4		8.8	10.6		15.5	18.0	

TABLE 4

Total solution time for the different versions of aggregation-based AMG and the MATLAB direct solver ("\"), reported per million of unknowns.

	S1	*S2*	*S3*	*S1*	*S2*	*S1*	*S2*	*S1*	*S2*
	MOD2D			LRFUST$_0$		LRFUST$_2$		LRFUST$_4$	
AGMG 2.3	4.7	5.3	6.3	6.9	10.9	9.2	21.1	162.2	–
AggGar	7.5	8.8	10.3	5.1	6.0	4.6	5.6	4.7	5.8
AGMGar	3.4	3.5	3.9	3.3	3.0	3.4	2.8	2.9	2.9
MATLAB "\"	6.2	6.6	–	8.5	10.8	10.0	11.8	9.8	12.5
	MOD3D			SPHRF$_{-3}$		SPHRF$_0$		SPHRF$_3$	
AGMG 2.3	5.0	6.3	6.9	11.0	10.8	10.6	10.9	10.6	10.9
AggGar	7.3	9.1	10.1	15.5	17.4	14.9	17.4	14.7	17.3
AGMGar	3.5	4.3	5.0	8.5	7.4	7.1	7.5	8.4	7.4

One sees that when AGMG 2.3 works well, it requires about 40% less time than the method presented here. However, like any heuristic method, AGMG 2.3 may occasionally fail; see the 2D example with strong local refinement. Then it pays off to use the method with guaranteed convergence rate (AggGar). Further, AGMGar seems to combine the advantages of both approaches. We explain this as follows. The relative slowness of AggGar (compared with AGMG 2.3) is mainly due to the constraints associated with the use of the AMLI-cycle, in particular the need to perform four inner iterations at each level. However, in practice (this cannot be proved), the K-cycle is as efficient as the AMLI-cycle in stabilizing the condition number once the two-grid scheme satisfies some minimal requirements [24] and such minimal requirements are

met when using the new aggregation with the parameters indicated above. Hence combining the new aggregation algorithm with the K-cycle allows AGMGar to be as fast as the heuristic method AGMG 2.3, while preserving the additional robustness associated with the new aggregation algorithm.

On the other hand, the comparison with MATLAB "\" allows us to consider the performances of aggregation-based methods from a more general viewpoint. One sees that AGMG 2.3 and AGMGar are both faster on the model Poisson problem in two dimensions, for which the direct solver is practically scalable and known as "hard to beat." Further, interestingly enough, the AMG variants based on the new coarsening algorithm are apparently more robust in the presence of strong local refinement.

7. Conclusions. We have presented a purely algebraic (black box) multigrid method that has a guaranteed convergence rate for any symmetric M-matrix with nonnegative row sum. Like many AMG schemes, our method has some uncertainty in the coarsening factor and, hence, in the computational cost per iteration. However, we showed that a sufficiently fast coarsening is guaranteed at least for a model 2D problem, whereas numerical results indicate that satisfactory complexities are obtained in quite diverse cases without any parameter tuning.

Moreover, the results obtained with AGMGar (see Table 4) show that the aggregation method presented here can also be useful in a more general context. Of course, AGMGar would deserve a more detailed presentation, and these results raise several further questions, such as the applicability to non–M-matrices. However, we do not pursue this discussion here, as it would lead us outside the scope of the present paper. We only briefly mention that the aggregation algorithms of section 5 *can* actually be applied to SPD matrices that are not M-matrices, i.e., that have some positive off-diagonal entries. If the matrix has nonnegative row sum, the main steps of our analysis still apply, except that the matrix A_r in the splitting $A = A_b + A_r$ referred to in section 3 is no longer guaranteed to be nonnegative definite; hence, the approach corresponds to a sensible heuristic in applications for which this matrix, which still has zero row sum, remains close enough to nonnegative definiteness. On the other hand, it is worth noting that the approach developed here has been further extended in [23] to *nonsymmetric* M-matrices.

Software. The results of this research have been integrated in the AGMG software [21] (released under the terms of the GNU General Public License), whose version 3.1.1 is nearly identical to the code referred to above as AGMGar.

Appendix A. Here we derive an explicit expression for the quality $\mu(G)$ of an aggregate $G = \{i, j\}$. In this context,

$$A_G = \begin{pmatrix} a_{ii} - s_i - a_{ij} & a_{ij} \\ a_{ji} & a_{jj} - s_j - a_{ji} \end{pmatrix} = -a_{ij} \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} + \begin{pmatrix} \delta_1 & \\ & \delta_2 \end{pmatrix},$$

where $s_i = -\sum_{k \neq i} a_{ik}$, $s_j = -\sum_{k \neq j} a_{jk}$, $\delta_1 = a_{ii} - s_i$, and $\delta_2 = a_{jj} - s_j$. Since we consider the block-diagonal (or the band) smoother,

$$M_G = \begin{pmatrix} a_{ii} + s_i + a_{ij} & a_{ij} \\ a_{ji} & a_{jj} + s_j + a_{ji} \end{pmatrix} = -a_{ij} \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} + \begin{pmatrix} \eta_1 & \\ & \eta_2 \end{pmatrix},$$

where $\eta_i = a_{ii} + s_i + 2a_{ij}$, $\eta_j = a_{jj} + s_j + 2a_{ij}$. Using these expressions in (3.8), some

elementary algebra leads to

$$\begin{aligned}\mu(G) &= \left(-a_{ij} + \frac{\eta_1\eta_2}{\eta_1+\eta_2}\right) \sup_{\mathbf{v}\neq\mathbf{1}} \frac{\mathbf{v}^T \left(\begin{smallmatrix} 1 & -1 \\ -1 & 1 \end{smallmatrix}\right)\mathbf{v}}{\mathbf{v}^T A_G \mathbf{v}} \\ &= \left(-a_{ij} + \frac{\eta_1\eta_2}{\eta_1+\eta_2}\right) \sup_{(v_1,v_2)\neq(1,1)} \frac{(v_1-v_2)^2}{-a_{ij}(v_1-v_2)^2 + \delta_1 v_1^2 + \delta_2 v_2^2}\,.\end{aligned}$$

Now, if $\delta_1 = \delta_2 = 0$, we have $\mu(G) = 1 + \frac{1}{-a_{ij}}\frac{\eta_1\eta_2}{\eta_1+\eta_2}$, whereas otherwise the supremum is reached in $(v_1, v_2) = (\delta_2, -\delta_1)$, which yields

$$\mu(G) \;=\; \left(-a_{ij} + \frac{\eta_1\eta_2}{\eta_1+\eta_2}\right) \Big/ \left(-a_{ij} + \frac{\delta_1\delta_2}{\delta_1+\delta_2}\right).$$

Eventually, replacing η_k, δ_k, $k = 1, 2$, by their expression yields the formula for $\mu(\{i,j\})$ in Algorithm 5.1, in which the denominator should be interpreted as equal to $-a_{ij}$ if either $\delta_1 = a_{ii} - s_i = 0$ or $\delta_2 = a_{jj} - s_j = 0$.

Appendix B. Here, we prove that

$$\mu(G_i \cup G_j) \;\geq\; \widetilde{\mu}(\{i,j\})\,, \tag{B.1}$$

where $\mu(\cdot)$ is defined as in Theorem 3.2 with respect to $A_{G_i\cup G_j}$ and $M_{G_i\cup G_j}$, where G_i and G_j are two disjoint sets, and where $\widetilde{\mu}(\{i,j\})$ is defined as in the *further pairwise aggregation* algorithm, with $\widetilde{s}_i = -\sum_{k\in G_i}\sum_{l\notin G_i} a_{kl}$, $\widetilde{s}_j = -\sum_{k\in G_j}\sum_{l\notin G_j} a_{kl}$, and $\widetilde{A} = P^T A P$ and with P satisfying (3.2) with respect to G_i, G_j.

Let

$$\widetilde{A}_{\{i,j\}} = \begin{pmatrix} \widetilde{a}_{ii} - \widetilde{s}_i - \widetilde{a}_{ij} & \widetilde{a}_{ij} \\ \widetilde{a}_{ji} & \widetilde{a}_{jj} - \widetilde{s}_j - \widetilde{a}_{ji} \end{pmatrix},$$
$$\widetilde{M}_{\{i,j\}} = \begin{pmatrix} \widetilde{a}_{ii} + \widetilde{s}_i + \widetilde{a}_{ij} & \widetilde{a}_{ij} \\ \widetilde{a}_{ji} & \widetilde{a}_{jj} + \widetilde{s}_j + \widetilde{a}_{ij} \end{pmatrix},$$

and

$$P_f = \begin{pmatrix} \mathbf{1}_{G_i} & \\ & \mathbf{1}_{G_j} \end{pmatrix}.$$

One has, as shown below,

$$\widetilde{A}_{\{i,j\}} = {P_f}^T A_{G_i\cup G_j} P_f, \tag{B.2}$$
$$\widetilde{M}_{\{i,j\}} = {P_f}^T M_{G_i\cup G_j} P_f. \tag{B.3}$$

Noting that $\mathbf{1}_{G_i\cup G_j} = P_f \mathbf{1}_{\{i,j\}}$, the inequality (B.1) follows from

$$\begin{aligned}&\mu(\,G_i \cup G_j\,) \\ &= \sup_{\mathbf{v}\notin\mathcal{N}(A_{G_i\cup G_j})} \frac{\mathbf{v}^T M_{G_i\cup G_j}(I - \mathbf{1}_{G_i\cup G_j}(\mathbf{1}_{G_i\cup G_j}^T M_{G_i\cup G_j}\mathbf{1}_{G_i\cup G_j})^{-1}\mathbf{1}_{G_i\cup G_j}^T M_{G_i\cup G_j})\mathbf{v}}{\mathbf{v}^T A_{G_i\cup G_j}\mathbf{v}} \\ &\geq \sup_{\mathbf{w}\notin\mathcal{N}(\widetilde{A}_{\{i,j\}})} \frac{\mathbf{w}^T \widetilde{M}_{\{i,j\}}(I - \mathbf{1}_{\{i,j\}}(\mathbf{1}_{\{i,j\}}^T \widetilde{M}_{\{i,j\}}\mathbf{1}_{\{i,j\}})^{-1}\mathbf{1}_{\{i,j\}}^T \widetilde{M}_{\{i,j\}})\mathbf{w}}{\mathbf{w}^T \widetilde{A}_{\{i,j\}}\mathbf{w}} \\ &= \widetilde{\mu}(\{i,j\}),\end{aligned}$$

where we have used $\mathbf{v} = P_f \mathbf{w}$ to obtain the inequality in the second line and where the last equality is shown in Appendix A.

We eventually prove (B.2), the proof of (B.3) following along the same lines. Regarding the first diagonal element of $\widetilde{A}_{\{i,j\}}$, we have, using (3.3) and the definition of $\widetilde{s}_i$,

$$\begin{aligned}(\widetilde{A}_{\{i,j\}})_{11} &= \widetilde{a}_{ii} - \widetilde{s}_i - \widetilde{a}_{ij} \\ &= \sum_{k\in G_i}\sum_{l\in G_i} a_{kl} + \sum_{k\in G_i}\sum_{l\notin G_i} a_{kl} - \sum_{k\in G_i}\sum_{l\in G_j} a_{kl} \\ &= \sum_{k\in G_i}\sum_{l\in G_i} a_{kl} - \sum_{k\in G_i}\sum_{l\notin G_i\cup G_j} |a_{kl}| \\ &= (P_f^T A|_{G_i\cup G_j} P_f)_{11} - (P_f^T \Sigma_{G_i\cup G_j} P_f)_{11}\,.\end{aligned}$$

Similarly, for the off-diagonal element,

$$(\widetilde{A}_{\{i,j\}})_{12} = \widetilde{a}_{ij} = \sum_{k\in G_i}\sum_{l\in G_j} a_{kl} = (P_f^T A|_{G_i\cup G_j} P_f)_{12}\,.$$

Hence, since $A_{G_i\cup G_j} = A|_{G_i\cup G_j} - \Sigma_{G_i\cup G_j}$ and $(P_f^T \Sigma_{G_i\cup G_j} P_f)_{12} = 0$, the equality (B.2) follows.

REFERENCES

[1] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. D. Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen, *LAPACK Users' Guide*, 3rd ed., SIAM, Philadelphia, 1999.

[2] O. Axelsson and P. S. Vassilevski, *Algebraic multilevel preconditioning methods.* I, Numer. Math., 56 (1989), pp. 157–177.

[3] O. Axelsson and P. S. Vassilevski, *Algebraic multilevel preconditioning methods*, II, SIAM J. Numer. Anal., 27 (1990), pp. 1569–1590.

[4] D. Braess, *Towards algebraic multigrid for elliptic problems of second order*, Computing, 55 (1995), pp. 379–393.

[5] A. Brandt, S. F. McCormick, and J. W. Ruge, *Algebraic multigrid (AMG) for sparse matrix equations*, in Sparsity and Its Applications, D. J. Evans, ed., Cambridge University Press, Cambridge, UK, 1985, pp. 257–284.

[6] M. Brezina, A. J. Cleary, R. D. Falgout, V. E. Henson, J. E. Jones, T. A. Manteuffel, S. F. McCormick, and J. W. Ruge, *Algebraic multigrid based on element interpolation (AMGe)*, SIAM J. Sci. Comput., 22 (2000), pp. 1570–1592.

[7] 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.

[8] M. Brezina, P. Vaněk, and P. S. Vassilevski, *An improved convergence analysis of smoothed aggregation algebraic multigrid*, Numer. Linear Algebra Appl., to appear.

[9] V. E. Bulgakov, *Multi-level iterative technique and aggregation concept with semi-analytical preconditioning for solving boundary value problems*, Comm. Numer. Methods Engrg., 9 (1993), pp. 649–657.

[10] T. Chartier, R. D. Falgout, V. E. Henson, J. Jones, T. Manteuffel, S. McCormick, J. Ruge, and P. S. Vassilevski, *Spectral AMGe (ρAMGe)*, SIAM J. Sci. Comput., 25 (2003), pp. 1–26.

[11] E. Cuthill and J. McKee, *Reducing the bandwidth of sparse symmetric matrices*, in Proceedings of the 24th National Conference of the Association for Computing Machinery, Brandon Press, NJ, 1969, pp. 157–172.

[12] M. Emans, *Performance of parallel AMG-preconditioners in CFD-codes for weakly compressible flows*, Parallel Comput., 36 (2010), pp. 326–338.

[13] R. D. Falgout and P. S. Vassilevski, *On generalizing the algebraic multigrid framework*, SIAM J. Numer. Anal., 42 (2004), pp. 1669–1693.

[14] W. Hackbusch, *Multi-grid Methods and Applications*, Springer, Berlin, 1985.

[15] V. E. Henson and P. S. Vassilevski, *Element-free AMGe: General algorithms for computing interpolation weights in AMG*, SIAM J. Sci. Comput., 23 (2001), pp. 629–650.
[16] J. E. Jones and P. S. Vassilevski, AMGe *based on element agglomeration*, SIAM J. Sci. Comput., 23 (2001), pp. 109–133.
[17] H. Kim, J. Xu, and L. Zikatanov, *A multigrid method based on graph matching for convection-diffusion equations*, Numer. Linear Algebra Appl., 10 (2003), pp. 181–195.
[18] J. Mandel, M. Brezina, and P. Vaněk, *Energy optimization of algebraic multigrid bases*, Computing, 62 (1999), pp. 205–228.
[19] A. C. Muresan and Y. Notay, *Analysis of aggregation-based multigrid*, SIAM J. Sci. Comput., 30 (2008), pp. 1082–1103.
[20] A. Napov and Y. Notay, *Algebraic analysis of aggregation-based multigrid*, Numer. Linear Algebra Appl., 18 (2011), pp. 539–564.
[21] Y. Notay, *AGMG*, software and documentation, http://homepages.ulb.ac.be/~ynotay/AGMG (25 July 2011).
[22] Y. Notay, *An aggregation-based algebraic multigrid method*, Electron. Trans. Numer. Anal., 37 (2010), pp. 123–146.
[23] Y. Notay, *Aggregation-based Algebraic Multigrid for Convection-Diffusion Equations*, Technical report GANMN 11–01, Université Libre de Bruxelles, Brussels, Belgium, 2011; available online at http://homepages.ulb.ac.be/~ynotay.
[24] Y. Notay and P. S. Vassilevski, *Recursive Krylov-based multigrid cycles*, Numer. Linear Algebra Appl., 15 (2008), pp. 473–487.
[25] 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.
[26] K. Stüben, *An introduction to algebraic multigrid*, in Multigrid, Academic Press, San Diego, CA, 2001, pp. 413–532.
[27] U. Trottenberg, C. W. Oosterlee, and A. Schüller, *Multigrid*, Academic Press, San Diego, CA, 2001.
[28] P. Vaněk, M. Brezina, and J. Mandel, *Convergence of algebraic multigrid based on smoothed aggregation*, Numer. Math., 88 (2001), pp. 559–579.
[29] P. Vaněk, J. Mandel, and M. Brezina, *Algebraic multigrid based on smoothed aggregation for second and fourth order elliptic problems*, Computing, 56 (1996), pp. 179–196.
[30] P. S. Vassilevski, *Hybrid V-cycle algebraic multilevel preconditioners*, Math. Comp., 58 (1992), pp. 489–512.
[31] P. S. Vassilevski, *Multilevel Block Factorization Preconditioners*, Springer, New York, 2008.
[32] P. S. Vassilevski, *General constrained energy minimization interpolation mappings for AMG*, SIAM J. Sci. Comput., 32 (2010), pp. 1–13.
[33] P. S. Vassilevski and L. T. Zikatanov, *Multiple vector preserving interpolation mappings in algebraic multigrid*, SIAM J. Matrix Anal. Appl., 27 (2006), pp. 1040–1055.
[34] M. Wabro, *AMGe—Coarsening strategies and application to the Oseen equations*, SIAM J. Sci. Comput., 27 (2006), pp. 2077–2097.
[35] J. Xu and L. Zikatanov, *On an energy minimizing basis for algebraic multigrid methods*, Comput. Vis. Sci., 7 (2004), p. 121–127.