ROBUSTNESS AND SCALABILITY OF ALGEBRAIC MULTIGRID

ANDREW J. CLEARY* , ROBERT D. FALGOUT* , VAN EMDEN HENSON* , JIM E. JONES*, THOMAS A. MANTEUFFEL† , STEPHEN F. MCCORMICK†, GERALD N. MIRANDA‡ AND JOHN W. RUGE§

Abstract. Algebraic multigrid (AMG) is currently undergoing a resurgence in popularity, due in part to the dramatic increase in the need to solve physical problems posed on very large, unstructured grids. While AMG has proved its usefulness on various problem types, it is not commonly understood how wide a range of applicability the method has. In this study, we demonstrate that range of applicability, while describing some of the recent advances in AMG technology. Moreover, in light of the imperatives of modern computer environments, we also examine AMG in terms of algorithmic scalability. Finally, we show some of the situations in which standard AMG does not work well, and indicate the current directions taken by AMG researchers to alleviate these difficulties.

Key words. algebraic multigrid, interpolation, unstructured meshes, scalability

1. Introduction.

Algebraic multigrid (AMG) was first introduced in the early 1980's [11, 8, 10, 12], and immediately attracted substantial interest [32, 28, 30, 29]. Research continued at a modest pace into the late 1980's and early 1990's [18, 14, 21, 25, 20, 26, 22]. Recently, however, there has been a major resurgence of interest in the field, for "classical" AMG as defined in [29], as well as for a host of other algebraic-type multilevel methods [3, 16, 34, 6, 2, 4, 5, 15, 33, 17, 35, 36, 37]. Largely, this resurgence in AMG research is due to the need to solve increasingly larger systems, with hundreds of millions or billions of unknowns, on unstructured grids. The size of these problems dictates the use of large-scale parallel processing, which in turn demands algorithms that scale well as problem size increases. Two different types of scalability are important. *Implementation scalability* requires that a single iteration be scalable on a parallel computer. Less commonly discussed is *algorithmic scalability*, which requires that the computational work per iteration be a linear function of the problem size and that the convergence factor per iteration be bounded below 1 with bound independent of problem size. This type of scalability is a property of the algorithm, independent of parallelism, but is a necessary condition before a scalable implementation can be attained.

Multigrid methods are well known to be scalable (both types) for elliptic problems on regular grids. However, many modern problems involve extremely complex geometries, making structured geometric grids extremely difficult, if not impossible, to use. Application code designers are turning in increasing numbers to very large unstructured grids, and AMG is seen by many as one of the most promising methods for solving the large-scale problems that arise in this context.

This study has four components. First, we examine the performance of "classical" AMG on a variety of problems having regular structure, with the intent of determining its robustness. Second, we examine the performance of AMG on the same suite of problems, but now with unstructured grids and/or irregular domains. Third, we study the algorithmic scalability of AMG by examining its performance on several of

* Center for Applied Scientific Computing (CASC), Lawrence Livermore National Laboratory, Livermore, CA. Email:{cleary, rfalgout, vhenson, jjones}@llnl.gov

† Department of Applied Mathematics, University of Colorado, Boulder, CO. Email: {tmanteuf, stevem}@boulder.colorado.edu

‡ USS Florida (SSBN-728), Naval Submarine Base, Silverdale, WA, Email: JerryTrish@aol.com

§ Front Range Scientific, Boulder, CO. Email: jruge@sobolev.Colorado.EDU

the problems using grids of increasing sizes. Finally, we introduce a new method for computing interpolation weights, and we show that in certain troublesome cases it can significantly improve AMG performance.

Our study differs from previous reports on the performance of AMG (e.g., [29, 30]) primarily by our examination of algorithmic scalability, our emphasis on unstructured grids, and the introduction of a new algorithm for computing interpolation weights. In Section 2, a description of some details of the AMG algorithm is given to provide an understanding of the results and later discussion. In Section 3, we present results of AMG applied to a range of symmetric scalar problems, using finite element discretizations on structured and unstructured 2D and 3D meshes. AMG is also tested on nonsymmetric problems, on both structured and unstructured meshes, and the results are presented in Section 4. A version of AMG designed for systems of equations is tested, with the focus on problems in elasticity. Results are discussed in Section 5. In Section 6, we introduce and report on tests of a new method for computing interpolation weights. We concluding with some remarks in Section 7.

2. The Scalar AMG Algorithm. We begin by outlining the basic principles and techniques that comprise AMG. Detailed explanations may be found in [29]. Consider a problem of the form

$$A\mathbf{u} = \mathbf{f}, \tag{1}$$

where A is an $n \times n$ matrix with entries a_{ij}. For convenience, the indices are identified with grid points, so that u_i denotes the value of $\mathbf{u}$ at point i, and the grid is denoted by $\Omega = \{1, 2, \ldots, n\}$. In any multigrid method, the central idea is that error $\mathbf{e}$ not eliminated by relaxation must be removed by coarse-grid correction. Applied to elliptic problems, for example, simple relaxations (Jacobi, Gauss-Seidel) reduce high frequency error components efficiently, but are very slow at removing smooth components. However, the smooth error that remains after relaxation can be approximated accurately on a coarser grid. This is done by solving the residual equation $A\mathbf{e} = \mathbf{r}$ on a coarser grid, then interpolating the error back to the fine grid and using it to correct the fine-grid approximation. The coarse-grid problem itself is solved by a recursive application of this method. One iteration of this process, proceeding through all levels, is known as a multigrid cycle. In geometric multigrid, standard uniform coarsening and linear interpolation are often used, so the main design task is to choose a relaxation scheme that reduces errors the coarsening process cannot approximate. One purpose of AMG is to free the solver from dependence on geometry, so AMG instead fixes relaxation (normally Gauss-Seidel), and its main task is to determine a coarsening process that approximates error that this relaxation cannot reduce.

An underlying assumption in AMG is that smooth error is characterized by small residuals, that is, $A\mathbf{e} \approx \mathbf{0}$, which is the basis for choosing coarse grids and defining interpolation weights. For simplicity of discussion here, we assume that A is a symmetric positive-definite M-matrix, with $a_{ii} > 0, a_{ij} \leq 0$ for $j \neq i$, and $\sum a_{ij} \geq 0$. This assumption is made for convenience; AMG will frequently work well on matrices that are not M-matrices. To define any multigrid method, several components are required. Using superscripts to indicate level number, where 1 denotes the finest level so that $A^1 = A$ and $\Omega^1 = \Omega$, the components that AMG needs are as follows:

1. "Grids" $\Omega^1 \supset \Omega^2 \supset \ldots \supset \Omega^M$.
2. Grid operators $A^1, A^2, \ldots, A^M$.
3. Grid transfer operators:
   Interpolation $I_{k+1}^k, k = 1, 2, \ldots M-1$,

Restriction $I_k^{k+1}, k = 1, 2, \dots M-1$.

4. Relaxation scheme for each level.

Once these components are defined, the recursively defined cycle is as follows:

```
Algorithm: $MV^k(\mathbf{u}^k, \mathbf{f}^k)$. The $(\mu_1, \mu_2)$ V-cycle.
    If $k = M$, set $\mathbf{u}^M = (A^M)^{-1}\mathbf{f}^M$.
    Otherwise:
        Relax $\mu_1$ times on $A^k\mathbf{u}^k = \mathbf{f}^k$ .
        Perform coarse grid correction:
            Set $\mathbf{u}^{k+1} = 0, \mathbf{f}^{k+1} = I_k^{k+1}(\mathbf{f}^k - A^k\mathbf{u}^k)$.
            "Solve" on level $k+1$ with $MV^{k+1}(\mathbf{u}^{k+1}, \mathbf{f}^{k+1})$.
            Correct the solution by $\mathbf{u}^k \leftarrow \mathbf{u}^k + I_{k+1}^k\mathbf{u}^{k+1}$.
        Relax $\nu_2$ times on $A^k\mathbf{u}^k = \mathbf{f}^k$.
```

For this cycle to work efficiently, relaxation and coarse-grid correction must work together to effectively reduce all error components. This gives two principles that guide the choice of the components:

P1: *Error components not efficiently reduced by relaxation must be well approximated by the range of interpolation.*

P2: *The coarse-grid problem must provide a good approximation to fine-grid error in the range of interpolation.*

Each of these affects a different set of components: given a relaxation scheme, **P1** determines the coarse grids and interpolation, while **P2** affects restriction and the coarse grid operators. In order to satisfy **P1**, AMG takes an algebraic approach: relaxation is fixed, and the coarse grid and interpolation are automatically chosen so that the range of the interpolation operator accurately approximates slowly diminishing error components (which may not always appear to be "smooth" in the usual sense). **P2** is satisfied by defining restriction and the coarse-grid operator by the *Galerkin* formulation:

$$I_k^{k+1} = \left(I_{k+1}^k\right)^T \qquad \text{and} \qquad A^{k+1} = I_k^{k+1} A^k I_{k+1}^k. \tag{2}$$

When A is symmetric positive definite, this ensures that the correction from the exact solution of the coarse-grid problem is the best approximation in the range of interpolation [23], where "best" is meant in the A-*norm*: by $||\mathbf{v}||_A \equiv \langle A\mathbf{v}, \mathbf{v}\rangle^{1/2}$.

The choice of components in AMG is done in a separate preprocessing step:

```
AMG Setup Phase:
    1. Set $k = 1$.
    2. Partition $\Omega^k$ into disjoint sets $C^k$ and $F^k$.
        (a) Set $\Omega^{k+1} = C^k$ .
        (b) Define interpolation $I_{k+1}^k$.
    3. Set $I_k^{k+1} = \left(I_{k+1}^k\right)^T$ and $A^{k+1} = I_k^{k+1} A^k I_{k+1}^k$.
    4. If $\Omega^{k+1}$ is small enough, set $M = k+1$ and stop. Otherwise,
       set $k = k + 1$ and go to step 2.
```

Step 2 is the core of the AMG setup process. Since the focus is on coarsening a particular level k, such superscripts are omitted here and c and f are substituted for $k+1$ and k where necessary to avoid confusion. The goal of the setup phase is to choose the set C of coarse-grid points and, for each fine-grid point $i \in F \equiv \Omega - C$, small set $C_i \subset C$ of interpolating points. Interpolation is then of the form:

$$(I_c^f \mathbf{u}^c)_i = \begin{cases} \mathbf{u}_i^c & \text{if } i \in C, \\ \sum_{j \in C_i} w_{ij} \mathbf{u}_j^c & \text{if } i \in F. \end{cases} \tag{3}$$

2.1. Defining Interpolation Weights. To define the interpolation weights w_{ij}, recall that slow convergence is equivalent to small residuals, $A\mathbf{e} \approx \mathbf{0}$. Thus, we focus on errors satisfying

$$a_{ii} e_i \approx -\sum_{j \neq i} a_{ij} e_j. \tag{4}$$

Now, for any a_{ij} that is relatively small, we could substitute e_i for e_j in (4) and this approximate relation would still hold. This motivates the definition of the set of *dependencies* of a point i, denoted by S_i, which consists of the set of points j for which a_{ij} is large in some sense. Hence, i depends on such j because, to satisfy the ith equation, the value of u_i is affected more by the value of u_j than by other variables. The definition used in AMG is

$$S_i \equiv \left\{ j \neq i : -a_{ij} \geq \alpha \max_{k \neq i} (-a_{ik}) \right\}, \tag{5}$$

with α typically set to be 0.25. We also define the set $S_i^T \equiv \{j : i \in S_j\}$, that is, the set of points j that depend on point i, and we say that S_i^T is the set of *influences* of point i. Note: our terminology here differs from the classical use in [29], which refers to i as being *strongly connected* to or *strongly dependent* on j if $j \in S_i$ and which uses no specific terminology for $j \in S_i^T$.

A basic premise of AMG is that relaxation smoothes the error in the direction of influence. Hence, we may select $C_i = S_i \cap C$ as the set of interpolation points for i, and adhere to the following criterion while choosing C and F:

P3: *For each $i \in F$, each $j \in S_i$ is either in C or $S_j \cap C_i \neq \emptyset$.*

That is, if i is a fine point, then the points influencing i must either be coarse points or must themselves depend on the coarse points used to interpolate u_i. This allows approximations necessary to define interpolation. For $i \in F$, (4) can be rewritten as:

$$a_{ii} e_i \approx -\sum_{k \in C_i} a_{ik} e_k - \sum_{j \notin C_i} a_{ij} e_j. \tag{6}$$

AMG interpolation is defined by making the following approximation in (6):

$$\forall j \notin C_i, \qquad e_j \approx \begin{cases} e_i & \text{if } j \in S_i \\ \dfrac{\sum_{k \in C_i} a_{jk} e_k}{\sum_{k \in C_i} a_{jk}} & \text{otherwise.} \end{cases} \tag{7}$$

Substituting this into (6) and solving for e_i gives the desired interpolation weights for point $i \in F$.

2.2. Selecting the Coarse Grid. The coarse grid is chosen to satisfy the criterion above, while attempting to control its size. We employ the two-stage process described in [29], modified slightly to reflect our modified terminology. The grid is first "colored", providing a tentative C/F choice. Essentially, a point with the largest number of influences ("influence count") is colored as a C point. The points depending on this C point are colored as F points. Other points influencing these F points are more likely to be useful as C points, so their influence count is increased. The process is repeated until all points are either C or F points.

Details of the initial C/F choice are as follows:

```
Repeat until $U = \emptyset$:
    Set $C = \emptyset, F = \emptyset, U = \Omega^k$. Set $\lambda_i = |S_i^T|$ (the number of
    points depending on the point $i$).
    Pick an $i \in U$ with maximal $\lambda_i$. Set $C = C \bigcap \{i\}$ and
    $U = U - \{i\}$.
    For all $j \in S_i^T$ (points depending on $\{i\}$) do:
        Set $F = F \bigcup \{j\}$ and $U = U - \{j\}$.
        For all $k \in S_j \bigcap U$ set $\lambda_k = \lambda_k + 1$ (Increment the
        $\lambda$ for points that influence the new $F$-points).
    For all $j \in S_i \bigcap U$ set $\lambda_j = \lambda_j - 1$
```

Next, a second pass is made, in which some F points may be recolored as C points to ensure that **P3** is satisfied. In this pass, each F-point i is examined. The *coarse interpolatory set* $C_i = S_i \bigcup C$ is defined. Then, if i depends on another F-point, j, the points influencing j are scanned, to see if any of them are in C_i. If this is *not* the case then j is tentatively converted into a C-point and added to C_i. The dependencies of i are then examined anew. If all F-points depending on i now depend on a point in C_i then j is permanently made a C-point and the algorithm proceeds to the next F-point and repeats. If, however, the algorithm finds *another* F-point dependent on i that is not dependent on a point in C_i then i itself is made into a C-point and j returned to the pool of F-points. This procedure is followed to minimize the number of F-points that are converted into C-points.

We make a brief comment about the computational and storage costs of the setup phase. Unlike geometric multigrid, these costs cannot be predicted precisely. Instead, computational cost must be estimated based on the average "stencil size" over all grids, the average number of interpolation points per F-point, the ratio of the total number of gridpoints on all grids to the number of points on the fine grid (grid complexity), and the ratio of the number of nonzero entries in all matrices to that of the fine-grid matrix (operator complexity). While a detailed analysis is beyond the scope of this work [29], a good rule of thumb is that the computational effort for the setup phase is typically equivalent to between four and ten V-cycles.

3. Results for Symmetric Problems.

In this section, results for AMG applied to symmetric scalar problems are presented. Initially, constant-coefficient diffusion problems in 2D are tested as a baseline for comparison as we begin to introduce complications, including unstructured meshes, irregular domains, and anisotropic and discontinuous coefficients. Results for 3D problems follow. All problems are run using the same AMG solver with fixed parameters. On many problems, it is possible to improve our results by tuning some of the input parameters (there are many), but the purpose here is to show AMG's basic behavior and robustness over a range of problems.

The primary indicator of the speed of the algorithm is the asymptotic convergence factor per cycle. This is determined by applying 20 cycles to the homogeneous problem, starting with a random initial guess, then measuring the reduction in the norm of the residual from one cycle to the next (we use the homogeneous problem to avoid contamination by machine representation). Generally, this ratio starts out very small for the first few cycles, then increases to some asymptotic value after 5-10 cycles, when the most slowly converging components become dominant. This asymptotic value is also a good indicator of the actual error reduction from one cycle to the next. We use the 2-norm of the residual, although it is easy to show that the asymptotic convergence factor is just the spectral radius of the AMG V-cycle iteration operator, and hence is independent of the choice of norm.

The times given are for the setup and a single (1,1) V-cycle. Setup time is what it takes to choose the coarser grids, define interpolation, and compute the coarse grid matrices. Cycle time is for one cycle, not the full solution time. Three machines are used in this study. The majority of the smaller tests are performed on a Pentium 166MHz PC, although some are performed on a Sun Sparc Ultra 1. For the larger problems that demonstrate scalability, we use a DEC Alpha. For this reason, timings should be compared only within individual problems. Additionally, timings for the smallest problems can have a high relative error, so the larger tests should give a better picture of performance. Grid complexity is defined as $\sum n_k/n_1$, where n_k is the number of grid points on level k. This gives an idea of how quickly the grids are reduced in size. For comparison, in standard multigrid, the number of points is reduced by a factor of 4 in 2D and 8 in 3D, yielding grid complexities of 4/3 and 8/7, respectively. AMG tends to coarsen more slowly. Operator complexity, which is a better indicator of the work per cycle, is defined as $\sum r_k n_k/r_1 n_1$, where r_k is the average number of non-zero entries per row (or "stencil size") on level k. Thus, the operator complexity is the ratio of the total number of nonzero matrix entries on all levels to those on the finest level. Since relaxation work is proportional to the number of matrix entries, this gives a good idea of the total amount of work in relaxation relative to relaxation work on the finest grid, and also of the total storage needed relative to that required for the fine grid matrix. In geometric multigrid, the grid and operator complexities are equal, but in AMG, operator complexity is usually higher since average stencil sizes tend to grow somewhat on coarser levels. Note that the convergence factors and complexities are entirely independent of the specific machine on which a test is performed.

In the tests reported here, the focus is on finite element discretizations of

$$\nabla \cdot (D\nabla u) = f \quad \text{with} \quad D = \begin{bmatrix} d_{11}(x,y) & d_{12}(x,y) \\ d_{21}(x,y) & d_{22}(x,y) \end{bmatrix}.$$

Several different meshes and diffusion coefficients D are used.

3.1. Regular domains, structured and unstructured grids. The first five problems are 2D Poisson equations, with $d_{11} = d_{22} = 1.0$ and $d_{12} = d_{21} = 0.0$. Different domains and meshes are used to demonstrate the behavior of AMG with simple equations.

We begin with the simplest 2D model problem. The success of AMG on the regular-grid Poisson problem is well-documented [30, 28, 29], so our purpose here is more to assess its scalability.

Problem 1 This is a simple 5-point Laplacian operator with homogeneous Dirichlet boundary conditions on the unit square. The experiment is run for uniform meshes

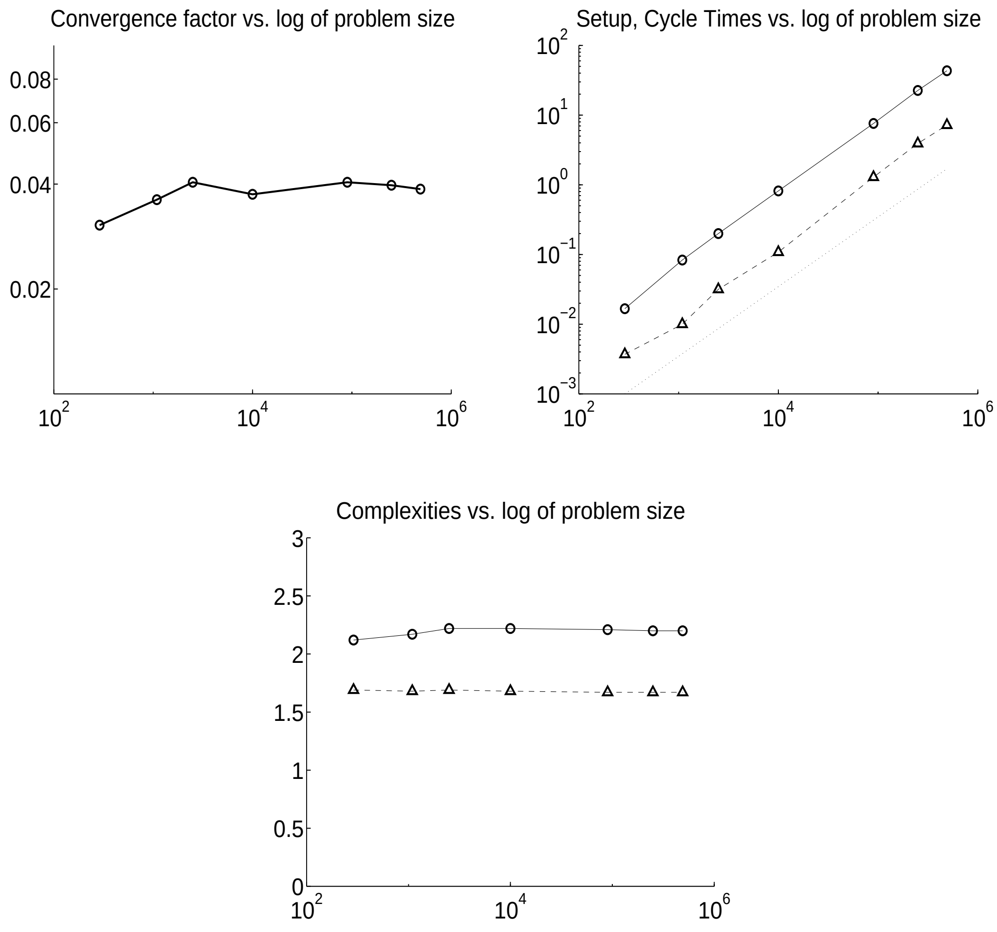


Fig. 1. Top Left: *Convergence factors, as a function of number of mesh points, for Problem 1 the uniform-mesh 5-point Laplacian.* Top Right: *Log-log plots of setup times (circles) and cycle times (triangles) for the uniform-mesh 5-point Laplacian. The dotted line, for reference, shows perfectly linear scaling.* Bottom: *Operator (circles) and grid (triangle) complexities for the uniform-mesh 5-point Laplacian.*

with $n \times n$ interior grid points, yielding mesh sizes $n^2 = N = 289$, 1089, 2500, 10000, 90000, 250000, and 490000.

Results for Problem 1 are displayed in Figure 1. The convergence factor (per cycle) is very stable at approximately 0.04 for all problem sizes. Both the setup and cycle time are very nearly linear in N (compare with the dotted line depicting a perfectly linear hypothetical data set). Here, setup time averages roughly the time of 6 cycles. As noted before, the operator complexities are higher than the corresponding grid complexities, but both appear to be unaffected by problem size. These data indicate that AMG (applied to the uniform-mesh Laplacian) is *algorithmically scalable*: the computational work is $O(N)$ per cycle and the convergence factor is $O(1)$ per cycle. An important component of our study is to determine to what extent this algorithmic scalability is retained as we increase problem complexity.

Problem 2 This is the same equation as Problem 1 ($-\Delta u = f$), but now discretized on an unstructured triangular mesh. These meshes are obtained from uniform triangulations by randomly choosing 15-20% of the nodes and "collapsing" them to neighboring nodes, then smoothing the resulting mesh. The resulting operators might be represented by M-matrices in some cases, but this is not generally the case. We use meshes with $N = 248$, 912, 3506, 13755, and 54518. A typical example is shown at top left in Figure 2.

Results of the experiments are displayed in Figure 2. On the unstructured meshes, convergence factors tend to show some dependence on mesh size, growing to around 0.35 on the finest grid. It should be noted, however, that these grids tend to be less structured than many found in practice, and no care was taken to ensure a "good" mesh; the meshes may have differing characteristics (such as aspect ratios), as there is a large degree of randomness in their construction. Complexities are also higher with the unstructured meshes, and the setup time increases correspondingly. The main point here is that AMG can deal effectively with unstructured meshes without too much degradation in convergence over the uniform case.

3.2. Irregular domains. We continue to use the Laplacian, but now with irregular domains. Since our emphasis here is the effects of this irregularity, we restrict our tests to two representative mesh sizes that give just a snapshot of algorithm scalability.

Problem 3 The computational domain is an unstructured triangular discretization of the torus $0.05 \leq \sqrt{x^2 + y^2} \leq 0.5$. Two different mesh sizes were used, resulting in grids with $N = 14700$ and 58445. Dirichlet boundary conditions around the hole are imposed, with Neumann conditions on the outer boundary.

Problem 4 The domain for this problem is shown in Figure 3. The boundary conditions are Neumann except that a Dirichlet condition is imposed around the small hole on the right. The meshes are uniform, with $h = 1/128$ and $1/256$, resulting in meshes with $N = 11419$ and 44227, respectively. The domain does not easily admit much coarser meshes.

Problem 5 The domain for this problem is shown on the bottom in Figure 3. Dirichlet conditions are imposed on the exterior boundary, and Neumann conditions are on the interior boundaries. A triangular unstructured mesh is used.

Results for Problems 3–5 are given in Table 1. Among these problems, Problem 3 has the simplest domain, but the least structured mesh and the slowest convergence. This indicates that domain configuration generally has little effect on AMG behavior, while the structure (and perhaps the quality) of the mesh is more important.

Table 1. Results for Problems 3–5.
Poisson problem on unstructured meshes, irregular domains

Problem	N	Convergence factor/cycle	Setup time (sec)	Cycle time (sec)	Grid complexity	Operator complexity
3	14700	0.232	2.530	0.370	1.840	3.100
3	58445	0.276	10.710	1.450	1.820	3.110
4	11419	0.134	0.990	0.160	1.690	2.230
4	44227	0.162	3.840	0.660	1.680	2.230
5	7971	0.122	1.370	0.170	1.720	2.500
5	30320	0.108	4.720	0.550	1.710	2.460

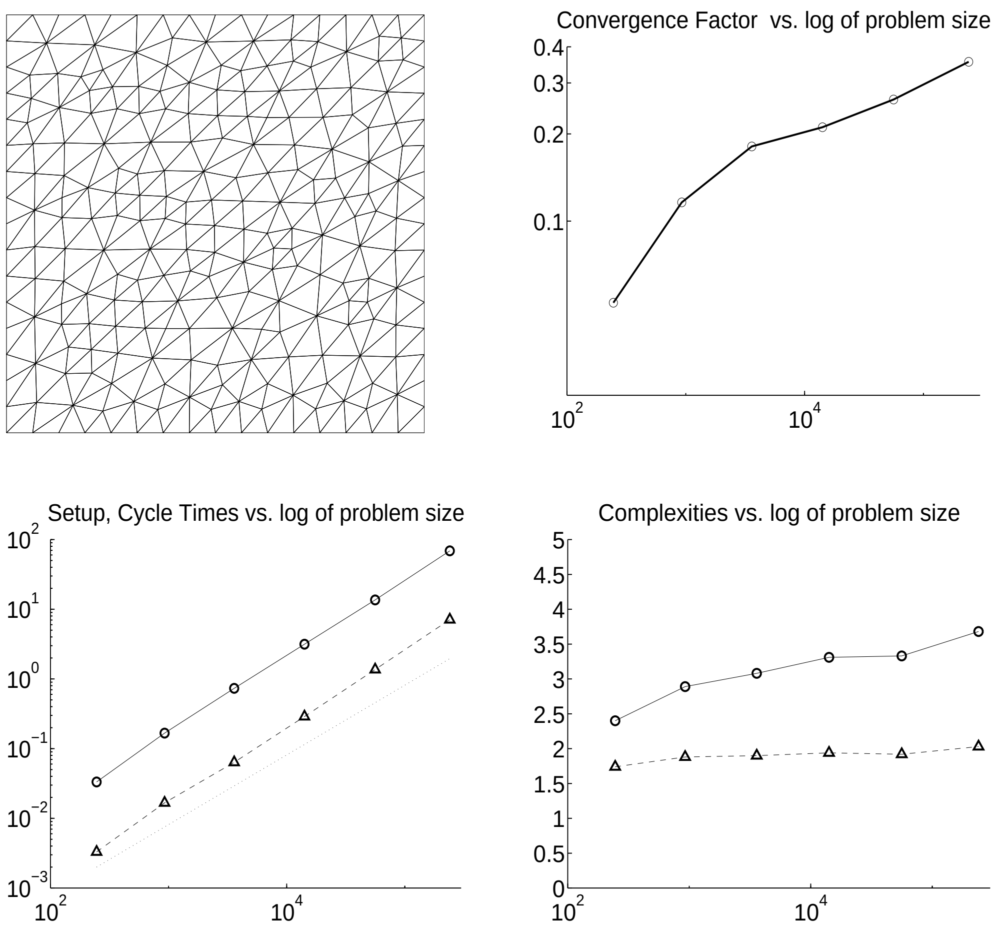


Fig. 2. Top Left: *A typical unstructured grid for Problem 2, obtained by randomly deleting 15% of the nodes in a regular grid and smoothing the result.* Top Right: *Convergence factors, as a function of number of mesh points, for the unstructured-mesh 5-point Laplacian.* Bottom Left: *Log-log plots of setup times (circles) and cycle times (triangles) for the unstructured-grid 5-point Laplacian. The dotted line, for reference, shows perfectly linear scaling.* Bottom Right: *Operator (circles) and grid (triangle) complexities for the unstructured-grid 5-point Laplacian.*

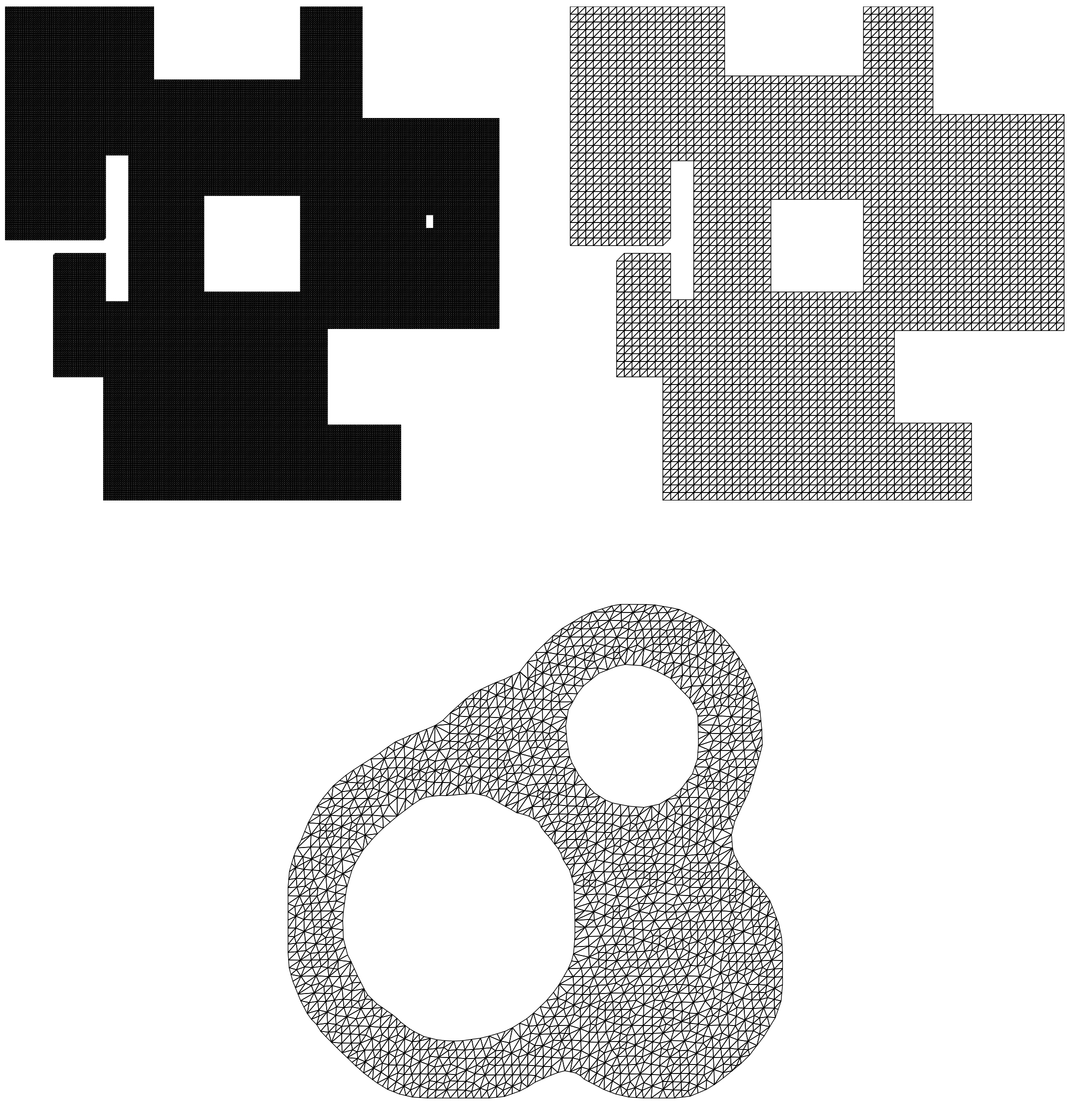


Fig. 3. *Domain* (Top Left) *and typical grid* (Top Right) *for Problem 4. Note that the mesh size necessary to display the triangulation is too coarse to observe the Dirichlet hole. Finer meshes are used for the calculations.* Bottom: *Typical grid for Problem 5.*

3.3. Isotropic diffusion. The next problem set deals with isotropic diffusion: $d_{11} = d_{22} = d(x, y)$ and $d_{12} = d_{21} = 0$. Discontinuous $d(x, y)$ can cause problems for many solution methods, including standard multigrid methods, although it is possible to get good results either by aligning the discontinuities along coarse grid lines, or by using operator-dependent interpolation [1]. In AMG, nothing special is required, since it is based on operator-dependent interpolation. The problems are categorized according to the diffusion coefficient used. The unit square is discretized on four meshes: two structured meshes with $N = 16642$ and 66049, and two unstructured meshes, with $N = 13755$ and 54518. The diffusion coefficients are defined in terms of a parameter, c, allowed to be either 10 or 1000, as follows:

PROBLEM 6 $d(x, y) = 1.0 + c|x - y|$.

PROBLEM 7 $d(x, y) = \begin{cases} 1.0 & x \leq 0.5 \\ c & x > 0.5. \end{cases}$

PROBLEM 8 $d(x, y) = \begin{cases} 1.0 & 0.125 \leq \max(|x - 0.5|, |y - 0.5|) \leq 0.25, \\ c & \text{otherwise.} \end{cases}$

PROBLEM 9 $d(x, y) = \begin{cases} 1.0 & 0.125 \leq \sqrt{(x - 0.5)^2 + (y - 0.5)^2} \leq 0.25, \\ c & \text{otherwise.} \end{cases}$

Results for these problems are presented in Table 2, which contains observed convergence factors and operator complexities for the various combinations of grid size and type, diffusion coefficient function, and discontinuity jump size. The overall results are fairly predictable. Convergence factors are fairly uniform. On the structured meshes, they tend to grow slightly with increasing grid size. They are noticeably larger for unstructured grids, and they appear to grow somewhat with increasing grid size. (As noted before, comparison among unstructured grids of various sizes must take into account that their generation involves some randomness, so they may differ in important ways.) The convergence factor does not seem to depend significantly on the size of the jump in the diffusion coefficient. In many cases, results were better with $c = 1000$ than with $c = 10$. Indeed, AMG has been applied successfully to problems with much larger jumps [28]; see also Problem 17. Note that there are only minor variations in operator complexity for the different problems and different grid sizes. The only significant effect on operator complexity appears to be whether the grid is structured or unstructured, with the latter showing complexity increases of about 30–40%. It should be noted, however, that even in these cases, the entire operator hierarchy can be stored in just over three times the storage required for the fine-grid alone.

Table 2. Results for Problems 6–9.
Poisson problem, variable and discontinuous coefficients

		Uniform mesh size				Unstructured mesh size			
Problem		16642		66049		13755		54518	
#	c	conv.	cmplxty	conv.	cmplxty	conv.	cmplxty	conv.	cmplxty
6	10	0.063	2.2	0.095	2.2	0.290	3.35	0.307	3.39
6	1000	0.097	2.21	0.180	2.2	0.264	3.32	0.369	3.36
7	10	0.111	2.25	0.126	2.23	0.266	3.29	0.287	3.38
7	1000	0.123	2.25	0.144	2.23	0.254	3.28	0.250	3.37
8	10	0.138	2.32	0.159	2.27	0.303	3.28	0.320	3.37
8	1000	0.220	2.30	0.188	2.26	0.286	3.32	0.336	3.38
9	10	0.165	2.33	0.179	2.28	0.280	3.32	0.311	3.38
9	1000	0.171	2.35	0.168	2.30	0.234	3.31	0.298	3.40

Problems 10–13 are designed to examine the case in which the diffusion coefficient is discontinuous and to determine whether the "scale" of the discontinuous regions affects performance. Accordingly, Problems 10–12 use a "checkerboard" pattern:

$$d(x,y) = \begin{cases} 1 \text{ if } i+j \text{ is even and } \dfrac{i}{n} \le x < \dfrac{i+1}{n},\ \dfrac{j}{n} \le y < \dfrac{j+1}{n}, \\[2ex] c \text{ if } i+j \text{ is odd and } \dfrac{i}{n} \le x < \dfrac{i+1}{n},\ \dfrac{j}{n} \le y < \dfrac{j+1}{n}, \end{cases}$$

where $i,j = 0,1,\ldots,n$. Specifically,

Problem 10 $n = 2$.

Problem 11 $n = 10$.

Problem 12 $n = 50$.

For the last problem of this group, we have

Problem 13 $d(x,y) = \text{random}(x,y)$.

Results for Problems 10–13 are displayed in Table 3. The overall trend is similar to the results for Problems 6–9, showing convergence factors that grow slightly with problem size and that are noticeably larger for unstructured grids.

Table 3. Results for Problems 10–13.
Poisson problem, variable and discontinuous coefficients

		Uniform mesh size				Unstructured mesh size			
Problem		16642		66049		13755		54518	
#	c	conv.	cmplxty	conv.	cmplxty	conv.	cmplxty	conv.	cmplxty
10	10	0.120	2.28	0.135	2.25	0.266	3.29	0.283	3.37
10	1000	0.110	2.28	0.119	2.24	0.252	3.28	0.290	3.37
11	10	0.225	2.74	0.275	2.56	0.295	3.25	0.328	3.34
11	1000	0.230	2.76	0.274	2.57	0.263	3.21	0.321	3.32
12	10	0.255	2.83	0.289	2.98	0.302	2.81	0.336	3.11
12	1000	0.238	2.67	0.283	2.98	0.263	2.66	0.403	3.01
13	10	0.199	2.86	0.290	2.94	0.272	3.23	0.324	3.31
13	1000	0.255	2.97	0.287	3.04	0.288	3.04	0.319	3.12

The best results for the isotropic diffusion problems are obtained for Problem 6, where the coefficient is continuous. The worst convergence factor is obtained for the 50×50 "checkerboard" pattern of Problem 12. Note that AMG performs well on Problem 8, where "smooth" functions are approximately constant in the center of the region, zero in the high-diffusion zone near the boundary, and smoothly varying in between. Good interpolation in the low-diffusion band is essential to good convergence. Overall, AMG appears to work quite well with discontinuous diffusion coefficients, even when they vary randomly by a large factor from point to point, as in Problem 13.

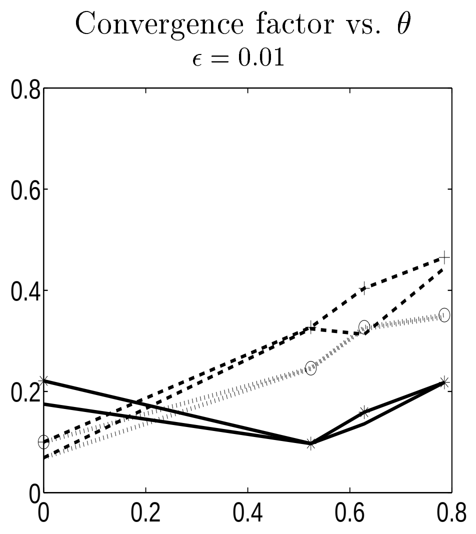
3.4. Anisotropic diffusion. The next series of problems deals with anisotropic diffusion, which can arise in several ways. Anisotropy can be introduced by the mesh being refined differently in each directions, perhaps to resolve a boundary layer or some other local phenomenon. Another case is a tensor product grid used in order to refine some area $[x_0, x_1] \times [y_0, y_1]$, with the mesh size small for $x \in [x_0, x_1]$ and for $y \in [y_0, y_1]$, but large elsewhere. This maintains a logically rectangular mesh, but causes anisotropic discretizations in different parts of the domain. This is relatively easy to deal with in geometric multigrid, where line relaxation and/or semi-coarsening can be used [9]. Non-aligned anisotropy, which is more difficult to handle with standard multigrid, arises from the operator itself, such as the case of the full potential operator in transonic flows. The performance of AMG on grid-induced (aligned) anisotropy has been reported previously [29], so we instead focus here on non-aligned anisotropy. Both types of anisotropy can be written in terms of the diffusion equation using the coefficient matrix:

$$D(x,y) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - (1-\epsilon) \begin{bmatrix} \cos^2\theta & -\cos\theta\sin\theta \\ -\cos\theta\sin\theta & \sin^2\theta \end{bmatrix}.$$

When θ is constant, this gives the operator $\epsilon\partial_{\xi\xi} + \partial_{\eta\eta}$, where ξ is in the direction θ. On a rectangular grid with mesh sizes h_x and h_y, the usual Poisson equation corresponds to the diffusion equation with $\theta = 0$ and $\epsilon = h_y^2/h_x^2$.

Problem 14 This problem features a non-aligned anisotropic operator on the unit square, with Dirichlet boundary conditions at $y = 0$ and $y = 1$ and Neumann conditions on the other two sides. The cases $\epsilon = 0.1$ and $\epsilon = 0.001$ were both examined with $\theta = 0, \pi/6, \pi/5$, and $\pi/4$. Each such combination is discretized on a uniform square mesh (with bilinear elements) and both uniform and unstructured triangular meshes (with linear elements). The uniform meshes have $N = 16642$ and 66049, while the unstructured meshes have $N = 13755$ and 54518.

Convergence factors for Problem 14, as shown in Figure 4, generally degrade with increasing θ. This is to be expected, as it indicates lessened alignment of anisotropy with the grid directions. The strong anisotropy case yields convergence factors as high as 0.745. As noted above, the non-aligned case is very difficult, even for standard multigrid, and is the subject of ongoing study. One encouraging result is that the unstructured grid formulations are relatively insensitive to grid anisotropy, with convergence factors that hover between 0.3 and 0.5 in all cases except $\theta = 0$. Overall, these results indicate that AMG is rather robust for anisotropic problems, although convergence factors are somewhat higher than those typically obtained with AMG on isotropic problems.



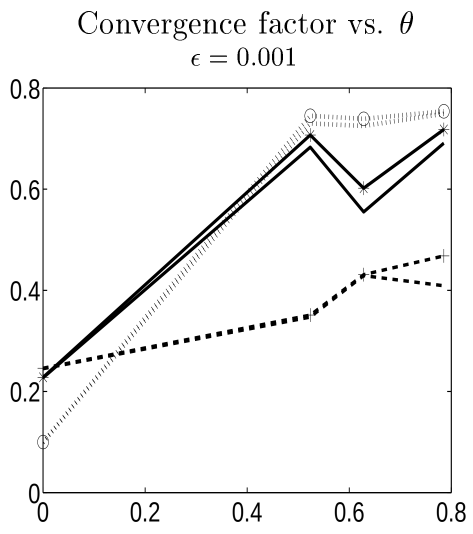


Fig. 4. *Convergence factors, plotted as a function of anisotropy direction θ.* Left: *Moderate anisotropy, $\epsilon = 0.01$* Right: *Strong anisotropy, $\epsilon = 0.001$. In each plot, the solid lines are the uniform meshes with square discretizations, the dotted lines are uniform meshes with triangular discretizations, and the dashed lines are the unstructured triangulations. In each case, the larger grid size is indicated by a symbol ("o", "+", or "*").*

Problem 15 We use the operator of Problem 4, $\nabla \cdot (D\nabla u)$, but with

$$D(x,y) = \frac{1}{r^2}\begin{bmatrix} \epsilon x^2 + y^2 & -xy \\ -xy & x^2 + \epsilon y^2 \end{bmatrix},$$

$r^2 = x^2 + y^2$, and $\epsilon = 100$. This yields a discretization such that, on any circle centered at the origin, there are dependencies in the tangential direction, but none in the radial direction.

This problem is very difficult to solve by conventional methods. Using the same meshes as in Problem 14, AMG produced the convergence factors given in Table 4.

Table 4. Results for Problem 15: Circular diffusion coefficient

uniform mesh (square)		uniform mesh (triangular)		unstructured mesh	
$N = 16642$	$N = 66049$	$N = 16641$	$N = 66049$	$N = 13755$	$N = 54518$
0.619	0.534	0.764	0.674	0.845	0.840

The convergence factors illustrate the difficulty with this problem, which cannot be handled easily by geometric methods, even on regular meshes. A polar-coordinate mesh would allow block relaxation over strongly coupled points, but would suffer from the difficulties of polar-coordinate grids (e.g., singularity at the origin) and would be useless for more general anisotropies. While convergence of AMG here is much slower than what we normally associate with multigrid methods, this example shows that AMG can be useful even for extremely difficult problems.

3.5. 3D problems. Turning our attention to three dimensions, we do not expect special difficulties here, since AMG is based on the algebraic relationships between the variables.

Problem 16 This is a 3D Poisson problem on the unit cube. Discretization is by trilinear finite elements on a rectangular mesh. Dirichlet boundary conditions are imposed at $y = 0$ and $y = 1$, and Neumann conditions are imposed at the other boundaries. Mesh line spacing and the number of mesh intervals were both varied to produce several grids with different spacings and extents in the three coordinate directions. The various combinations of mesh sizes used, convergence factors, and operator complexities are shown in Table 5.

Table 5. Results for Problem 16.
3D Poisson problem, regular rectangular mesh

N_x	h_x	N_y	h_y	N_z	h_z	N	Convergence factor/cycle	Operator Complexity
10	1/10	10	1/10	10	1/10	1089	0.050	4.10
20	1/20	20	1/20	20	1/20	8379	0.064	5.21
25	1/25	25	1/25	25	1/25	16224	0.068	5.26
20	1/20	20	1/20	20	1/200	8379	0.315	1.75
20	1/20	20	1/200	20	1/200	8379	0.151	1.28
20	1/200	20	1/20	20	1/200	8379	0.171	1.31
20	1/200	20	1/20	20	1/2000	8379	0.324	1.75

Problem 17 This is a 3D unstructured mesh problem, generated by a code used at Lawrence Livermore National Laboratory, for the diffusion problem $\nabla\cdot(a(\vec{x})\nabla u(\vec{x})) = g(\vec{x})$. The domain is a segment of a sphere, from $r = 0.02$ to $r = 0.1$, $\theta = 0$ to $\pi/2$, and $\phi = \pi/4$ to $\pi/2$. The coefficient $a(\vec{x})$ is a large constant for $r \leq 0.05$ and a small constant for $r > 0.05$, with a step discontinuity of 1.0×10^{26}. The boundaries $r = 0.02$ and $r = 0.01$ are Dirichlet, while the boundaries $\phi = \pi/4$ and $\phi = \pi/2$ are surfaces of symmetry. Discretization is by finite elements using hexahedral elements.

Figure 5 shows the locations of the nodes at the element corners for one of the problems. Three problem sizes are given, with $N = 500$, 4000, and 8000. Convergence factors and operator complexities for these problems are given in Table 6.

Table 6. Results for Problem 17.
3D unstructured diffusion problem

N	Convergence factor/cycle	Operator complexity
500	0.070	1.99
4000	0.165	2.49
8000	0.166	2.84

AMG apparently works quite well for 3D problems, including those with discontinuous coefficients. The convergence factors are good in all cases. Complexity varies significantly, with the highest values for the uniform grids, but decreasing markedly with increasing grid anisotropy. This may be taken as further evidence that AMG automatically takes advantage of directions of influence.

Overall, AMG performed well on this suite of symmetric scalar test problems. Many of these problems are designed to be very difficult, often unrealistically so, especially those with the circular anisotropic diffusion pattern and the random diffusion coefficients. Recall that the same AMG algorithm, with no parameter tuning, was used in all cases. There are a number of tools for increasing the efficiency of AMG, especially on symmetric problems, that have proved useful in many cases. One

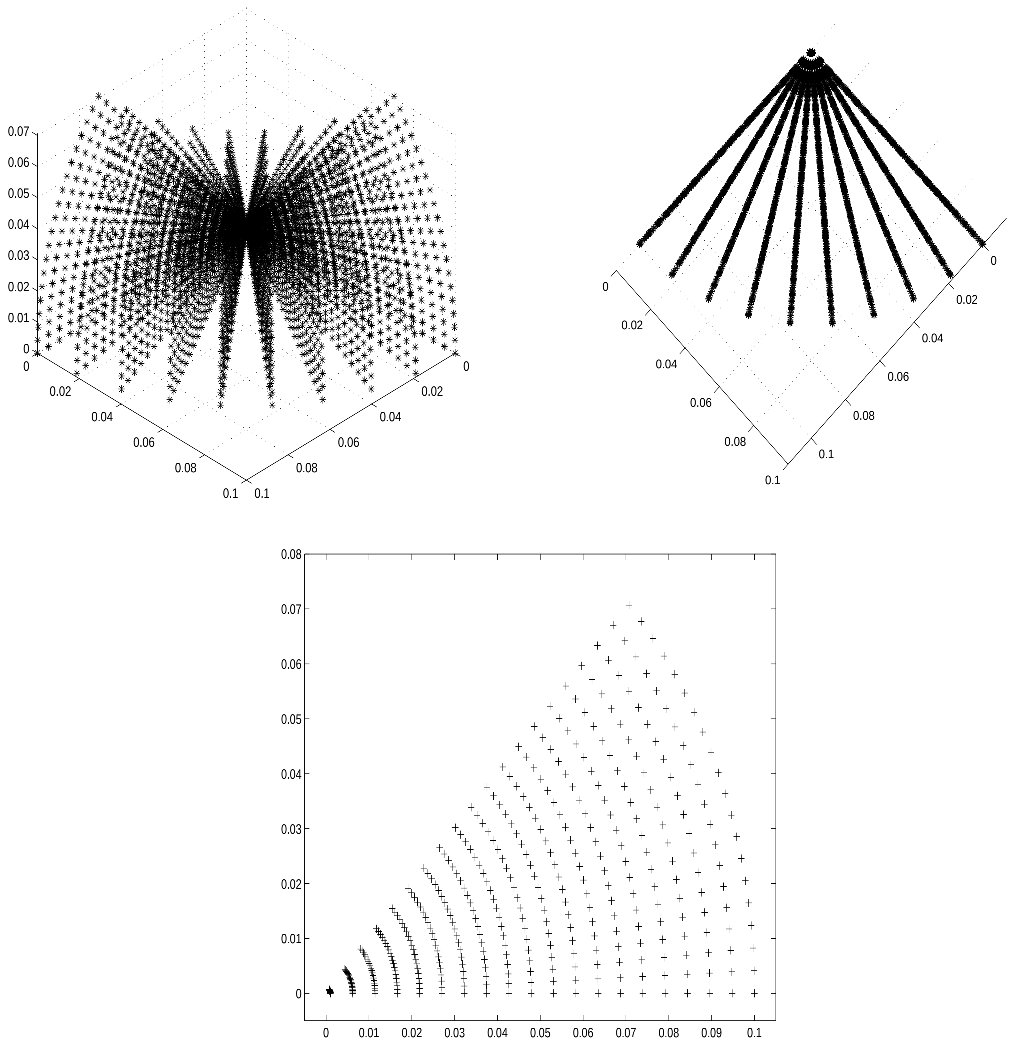


Fig. 5. *Nodes at element corners, 3D diffusion problem.* Top Left: *View of all the node locations.* Top Right: *View from directly overhead, showing radial lines of nodes in the azimuthal direction.* Bottom: *Distribution of nodes within a plane of constant azimuth.*

is the so-called $V^\star$ cycle [30], in which the coarse-grid corrections are multiplied by an optimal parameter, determined by minimizing the A-norm of the corrected error. Another, which has been successful in applying AMG to Maxwell's equations [27], is to use an outer conjugate gradient iteration, with AMG cycling as a preconditioner. Often, when AMG fails to perform well, the problem lies in a small number of components that are not reduced efficiently by relaxation or coarse-grid correction, and conjugate gradients can be very efficient in such cases. Other methods for improving efficiency include the $F-$cycle [7, 31] and the full multigrid (FMG) method, whose applicability to AMG is the subject for future research.

4. AMG applied to Nonsymmetric Scalar Problems.

Although much of the motivation and theory for AMG is based on symmetry of the matrix, this is not at all a requirement for good convergence behavior. Mildly nonsymmetric problems behave essentially like their symmetric counterparts. Such cases arise when a nonsymmetric discretization of a symmetric problem is used or when the original problem is predominantly elliptic. An important requirement for current versions of AMG is that point Gauss-Seidel relaxation converge, however slowly. Thus, central differencing of first-order terms, when they dominate, cannot be used because of severe loss of diagonal dominance. Even in these cases, successful versions of AMG can be developed using Kaczmarz relaxation [9]. Nevertheless, we restrict ourselves here to upstream differencing so that we can retain our use of Gauss-Seidel relaxation.

Problem 18 This is a convection-diffusion problem of the form

$$\epsilon \Delta u + \cos\theta u_x + \sin\theta u_y = f,$$

with Dirichlet boundary conditions. Triangular meshes are used, both structured ($N = 16642$ and 66049) and unstructured ($N = 13755$ and 54518). The diffusion term is discretized by finite elements. The convection term is discretized using upstream differencing, that is, the integral of the convection term is computed over the triangle and added to the equation corresponding to the node with the largest coefficient (the node "most upstream"). Note that this can result in a matrix that has off-diagonal entries of both signs. Two choices for ϵ are employed: $\epsilon = 0.1$ and $\epsilon = 0.0001$. Tests are conducted with $\theta = k\pi/8$ for $k = 0, 1, \ldots, 15$.

Results are presented in Figure 6. The curves in the top left graph are for $\epsilon = 0.1$; the structured grid results are displayed with solid lines, and the unstructured-grid results are displayed with dashed lines. For each pair of curves, the curve with the marker ("o" or "$*$") indicates the mesh with larger N. The convection-dominated case $\epsilon = 0.0001$ is shown at the top right (structured grids) and on the bottom (unstructured grids). In each case, the smaller N is shown with solid lines and the larger N with dashed lines. Note that convergence is generally good and fairly uniform, particularly for the unstructured cases. Results on the smaller uniform mesh are especially good when the flow is aligned with the directions $\theta = 0, \pi/4$, or $\pi/2$. This is due to the triangulation: to obtain the uniform mesh, the domain is partitioned into squares, and then each square is split into two triangles, with the diagonal going from the lower left to the upper right; the "good" directions are aligned with the edges of the triangles. This also has an effect on the quality of the discretization, and on convergence, when the flow is in the direction $3\pi/4$ and $7\pi/4$. Here, the discretization used for the convection term causes a rather severe loss of positivity in the off-diagonals. This is more the fault of the discretization than AMG. For $\theta = 11\pi/8$ with the smaller uniform mesh,

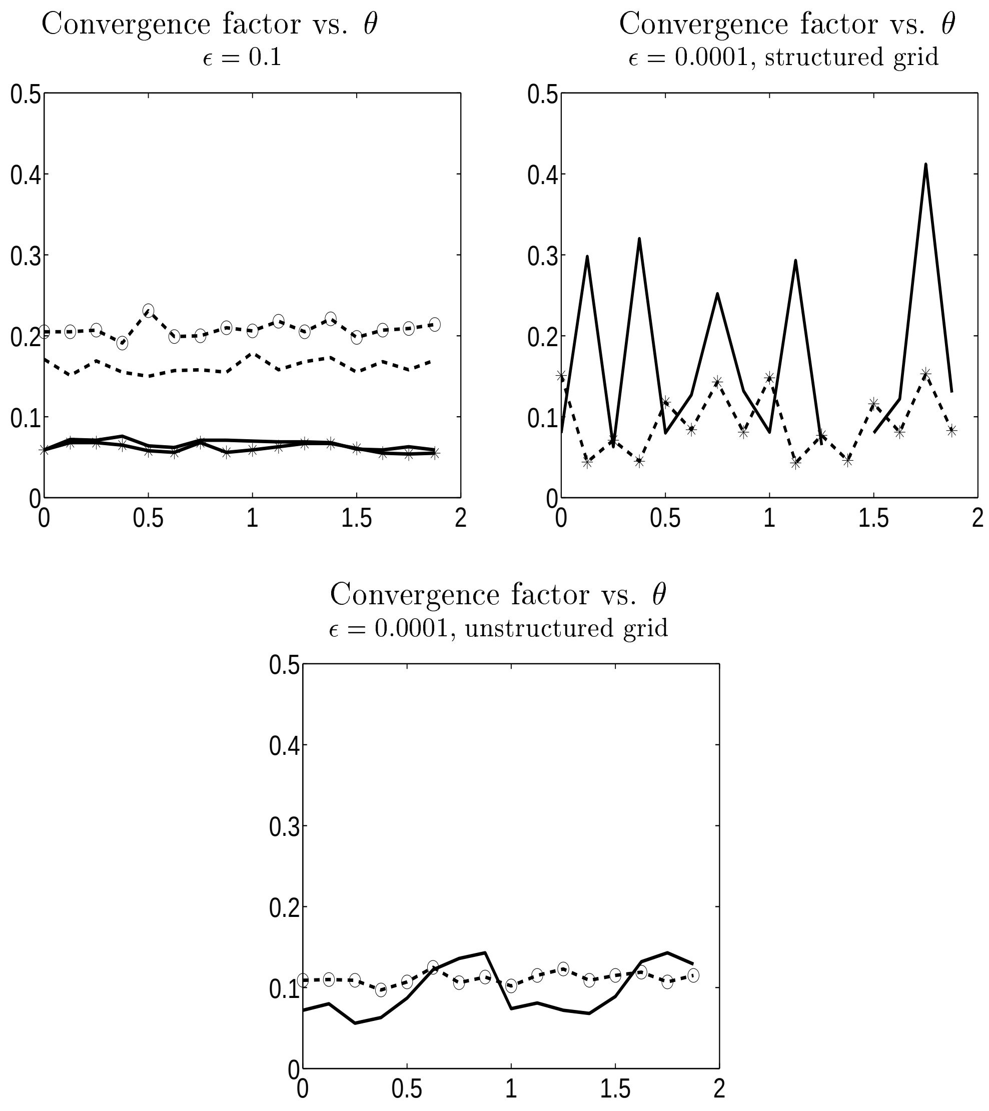


FIG. 6. *Convergence factors, plotted as a function of direction, θ, for the nonsymmetric problem.* Top Left: *Weaker convection case, $\epsilon = 0.1$ The solid lines are the uniform meshes, and the dashed lines are the unstructured triangulations. In each case the larger grid size is indicated by the placement of a symbol ("o" or "$*$").* Top Right: *Convection-dominated case, $\epsilon = 0.0001$, structured grid. The solid line is the smaller N, the larger N is indicated by the dashed line.* Bottom: *Convection-dominated case, $\epsilon = 0.0001$, unstructured grid. The solid line is the smaller N, the larger N is indicated by the dashed line.*

AMG was unable to handle the discretization produced and failed in the setup phase. Often in multigrid applications, problems in convergence indicate problems with the discretization, as is the case here. Note that for unstructured grids where flow cannot align with (or against) the grid, convergence is generally more uniform. An interesting point is that, in many cases, a smaller diffusion coefficient reduces the convergence factor. This is particularly striking with the unstructured meshes. Finally, note that there is generally not much difference between results for θ and for $\theta + \pi$, so there is no benefit from accidental alignment of relaxation with the flow direction, and, conversely, there is no slowing of convergence due to upstream relaxation. This is due to the C/F ordering of relaxation. These tests show that AMG can be applied to nonsymmetric problems. While the convergence factors in this test are generally less than 0.2–0.25, which is certainly acceptable, some concern may be raised about the scalability issue, since the convergence factors for the larger unstructured mesh are noticeably greater than those for the smaller unstructured grids. It remains to be determined if the convergence factors continue to grow with increasing problem size, or if they reach an asymptotic limit.

5. AMG for Systems of Equations.

The extension of AMG to "systems" problems, where more than one function is being approximated, is not straightforward. Many different approaches can be formulated. Consider a problem with two unknown functions of the form

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ \mathbf{g} \end{bmatrix}. \tag{8}$$

The scalar algorithm could work in special circumstances (for example, if B and C are relatively small in some sense), but, generally, the scalar ideas of smoothness break down. One approach would be to iterate in a block fashion on the two equations, with two separate applications of AMG, one using A as the matrix (solving for $\mathbf{u}$, holding $\mathbf{v}$ fixed) and one using B (solving for $\mathbf{v}$, holding $\mathbf{u}$ fixed), and repeating until convergence. This is often very slow. An alternative is to use this block iteration as a preconditioner in an outer conjugate gradient solution process.

Another fairly simple alternative is to couple the block iteration process on all levels, that is, to coarsen separately for each function, obtaining two interpolation operators $I_{\mathbf{u}}$ and $I_{\mathbf{v}}$, then define a full interpolation operator of the form

$$I = \begin{bmatrix} I_{\mathbf{u}} & 0 \\ 0 & I_{\mathbf{v}} \end{bmatrix}. \tag{9}$$

The Galerkin approach can then be used to construct the coarse grid operator,

$$\begin{bmatrix} I_{\mathbf{u}}^T A I_{\mathbf{u}} & I_{\mathbf{u}}^T B I_{\mathbf{v}} \\ I_{\mathbf{v}}^T C I_{\mathbf{u}} & I_{\mathbf{v}}^T D I_{\mathbf{v}} \end{bmatrix}. \tag{10}$$

Once the setup process is completed, multigrid cycles are performed as usual. We will call this the *function approach* since it treats each function separately in determining coarsening and interpolation. When $\mathbf{u}$ and $\mathbf{v}$ are defined on the same grid, it is also possible to couple the coarse-grid choices for both, allowing for nodal relaxation, where both unknowns are updated simultaneously at a point. Following are results for the function approach applied to several problems in 2D and 3D elasticity.

PROBLEM 19 This problem is plane-stress elasticity:

$$\begin{aligned} u_{xx} + \frac{1-\nu}{2}\, u_{yy} + \frac{1+\nu}{2}\, v_{xy} &= f_1, \\ \frac{1+\nu}{2}\, u_{xy} + \frac{1-\nu}{2}\, v_{xx} + v_{yy} &= f_2, \end{aligned}$$

where u and v are displacements in the x and y directions, respectively. This can be a difficult problem for standard multigrid methods, especially when the domain is long and thin. The problem is discretized on a rectangular grid using bilinear finite elements, and several different problem sizes and domain configurations are used.

PROBLEM 20 This problem is 3D elasticity:

$$\begin{aligned} u_{xx} + \frac{1-\nu}{2}\,(u_{yy} + u_{zz}) + \frac{1+\nu}{2}\,(v_{xy} + w_{xz}) &= f_1, \\ v_{yy} + \frac{1-\nu}{2}\,(v_{xx} + v_{zz}) + \frac{1+\nu}{2}\,(u_{xy} + w_{yz}) &= f_2, \\ w_{zz} + \frac{1-\nu}{2}\,(w_{xx} + w_{yy}) + \frac{1+\nu}{2}\,(u_{xz} + v_{yz}) &= f_3, \end{aligned}$$

where u, v, and w are displacements in the three coordinate directions. The problem is discretized on a 3D rectangular grid using trilinear finite elements. Several different problem sizes and domain configurations are used.

In all tests, we take $\nu = 0.3$. The function approach with (1,1) V-cycles is used in all tests. Results for Problem 19 are contained in Table 7, and for Problem 20 in Table 8. Note that complexities are stable in 2D, with some dependence on problem size in 3D. Convergence depends fairly heavily on the number of fixed boundaries in both 2D and 3D, with convergence degrading as the number of free boundaries increases.

Table 7. Results for 2D elasticity, Problem 19

h	n	# of fixed boundaries	Convergence factor	Operator complexity
1/32	1056	1	0.398	2.00
1/32	1023	2	0.253	1.91
1/32	961	4	0.202	1.85
1/64	4160	1	0.657	1.95
1/64	4095	2	0.292	1.91
1/64	3969	4	0.204	1.92

Table 8. Results for 3D elasticity, Problem 20

h	n	# of fixed boundaries	Convergence factor	Operator complexity
1/8	648	1	0.631	2.29
1/8	567	2	0.309	2.43
1/8	441	4	0.124	2.11
1/8	343	6	0.052	2.20
1/12	1859	2	0.326	2.76
1/12	1573	4	0.137	2.56
1/12	1331	6	0.084	2.94

6. Iterative Interpolation Weights. Occasionally, we encounter situations where convergence of AMG is poor, yet no specific reason is apparent. Our experience leads us to believe that the fundamental problem, in many cases, stems from the limitation of the matrix entry a_{jk} to reflect the true "smoothness" between e_j and e_k. Often the true influences between variables are not clear. One case where this limitation is quite evident is where finite elements with extreme aspect ratios are used, especially in cases of extreme grid anisotropy or of thin-body elasticity. As a simple example, consider the 2D nine-point negative Laplacian based on quadrilateral elements that are stretched in the x-direction. The stencil changes as follows:

$$\frac{1}{8}\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \Longrightarrow \frac{1}{8}\begin{bmatrix} -1 & -4 & -1 \\ 2 & 8 & 2 \\ -1 & -4 & -1 \end{bmatrix}, \quad \text{as} \quad \frac{\Delta x}{\Delta y} \longrightarrow \infty. \tag{11}$$

The limiting case is no longer an M-matrix. Indeed, even moderate aspect ratios (e.g., $\Delta x = 10\Delta y$) have off-diagonal entries of both signs. It is not immediately clear how the neighbors to the east and west of the central point should be treated. Do they influence the central point? Should they be in S_i? Even if they are not treated as influences, similar questions arise about how the corner points relate to the central point. Geometric intuition indicates they are decoupled from the center, and should not be treated as influences. Yet, for the most common choice of α in (5), AMG treats them as influences. Another difficulty arises when two F points i and j influence each other. Then e_j must be approximated in the second sum on the right side of (6) to determine the weights for i, while e_i must be approximated, in (6) but with the roles of i and j reversed, to determine the weights for j. However, since both e_i and e_j are to be interpolated (being F-point values), it makes sense to *use* the interpolations to obtain these approximations, that is, the approximations for e_i and e_j in (6) should be

$$e_j \approx \frac{\sum\limits_{k\in C_i} w_{jk}e_k}{\sum\limits_{k\in C_i} w_{jk}} \quad \text{and} \quad e_i \approx \frac{\sum\limits_{k\in C_j} w_{ik}e_k}{\sum\limits_{k\in C_j} w_{ik}}, \tag{12}$$

respectively. Note that the approximations for any points in C are unchanged in these equations.

This gives an implicit system for the interpolation weights, which is solved by an iterative scheme with the initial approximation $w_{ij} = a_{ij}$. The new interpolation weights are then calculated in a Gauss-Seidel-like manner, using the most recently computed weights to make the approximations in (12). Two sweeps are generally sufficient. An important addition to the process is that, after the first sweep, the interpolation sets are modified by removing from C_i any point for which a negative interpolation weight is computed. The second sweep is then used to compute the final interpolation weights.

We present results of two experiments illustrating the effectiveness, on certain types of problems, of using this *iterative weight definition* scheme. Other examples may be found in [24].

Problem 21 This operator is the "stretched quadrilateral" Laplacian mentioned above, discretized on an $N = n \times n$ grid, for $N = 400,\ 900,\ 1225$, and 10000. The stretching factor ε represents the ratio of the x-dimension to the y-dimension of the quadrilateral. Values used are $\varepsilon = 10,\ 25,\ 50$, and 100. In each case, the convergence factor is computed for the choices $\alpha = 0.25$ and $\alpha = 0.5$. In the latter case, the corner points are not treated as influences, and AMG selects a semi-coarsened coarse grid, which is the method geometric multigrid would take.

Table 9. Results for problem 21, $\Delta x = \varepsilon \Delta y$

		Convergence Factor			
		standard weights		iterative weights	
ε	N	$\alpha = 0.25$	$\alpha = 0.5$	$\alpha = 0.25$	$\alpha = 0.5$
10	900	0.45	0.23	0.13	0.13
10	10000	0.47	0.24	0.14	0.14
25	400	0.14	0.14	0.14	0.14
50	1225	0.25	0.14	0.15	0.14
100	900	0.83	0.53	0.82	0.23
100	10000	0.93	0.55	0.93	0.28

Results are displayed in Table 9. On the smallest problem, iterative weights have no effect. However, the convergence rate on that problem is quite good, even for standard weights. For moderate stretching ($\varepsilon < 100$), the effect of iterative weighting is to correct for misidentified influence (i.e., improvement for $\alpha = 0.25$) *and* to improve the results even for correctly identified influence ($\alpha = 0.5$). For extreme stretching, only the latter effect applies.

Problem 22 Here we use the unstructured 3D diffusion operator from Problem 17, whose grid is displayed in Figure 5. Problem sizes are $N = 500$, 4000, and 8000. The problem includes a very large jump discontinuity, $O(10^{26})$, in the diffusion coefficients.

Table 10. Results for problem 22

	Convergence Factor			
	standard weights		iterative weights	
N	$\alpha = 0.25$	$\alpha = 0.5$	$\alpha = 0.25$	$\alpha = 0.5$
500	0.24	0.16	0.24	0.11
4000	0.71	0.42	0.17	0.17
8000	0.69	0.46	0.17	0.26

Results are displayed in Table 10. Again we see that, on the smallest problem, iterative weights have no effect, but that convergence there is fairly good anyway, even for standard weights. On the two larger problems, iterative weights produce significant improvement for both choices of α. Apparently, iterative weighting is countering the effects of both poor element aspect ratios near the boundaries and jump discontinuities in identifying influences among variables.

On these problems, and similar problems characterized by coefficient discontinuities and/or extreme aspect ratios in the elements, iterative weight definition proves to be quite effective. However, iterative weighting is not always effective at improving slow AMG convergence, and in a few cases it can actually cause very minor degradation in performance [24]. We study a new approach in [13], called element interpolation (AMGe), which has the promise of overcoming the difficulties associated with poor aspect ratios, misidentified influences, and thin-body elasticity, provided the individual element stiffness matrices are available.

7. Conclusions.

The need for fast solvers for many types of problems, especially those discretized on unstructured meshes, is a clear indication that there is a market for software with the capabilities that AMG offers. Our study here demonstrates the robustness of AMG as a solver over a wide range of problems. Our tests indicate that it can be further extended, and that robust, efficient codes can be developed for problems that are very difficult to solve by other techniques. AMG is also shown to have good scalability on model problems. This scalability does tend to degrade somewhat with increasing problem complexity, but the convergence factors remain tractable even in the worst of these situations.

REFERENCES

[1] R. E. ALCOUFFE, A. BRANDT, J. E. DENDY, AND J. W. PAINTER, *The multi–grid methods for the diffusion equation with strongly discontinuous coefficients*, SIAM J. Sci. Stat. Comput., 2 (1981), pp. 430–454.

[2] O. AXELSSON, *Stabilization of algebraic multilevel iteration; additive methods*, in AMLI'96: Proceedings of the Conference on Algebraic Multilevel Iteration Methods with Applications, vol. 1, Nijmegan, The Netherlands, 1996, University of Nijmegan, pp. 49–62.

[3] O. AXELSSON AND M. NEYTCHEVA, *The algebraic multilevel iteration methods - theory and applications*, in Proceedings of the Second International Colloquium in Numerical Analysis, August 14-18, 1993, Plovdiv, Bulgaria, 1993, pp. 13–23.

[4] Z.-Z. BAI, *A class of hybrid algebraic multilevel preconditioning methods*, Appl. Numer. Math., 19 (1996), pp. 389–399.

[5] Z.-Z. BAI AND O. AXELSSON, *A unified framework for the construction of various algebraic multilevel preconditioning methods*, in AMLI'96: Proceedings of the Conference on Algebraic Multilevel Iteration Methods with Applications, vol. 1, Nijmegan, The Netherlands, 1996, University of Nijmegan, pp. 63–76.

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

[7] A. BRANDT, *Guide to multigrid development*, in Multigrid Methods, W. Hackbusch and U. Trottenberg, eds., vol. 960 of Lecture Notes in Mathematics, Springer-Verlag, Berlin, 1982, pp. 220–312.

[8] A. BRANDT, *Algebraic multigrid theory: The symmetric case*, in Preliminary Proceedings for the International Multigrid Conference, Copper Mountain, Colorado, April 1983.

[9] A. BRANDT, *Multigrid techniques: 1984 guide with applications to fluid dynamics*, GMD–Studien Nr. 85, Gesellschaft für Mathematik und Datenverarbeitung, St. Augustin, 1984.

[10] ———, *Algebraic multigrid theory: The symmetric case*, Appl. Math. Comput., 19 (1986), pp. 23–56.

[11] A. BRANDT, S. F. MCCORMICK, AND J. W. RUGE, *Algebraic multigrid (AMG) for automatic multigrid solutions with application to geodetic computations.* Report, Inst. for Computational Studies, Fort Collins, Colo., October 1982.

[12] ———, *Algebraic multigrid (AMG) for sparse matrix equations*, in Sparsity and Its Applications, D. J. Evans, ed., Cambridge University Press, Cambridge, 1984.

[13] 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).* Submitted to the SIAM Journal on Scientific Computing special issue on the Fifth Copper Mountain Conference on Iterative Methods, 1998.

[14] W. DAHMEN AND L. ELSNER, *Algebraic multigrid methods and the Schur complement*, in Robust Multi–Grid Methods, W. Hackbusch, ed., vol. 23 of Notes on Numerical Fluid Mechanics, Braunschweig, 1989, Vieweg, pp. 58–68.

[15] J. FUHRMAN, *Outlines of a modular algebraic multilevel method*, in AMLI'96: Proceedings of the Conference on Algebraic Multilevel Iteration Methods with Applications, vol. 1, Nijmegan, The Netherlands, 1996, University of Nijmegan, pp. 141–152.

[16] G. GOLUBOVICI AND C. POPA, *Interpolation and related coarsening techniques for the algebraic multigrid method*, in Multigrid Methods IV, Proceedings of the Fourth European Multigrid Conference, Amsterdam, July 6-9, 1993, vol. 116 of ISNM, Basel, 1994, Birkhäuser, pp. 201–213.

[17] T. GRAUSCHOPF, M. GRIEBEL, AND H. REGLER, *Additive multilevel-preconditioners based on bi-*

linear interpolation, matrix dependent geometric coarsening and algebraic multigrid coarsening for second order elliptic pdes, Appl. Numer. Math., 23 (1997), pp. 63–96.

[18] R. Hempel and C. P. Thompson, *A note on the vectorization of algebraic multigrid algorithms*, Appl. Math. Comput., 26 (1988), pp. 245–256.

[19] R. A. Horn and C. R. Johnson, *Matrix Analysis*, Cambridge University Press, Cambridge, England, 1985.

[20] W. Z. Huang, *Convergence of algebraic multigrid methods for symmetric positive definite matrices with weak diagonal dominance*, Appl. Math. Comput., 46 (1991), pp. 145–164.

[21] Y. A. Kuznetsov, *Algebraic multigrid domain decomposition methods*, Sov. J. Numer. Anal. Math. Modeling, 4 (1989), pp. 361–392.

[22] ———, *Overlapping domain decomposition methods for FE–problems with elliptic singular perturbed operators*, in Fourth International Symposium on Domain Decomposition Methods for Partial Differential Equations, R. Glowinski, Y. A. Kuznetsov, G. A. Meurant, J. Périaux, and O. B. Widlund, eds., Philadelphia, 1991, SIAM, pp. 223–241.

[23] S. F. McCormick, *Multigrid methods for variational problems: general theory for the V–cycle*, SIAM J. Numer. Anal., 22 (1985), pp. 634–643.

[24] G. N. Miranda, *Interpolation weights of algebraic multigrid*, Master's thesis, Naval Postgraduate School, Monterey, CA, June 1997.

[25] C. Popa, *On smoothing properties of SOR relaxation for algebraic multigrid method*, Studii si Cerc. Mat. Ed. Academiei Române, 5 (1989), pp. 399–406.

[26] G. Robinson, *A simple parallel algebraic multigrid*, in Occam and the Transputer, IOS Press, 1991, pp. 62–75.

[27] J. W. Ruge, B. Lee, and S. F. McCormick, *Multigrid methods for solving the time- harmonic Maxwell equations with variable material parameters.* Final report, Dassault Aviation project, 1995.

[28] J. W. Ruge and K. Stüben, *Efficient solution of finite difference and finite element equations by algebraic multigrid (AMG)*, in Multigrid Methods for Integral and Differential Equations, D. J. Paddon and H. Holstein, eds., The Institute of Mathematics and its Applications Conference Series, Clarendon Press, Oxford, 1985, pp. 169–212.

[29] ———, *Algebraic multigrid (AMG)*, in Multigrid Methods, S. F. McCormick, ed., vol. 3 of Frontiers in Applied Mathematics, SIAM, Philadelphia, PA, 1987, pp. 73–130.

[30] K. Stüben, *Algebraic multigrid (AMG): experiences and comparisons*, Appl. Math. Comput., 13 (1983), pp. 419–452.

[31] K. Stüben and U. Trottenberg, *Multigrid methods: Fundamental algorithms, model problem analysis and applications*, in Multigrid Methods, W. Hackbusch and U. Trottenberg, eds., vol. 960 of Lecture Notes in Mathematics, Berlin, 1982, Springer–Verlag, pp. 1–176.

[32] K. Stüben, U. Trottenberg, and K. Witsch, *Software development based on multigrid techniques*, in Proc. IFIP–Conference on PDE Software, Modules, Interfaces and Systems, B. Enquist and T. Smedsaas, eds., Sweden, 1983, Söderköping.

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

[34] P. Vaněk, J. Mandel, and M. Brezina, *Algebraic multigrid on unstructured meshes.* UCD/CCM Report 34, Center for Computational Mathematics, University of Colorado at Denver, December 1994. http://www-math.cudenver.edu/ccmreports/rep34.ps.gz.

[35] ———, *Solving a two-dimensional Helmholtz problem by algebraic multigrid.* UCD/CCM Report 110, Center for Computational Mathematics, University of Colorado at Denver, October 1997. http://www-math.cudenver.edu/ccmreports/rep110.ps.gz.

[36] ———, *Convergence of algebraic multigrid based on smoothed aggregation.* UCD/CCM Report 126, Center for Computational Mathematics, University of Colorado at Denver, February 1998. http://www-math.cudenver.edu/ccmreports/rep126.ps.gz.

[37] W. L. Wan, T. F. Chan, and B. Smith, *An energy-minimizinginterpolation for robust multigrid methods.* UCLA CAM Report 98-6, Department of Mathematics, UCLA, February 1998.