CHAPTER 4

Algebraic Multigrid

J. W. RUGE AND K. STÜBEN

4.1. Introduction. The focus in the application of standard multigrid methods is on the continuous problem to be solved. With the geometry of the problem known, the user discretizes the corresponding operators on a sequence of increasingly finer grids, each grid generally being a uniform refinement of the previous one, with transfer operators between the grids. The coarsest grid is sufficiently coarse to make the cost of solving the (residual) problem there negligible, while the finest is chosen to provide some desired degree of accuracy. The solution process, which involves relaxation, transfer of residuals from fine to coarse grids, and interpolation of corrections from coarse to fine levels, is a very efficient solver for the problem on the finest grid, provided the above "multigrid components" are properly chosen.

Roughly, the efficiency of proper multigrid methods is due to the fact that error only slightly affected by relaxation (*smooth* error) can be easily approximated on a coarser grid by solving the residual equation there, where it is cheaper to compute. This error approximation is interpolated to the fine grid and used to correct the solution. Generally, uniform coarsening and linear interpolation are used, so the key to constructing an efficient multigrid algorithm is to pick the relaxation process that quickly reduces error not in the range of interpolation.

The algebraic multigrid (AMG) approach is developed to solve *matrix equations* using the principles of usual multigrid methods. In contrast to "geometric" multigrid methods, the relaxation used in AMG is *fixed.* The coarsening process (picking the coarse "grid" and defining interpolation) is performed *automatically* in a way that ensures the range of interpolation approximates those errors not efficiently reduced by relaxation. From a theoretical point of view, the process is best understood in the context of symmetric M-matrices, although, in practice, its use is not restricted to such cases. The underlying idea of the coarsening process is to exploit the fact that the form of the error after relaxation can be approximately expressed using the equations themselves, so that the coarse grid can be chosen and

interpolation defined if the equations are used directly. This makes AMG attractive as a "black box" solver. In addition, AMG can be used for many kinds of problems, described below, where the application of standard multigrid methods is difficult or impossible.

(1) The first kind of problem that presents difficulties for geometric multigrid methods is one in which the domain of the problem is complex enough so that any sensible discretization is too fine to serve as the coarsest grid. Even if the finest grid can be coarsened sufficiently, the necessary pattern of coarsening may be quite complex, and the work required to write the needed interpolation routines would be prohibitive. AMG can be used in such situations, since coarsening is automatic. As long as solution of the problem by relaxation remains inefficient, a coarser grid that will help the process can be chosen, without regard to the underlying geometry.

(2) A second set of problems, related to the first, consists of those for which the discretization on the finest grid does not allow uniform coarsening at all. For example, finite element discretizations using irregular triangulations result in such problems. These often arise when finite element applications packages produce meshes adapted to particular domains or structures. These packages were not written with multigrid solution methods in mind, and to change them would be impractical. Again, AMG deals effectively with such problems, since uniform grids are not at all necessary.

(3) A third case consists of problems caused by the operator itself, not the domain. In such cases, when uniform coarsening is used with linear interpolation, it may be difficult to find a relaxation process that smoothes the error sufficiently to admit a good coarse grid correction. At times this may not even be possible; for example, when we are solving diffusion problems with discontinuous coefficients, it is necessary to change the definition of interpolation across the discontinuities (cf. [5]). Another example is the skewed Laplace operator, for which no smoothing in the usual sense is possible. In effect, the discretization splits the grid into two sets (in a checkerboard fashion) that are not connected at all by the matrix, and interpolation between the two sets is useless. AMG, however, is not affected in such cases. Interpolation is defined only from points that are "strongly connected" as defined by the matrix entries, and weights are chosen that reflect the relative strength of each connection.

(4) Finally, some problems are purely discrete. It is even possible (though unlikely) that such a problem has *no* geometric background. Examples of discrete problems arise in geodesy (cf. [Po1]), structural mechanics (trusses and frames, cf. [Ch1]), and economics. Often simplified models of such problems resemble those arising from elliptic partial differential equations, so multigrid processes should provide efficient solutions, but a straightforward application of usual multigrid methods is impossible. In such cases, AMG can be used, since it does not rely on an underlying continuous problem in order to obtain the coarse grid operators.

Section 4.2 of this chapter introduces the general AMG components and contains some introductory remarks. Sections 4.3–4.5 cover theoretical aspects of AMG, while §§ 4.6–4.8 concentrate on practical considerations. These theoretical and practical parts can be read independently; references between them are provided when necessary.

Sections 4.3–4.5 discuss in detail the interaction between the relaxation process and the coarse grid correction necessary for proper behavior of the solution process, adapting and developing theoretical results of Brandt, Mandel, McCormick and others to suit our purposes. Section 4.3 gives sufficient conditions on relaxation and interpolation for the convergence of the V-cycle. Section 4.4 focuses on the relaxation used in AMG, what smoothing means in an algebraic setting, and how it relates to the existing theory. Section 4.5 discusses the coarse grid correction and the definition of interpolation. Some properties of the coarse grid operators are discussed, and results on the convergence of two-level and multi-level convergence are given.

In § 4.6, we give details of an algorithm particularly suited for problems obtained by discretizing a single elliptic, second order partial differential equation (PDE), as well as problems of a similar nature, which we call "scalar" problems. We also give some information on the work and storage requirements. Results of experiments with such problems using both finite difference and finite element discretizations are presented. In addition, we discuss several modifications to the method, including the explicit use of geometric information.

Although the convergence behavior of AMG is insensitive to complex domains and discretizations, the algorithm described in § 4.6 can fail when more than one function is being approximated, as in a discretized system of PDEs ("system" problems). In § 4.7, we discuss modifications of the algorithm that account for this fact, and give results for several examples, including some discrete problems and some problems from elasticity and structural mechanics.

The last section contains concluding remarks on the method and directions of present and future research. These directions include the use of AMG for solving chains of linear problems efficiently. Such chains arise in the solution of time-dependent and nonlinear problems. The approach is to update the coarser grids and grid transfer operators only when necessary, and only locally.

4.2. Terminology, assumptions and notation. Formally, an AMG cycle can be described in the same way as any other multigrid cycle. Differences are mainly due to the different meaning of the terms *grid, grid points,* etc. In this section we will give a formal description, specify certain assumptions and introduce some notation. In particular, in § 4.2.3 we introduce three inner

products, important mainly because their related norms behave differently if applied to "smooth" error vectors. This makes it possible to identify smooth errors by simply comparing different corresponding norm values. Also, and for the same reason, these norms will be used in convergence theorems to formulate reasonable assumptions on the smoothing operators and the coarse grid correction operators *separately*.

4.2.1. AMG components in general. In order to solve a (sparse) linear system of equations

$$A\mathbf{u} = b \quad \text{or} \quad \sum_{j=1}^{n} a_{ij}\mathbf{u}_j = b_i \quad (i = 1, \cdots, n) \tag{2.1}$$

by means of a "multigrid-like" cycling process, we first have to generate a sequence of smaller and smaller systems of equations:

$$A^m\mathbf{u}^m = b^m \quad \text{or} \quad \sum_{j=1}^{n_m} a_{ij}^m\mathbf{u}_j^m = b_i^m \quad (i = 1, \cdots, n_m) \tag{2.2}$$

($m = 1, 2, \cdots, q$ with $n = n_1 > n_2 > \cdots > n_q$) which, for $m > 1$, will formally play the same role as coarse grid equations in geometric multigrid methods.[1] In particular, b^m and u^m will be residuals and corrections, respectively. For $m = 1$, (2.2) is identical to (2.1). Thus, in the context of AMG, a *grid* can simply be regarded as a set of unknowns. Corresponding *grid functions* u^m and b^m are just vectors in $\mathbf{R}^{n_m}$ and the *coarse grid operators* A^m are matrices mapping $\mathbf{R}^{n_m}$ into itself.

Furthermore, we need (linear) transfer operators

$$I_{m+1}^m : \mathbf{R}^{n_{m+1}} \to \mathbf{R}^{n_m} \quad \text{and} \quad I_m^{m+1} : \mathbf{R}^{n_m} \to \mathbf{R}^{n_{m+1}}$$

called *interpolation* and *restriction*, respectively, and smoothing processes of the form

$$u_{\text{new}}^m = G^m u_{\text{old}}^m + (I^m - G^m)(A^m)^{-1}b^m,$$

with corresponding (linear) *smoothing operators* $G^m : \mathbf{R}^{n_m} \to \mathbf{R}^{n_m}$.

Once the above components are known, we can set up a multigrid cycle in the usual way (cf. [538,§ 4.4.2]). Our main concern in §§ 4.3–4.5 will be to get some theoretical insight and to derive some general rules on how to proceed in constructing these components. In § 4.6 we will give an explicit algorithm that has turned out to be quite efficient for many applications.

[1] Compared to geometric multigrid descriptions, the indexing is "backwards" in AMG. Here it is more convenient to use *increasing* numbers m for denoting coarser and coarser "levels," since, starting from a given finest level, the number of coarser levels to be used is not known a priori.

4.2.2. Assumptions and remarks. A striking difference between a geometric and an algebraic multigrid solver is the fact that the latter includes a separate (A-dependent) preparation phase in which the *complete* coarsening process is *fully automatic.* No geometric grid structure is needed. In fact, in the ideal case, an AMG algorithm uses only information explicitly contained in the given matrix A (e.g., in terms of the size of the matrix entries). The coarsening performed by AMG is required to be such that the resulting cycle will become efficient, i.e., it should exhibit typical multigrid convergence speed and should need $O(n)$ operations only. (Of course, the preparation phase itself should also not need more than $O(n)$ operations.)

There is no general algorithm which allows for this without any further assumptions on the matrix A. For our theoretical discussion we will always assume A to be *symmetric* and *positive definite.*[2] (These terms, as well as the definition of the transpose of a matrix, always refer to the *Euclidean inner product,* if there is no explicit statement to the contrary.) Concerning the AMG components, we generally assume the interpolation operators I_{m+1}^m to have *full rank* and the restriction and coarse grid operators to be defined by

$$I_m^{m+1} = (I_{m+1}^m)^T \quad \text{and} \quad A^{m+1} = I_m^{m+1} A^m I_{m+1}^m, \tag{2.3}$$

respectively. These special choices are very convenient, as the symmetry and positive definiteness of A imply that of A^m for all m and all intermediate $(m, m+1)$ *coarse grid correction operators* (cf. [538, § 2.3]),

$$T^m = I^m - I_{m+1}^m (A^{m+1})^{-1} I_m^{m+1} A^m, \tag{2.4}$$

become orthogonal projectors (see § 4.3.1). Multigrid methods using (2.3) are often referred to as *Galerkin type* methods due to their origin in finite element formulations, where these equalities hold naturally in an implicit way. While in geometric multigrid methods (applied to finite difference equations) there are several different possible ways to choose the restriction and coarse grid operators, in a purely algebraic setting there is hardly an alternative to the above choice.

As a smoothing process we usually take the simplest one, namely *Gauss–Seidel relaxation.* Summarizing, we see that in the preparation phase of AMG *only the interpolation operators remain to be defined recursively.*

4.2.3. Some further notation. For any square matrix A we write $A > 0$ $(A \geq 0)$ if A is symmetric and positive definite (positive semi-definite). Correspondingly, $A > B$ $(A \geq B)$ stands for $A - B > 0$ $(A - B \geq 0)$.

[2] We could also allow for positive semi-definite row sum zero matrices. For simplicity, however, we exclude this case from our theoretical discussion. The concrete AMG algorithm which will be presented in § 4.6 does not require A to be necessarily symmetric and positive definite. Limitations of this algorithm will be discussed.

Important parts of our theoretical discussion refer to the "model class" of *symmetric M-matrices,* where a symmetric matrix A is defined to be an M-matrix if it is positive definite and off-diagonally nonpositive. For vectors, $u > 0$ and $u \geq 0$ mean that the corresponding inequalities hold componentwise.

We will usually use the letter u to denote *solution* quantities and the letters v or e to denote *correction* or *error* quantities. The *range* of some matrix operator L is denoted by $R(L)$ and its *spectral radius* by $\rho(L)$. In addition to the *Euclidean inner product* $\langle \cdot, \cdot \rangle$, we will need three different inner products

$$\langle u, v \rangle_0 = \langle Du, v \rangle, \quad \langle u, v \rangle_1 = \langle Au, v \rangle, \quad \langle u, v \rangle_2 = \langle D^{-1}Au, Av \rangle,$$

along with their corresponding norms $\| \cdot \|_i$ $(i = 0, 1, 2)$. Here $A > 0$ is assumed and $D = \operatorname{diag}(A)$. The Euclidean norm is denoted by $\| \cdot \|$. Although these definitions will be used on different levels in the multigrid hierarchy (with A^m instead of A), we will, for simplicity, always use the same symbols.[3]

Usually, when no confusion can arise, we simplify notation. For instance, when we are considering one multigrid level at a time, we omit the indices $m, m+1, \cdots$. On the other hand, when we are considering two consecutive levels only, we use indices h and H instead of m and $m+1$, respectively, to distinguish fine and coarse level quantities. For instance, $A^h \mathbf{u}^h = b^h$ denotes the current fine grid problem, and the corresponding coarse grid operator is denoted by $A^H = I_h^H A^h I_H^h$ with $I_h^H = (I_H^h)^T$ (cf. (2.3)). These particular indices have been chosen only to have a formal similarity to known geometric two-level descriptions; generally, h and H have nothing to do with any kind of discretization parameter.

4.3. General results on convergence. In this section we give some general convergence theorems. This will be in terms of estimates of the V-cycle convergence factor, assuming a fixed A and that corresponding multigrid components are given. Clearly, we are not interested in having

[3] Using the energy inner product $\langle \cdot, \cdot \rangle_1$, we find that these inner products actually belong to a *scale of inner products* defined by

$$\langle u, v \rangle_\alpha := \langle (D^{-1}A)^{\alpha - 1} u, v \rangle_1.$$

A more general scale would be given by using some matrix $B > 0$ instead of D. (Note that $B^{-1}A$ is always symmetric and positive definite with respect to the energy inner product $\langle \cdot, \cdot \rangle_1$.) While, in particular, the corresponding norm $\| \cdot \|_\alpha$ for $\alpha = 2$ plays an important role in geometric multigrid (V-cycle) theory for regular elliptic partial differential problems, the norms for $\alpha < 2$ allow for a more general (W-cycle) theory in less regular cases [375]. However, we will not consider "fractional norms" in this chapter, nor will we consider any B different from D.

convergence for one particular A only, but rather in having *uniform* convergence if A ranges over some reasonable *class* of matrices. Corresponding to the fact that in geometric multigrid the convergence factor of a cycle does not depend on a discretization parameter, AMG interpolation should be such that, in particular, convergence speed does not depend on the size of A. This has to be kept in mind in interpreting the theorems of the next sections which are usually formulated assuming a fixed A.

The results below are contained in [375], [395], but they have been included here to serve as a starting point for a theoretical discussion. Also, they have been reformulated to better suit our purposes and our terminology.

4.3.1. General estimates of the V-cycle convergence factor. In the symmetric and positive definite cases (and assuming (2.3)), it is easiest to investigate convergence with respect to the *energy norm* $\|\cdot\|_1$. This is because all $(m, m+1)$ coarse grid correction operators (2.4) are orthogonal projectors with respect to the energy inner product $\langle\cdot,\cdot\rangle_1$ with $R(T^m)$ being orthogonal to $R(I_{m+1}^m)$. In particular, $\|T^m\|_1 = 1$ and the energy norm of errors after an $(m, m+1)$ coarse grid correction step is minimum (with respect to changes in $R(I_{m+1}^m)$). The following theorem gives a quantitative estimate for the V-cycle convergence factor.

THEOREM 3.1. *Let $A > 0$. Assume that the interpolation operators I_{m+1}^m have full rank and that the restriction and coarse grid operators are defined by* (2.3) *Furthermore, suppose that, for all e^m,*

$$\|G^m e^m\|_1^2 \le \|e^m\|_1^2 - \delta \|T^m e^m\|_1^2 \tag{3.1a}$$

holds with some $\delta > 0$ independently of e^m and m. Then $\delta \le 1$, and—provided that the coarsest grid equation is solved and that at least one smoothing step is performed after each coarse grid correction step—the V-cycle to solve (2.1) *has a convergence factor (with respect to the energy norm) bounded above by $\sqrt{1-\delta}$.*

Proof. The proof is recursive. Let us therefore consider any two consecutive levels of the multigrid hierarchy described in § 4.2.1 (denoted by indices h and H instead of m and $m+1$, respectively), and let us suppose the convergence factor of the V-cycle on the H-level to be $0 \le \eta_H < 1$. In order to derive a bound η_h for the convergence factor of the V-cycle on the h-level as a function of η_H, we first note that (with $\bar{T}^h$ denoting the corresponding coarse grid correction operator) the error after a coarse grid correction step can be written as

$$\bar{T}^h e^h = e^h - I_H^h \tilde{v}^H = e^h - I_H^h v^H + I_H^h (v^H - \tilde{v}^H) = T^h e^h + I_H^h (v^H - \tilde{v}^H).$$

Here $\tilde{v}^H$ and v^H denote the *V-cycle* correction and the corresponding *(h, H)-two-level* correction (i.e., the exact solution of $A^H v^H = I_h^H A^h e^h$),

respectively, and T^h is the (h, H)-two-level correction operator (2.4). Due to our above assumption on the V-cycle convergence on the H-level, we can estimate the following:

$$\|I_H^h(v^H - \tilde{v}^H)\|_1 = \|v^H - \tilde{v}^H\|_1 \leq \eta_H \|v^H\|_1 = \eta_H \|I_H^h v^H\|_1.$$

(Note that because of (2.3), $\|w^H\|_1 = \|I_H^h w^H\|_1$ for every w^H.) As $R(T^h)$ is orthogonal to $R(I_H^h)$ with respect to $\langle \cdot, \cdot \rangle_1$, we obtain

$$\begin{aligned} \|\tilde{T}^h e^h\|_1^2 &= \|T^h e^h\|_1^2 + \|I_H^h(v^H - \tilde{v}^H)\|_1^2 \\ &\leq \|T^h e^h\|_1^2 + \eta_H^2 \|I_H^h v^H\|_1^2 \\ &= \|T^h e^h\|_1^2 + \eta_H^2(\|e^h\|_1^2 - \|T^h e^h\|_1^2). \end{aligned} \tag{3.2}$$

Using assumption (3.1a), and because $T^h \tilde{T}^h = T^h$ and $\|T^h\|_1 = 1$, we find that the following estimation now shows $\eta_h \leq \max\{\eta_H, \sqrt{1-\delta}\}$:

$$\begin{aligned} \|G^h \tilde{T}^h e^h\|_1^2 &\leq \|\tilde{T}^h e^h\|_1^2 - \delta \|T^h \tilde{T}^h e^h\|_1^2 = \|\tilde{T}^h e^h\|_1^2 - \delta \|T^h e^h\|_1^2 \\ &\leq (1 - \delta - \eta_H^2) \|T^h e^h\|_1^2 + \eta_H^2 \|e^h\|_1^2 \leq \|e^h\|_1^2 \max\{\eta_H^2, 1 - \delta\}. \end{aligned}$$

A recursive application of this result proves the theorem.

While (3.1a) is a natural requirement if smoothing is performed *after* the coarse grid correction steps (cf. the next section), a slightly different condition is natural if smoothing is done *before* the coarse grid correction steps:

Remark 3.2 If, instead of (3.1a), we have

$$\|G^m e^m\|_1^2 \leq \|e^m\|_1^2 - \delta \|T^m G^m e^m\|_1^2, \tag{3.1b}$$

the V-cycle convergence factor is bounded above by $1/\sqrt{1+\delta}$ if at least one smoothing step is performed before each coarse grid correction step.

Proof. The proof is similar to the one of the previous theorem. From (3.2) (applied to $G^h e^h$ instead of e^h), and from our assumption (3.1b), we obtain the following two inequalities:

$$\|\tilde{T}^h G^h e^h\|_1^2 \leq (\xi + \eta_H^2(1 - \xi)) \|G^h e^h\|_1^2, \qquad (1 + \delta\xi) \|G^h e^h\|_1^2 \leq \|e^h\|_1^2,$$

respectively, with $\xi = \xi(e^h) = \|T^h G^h e^h\|_1^2 / \|G^h e^h\|_1^2$. By combining these inequalities, we see that $\|\tilde{T}^h G^h e^h\|_1 \leq \eta_h \|e^h\|_1$ with

$$\eta_h^2 = \max_{0 \leq \xi \leq 1} \frac{\xi + \eta_H^2(1 - \xi)}{1 + \delta\xi} = \max\left\{\eta_H^2, \frac{1}{1+\delta}\right\},$$

which proves the remark.

In practice, we usually perform one smoothing step both before *and* after the coarse grid correction. Then, clearly, either one of the conditions (3.1a) and (3.1b) can be used. Moreover, if *both* conditions hold with constants δ_1 and δ_2, respectively, it is easy to see that the convergence factor of the

V-cycle is bounded above by $\sqrt{(1-\delta_1)/(1+\delta_2)}$. The proof is the same as that of Theorem 3.1 (with $G^h e^h$ instead of e^h) with the only exception that, just before taking the maximum, we apply (3.1b) in a similar way as in the proof of Remark 3.2.

4.3.2. Interpretation and sufficient conditions. The conditions (3.1), which reflect the basic interplay between the two essential multigrid processes, smoothing and coarse grid correction, have a simple interpretation. For instance, (3.1a) means that error components e^m that cannot efficiently be reduced by T^m(i.e., for which $\|T^m e^m\|_1 \approx \|e^m\|_1$) have to be effectively and uniformly reducible by G^m. On the other hand, for components that *can* be efficiently reduced by T^m (i.e., for which $\|T^m e^m\|_1 \ll \|e^m\|_1$, or, in other words, that are approximately in $R(I^m_{m+1})$), the smoothing operator G^m is allowed to be ineffective.

Errors for which smoothing is ineffective (i.e., for which $G^m e^m \approx e^m$) are called *smooth*. Thus, the essence of the above remark is, roughly, that *smooth errors have to be approximately in* $R(I^m_{m+1})$. Achieving this will be one main objective in the explicit construction of interpolation operators in AMG. Note that this is the most basic conceptual difference between a usual geometric multigrid solver and an AMG solver. In geometric solvers, all coarser level components are predefined, and the smoothing process has to be *selected* so that the above objective is satisfied. On the other hand, in AMG the smoothing process is *fixed,* and, by exploiting properties of corresponding smooth errors, we explicitly construct suitable (operator-dependent) interpolation operators. Of course, we still need a handy algebraic characterization of smooth errors. For this see § 4.4.3.

In applications, the assumption (3.1a) (and similarly (3.1b)) is usually replaced by two separate inequalities for G^m and T^m which are often called the *smoothing assumption* and the *approximation assumption,* respectively. For this, observe that $\|T^m e^m\|_1$ can be regarded as a (semi-) norm for e^m which, as we mentioned above, necessarily has to be $\ll |e^m\|_1$ for smooth errors. We now replace this norm by a simpler one which does not depend on T^m, namely by $\|e^m\|_2$, but which will turn out to still have a similar property. It is obvious that the following two conditions imply (3.1a) with $\delta = \alpha/\beta$:

$$\begin{aligned} &\|G^m e^m\|_1^2 \le \|e^m\|_1^2 - \alpha \|e^m\|_2^2, \\ &\|T^m e^m\|_1^2 \le \beta \|e^m\|_2^2 \quad (\alpha, \beta > 0). \end{aligned} \tag{3.3a}$$

Similarly,

$$\begin{aligned} &\|G^m e^m\|_1^2 \le \|e^m\|_1^2 - \alpha \|G^m e^m\|_2^2, \\ &\|T^m e^m\|_1^2 \le \beta \|e^m\|_2^2 \quad (\alpha, \beta > 0) \end{aligned} \tag{3.3b}$$

are sufficient for (3.1b) to hold.

The first inequalities in (3.3a) and (3.3b) hold under quite general circumstances, as the next section shows. It is the corresponding second inequalities that present the most problems (cf. § 4.5.5). In § 4.5.3 we will consider a weaker condition that can easily be satisfied under reasonable assumptions on A and still allow for a *two-level* convergence theory.

4.4. The smoothing property. In this section, we will consider smoothing only. Omitting the index m, we will see that, for typical relaxation operators G like that of Gauss–Seidel relaxation, the inequalities (cf. (3.3))

$$\|Ge\|_1^2 \leq \|e\|_1^2 - \alpha \|e\|_2^2, \tag{4.1a}$$

$$\|Ge\|_1^2 \leq \|e\|_1^2 - \alpha \|Ge\|_2^2 \tag{4.1b}$$

hold for all e with some constant $\alpha > 0$ (which is usually different for the two different inequalities). Moreover, under reasonable assumptions on A, they also hold *uniformly* with respect to A.

4.4.1. An auxiliary result. Note first that $\alpha \|e\|_2^2 \leq \|e\|_1^2$ and $\alpha \|Ge\|_2^2 \leq \|e\|_1^2$, respectively, are *necessary* for the inequalities in (4.1) to hold. In order to have smoothing in the sense of (4.1a), for instance, uniformly for A ranging over a certain class $\mathcal{A}$ of matrices, this means that the 2-norm has to be weaker than the 1-norm, uniformly for $A \in \mathcal{A}$. Because $\rho(D^{-1}A) = \sup \{\|e\|_2^2/\|e\|_1^2\}$, this is equivalent to the uniform boundedness of $\rho(D^{-1}A)$ which is satisfied for all important cases under consideration (cf. Examples 4.3).

To be more specific, let G be of the general form

$$G = I - Q^{-1}A, \tag{4.2}$$

with some nonsingular matrix Q. We then have [377] the following lemma.

LEMMA 4.1. *Given any $A > 0$, the two inequalities in* (4.1) *are equivalent to*

$$\alpha Q^T D^{-1} Q \leq Q + Q^T - A$$

and

$$\alpha (A - Q^T) D^{-1} (A - Q) \leq Q + Q^T - A,$$

respectively.

Proof. Using the definition of G, we obtain

$$\|Ge\|_1^2 = \|e\|_1^2 - \langle (Q + Q^T - A) Q^{-1} Ae, Q^{-1} Ae \rangle.$$

Thus, the two inequalities in (4.1) are equivalent to the following:

$$\alpha \|e\|_2^2 \leq \langle (Q + Q^T - A) Q^{-1} Ae, Q^{-1} Ae \rangle,$$

$$\alpha \|Ge\|_2^2 \leq \langle (Q + Q^T - A) Q^{-1} Ae, Q^{-1} Ae \rangle,$$

which in turn are equivalent to

$$\alpha\langle D^{-1}Qe, Qe\rangle \le \langle (Q+Q^T-A)e, e\rangle,$$
$$\alpha\langle D^{-1}(Q-A)e, (Q-A)e\rangle \le \langle (Q+Q^T-A)e, e\rangle,$$

respectively. This proves the lemma.

4.4.2. Specific cases. We first look at standard *Gauss–Seidel relaxation* (cf. [103]). In this case, Q is just the lower triangular part of A (including the diagonal).

THEOREM 4.2. *Let $A>0$ and define, with any positive vector $w=(w_i)$,*

$$\gamma_- = \max_{1\le i\le n}\left\{\frac{1}{w_i a_{ii}}\sum_{j<i} w_j|a_{ij}|\right\}, \qquad \gamma_+ = \max_{1\le i\le n}\left\{\frac{1}{w_i a_{ii}}\sum_{j>i} w_j|a_{ij}|\right\}.$$

Then Gauss–Seidel relaxation satisfies (4.1a) *and* (4.1b) *if* $\alpha \le 1/(1+\gamma_-)(1+\gamma_+)$ *and* $\alpha \le 1/\gamma_-\gamma_+$, *respectively.*

Proof. We first observe that $Q+Q^T-A=D$. Thus, because of Lemma 4.1, the two inequalities (4.1) are equivalent to

$$\alpha \le 1/\rho(D^{-1}QD^{-1}Q^T) \quad \text{and} \quad \alpha \le 1/\rho(D^{-1}(A-Q)D^{-1}(A-Q^T)),$$

respectively, which in turn are implied by

$$\alpha \le 1/|D^{-1}Q|\,|D^{-1}Q^T| \quad \text{and} \quad \alpha \le 1/|D^{-1}(A-Q)|\,|D^{-1}(A-Q^T)|,$$

respectively. Here $|\cdot|$ stands for any arbitrary matrix norm induced by a vector norm. The special choice

$$|L| = |L|_w = \max_{1\le i\le n}\left\{\frac{1}{w_i}\sum_{j=1}^{n} w_j|l_{ij}|\right\} \tag{4.3}$$

proves the theorem.

Examples 4.3. The inequalities (4.1) hold uniformly for essentially all matrices that are of interest here. For instance, for any sparse $A>0$ with no more than a *fixed* number l of nonvanishing off-diagonal entries per row, the choice $w_i = 1/\sqrt{a_{ii}}$ gives $\gamma_- < l$ and $\gamma_+ < l$ because $a_{ij}^2 < a_{ii}a_{jj}$ $(i\ne j)$. Thus, for any such A, (4.1a) and (4.1b) are satisfied with $\alpha=(1+l)^{-2}$ and $\alpha=l^{-2}$, respectively. From a practical point of view, however, these values are far too pessimistic in typical situations. In practice, we usually have $\sum_{j\ne i}|a_{ij}| \approx a_{ii}$, which means that by selecting $w_i=1$ we can expect γ_- and γ_+ to be close to or even less than 1. An important class of matrices in which we have uniform smoothing is the class of *symmetric M-matrices*. To see this, note that for any such matrix A there exists a vector $z>0$ with $Az>0$ (cf. [Sc1]). By choosing $w=z$ in Theorem 4.2, we obtain

$$\gamma_- = \max_{1\le i\le n}\left\{\frac{1}{z_i a_{ii}}\sum_{j<i} z_j|a_{ij}|\right\} = \max_{1\le i\le n}\left\{1-\frac{1}{z_i a_{ii}}\sum_{j\le i} z_j a_{ij}\right\} < 1.$$

Similarly, we obtain $\gamma_+ < 1$. Thus, (4.1a) and (4.1b) are satisfied with $\alpha = \frac{1}{4}$ and $\alpha = 1$, respectively.

Gauss–Seidel relaxation is the scheme we usually use for smoothing in AMG. With respect to vector and parallel computers, Jacobi-type relaxations are also of interest. In particular, standard *Jacobi relaxation* has properties similar to Gauss–Seidel relaxation, if some relaxation parameter ω is used, i.e., if $Q = D/\omega$.

THEOREM 4.4. *Let $A > 0$ and $\gamma_0 \geq \rho(D^{-1}A)$. Then Jacobi relaxation with relaxation parameter $0 < \omega < 2/\gamma_0$ satisfies* (4.1a) *and* (4.1b) *if $\alpha \leq \omega(2 - \omega\gamma_0)$ and $\alpha \leq \omega \min\{2, (2 - \omega\gamma_0)/(1 - \omega\gamma_0)^2\}$, respectively.*

Proof. Using Lemma 4.1, we see that the two inequalities (4.1) are equivalent to

$$\rho\left(D^{-1}A + \frac{\alpha}{\omega^2}I\right) \leq \frac{2}{\omega} \quad \text{and} \quad \rho\left(D^{-1}A + \alpha\left(D^{-1}A - \frac{1}{\omega}I\right)^2\right) \leq \frac{2}{\omega},$$

which in turn are implied by

$$\gamma_0 + \frac{\alpha}{\omega^2} \leq \frac{2}{\omega} \quad \text{and} \quad \max_{0 \leq \lambda \leq \gamma_0} \left\{\lambda + \alpha\left(\lambda - \frac{1}{\omega}\right)^2\right\} \leq \frac{2}{\omega},$$

respectively. From this, our statement follows by a simple calculation.

As a bound for $\rho(D^{-1}A)$ we can use, for instance, $\gamma_0 = |D^{-1}A|_w$ with any vector $w > 0$ (see (4.3)). For scalar PDE applications we typically have $\gamma_0 \approx 2$. Also, for symmetric M-matrices we can choose $\gamma_0 = 2$ (cf. Examples 4.3). In terms of γ_0, the optimal values for ω (which give the largest values of α) are easily computed to be $\omega^* = 1/\gamma_0$ in the case of (4.1a) and $\omega^* = 3/2\gamma_0$ in the case of (4.1b). For these optimal parameters, the smoothing inequalities are satisfied with $\alpha = 1/\gamma_0$ and $\alpha = 3/\gamma_0$, respectively, giving $\alpha = \frac{1}{2}$ and $\alpha = \frac{3}{2}$ if $\gamma_0 = 2$.

4.4.3. Interpretation of algebraic smoothness. In AMG we *define* an error e to be smooth if it is slow to converge with respect to G, i.e., if $\|Ge\|_1 \approx \|e\|_1$. Note that, in this generality, the term "smooth" is not always related to what is called smooth in a geometric environment. Actually, e being smooth does not mean much more here than that e has to be taken care of by the coarse grid correction in order to obtain an efficient multigrid method. From an algebraic point of view, this is the important point in distinguishing smooth and nonsmooth error. Geometrically, however, the term smooth is used in a more restrictive (viz. the "natural") way. In fact, without any further assumptions on A (besides $A > 0$), an algebraically smooth error may well be highly oscillating geometrically, e.g., in certain cases of off-diagonally nonnegative matrices. Also, there may not be any algebraically smooth error at all, e.g., if A is a strongly diagonally dominant

M-matrix. Such cases, however, are generally not those that typically arise from interesting PDE applications or those that require a particular fast solution method.

For common relaxation schemes, a smooth error e is algebraically characterized by the fact that the residual $r = Ae$ is small in some sense compared to the error itself. This can be seen either directly from their definition (4.2) or from the properties (4.1). The latter, for instance, show that a smooth error necessarily has to satisfy $\|e\|_2 \ll \|e\|_1$, or, more explicitly, $\sum_i r_i^2/a_{ii} \ll \sum_i r_i e_i$. Therefore, on the average, we can expect $|r_i| \ll a_{ii}\,|e_i|$ for all i. Consequently, we obtain a quite good approximation for e_i as a function of its "neighboring" error values e_j ($j \in N_i$) by evaluating

$$(r_i =)\ a_{ii}e_i + \sum_{j \in N_i} a_{ij}e_j = 0, \tag{4.4}$$

where $N_i = \{j \neq i : a_{ij} \neq 0\}$ denotes the *neighborhood* of i. This simple fact about smooth errors gives the basic information needed for the construction of interpolation, one goal of which is to guarantee that smooth error is approximately in the range of interpolation (cf. § 4.3.2). However, as (4.4) is usually not "local enough," it cannot be used directly in order to obtain an efficient interpolation formula for AMG (see Example 5.1 below), but rather has to be "truncated." In § 4.5 we will discuss some possibilities.

In the important case of symmetric M-matrices one easily obtains some more information on what smooth error looks like (cf. [103]). For this, note first that

$$\|e\|_1^2 \leq \|e\|_0\,\|e\|_2, \tag{4.5}$$

which follows from $\langle Ae, e\rangle = \langle D^{-1/2}Ae, D^{1/2}e\rangle$ by applying Schwarz' inequality. This shows that $\|e\|_2 \ll \|e\|_1$ implies $\|e\|_1 \ll \|e\|_0$, or, more explicitly,

$$\langle Ae, e\rangle = \frac{1}{2}\sum_{i,j}(-a_{ij})(e_i - e_j)^2 + \sum_i \left(\sum_j a_{ij}\right)e_i^2 \ll \sum_i a_{ii}e_i^2.$$

For the most important case that $\sum_{j\neq i}|a_{ij}| \approx a_{ii}$, this means that, on the average for each i,

$$\sum_{j\neq i}\frac{|a_{ij}|}{a_{ii}}\frac{(e_i - e_j)^2}{e_i^2} \ll 1.$$

In other words, smooth error generally *varies slowly in the direction of strong connections,* i.e., from e_i to e_j if $|a_{ij}|/a_{ii}$ is relatively large. This observation is important in connection with the question of the truncation mentioned above (see §§ 4.5.4 and 4.6.1).

4.5. The coarse grid approximation and convergence. The main topic of this section is how the coarsening enters the question of smoothing,

which was not considered in the previous section. As the corresponding discussion will refer to only two consecutive levels at a time, we will, according to our convention, use indices h and H to distinguish fine and coarse level quantities.

All of the following considerations could be applied in the framework of vectors, matrices, etc. The transfer of information between different levels, however, becomes (at least formally) more transparent and easier to describe if we use some kind of grid terminology as introduced in § 4.5.1. Also, an AMG method then becomes formally completely analogous to a usual geometric multigrid method.

We have seen in § 4.3 that the approximation property (second inequality in (3.3)) is sufficient for a V-cycle convergence theory. However, even if we know that coarsening strategies with this property exist (e.g., in regular elliptic scalar PDE problems), it is quite difficult to determine one in an automatic AMG process that uses *only algebraic information.* This will be discussed in § 4.5.5. In § 4.5.3, we first consider a weaker condition, due to Brandt [103], which will turn out to be sufficient for a *two-level* convergence theory. As an application, we will treat symmetric M-matrices in § 4.5.4.

Some major results given in §§ 4.5.3 and 4.5.4 are also contained in [103]. However, we consider some additional results which are relevant for the practical development of an AMG algorithm and we also have a different point of view here in that we stress connections to the general convergence theorems of § 4.3.

4.5.1. Grid terminology. As was mentioned earlier, we can, for instance, introduce grids as sets of unknowns. By accepting some loss in generality, however, we can go one step further, namely, if we regard the sets of coarse level variables as being "subsets" of the finer level variables: in terms of any two consecutive levels, h and H, we assume that each coarse level variable u_k^H is used to directly correct some uniquely defined finer level variable $u_{i(k)}^h$.[4] Thus, the set of variables on level h can be split into two disjoint subsets: the first one contains the variables also represented in the coarser level (*C-variables*), and the second one is just the complementary set (*F-variables*). The corresponding sets of indices are denoted by C^h and F^h, respectively. By renumbering the coarse level unknowns (and all related quantities) such that they have the same index as their finer grid analogues, we can now, for instance, write the H-level equations in the form

$$\sum_{j \in C^h} a_{ij}^H \mathbf{u}_j^H = b_i^H \qquad (i \in C^h). \tag{5.1}$$

[4] This assumption is "natural" for all cases we have in mind in this paper, and it corresponds formally to the geometric multigrid case if coarser grids are defined to be subsets of the finer ones.

By identifying each $i \in \Omega^h = C^h \cup F^h$ with some fictitious *point* in the plane, we interpret the equations $A^h\mathbf{u}^h = b^h$ and $A^H\mathbf{u}^H = b^H$ as *grid equations* on the fictitious *grid* Ω^h and the subgrid $\Omega^H = C^h$, respectively. In this sense, referring to a point $i \in \Omega^h$ means nothing else than referring to the variable u_i^h.

According to the above terminology, the actual h to H coarsening proceeds in three steps. First we have to decide which variables (points) are to become C- and which F-variables (-points), i.e., we have to define a splitting $\Omega^h = C^h \cup F^h$. Then, with $\Omega^H = C^h$, the weights $w_{ik} = w_{ik}^h$ of the interpolation

$$(5.2) \qquad (I_H^h e^H)_i = \sum_{k \in C^h} w_{ik} e_k^H \quad (i \in \Omega^h) \quad \text{with } w_{ik} = \delta_{ik} \text{ if } i \in C^h$$

have to be defined (δ_{ik} denotes the Kronecker symbol). Coarsening is concluded by the computation of $I_h^H = (I_H^h)^T$ and $A^H = I_h^H A^h I_H^h$. Clearly, for reasons of efficiency, interpolation should be "local," i.e., for each $i \in F^h$, we must have $w_{ik} = 0$ unless $k \in C_i^h$, where $C_i^h \subset C^h$ is some reasonable small set of interpolation points. For reasons of easy referencing, we call an interpolation of the above form *standard interpolation*. Note that each standard interpolation has full rank. In practice, the selection of a coarser grid and the definition of a corresponding interpolation formula are closely coupled processes. Whenever we talk about "construction of interpolation," we actually mean both processes.

When we are setting up more than two levels, all of the above applies recursively in a straightforward manner starting from the given finest level.

Finally, we define *connections* between grid points in a natural way in the sense of the directed graph which is associated with any matrix. That is, for any level h, we define a point $i \in \Omega^h$ to be (directly) *connected* to a point $j \in \Omega^h$ if $a_{ij}^h \neq 0$. Correspondingly, we define the (direct) *neighborhood* of a point $i \in \Omega^h$ by

$$N_i^h = \{j \in \Omega^h : j \neq i,\ a_{ij}^h \neq 0\}.$$

4.5.2. Convergence versus numerical work. Before we investigate the question of coarsening, we want to point out that, in the context of AMG, it is not enough to merely look at convergence: convergence does not mean anything if *numerical work* is not taken into account. This point is much more crucial here than it is in standard geometric multigrid applications where the numerical work (per cycle, say) is approximately known a priori. In an algebraic multigrid process, in general, nothing definite is known a priori about the numerical work. Consequently, it is very important to find conditions (for the practical construction of interpolation) that not only give good convergence, but also allow for a reasonable control of the numerical work by the algorithm (cf. § 4.6.3). The following example stresses the importance of this remark.

Example 5.1. It is easy to develop an AMG solver that has a convergence factor of 0, i.e., that solves (2.1) exactly in one cycle only. Assuming any $A > 0$, we find that the preparation phase of such an algorithm runs as follows. First, the set of all n variables is subdivided somehow into two nonempty sets of C- and F-variables (cf. § 4.5.1) in such a way that each F-variable has *no connection to any other F-variable.* (Note that this is always possible unless $n = 1$.) The interpolation (5.2) is defined by $w_{ik} = -a_{ik}/a_{ii}$ $(i \in F,\ k \in C)$. After the computation of (2.3), this process is recursively repeated until only very few unknowns are left. As there are no F-to-F connections on any level, it is easy to see that $R(G^m) \subseteq R(I^m_{m+1})$ holds for all m, if only the order of (Gauss–Seidel) relaxation is arranged such that *F-variables are always relaxed last.* Because $T^m e^m = 0$ if $e^m \in R(I^m_{m+1})$, we therefore have $T^m G^m e^m \equiv 0$. As a consequence, (3.1b) is satisfied for any arbitrarily large δ, which proves our statement. However, a closer examination immediately shows that such an algorithm is completely useless for practical purposes: with increasing m, not only does the reduction of the number of unknowns usually become extremely slow from one level to the next, but there is also a tremendous fill-in in the coarse grid matrices. The major drawback of the above coarsening process is that it is actually defined just to give $R(G^m) \subseteq R(I^m_{m+1})$. *Smoothing effects are not really exploited,* and this condition is too strong to be practical.

4.5.3. Two-level convergence. In this section we will consider two-level methods only. Without essential loss of generality, we therefore restrict our investigation to the case where smoothing is performed *after* each coarse grid correction step. Assuming that the corresponding smoothing property (cf. (3.3a))

$$\|G^h e^h\|_1^2 \leq \|e^h\|_1^2 - \alpha \|e^h\|_2^2 \qquad (\alpha > 0) \tag{5.3}$$

holds, we will give a general condition on I^h_H that allows for an estimation of the two-level convergence factor $\|G^h T^h\|_1$. As a motivation recall that, due to (5.3), error reduction by smoothing becomes inefficient if $\|e^h\|_2 \ll \|e^h\|_1$. Therefore, in order to obtain reasonable two-level convergence, the least we have to require is that the coarse grid correction T^h raises the error norm $\|e^h\|_2$ relative to $\|e^h\|_1$. More precisely, we need an inequality of the form

$$\|T^h e^h\|_1^2 \leq \beta \|T^h e^h\|_2^2 \qquad (\beta > 0). \tag{5.4}$$

Note that this is just the approximation property in (3.3), required for $e^h \in R(T^h)$ only. Instead of (5.4), a more practical, sufficient condition is used in the following theorem.

THEOREM 5.2. *Let $A^h > 0$ and let G^h satisfy* (5.3). *Suppose that the interpolation I_H^h has full rank and that, for each e^h,*

$$\min_{e^H} \|e^h - I_H^h e^H\|_0^2 \leq \beta \|e^h\|_1^2, \tag{5.5}$$

with $\beta > 0$ independent of e^h. Then $\beta \geq \alpha$, and the (h, H)-two-level convergence factor satisfies $\|G^h T^h\|_1 \leq \sqrt{1 - \alpha/\beta}$.

Proof. We first show that assumption (5.5) implies (5.4). As $R(T^h)$ is orthogonal to $R(I_H^h)$ with respect to $\langle \cdot, \cdot \rangle_1$, we have, for all $e^h \in R(T^h)$ and for all e^H,

$$\|e^h\|_1^2 = \langle A^h e^h, e^h - I_H^h e^H \rangle.$$

A simple calculation using Schwarz' inequality (writing A and D instead of A^h and D^h, respectively, and observing that $D^{-1}A > 0$ with respect to $\langle \cdot, \cdot \rangle_1$) yields

$$\begin{aligned} \|e^h\|_1^2 &= \langle A^{1/2}(D^{-1}A)^{1/2} e^h, A^{1/2}(D^{-1}A)^{-1/2}(e^h - I_H^h e^H) \rangle \\ &\leq \|A^{1/2}(D^{-1}A)^{1/2} e^h\| \, \|A^{1/2}(D^{-1}A)^{-1/2}(e^h - I_H^h e^H)\| \\ &= \|e^h\|_2 \, \|e^h - I_H^h e^H\|_0. \end{aligned}$$

Because of (5.5), we therefore obtain $\|e^h\|_1^2 \leq \beta \|e^h\|_2^2$ for all $e^h \in R(T^h)$, which proves (5.4). The convergence estimate of the theorem is now an immediate consequence of (5.4) and (5.3):

$$\begin{aligned} \|G^h T^h e^h\|_1^2 &\leq \|T^h e^h\|_1^2 - \alpha \|T^h e^h\|_2^2 \leq \|T^h e^h\|_1^2 - \frac{\alpha}{\beta} \|T^h e^h\|_1^2 \\ &= \left(1 - \frac{\alpha}{\beta}\right) \|T^h e^h\|_1^2 \leq \left(1 - \frac{\alpha}{\beta}\right) \|e^h\|_1^2. \end{aligned}$$

Assume the interpolation to be of the standard type (5.2); the following theorem gives a condition sufficient for (5.5) and relating the interpolation weights w_{ik} to the matrix entries a_{ij}^h. Here and in the following we use the notation $s_i = \sum_{k \in C} w_{ik}$ for the ith row sum of interpolation.

THEOREM 5.3. *Let $A^h > 0$ and assume, for any given set $C = C^h$ of C-points, that I_H^h is of the form* (5.2) *with $w_{ik} \geq 0$ and $s_i \leq 1$. Then property* (5.5) *is satisfied if the following two inequalities hold with $\beta > 0$ independently of $e = e^h$:*

$$\begin{aligned} &\sum_{i \in F} \sum_{k \in C} a_{ii}^h w_{ik} (e_i - e_k)^2 \leq \frac{\beta}{2} \sum_{i,j} (-a_{ij}^h)(e_i - e_j)^2, \\ &\sum_{i \in F} a_{ii}^h (1 - s_i) e_i^2 \leq \beta \sum_i \left(\sum_j a_{ij}^h \right) e_i^2. \end{aligned} \tag{5.6}$$

Proof. Let e^h be arbitrary and define $e_k^H = e_k^h (k \in C)$. With this particular e^H and omitting the index h, inequality (5.5) reads

$$\sum_{i \in F} a_{ii} \Big(e_i - \sum_{k \in C} w_{ik} e_k \Big)^2 \leq \beta \Big(\frac{1}{2} \sum_{i,j} (-a_{ij})(e_i - e_j)^2 + \sum_i \Big(\sum_j a_{ij} \Big) e_i^2 \Big).$$

Estimating the left-hand side by

$$\begin{aligned} \sum_{i \in F} a_{ii} \Big(e_i - \sum_{k \in C} w_{ik} e_k \Big)^2 &= \sum_{i \in F} a_{ii} \Big(\sum_{k \in C} w_{ik} (e_i - e_k) + (1 - s_i) e_i \Big)^2 \\ &\leq \sum_{i \in F} a_{ii} \Big(\sum_{k \in C} w_{ik} (e_i - e_k)^2 + (1 - s_i) e_i^2 \Big), \end{aligned}$$

we see that (5.6) is sufficient for (5.5).

4.5.4. Application to M-matrices. In this section, we will discuss interpolation in some detail for symmetric M-matrices A^h. Recall that, in this class of matrices, we have uniform smoothing (cf. § 4.4.2). Under the additional assumption of *weak diagonal dominance,* i.e., $\sum_j a_{ij}^h \geq 0$ for all i,[5] we will consider ways to interpolate such that (5.5) also holds uniformly. Theorem 5.2 then guarantees uniform two-level convergence. The following considerations will be mainly theoretical; for an efficient algorithm and practical applications we refer to §§ 4.6.1, 4.6.2 and 4.6.4.

As we already mentioned in § 4.4.3, the general idea in deriving an interpolation formula for some F-point i is to use the ith residual equation (4.4) of A^h. In particular, for each $i \in F$ with $N_i \subseteq C$, a good formula is obtained by defining the set of interpolation points by $C_i = N_i$ with corresponding weights $w_{ik} = |a_{ik}^h| / a_{ii}^h$ $(k \in C_i)$ (cf. Example 5.1). For reasons of localness, however, we want C_i to be small. Therefore, we generally allow $D_i = N_i - C_i \neq \emptyset$ and we have to make a decision about how to "distribute" the noninterpolatory connections a_{ij}^h $(j \in D_i)$ in interpolating to i. This distribution is the most crucial point in the development of a good interpolation: the goal of obtaining local interpolation has to be achieved without losing too much precision compared to (4.4).

The interpolation formula introduced in § 4.5.4.1 below is quite simple in that all a_{ij}^h $(j \in D_i)$ are added to the diagonal element a_{ii}^h. In terms of (4.4), this means that we replace e_j by e_i for all $j \in D_i$. If $|a_{ij}^h|$ is small, this replacement should not cause too much harm. If, on the other hand, $|a_{ij}^h|$ is large, we generally can expect this replacement to be reasonable due to smoothness arguments (cf. § 4.4.3). Although this approach leads to a quite acceptable AMG algorithm (cf. [536], [486]), it can be improved considerably by interpolating in a different way which is described in § 4.5.4.2. The main idea here is to distribute the noninterpolatory connections a_{ij}^h

[5] Such matrices are called *positive type* matrices in [103].

($j \in D_i$) in a weighted average way to the interpolation points in C_i instead of simply adding them to the diagonal.

A combination of the above two approaches is used in actual practice. This is, in general, considerably better for geometrically posed problems than the simple approach of § 4.5.4.1. The reasons, however, are not apparent from merely algebraic estimates as given below; rather, we must take certain geometric arguments into account (see § 4.6.1).

4.5.4.1. Interpolation along direct connections. In this section we consider interpolation along *direct* connections. That is, we consider only interpolation points C_i with $C_i \subseteq N_i \cap C$ and corresponding weights of the form

$$w_{ik} = \eta_i |a_{ik}^h| \qquad (i \in F,\ k \in C_i), \tag{5.7}$$

with $0 \le \eta_i \le 1/\sum_{l \in C_i} |a_{il}^h|$, which just ensures $s_i \le 1$. In order for property (5.5) to hold, it is, due to Theorem 5.3, obviously sufficient to require for every $i \in F$, $k \in C_i$

$$0 \le a_{ii}^h w_{ik} \le \beta |a_{ik}^h|, \qquad 0 \le a_{ii}^h (1 - s_i) \le \beta \sum_j a_{ij}^h. \tag{5.8}$$

From these simple inequalities, we can easily derive more practical conditions, which can be used to effectively develop an automatic coarsening algorithm that has β as input parameter and chooses interpolation with weights (5.7) satisfying (5.8). An example which has been tested in practical applications is given in the following theorem.

THEOREM 5.4. *Let $\beta \ge 1$ be fixed. Assume for any symmetric, weakly diagonally dominant M-matrix A^h the C-points are picked such that, for each $i \in F$, there is a nonempty set $C_i \subseteq N_i \cap C$ with*

$$\beta \sum_{j \notin C_i} a_{ij}^h \ge a_{ii}^h \tag{5.9}$$

and define the interpolation weights (5.7) *by $\eta_i = 1/\sum_{j \notin C_i} a_{ij}^h$. Then property* (5.5) *is satisfied.*

For fixed β, there are usually many possible ways to choose C and C_i so that they satisfy (5.9). Aside from this purely algebraic condition, there are, however, others that should be met (at least approximately) to ensure efficiency in practice. For instance (further requirements are discussed in § 4.6.1), we want C to be as small as possible. Consequently, we should try to satisfy (5.9) with sets C_i as small as possible. This, in turn, requires arranging a concrete algorithm such that each F-point i is *strongly* connected to its interpolation points, i.e., we actually want each $|a_{ij}|$ ($j \in C_i$) to be comparable in size to the largest of the $|a_{ik}|$ ($k \in N_i$).

We want to make a few remarks concerning the choice of the parameter β. Clearly, the larger β, the weaker the assumption (5.9) is. In particular, for large β, (5.9) allows for rapid coarsening, but, due to Theorem 5.2, the two-level convergence speed will be very slow. On the other hand, the choice $\beta = 1$ gives best convergence, but will lead to an extremely expensive method (cf. Example 5.1). For real computations, experience shows that, on the average, a reasonable compromise is given by $\beta = 2$, which means that about 50% of the total strength of connections of every F-point will be represented on the coarser level (and used for interpolation). If applied recursively in a real multi-level cycle (cf. § 4.5.4.4), this still limits the efficiency of coarsening so that, in practice, a corresponding AMG algorithm is not fully satisfactory (in particular, if the occurring matrices have many row entries of similar size (see [486])).

4.5.4.2. More general interpolation. Instead of (5.7), we now consider weights of the more general form

$$w_{ik} = \eta_i \left(|a_{ik}^h| + \sum_{j \in D_i} \eta_{ik}^{(j)} |a_{ij}^h| \right) \qquad (i \in F,\ k \in C_i), \tag{5.10}$$

with nonnegative η_i and $\eta_{ik}^{(j)}$. This form is obtained if we perform, for every F-point i, a *weighted distribution* of all noninterpolatory connections a_{ij}^h $(j \in D_i)$ to the interpolation points $k \in C_i$ using the distribution weights $\eta_{ik}^{(j)}$. Here we do not require C_i to be a subset of N_i, which means that we allow *long-range interpolation*, i.e., interpolation from points to which there is no *direct* connection.

Again, there are several reasonable ways to specify η_i and the distribution weights $\eta_{ik}^{(j)}$. In the following theorem, we choose $\eta_{ik}^{(j)}$ to be proportional to a_{jk}^h, i.e., we distribute only along direct connections. The essential condition for this to be reasonable is that, for every F-point i, the *total* connection of each noninterpolatory point j to the set of interpolation points C_i is strong enough, i.e., $\sum_{l \in C_i} |a_{jl}^h|$ is sufficiently large. The following theorem expresses this more quantitatively.

THEOREM 5.5. *Let $\zeta > 0$ be a fixed number. Assume, for any symmetric, weakly diagonally dominant M-matrix A^h, that the C-points and $C_i \subseteq C$ $(i \in F)$ are picked such that, for each j and $k \in N_j \cap C$, we have*

$$\zeta \sum_{l \in C_i} |a_{jl}^h| \geq \sum_{v \in F_{jk}} |a_{jv}^h| \qquad (i \in F_{jk}), \tag{5.11}$$

where $F_{jk} = \{i \in F : k \in C_i \text{ and } j \in D_i\}$. Then, if the weights (5.10) *of interpolation are defined by $\eta_i = 1/a_{ii}^h$ and $\eta_{ik}^{(j)} = |a_{jk}^h| / \sum_{l \in C_i} |a_{jl}^h|$, property* (5.5) *is satisfied with some β that depends only on ζ but not on A^h.*

Proof. For the proof, we apply Theorem 5.3. We first observe that the second inequality in (5.6) is satisfied for every $\beta \geq 1$ because $a_{ii}(1 - s_i) =$

$\sum_j a_{ij}$ $(i \in F)$. The first inequality in (5.6) follows from the estimate

$$a_{ii}w_{ik}(e_i - e_k)^2 = |a_{ik}|\,(e_i - e_k)^2 + \sum_{j\in D_i} \eta_{ik}^{(j)}\,|a_{ij}|\,(e_i - e_k)^2$$

$$\leq |a_{ik}|\,(e_i - e_k)^2 + 2\sum_{j\in D_i} \eta_{ik}^{(j)}\,|a_{ij}|\,((e_i - e_j)^2 + (e_j - e_k)^2)$$

by observing that

$$\sum_{i\in F}\sum_{k\in C_i}\sum_{j\in D_i} \eta_{ik}^{(j)}\,|a_{ij}|\,(e_i - e_j)^2 = \sum_{i\in F}\sum_{j\in D_i}\Big(\sum_{k\in C_i} \eta_{ik}^{(j)}\Big)\,|a_{ij}|\,(e_i - e_j)^2$$

$$= \sum_{i\in F}\sum_{j\in D_i} |a_{ij}|\,(e_i - e_j)^2$$

and

$$\sum_{i\in F}\sum_{k\in C_i}\sum_{j\in D_i} \eta_{ik}^{(j)}\,|a_{ij}|\,(e_j - e_k)^2 = \sum_{j}\sum_{k\in C}\Big(\sum_{i\in F_{jk}} \frac{|a_{ji}|}{\sum_{l\in C_i}|a_{jl}|}\Big)\,|a_{jk}|\,(e_j - e_k)^2$$

$$\leq \zeta \sum_{j}\sum_{k\in C}\Big(\sum_{i\in F_{jk}} \frac{|a_{ji}|}{\sum_{v\in F_{jk}}|a_{jv}|}\Big)\,|a_{jk}|\,(e_j - e_k)^2$$

$$= \zeta \sum_{j}\sum_{k\in C} |a_{jk}|\,(e_j - e_k)^2.$$

4.5.4.3. Extensions and further remarks. In constructing interpolation formulas in the previous two sections, we used Theorem 5.3 as our theoretical tool. From the basic inequalities (5.6), it can be seen that the essential ideas carry over to matrices that also contain some (small) *positive* off-diagonal elements. For instance, for the construction of interpolation in § 4.5.4.1, it is sufficient to require that A^h satisfies $a_{ij}^h < 0$ $(i \in F,\ j \in C_i)$ and, for all e^h,

$$\sum_{i,j}(-a_{ij}^h)(e_i^h - e_j^h)^2 \geq \zeta \sum_{i\in F}\sum_{j\in C_i}(-a_{ij}^h)(e_i^h - e_j^h)^2$$

with $\zeta > 0$.[6] For cases involving higher order difference approximations to second order elliptic problems, or for certain problems involving mixed derivatives, such relations can usually be derived by using simple estimates like

$$\frac{\phi\psi}{\phi + \psi}(\alpha + \beta)^2 \leq \phi\alpha^2 + \psi\beta^2 \qquad (\phi,\ \psi > 0).$$

A last remark refers to the assumption of *weak diagonal dominance*. While uniform smoothing is guaranteed in the class of *all* symmetric M-matrices A^h, (5.5) cannot be expected to hold in this class with β being independent

[6] Such matrices are called *essentially positive type* in [103].

of A^h. This becomes obvious if, e.g., we consider the subclass of matrices A^h_μ ($\mu < \lambda_0$) defined by discretizing the Helmholtz operator $-\Delta - \mu I$ on the unit square with fixed mesh size h and Dirichlet boundary conditions. Here, λ_0 is the first eigenvalue of A^h_0. If we select $e^h = e^h_0$ as the corresponding (normalized) eigenfunction, we have for each $A^h = A^h_\mu$ that $\|e^h\|_1^2 = \lambda_0 - \mu$. Thus, for (5.5) to hold independently of μ if $\mu \to \lambda_0$, its left side necessarily has to approach zero, i.e., I^h_H has to be such that this particular eigenfunction e^h is arbitrarily close to $R(I^h_H)$, which is generally not true unless special techniques are used (see § 4.6.6).

Note, however, that A^h can be allowed to have *zero row sums*. This is because the first eigenvector, which *is* in the range of interpolation by definition, is then constant. In fact, as is easily seen, constants are always interpolated to constants and the zero row sum property is preserved on all coarser grids.

4.5.4.4. Recursive application in multi-level cycles. In order to recursively set up a multi-level cycle, we need, first of all, that the finest grid operator's essential properties, which have been exploited in our considerations, carry over to the coarser grids. The following theorem shows that this can be achieved for M-matrices.

THEOREM 5.6. *Let A^h be any symmetric, weakly diagonally dominant M-matrix and let the interpolation weights satisfy* (5.8) *with some $\beta \le 2$. Then A^H is also a symmetric, weakly diagonally dominant M-matrix.*

Proof. First, because of the Galerkin formulation, A^H is symmetric and positive definite. In order to show that A^H is off-diagonally nonpositive, we write its entries in their explicit form:

$$a^H_{kl} = \sum_{i,j} w_{ik} a^h_{ij} w_{jl} \qquad (k, l \in C).$$

Because $w_{ik} = \delta_{ik}$ (if $i, k \in C$) and for reasons of symmetry, we can rewrite this in the following way:

$$\begin{aligned} a^H_{kl} &= a^h_{kl} + \sum_{i \in F} (w_{ik} a^h_{il} + w_{il} a^h_{ik}) + \sum_{i \in F} \sum_{j \in F} w_{ik} a^h_{ij} w_{jl} \\ &= a^h_{kl} + \sum_{i \in F} w_{ik} \left(a^h_{il} + \frac{1}{2} \sum_{j \in F} w_{jl} a^h_{ij} \right) + \sum_{i \in F} w_{il} \left(a^h_{ik} + \frac{1}{2} \sum_{j \in F} w_{jk} a^h_{ij} \right). \end{aligned}$$

Due to our assumption that $w_{ik} a^h_{ii} \le 2 |a^h_{ik}|$ ($i \in F$, $k \in C$), we have

$$a^h_{ik} + \frac{1}{2} \sum_{j \in F} w_{jk} a^h_{ij} \le a^h_{ik} + \frac{1}{2} w_{ik} a^h_{ii} \le 0 \qquad (i \in F, k \in C),$$

which implies $a^H_{kl} \le 0$ ($k, l \in C$, $k \ne l$). To show that A^H is weakly diagonally

dominant, we compute $\sum_{l\in C} a_{kl}^H$ by adding up the above equations for a_{kl}^H:

$$\sum_{l\in C} a_{kl}^H = \sigma_k^C + \sum_{i\in F} w_{ik}\Big(\sigma_i^C + \frac{1}{2}\sum_{j\in F} s_j a_{ij}^h\Big) + \sum_{i\in F} s_i\Big(a_{ik}^h + \frac{1}{2}\sum_{j\in F} w_{jk} a_{ij}^h\Big)$$

$$= \sigma_k + \sum_{i\in F} w_{ik}\Big(\sigma_i + \frac{1}{2}\sum_{j\in F}(s_j - 1)a_{ij}^h\Big) + \sum_{i\in F}(s_i - 1)\Big(a_{ik}^h + \frac{1}{2}\sum_{j\in F} w_{jk} a_{ij}^h\Big).$$

Here we used the abbreviations $\sigma_i = \sum_j a_{ij}^h$ and $\sigma_i^C := \sum_{l\in C} a_{il}^h$. Using (5.8), we obtain

$$\sigma_i + \frac{1}{2}\sum_{j\in F}(s_j - 1)a_{ij}^h \geq \sigma_i + \frac{1}{2}(s_i - 1)a_{ii}^h \geq 0,$$

which implies the diagonal dominance.

The situation is more involved if the interpolation strategy of § 4.5.4.2 is used. Instead of going into detail, we just mention that the coarsening process, if arranged as described in § 4.6, will then generate coarse grid matrices which are generally close to being weakly diagonally dominant M-matrices. In any case, as long as the remarks of the previous section apply, there will be no essential deterioration in practice.

4.5.5. Remarks on multi-level convergence. So far, we have given conditions which, e.g., in our model class of symmetric, weakly diagonally dominant M-matrices or certain perturbations thereof, guarantee uniform two-level convergence. We have also seen that, in these cases, a recursive application in a real multi-level cycle is formally possible in the sense that the important properties of the finest level operator carry over to the coarser level ones. In this section we now discuss the question of multi-level convergence.

We first note that the two-level theory does not directly carry over to a multi-level theory. That is, requiring (5.5) to hold on each of the intermediate levels with the same β is not sufficient to guarantee that the V-cycle convergence factor is independent of q, the number of levels. This can be seen from the following simple counterexample (cf. [103]).

Example 5.7. Let A^h be derived from discretizing $-u''$ on the unit interval with mesh size h, i.e., the rows of A^h correspond to the difference stencil $1/h^2[-1\ \ 2\ \ -1]_h$. (For our purpose we may ignore the boundary in the following.) One possibility for satisfying (5.5) with $\beta = 2$ is to coarsen the grid by doubling the mesh size, and to define *strictly one-sided* interpolation to each F-point (using only the respective neighboring C-point to the right, say, with the interpolation weight being 1). The corresponding coarse grid operator A^{2h} is then easily computed to correspond to the difference stencil $1/(2h)^2[-4\ \ 8\ \ -4]_{2h}$, which, after proper scaling of the

restriction operator by $\frac{1}{2}$, is seen to be off by a factor of 2 compared to the "natural" $2h$-discretization of $-u''$. Due to this, for a *very smooth* error frequency, $\sin(\pi x)$ say, we obtain $T^h e^h \approx \frac{1}{2} e^h$. Consequently, as smoothing hardly affects this frequency, we cannot expect the asymptotic two-level convergence factor to be better than $\frac{1}{2}$. If the same coarsening strategy is now used recursively to introduce coarser and coarser levels, the above arguments carry over to each of the intermediate levels. In particular, each coarser grid operator is off by a factor of 2 compared to the previous one. A simple recursive argument, applied to the same frequency as above, shows that errors are accumulated from grid to grid, and that the asymptotic (V-cycle) convergence factor cannot be expected to be better than $1 - 1/2^{q-1}$.

Clearly, one standard way to overcome q-dependent convergence factors is to use "better" cycles such as F- or W-cycles (cf. [101]). Apart from the fact that such cycles are more expensive (expense may be considerable in AMG, depending on the actual coarsening), they will have at best the same convergence factor as the corresponding two-level method. This method, in turn and for the same reasons that may lead to q-dependent convergence, may not be satisfactory. Therefore, a sufficient understanding of the reasons for q-dependent convergence is necessary.

As seen from the above example, a basic deficiency of the essential condition (5.5) is that it, by itself, does not guarantee a sufficiently good interpolation for smooth errors. A stronger condition that has been shown to be suitable for a V-cycle convergence theory is the approximation property given in § 4.3.2, namely

$$\|T^h e^h\|_1^2 \leq \beta \|e^h\|_2^2. \tag{5.12}$$

Its essential relation to condition (5.5) is seen from the following lemma.

LEMMA 5.8. *If* (5.12) *holds, then we also have*

$$\min_{e^H} \|e^h - I_H^h e^H\|_0^2 \leq \beta^2 \|e^h\|_2^2. \tag{5.13}$$

Proof. We first observe that (5.12) is equivalent to

$$\|T^h e^h\|_0^2 \leq \beta \|e^h\|_1^2, \tag{5.14}$$

which, omitting the index h, follows from

$$\sup_e \frac{\|Te\|_1^2}{\|e\|_2^2} = \rho(E^{-1/2} T E^{-1/2}) = \rho(TE^{-1}T) = \sup_e \frac{\|Te\|_0^2}{\|e\|_1^2},$$

with $E = D^{-1}A$. Applying (5.14) to $T^h e^h$ and using (5.12), we have

$$\|T^h e^h\|_0^2 \leq \beta \|T^h e^h\|_1^2 \leq \beta^2 \|e^h\|_2^2,$$

which, in particular, implies (5.13).

For smooth errors e^h, (5.13) is much stronger than (5.5) because $\|e^h\|_2 \ll \|e^h\|_1$. For instance, in the case of second order scalar PDE problems and smooth error frequencies on a regular grid of mesh size h, we typically have that $\|e^h\|_2 = O(h)\,\|e^h\|_1$. In such situations, (5.13) *increases the order of interpolation* compared to (5.5).

In summary, it becomes clear that, in order to obtain robust and efficient V-cycle convergence, we have to increase the accuracy of interpolation for smooth errors. Theoretically, we could achieve this by constructing interpolation that satisfies (5.13) rather than (5.5), a sufficient condition being that the inequality in (5.13) must hold for each e^h with the special choice $e_k^H = e_k^h$ $(k \in C)$:

$$\sum_{i \in F} a_{ii}^h \Big(e_i^h - \sum_{k \in C} w_{ik} e_k^h \Big)^2 \leq \beta^2 \sum_i a_{ii}^h \Big(\frac{r_i^h}{a_{ii}^h} \Big)^2,$$

where $r^h = A^h e^h$. Unfortunately, it is hardly possible to construct the interpolation so that we satisfy this inequality *exactly* by using *only algebraic information* (such as the matrix entries), unless we define it in the inefficient way described in Example 5.1.

In practice, however, it turns out that it is not really necessary to satisfy this inequality exactly. For geometrically posed problems, a sufficient improvement of the accuracy of interpolation is generally obtained if we add certain objectives to the interpolation requirements that result from § 4.5.4. These objectives (see criteria (C1) and (C2) in § 4.6.1), although partly motivated by geometric arguments, do not explicitly use geometric information. In particular, extremely one-sided interpolation like the one in Example 5.7 will be avoided.

Of course, if there is really no geometric background to a problem, or if the underlying structure of the connections is far from being local, there is no a priori guarantee of highest efficiency. However, a large number of practical tests, including complicated problems with random matrices and quite irregular finite element triangulations, show that obeying the objectives mentioned above results in a very robust and efficient method. For more information and, in particular, a collection of examples, see the following numerical sections.

For solving problems with underlying geometry, there are ways to improve interpolation further. One possibility is to supply AMG with some minimum geometric information, e.g., point locations. Along with other possibilities, this will also be discussed in the following section. All these "more sophisticated" techniques are not really needed in connection with the problems we had in mind up to now: the extra work does not pay. They become, however, quite important for certain "systems" problems.

4.6. AMG for scalar problems. AMG, as we stated previously, is designed to use the principles of geometric multigrid methods without

relying on the geometry of the problem being solved. In the developmental phase, design choices were often based on experiences with discretized second order elliptic PDEs, but an effort was made to keep the explicit dependence on the underlying geometry to a minimum. As a result, the method is applicable to a wide class of problems including, but not limited to, those involving symmetric M-matrices. The method developed for such problems, called the *scalar algorithm,* only makes explicit use of the information contained in the matrix itself and does not rely on the geometry of the problem.

The application of AMG to a matrix equation is a two-part process. The first part, called the *setup phase,* consists of choosing the coarser grids and defining the grid transfer and coarse grid operators. The second, called the *solution phase,* uses the components defined in order to perform standard multigrid cycling until a desired level of tolerance is reached.

The purpose of this section is to describe the setup phase in some detail, to give the work and storage requirements, and to give results for a number of sample problems. First, in § 4.6.1, an overview of the principles and the reasons for our choice of the algorithm will be given. Section 4.6.2 covers the technical details of the basic setup algorithm. This will be of interest primarily to the reader wishing to program AMG. Section 4.6.3 contains a discussion of the work and storage required for the AMG algorithm. Numerical results for a number of sample problems are given in § 4.6.4. Section 4.6.5 contains a simple method for computing a posteriori error estimates for problems involving symmetric M-matrices. Finally, § 4.6.6 outlines ways to improve the performance of the algorithm by, in some cases, explicitly using geometric information, which is usually readily available.

In the remainder of the paper, we assume that the reader is familiar with the notation and terminology introduced in §§ 4.2 and 4.5.1.

4.6.1. The setup phase: a general discussion. In any multigrid approach, relaxation and coarse grid correction are used in conjunction to eliminate the error. This requires that any error which cannot be corrected by appealing to a coarser grid problem (i.e., any error not approximately represented in the range of interpolation) must be effectively reduced by relaxation. In geometric multigrid methods, linear interpolation is normally used; then relaxation must be chosen that smoothes the error in the usual geometric sense. The approach in AMG is just the opposite. The relaxation method is fixed, and the coarse grid and interpolation operator are chosen so that the range of interpolation is forced to (approximately) contain those functions unaffected by relaxation. The other multigrid components needed are then defined as in § 4.2.2.

Consider the application of AMG to a problem $A\mathbf{u} = b$. An outline for the setup algorithm, provided $A^1 = A$, is as follows:

A1. *The setup phase:*
  Step 1. Set $m = 1$.
  Step 2. Choose the coarse grid Ω^{m+1} and define the interpolation operator I_{m+1}^m.
  Step 3. Set $I_m^{m+1} = (I_{m+1}^m)^T$ and $A^{m+1} = I_m^{m+1} A^m I_{m+1}^m$.
  Step 4. If Ω^{m+1} is small enough to allow for inexpensive inversion of A^{m+1}, set $q = m + 1$ and stop. Otherwise, set $m = m + 1$ and go to Step 2.

Step 2, called the *coarsening step,* is the only process not fully specified. In this section, we define the interpolation formula and determine the properties that the coarse grid should have. This will be done for any fixed level, denoted by the index h, since the method is the same for all levels. The corresponding coarser level to be constructed will be denoted by the index H. All sets of points used below (C, F, S, D and N) actually depend on h, but the index will be omitted for convenience.

Now, to determine what is required of the coarsening procedure, we consider the matrix equation

$$A^h \mathbf{u}^h = b^h, \tag{6.1}$$

where, for ease of motivation, we assume that A^h is a symmetric M-matrix. In this case, the theoretical results of the previous section apply. (The algorithm itself, however, does not require this to be true, and it can formally be applied in more general situations. Some examples of such situations are presented in § 4.6.4.) In particular, we have shown in § 4.4.3 that Gauss–Seidel relaxation converges slowly if and only if the (properly scaled) residuals are small compared to the error, e^h, between the current approximation and the exact solution. Thus, error that is slow to converge is characterized by

$$a_{ii}^h e_i^h \approx -\sum_{j \in N_i} a_{ij}^h e_j^h \qquad (i \in \Omega^h). \tag{6.2}$$

Such error is called *algebraically smooth.* Functions with this property should be well represented by the range of interpolation.

We assume that the interpolation operator is constructed as in § 4.5.1. That is, the grid Ω^h is partitioned into two sets, C and F, called C- and F-points, respectively. C is identified with the coarse grid Ω^H. Then, for any function e^H,

$$(I_H^h e^H)_i = \begin{cases} e_i^H & \text{if } i \in C, \\ \displaystyle\sum_{k \in C_i} w_{ik} e_k^H & \text{if } i \in F, \end{cases} \tag{6.3}$$

with some small sets of interpolation points $C_i \subset C$. Formula (6.2) is used to obtain the interpolation weights $w_{ik} = w_{ik}^h$ for the points in F. For instance, if C and F are chosen so that, for each $i \in F$, $N_i \subseteq C$ and we set $C_i = N_i$, then (6.2) can be used directly to define the weights (i.e., set $w_{ik} = -a_{ik}^h / a_{ii}^h$ for $i \in F$, $k \in C$). However, as noted in Example 5.1, this is seldom practical, and we would generally like to interpolate to $i \in F$ from as small a set C_i as possible. Then, in order to obtain an interpolation formula from (6.2), for each $j \in D_i = N_i - C_i$, e_j^h must be approximated in terms of e^h at points in C_i.

Formula (6.2) states that, at each point i, e_i^h is essentially determined by those e_j^h for which $-a_{ij}^h$ is large. For this reason, we introduce the following definition. A point i is *strongly connected to* j, or *strongly depends on* j, if $-a_{ij}^h \geq \theta \max_{l \neq i} \{-a_{il}^h\}$ with some $0 < \theta \leq 1$ (usually taken to be .25 in practice). With S_i denoting the set of all strong connections of point i, we will take $C_i = C \cap S_i$.[7] (Here, C is assumed to be given. Criteria for how to actually choose C will be derived below.) Then let $D_i^s = D_i \cap S_i$ and $D_i^w = D_i - S_i$. These are called the *strong* and *weak noninterpolatory connections*, respectively. Now, for points $j \in D_i^w$, we can replace e_j^h by e_i^h in (6.2). For points $j \in D_i^s$, however, this is generally not sufficient. Since the value of e_j^h is also determined by e^h at the points to which j is strongly connected, we make the approximation

$$e_j^h \approx \Big(\sum_{k \in C_i} a_{jk}^h e_k^h \Big) \Big/ \Big(\sum_{k \in C_i} a_{jk}^h \Big). \tag{6.4}$$

These approximations are substituted into (6.2), which is then solved for e_i^h, yielding the interpolation weights for point i. The more strongly connected j is to the points in C_i, the more accurate we can expect (6.4) to be (cf. Theorem 5.5). In practice, we only require at least one such strong connection. This requirement is considerably weaker than the corresponding one of Theorem 5.5, but it turned out to be sufficient in all our applications and gave highest efficiency in terms of overall work. Thus, the following criterion is suggested for choosing the sets C and F:

(C1) For each $i \in F$, each point $j \in S_i$ should either be in C, or should be strongly connected to at least one point in C_i ($= C \cap S_i$).

A primary concern in the coarsening process is to produce multigrid components that result in an efficient solution process. The efficiency is

[7] Note that if, in contrast to our assumption, A^h is not an M-matrix, but rather contains *few positive off-diagonal entries* per row, the corresponding connections are not defined to be strong. Correspondingly, an F-point i does not interpolate from any point j with $a_{ij}^h > 0$. Unless such coefficients become substantial, AMG performance will not seriously deteriorate.

determined by the convergence factors and the amount of work needed per cycle. As a general rule, the more points Ω^H has, the better h-level convergence can be. On the other hand, the amount of work necessary for relaxation on the H-level increases with the size of A^H, so it is advantageous to limit the number of C-points while still enforcing (C1). We do this by introducing the following additional criterion:

(C2) C should be a maximal subset of all points with the property that no two C-points are strongly connected to each other.

It is easy to construct examples for which it is impossible to strictly satisfy both (C1) and (C2). However, (C2) is used as a guideline for constructing C while ensuring that (C1) holds. The details of the C-point choice, and how it relates to these criteria, are discussed in the next section.

Remark 6.1. The coarse grids chosen according to (C1) and (C2) tend to produce an especially efficient solution process in problems with a geometric background, such as discretized PDEs. Since the work of relaxation, which dominates the cycling time, is proportional to the number of nonzero entries in the matrices, it is important to keep the "stencil size" as small as possible on all levels. This is generally achieved by the criteria used. Another consequence of the coarsening produced is that the grids chosen generally avoid "one-sided" interpolation, such as that discussed in Example 5.7. Lemma 5.8 essentially states that, in solving a regular, second order elliptic partial differential equation, second order interpolation is required for proper V-cycle convergence. Although this cannot be guaranteed without the explicit use of geometric information, grids chosen according to (C1) and (C2) typically give balanced sets of interpolation points, allowing for interpolation that is sufficiently accurate for the problems tested.

4.6.2. Technical details of the coarsening process. This section is not essential for the remainder of the paper and can be skipped by the casual reader. Its purpose is to describe the coarsening process (Step 2) of the setup phase A1 in the previous section in detail. In addition to the notation used there, we define the set of points strongly connected *to* i by $S_i^T = \{j : i \in S_j\}$. Furthermore, for a set P, $|P|$ denotes the number of elements in P.

In order to keep the work involved in the choice of the coarse grid small, a two-part process is used. First, a relatively quick C-point choice is performed; this attempts to enforce the criterion (C2) by distributing C-points uniformly over the grid. In the second part, which is combined with the computation of interpolation weights, the tentative F-points resulting from the first part are tested to ensure that (C1) holds, with new C-points added as necessary. The first part is given below.

A2. *Preliminary C-point choice:*

Step 1. Set $C = \emptyset$, $F = \emptyset$, $U = \Omega^h$, and $\lambda_i = |S_i^T|$ for all i.

Step 2. Pick an $i \in U$ with maximal λ_i. Set $C = C \cup \{i\}$ and $U = U - \{i\}$.

Step 3. For all $j \in S_i^T \cap U$, perform Steps 4 and 5.

Step 4. Set $F = F \cup \{j\}$ and $U = U - \{j\}$.

Step 5. For all $l \in S_j \cap U$, set $\lambda_l = \lambda_l + 1$.

Step 6. For all $j \in S_i \cap U$ set $\lambda_j = \lambda_j - 1$.

Step 7. If $U = \emptyset$, stop. Otherwise, go to Step 2.

Roughly speaking, the tendency is to "build" the grid starting from one point and continuing outwards until the domain is covered. In Step 2, λ_i, which can be written as $|S_i^T \cap U| + 2|S_i^T \cap F|$, measures the value of the point $i \in U$ as a C-point given the current status of C and F, since a point that can be used for interpolation to a large number of F-points should help to satisfy (C1) with smaller C. Initially, points with many others strongly connected to them become C-points, while later the tendency is to pick as C-points those on which many F-points depend. This tends to produce a grid with a maximal number of C-points subject to the restriction that there are few, if any, strong C-C connections. In addition, this guarantees that each F-point produced has at least one strong connection to a C-point.

The algorithm for performing the final C-point choice and defining the interpolation weights is as follows:

A3. *Final C-point choice and definition of interpolation weights:*

Step 1. Set $T = \emptyset$.

Step 2. If $T \supseteq F$, stop. Otherwise, pick $i \in F - T$ and $T = T \cup \{i\}$.

Step 3. Set $C_i = S_i \cap C$, $D_i^s = S_i - C_i$, $D_i^w = N_i - S_i$ and $\tilde{C}_i = \emptyset$.

Step 4. Set $d_i = a_{ii}^h + \sum_{j \in D_i^w} a_{ij}^h$ and for $k \in C_i$, set $d_k = a_{ik}^h$.

Step 5. For each $j \in D_i^s$ do Steps 6 through 8.

Step 6. If $S_j \cap C_i \neq \emptyset$, then go to Step 8.

Step 7. If $\tilde{C}_i \neq \emptyset$, set $C = C \cup \{i\}$, $F = F - \{i\}$, and go to Step 2. Otherwise, set $\tilde{C}_i = \{j\}$, $C_i = C_i \cup \{j\}$, $D_i^s = D_i^s - \{j\}$, and go to Step 4.

Step 8. Set $d_k = d_k + a_{ij}^h a_{jk}^h / \sum_{l \in C_i} a_{jl}^h$ for $k \in C_i$.

Step 9. Set $C = C \cup \tilde{C}_i$, $F = F - \tilde{C}_i$, and $w_{ik} = -d_k/d_i$ for each $k \in C_i$, and go to Step 2.

The main idea of this algorithm is to sequentially test each F-point i in order to ensure that each point in D_i^s has a strong connection to at least one point in C_i. The set of F-points tested is denoted by T, and when all F-points have been tested, the algorithm terminates. When an F-point i is found to have a noninterpolatory strong connection $j \in D_i^s$ that does *not* strongly depend on C_i, then j is tentatively made into a C-point (it is put in

the sets $\tilde{C}_i$ and C_i), and testing of the points in D_i^s (which no longer contains j) starts again. If all those points now depend strongly on C_i, then interpolation is defined for i (Step 8) and j is put into C (Step 9). However, if the algorithm finds another point in D_i^s that does not depend strongly on C_i, then the point i itself is put into C. This is done in an effort to minimize the number of additional C-points introduced. From this viewpoint, it is better to make i itself a C-point than to force more than one of its strong connections to be C-points.

In practice, this two-part coarsening process performs well. The preliminary choice is fast, and the number of C-points added in the final choice is normally small.

4.6.3. Work and storage requirements. It is important to have some idea of the work (in terms of floating point operations) and the storage necessary to solve a given problem with AMG. Since the process is fully automatic, this cannot be predicted exactly. However, the total work necessary for both setup and cycling can be estimated in terms of several basic quantities. Ideally, this work should be $O(n)$, and the constant should depend only on the nature of the problem, not the size.

The quantities used in the estimates are: κ^A, the average "stencil size" over all grids; κ^I, the average number of interpolation points per F-point, σ^Ω, the ratio of the total number of points on all grids to that on the fine grid (called the *grid complexity*); and σ^A, the ratio of the total number of nonzero entries in all the matrices to that in the fine grid matrix (called the *operator complexity*). The goal of the AMG setup phase is to keep these values as small as possible in order to ensure efficient cycling. Typical values are given in § 4.6.4 for a number of problems.

In usual geometric multigrid, these quantities are known in advance. The grids are generally coarsened by a factor of 2 in all directions, so the grid complexity σ^Ω is $1/(1-2^{-d})$, where d is the dimension of the domain. If the same type of approximation to the continuous operator is used on all grids, κ^A is equal to the stencil size of the fine grid problem, and the operator complexity σ^A is also $1/(1-2^{-d})$. For linear interpolation, we have $\kappa^I = d/(1-2^{-d})$. Thus, for example, a PDE defined on a region of the plane discretized with a 5-point stencil will give the values $\sigma^A = \sigma^\Omega = 4/3$, $\kappa^A = 5$, and $\kappa^I = 8/3$.

Numerical work in the setup phase. The floating point arithmetic needed for the setup phase represents only a small amount of the total time required, since the dominant part involves sorting, keeping counters, looping, maintaining linked lists, dynamic storage manipulation, garbage collection, etc. For this reason, we do not go into detail, but only present the final estimates for the total number of floating point operations needed to define the interpolation weights and the coarse grid operators. These are

denoted by ω^I and ω^A, respectively, and are as follows[8]:

$$\omega^I = n\kappa^I(3(\kappa^A - \kappa^I) - 2), \qquad \omega^A = n\kappa^I(2\kappa^I(\kappa^A - \kappa^I) + 3\kappa^I + \kappa^A). \tag{6.5}$$

Numerical work per cycle. The approximate numbers of floating point operations on level m required for one relaxation sweep, residual transfer, and interpolation are

$$2n_m^A, \quad 2n_m^A + 2\kappa^I n_m^F, \quad n_m^C + 2\kappa^I n_m^F,$$

respectively. Here, n_m^A is the number of nonzero entries of A^m, and n_m^C and n_m^F denote the numbers of C- and F-points on Ω^m. The actual work per cycle, of course, depends on its type. The most common one is the (ν_1, ν_2) V-cycle, in which ν_1 and ν_2 are the numbers of relaxation sweeps performed before and after each coarse grid correction. Then, for example, the total amount of work per V-cycle, if we let $\nu = \nu_1 + \nu_2$ (and note that $\sum_m n_m^F \approx n$), is approximately

$$n(2(\nu + 1)\kappa^A \sigma^\Omega + 4\kappa^I + \sigma^\Omega - 1). \tag{6.6}$$

Remark 6.2. The number of operations required for relaxation dominates the work per V-cycle and is proportional to the number of nonzero entries in all the matrices. Thus, the operator complexity, σ^A, which does not occur in the above estimate, is actually a convenient approximation of the ratio of the total work per V-cycle to the relaxation work on the finest grid.

Remark 6.3. Some cycling schemes used in usual multigrid methods may not be appropriate at times for AMG, because the reduction of the problem size from one level to the next is not fast enough. For example, for a 2-D anisotropic problem, this reduction is given by a factor of around .5 (cf. Table 4.2). In this case, the total work per point in a W-cycle is known to depend on the size of the fine grid problem (cf. [538, § 4.4]).

Storage requirements. The most convenient storage scheme for AMG is that used in the Yale Sparse Matrix Package (cf. [Ei1]). This form allows the use of general matrices and does not require storage of the zero entries. Three one-dimensional arrays are used to specify a matrix: A, IA, and JA. We assume that the rows of the matrix are ordered so that the ith equation corresponds to the ith variable. The diagonal entry a_{ii} is stored in the real vector A at the position indicated by the ith entry of IA, with the nonzero off-diagonals following in any order. The rows are stored consecutively. The vector JA, which has the same length as A, contains the column number of the corresponding entries of A. The interpolation operators are stored in a

[8] This estimate for ω^I assumes that *all* connections are strong, which is the worst case. In addition, there has been no effort to take advantage of the symmetry in defining the coarse grid operator. An algorithm so designed could cut the work for the operator construction by close to a factor of 2.

similar form. The restriction operator does not need to be stored, since it is simply the transpose of interpolation. In addition to the matrices and the interpolation operators, we need storage for the solution approximation and the right-hand side, and pointers giving the correspondence between points on different grids. We need a total of $(\kappa^A + 2)\sigma^\Omega + \kappa^I$ real and $(\kappa^A + 3)\sigma^\Omega + \kappa^I + 1$ integer storage locations per point on the finest grid.[9]

Remark 6.4. The values of σ^A, σ^Ω, κ^A, and κ^I for geometric multigrid applied to a given problem can generally be regarded as "best-case" values for AMG applied to the same problem. Consequently, using the values given before in the above formulas, we can expect the solution of a PDE discretized on a uniform grid with a 5-point stencil AMG to require at least 12 real storage locations and 14 integer locations per point on the finest grid. In practice, values may be higher, but will seldom exceed about 19 real and 22 integer locations.

Remark 6.5. During the setup phase some work space is necessary. However, at any stage of the process, there is available storage, which will be needed for matrices and pointers for levels not yet determined; this usually suffices.

4.6.4. Numerical results. In this section we present results of experiments with AMG on a number of discretized partial differential equations. They demonstrate the robustness and efficiency of the method in a variety of cases, including irregular domains and discretizations, anisotropic and discontinuous coefficients and nonsymmetric operators.

In all cases, the parameter θ defining strong connections and described in § 4.6.2 was taken as 0.25. (1, 1) V-cycles with Gauss–Seidel relaxation and C/F-ordering of points were used in all tests, and the convergence factors were computed by a von Mises vector iteration. The values given in Tables 4.1–4.7 are as follows (cf. § 4.6.3).[10]

ρ	Asymptotic V-cycle convergence factor,
t_V	Time required for one V-cycle,
t_{prep}	Time required for the setup phase,
σ^A	Operator complexity,
σ^Ω	Grid complexity,
κ^A	Average stencil size,
κ^I	Average number of interpolation points per F-point.

[9] Again, there can be some savings if we take advantage of symmetry. The setup phase is most efficient when both the upper and lower off-diagonal nonzero matrix entries are stored, but it is possible to keep all nonzeros stored only for the current working level, and to discard the lower (or upper) off-diagonal elements before proceeding.

[10] All the time measurements reported in this section were obtained on an IBM 3083 computer at the Gesellschaft für Mathematik und Datenverarbeitung, St. Augustin, West Germany.

The Laplace operator. First, in order to establish the behavior of the method in the absence of any irregularities, results are given for several different finite difference discretizations of the Laplace operator on the unit square with Dirichlet boundary conditions. In all cases, a uniform square grid with mesh size $h = 1/64$ is used. The stencils used are as follows:

$$(1)\ \frac{1}{h^2}\begin{bmatrix} & -1 & \\ -1 & 4 & -1 \\ & -1 & \end{bmatrix}, \qquad (2)\ \frac{1}{2h^2}\begin{bmatrix} -1 & & -1 \\ & 4 & \\ -1 & & -1 \end{bmatrix},$$

$$(3)\ \frac{1}{8h^2}\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}, \qquad (4)\ \frac{1}{20h^2}\begin{bmatrix} -1 & -4 & -1 \\ -4 & 20 & -4 \\ -1 & -4 & -1 \end{bmatrix}.$$

Table 4.1 contains the results for these problems, which are typical for isotropic operators. There are some important points to note here.

First, although the 9-point stencils require more time for relaxation on the finest grid than the 5-point stencils, the overall times per cycle are somewhat less. The reason is that the complexity values and average numbers of interpolation points are lower for the 9-point stencil. Quantitatively, this can be seen when the complexities and average values shown in Table 4.1 are inserted in (6.6) to approximate the number of floating point operations. Both Problem 1, with a 5-point stencil, and Problem 3, with a 9-point stencil, require approximately 79 operations per cycle and per point on the finest grid. It is generally true that larger stencils result in faster coarsening, lowering the complexity, since (C1) can be satisfied with a smaller percentage of the strong connections of each F-point in C. For example, for the 5-point stencil, *all* the neighbors of each F-point must be in C, while for the 9-point stencil, as few as 2 can suffice.

Remark 6.6. For comparison, a standard geometric multigrid cycle for isotropic problems, using full weighting for the residual transfer, would require approximately 48 operations per fine grid point for a (general) 5-point stencil and 80 for a 9-point stencil. Thus, in these examples, the

TABLE 4.1

Problem	ρ	Times		Complexity		Averages	
		t_V	t_{prep}	σ^A	σ^Ω	κ^A	κ^I
1	0.054	0.29	1.63	2.21	1.69	6.46	3.29
2	0.067	0.27	1.52	2.12	1.64	6.28	3.12
3	0.078	0.26	1.83	1.30	1.31	8.73	2.51
4	0.109	0.26	1.83	1.30	1.31	8.73	2.51

work per AMG cycle is comparable for 5-point stencils and for 9-point stencils it is nearly optimal.

Second, AMG works well when applied to the skewed Laplacian, while geometric multigrid methods do not. This is because the grid actually consists of two sets of points with no connections between these sets. We cannot define any relaxation process that results in smoothing of the error between the two sets, so usual geometric coarsening and linear interpolation cannot give meaningful coarse grid corrections. AMG, on the other hand, automatically treats the problem as two decoupled problems, since the coarsening process is based on interpolation along strong connections.

Remark 6.7. The average ratio of setup time to V-cycle time for the above problems is about 6.3, while that of the corresponding numbers of floating point operations (computed from (6.5), (6.6)) is around 2. This clearly shows that real arithmetic actually takes up a relatively small amount of the total setup time.

Anisotropic operators and discontinuous diffusion coefficients. Although somewhat artificial, the example of the skewed Laplace operator demonstrates the ability of AMG to tailor coarsening to the problem at hand. It is easy to see from the algorithm described in § 4.6.2 that coarsening must occur in the direction of strong connections, so that, for example, 2-D problems that are strongly anisotropic in a direction aligned with the grid are actually treated as a number of separate 1-D problems.

Here, we report on experiments with three examples of operators involving "problem" coefficients. As before, the domain for each is the unit square, Dirichlet boundary conditions are imposed, and each is discretized on a uniform grid with $h = 1/64$, using 5-point stencils. The first problem is

$$-\varepsilon u_{xx} - u_{yy} = f,$$

and several choices for ε are used. In geometric multigrid, a line relaxation must be used in order to ensure smoothing when standard coarsening is used, with the direction of the relaxation depending on the size of ε. An alternative is to coarsen only in one direction (and to use point relaxation).

The second problem, more complex than the previous one, is

$$-(100^{x+y-1}u_x)_x - u_{yy} = f.$$

Here, the direction and strength of the anisotropy varies over the domain. In order to solve this problem by usual multigrid methods, alternating direction line relaxation is necessary. In this case coarsening in one direction is not useful, since the direction would have to change over the domain. This is not a problem in AMG, since the type of coarsening in each part of the domain is automatically adapted to the direction of strong connections there. Finally, we present a problem that we cannot solve with geometric multigrid methods simply by choosing the proper relaxation scheme. This is

a diffusion problem with a discontinuous diffusion coefficient, and is as follows:

$$-\nabla(d(x, y)\nabla u) = f \text{ with } d(x, y) = \begin{cases} 1 & 0.0 \le x \le 0.5 \quad 0.0 \le y \le 0.5, \\ 10 & 0.0 \le x \le 0.5 \quad 0.5 < y \le 1.0, \\ 100 & 0.5 < x \le 1.0 \quad 0.5 < y \le 1.0, \\ 1000 & 0.5 < x \le 1.0 \quad 0.0 \le y \le 0.5. \end{cases}$$

In the application of usual multigrid to such problems, error is not smoothed across the discontinuities and interpolation must be adapted at interfaces (cf. [5]).

The results for the application of AMG to these problems are given in Table 4.2. Note that the convergence factors do not vary much, and are comparable to those obtained for the Laplace operator, even when the direction of strong connections changes or the coefficients are discontinuous.

For Problem 1, there are two points to note. First, the grid complexity tends to 2 as ε becomes very small or very large. This is because coarsening is basically in one direction only for such problems, so about half the grid points are chosen as C-points. On coarser grids, the strength of connections in all directions becomes more uniform, and faster coarsening, such as that obtained for the 9-point stencils, results, thereby decreasing the complexity. For problems with very small or large ε, however, this only occurs on very coarse grids, if at all, and the grid complexity stays around 2.

Second, the operator complexity, and thus the time per cycle, becomes quite high for $\varepsilon = .1$ or 10. This is mainly due to the coarsening obtained

TABLE 4.2

ε	Differential operator	ρ	Times		Complexity		Averages	
			t_V	t_{prep}	σ^A	σ^Ω	κ^A	κ^I
0.001	$-\varepsilon u_{xx} - u_{yy}$	0.082	0.31	1.05	2.54	1.92	6.55	1.84
0.01		0.094	0.32	1.20	2.72	1.93	6.96	2.03
0.1		0.063	0.37	1.82	3.33	1.87	8.80	2.64
0.5		0.071	0.28	1.62	2.19	1.68	6.44	3.27
1		0.054	0.29	1.63	2.21	1.69	6.46	3.29
2		0.059	0.28	1.61	2.19	1.68	6.44	3.28
10		0.079	0.36	1.84	3.37	1.87	8.90	2.62
100		0.095	0.31	1.18	2.68	1.92	6.90	2.00
1000		0.083	0.30	1.07	2.54	1.92	6.55	1.84
	$-(100^{x+y-1}u_x)_x - u_{yy}$	0.089	0.30	1.68	2.35	1.72	6.76	3.31
	$-\nabla(d(x, y)\nabla u)$	0.082	0.30	1.45	2.45	1.79	6.75	2.72

from grid 2 to grid 3, and is a result of the choice of θ in the definition of strong connections. For this problem, the complexity can be reduced to a more reasonable level by a different choice of the parameter θ.

The solution of the above problems requires an average of 89 operations per fine grid point per V-cycle. For comparison, a robust geometric multigrid algorithm capable of solving these problems without changes (with the exception of Problem 3), using a V-cycle, standard coarsening, full weighting, and two alternating line relaxation sweeps per level (assuming that the decompositions have been previously computed and stored), requires 70 operations per fine grid point per cycle.

Problem grids and domains. Irregular domains and grids can cause problems for usual geometric multigrid methods in several ways. The domain may contain features that cannot be resolved easily on coarse grids, so that the operators defined cannot provide good coarse grid corrections. Irregular grids are used for a variety of reasons, often in combination with finite elements. In such cases, it may not be possible to find a natural sequence of coarser grids, and it may be inconvenient (or impossible) to define a finest grid that does admit a natural coarsening. Here we give results for finite element discretizations of the Laplace operator on several such domains. These are shown in Figs. 4.1–4.4. (Some of the figures show grids coarser than the ones actually used so that the grid details are not obscured.) Each of these problems presents special difficulties for usual multigrid methods.

Problem 1. Figure 4.1 shows a regular discretization of a complex domain. Small holes in the region with a size equal to the mesh size, and

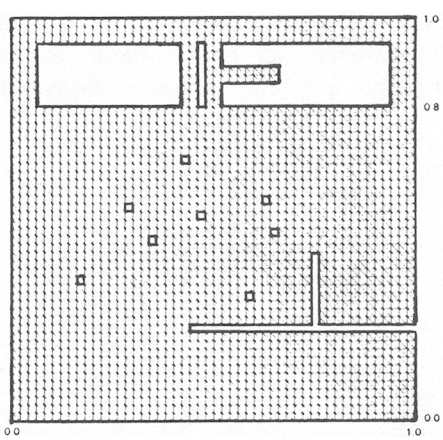


FIG. 4.1

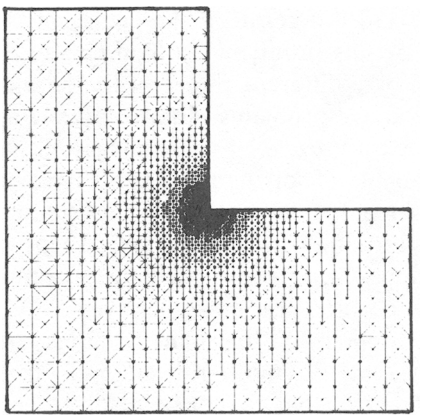


FIG. 4.2

detailed structures, make usual coarsening impossible. The re-entrant corners also present singularities.

Problem 2. The domain and grid in Fig. 4.2 cause several problems. First, there is a singularity due to the re-entrant corner. Also the grid near the corner is locally refined, resulting in irregular stencils at a number of points. The difference in mesh sizes over the region can cause problems for other iterative methods. In usual multigrid methods, the approach to local refinement is to define a sequence of successively finer uniform grids, each covering a smaller area. With the grid given, though, choosing the coarser grids will be difficult.

Problem 3. The triangle in Fig. 4.3 is discretized on a "random" grid, which would cause many problems in the geometric approach. This example actually does have "natural" coarser grids, but determining these grids can be impractical. AMG, however, uses no knowledge of the grid structure, so that finding a natural coarsening, if one even exists, is not necessary.

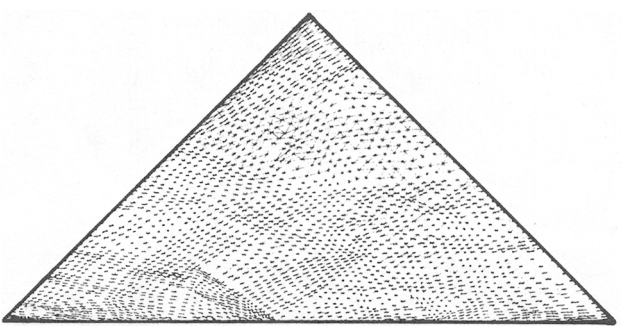


FIG. 4.3

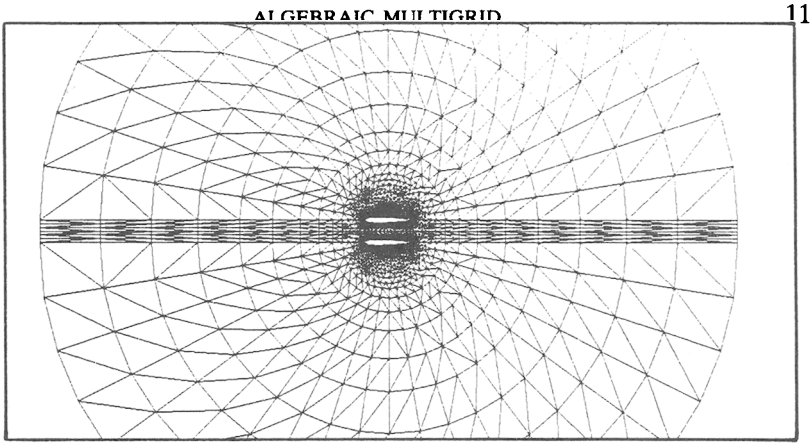


FIG. 4.4(a)

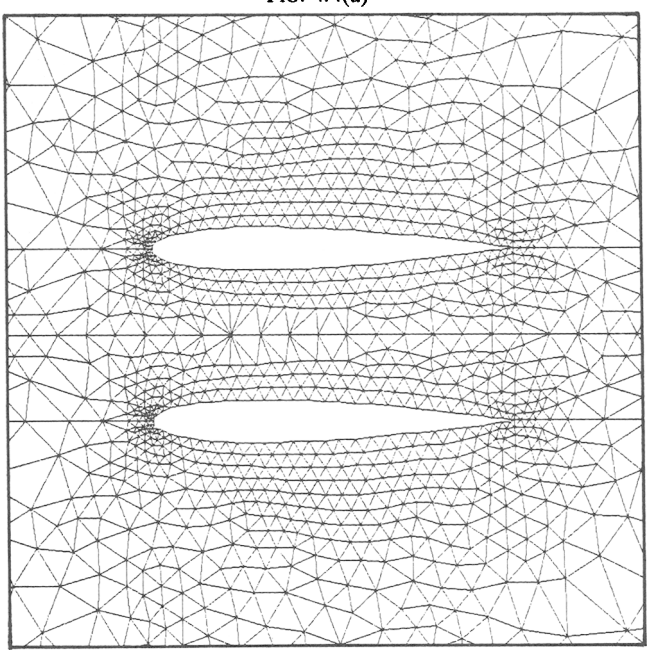


FIG. 4.4(b)

Problems 4 and 5. Finally, Figs. 4.4(a) and (b) show a grid generated at INRIA to model flow around two NACA012 airfoils (representing a cross section of an air intake). The outer boundary is a circle, and the figures show the details of the grid around the airfoils. This example incorporates

many of the difficulties of the previous problems. The grid is refined close to the airfoils, the trailing edges cause singularities, and there is no natural coarsening. For Problem 5, another similar domain not shown here, is used to model flow around a single NACA012 airfoil. Neumann boundary conditions are imposed on the surface of the airfoils, and Dirichlet conditions are used at the outer boundary (to specify the flow at "infinity").

The results of the application of AMG to each of these problems is given in Table 4.3. The number of points on the finest grid is given by n. The increase of the convergence factors over those for the finite difference discretizations is small, in spite of the difficulties in the problems. It should be noted that the matrices obtained for Problems 3–5 are not M-matrices (off-diagonals of both signs result), so the assumptions made to motivate the coarsening process do not have to hold strictly.

Nonsymmetric operators. Although AMG was developed primarily for symmetric problems, some nonsymmetric problems can be handled as well with no modifications to the algorithm. We show this here using the convection-diffusion operator

$$-\varepsilon\Delta u + a(x, y)u_x + b(x, y)u_y = f(x, y).$$

Again, the domain is the unit square and Dirichlet boundary conditions are used. The problem is discretized on a uniform grid with $h = 1/64$. The discrete operator is the 5-point finite difference approximation of the form

$$\frac{1}{h^2}\begin{bmatrix} & -\varepsilon + bh\mu_y & \\ -\varepsilon + ah(\mu_x - 1) & -\Sigma & -\varepsilon + ah\mu_x \\ & -\varepsilon + bh(\mu_y - 1) & \end{bmatrix},$$

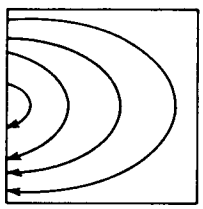
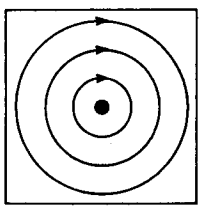
where

$$\mu_x = \begin{cases} \varepsilon/2ah & \text{if } ah > \varepsilon, \\ 1 + \varepsilon/2ah & \text{if } ah < -\varepsilon, \\ \frac{1}{2} & \text{if } |ah| \le \varepsilon, \end{cases} \quad \text{and} \quad \mu_y = \begin{cases} \varepsilon/2bh & \text{if } bh > \varepsilon, \\ 1 + \varepsilon/2bh & \text{if } bh < -\varepsilon, \\ \frac{1}{2} & \text{if } |bh| \le \varepsilon. \end{cases}$$

TABLE 4.3

Problem	n	ρ	Times t_V	Times t_{prep}	Complexity σ^A	Complexity σ^Ω
1	4064	0.089	0.29	2.22	1.93	1.71
2	2232	0.124	0.17	1.10	2.48	1.68
3	1953	0.153	0.20	1.26	2.84	1.87
4	1650	0.229	0.16	0.83	2.54	1.73
5	960	0.153	0.09	0.42	2.50	1.82

TABLE 4.4

Problem		ε	ρ	Times t_V	Times t_{prep}	Complexity σ^A	Complexity σ^Q
1	$l=0$	10^{-5}	0.00005	0.29	0.78	2.30	1.97
	$l=1$	10^{-5}	0.005	0.45	1.67	4.45	2.19
	$l=2$	10^{-5}	0.0004	0.46	2.01	4.63	2.07
	$l=3$	10^{-5}	0.002	0.41	1.57	3.81	2.05
	$l=4$	10^{-5}	0.00005	0.29	0.78	2.30	1.97
2		0.1	0.060	0.29	1.62	2.21	1.69
		0.001	0.055	0.40	1.49	3.68	2.10
		0.00001	0.030	0.39	1.41	3.57	2.10
3		0.1	0.056	0.29	1.63	2.21	1.68
		0.001	0.160	0.40	1.51	3.76	2.10
		0.00001	0.173	0.40	1.46	3.72	2.13

Σ denotes the sum of the surrounding coefficients. Simple central differencing cannot be used when ah or bh become large compared to ε, since the operator is no longer stable.

Table 4.4 contains the convergence factors for AMG applied to several different choices of a, b, ε. The functions a and b used are as follows:

(1) $\quad a(x, y) \equiv \sin l\frac{\pi}{8}, \quad b(x, y) \equiv \cos l\frac{\pi}{8}, \quad (l = 0, 1, 2, 3, 4),$

(2) $\quad a(x, y) = (2y - 1)(1 - x^2), \qquad b(x, y) = 2xy(y - 1),$

(3) $\quad a(x, y) = 4x(x - 1)(1 - 2y), \qquad b(x, y) = -4y(y - 1)(1 - 2x).$

In Problem 1, $\varepsilon = 10^{-5}$ was used. The characteristic directions given in Problems 2 and 3 are shown in Table 4.4, and the values of ε used are .1, .001, and .00001.

For Problem 1, where the characteristic directions are straight lines, convergence is extremely fast. This is especially true when the characteris-

tics are aligned with the grid lines. In this case, if ε were 0, AMG would act as a direct solver, giving the exact solution in one cycle. For small ε, the convergence factors actually decrease to zero with cycling, in contrast to the behavior for symmetric problems, where the factors generally increase to some asymptotic value. For this reason, the convergence factors given are geometric averages over a number of cycles.

For Problems 2 and 3, convergence factors are similar to those obtained for the symmetric problems studied when ε is large. As ε is decreased, Problem 2 behaves in a manner similar to Problem 1, with decreasing convergence factors. However, the factors increase in Problem 3. This may be due to the fact that as $\varepsilon \to 0$ the problem becomes more and more singular. The continuous problem for $\varepsilon = 0$ is not well defined, since the null space includes functions with arbitrary constant values on any characteristic circle that does not intersect the boundary.

Remark 6.8. The behavior of AMG as a direct solver for some problems with $\varepsilon = 0$ is a consequence of the coarsening. Since strong connections are in the upstream direction, for any point i, there are generally few, if any, strong connections between points in S_i. This results in choosing $C_i = S_i$ for all $i \in F$. The C/F-relaxation then produces error that lies exactly in the range of interpolation, and the coarse grid correction is exact. Thus, the reason for convergence is purely algebraic, and not a consequence of smoothing in the usual sense. When the characteristics change direction over the region, as in Problems 2 and 3, the directionality of the strong connections on coarser grids is lost, and AMG can no longer function as a direct solver.

Remark 6.9. In all cases, except in Problem 1 for $l = 0$ and $l = 4$, when ε is small, complexity is higher than for symmetric operators. This is because the algorithm produces coarse grid lines that are roughly perpendicular to the characteristic directions, and the number of grid points is reduced by around a factor of 2. Unless the flow is actually aligned with the grid lines, however, the strong connections of each point tend to "fan out" so that each point is strongly connected to an increasing number of points on the upstream grid line.

4.6.5. A posteriori error estimates. In this section we briefly recall a possibility of obtaining realistic a posteriori error bounds if A in (2.1) is an M-matrix. The idea behind the procedure described below is well known and relies on monotonicity principles (cf. [Sc1]). We first state the following lemma.

LEMMA 6.10. *For every M-matrix A,*

(i) $Au \geq 0 \Rightarrow u \geq 0$, *and*

(ii) $Au > 0 \Rightarrow u > 0$.

Let u now be any approximate solution of $A\mathbf{u} = b$ with corresponding error $e = \mathbf{u} - u$ and residual $r = b - Au = Ae$. Furthermore, suppose any z with $Az > 0$ to be given. Then an immediate consequence of the above lemma is that

$$\alpha Az \le r \le \beta Az \Rightarrow \alpha z \le e \le \beta z.$$

Thus, to obtain lower and upper bounds for the error e, we only have to find a vector z with $Az > 0$ and compute α and β. This can be done as sketched in the following procedure.

(1) Perform one AMG cycle to get an approximate solution z of the linear system $A\mathbf{u} = 1$ starting with the first approximation $z^{(0)} = 1$, say.

(2) Check whether $Az > 0$. If this is the case, go to (3). Otherwise improve z by performing another cycle and repeat the check. Usually one or two cycles will be enough to obtain $Az > 0$.

(3) The error estimate $\alpha z \le e \le \beta z$ holds with $\alpha = \min_i \{r_i/(Az)_i\}$ and $\beta = \max_i \{r_i/(Az)_i\}$.

The above procedure costs only little extra work and usually will give quite realistic error estimates.

4.6.6. Variations of the coarsening strategy. The coarsening procedure presented in § 4.6.2 gives satisfactory results for a wide variety of problems. However, for some cases we have not yet considered here, coarsening may not be satisfactory, either because the coarse grids are too fine, resulting in high complexity and increased work in cycling; or because the error after relaxation may not be approximated well enough by the range of interpolation, which results in slow convergence per cycle. For example, the application of AMG to finite difference equations in 3-D often results in high operator complexity. Also, some of the problems of § 4.7 require a more careful determination of the interpolation weights in order to maintain efficiency.

In the following, we discuss several modifications to the coarsening process given in § 4.6.2 which can be used to improve efficiency in such situations. These can be split into two groups: modifications in the coarse grid choice, and changes in the definition of the interpolation operator. Often, these are used in conjunction in an effort to get both faster coarsening and improved interpolation accuracy.

Although the trend in AMG has been to set up the necessary multigrid components with a minimal amount of user-supplied information, many problems do have an underlying geometric structure (although it may be quite irregular), and this information can often be used to improve performance. In particular, the coordinates of the points at which the variables are defined are often readily available, since they are frequently used in the generation of the problem itself. The use of this geometric

information, along with that contained in the matrix, provides the basis for some of the approaches given below.

4.6.6.1. Modifications to the C-point choice.

Long-range interpolation. Long-range interpolation is simply the use, for interpolation to an F-point i, of points not necessarily in S_i, the set of points to which i is *directly* strongly connected. The primary reason for considering this generalization is to reduce the operator complexity. It is convenient from a programming standpoint to require that $C_i \subset S_i$, but this can be too restrictive in terms of coarsening. This is particularly true in the case of finite difference equations in 3-D. For example, the use of 7-point stencils allows only coarsening by a factor of 2, but the stencil size grows to about 19 points on the second grid. Coarsening becomes more efficient afterwards, but the overall operator complexity is quite high. In such problems it is possible to coarsen faster while not seriously hurting the quality of the correction.

Clearly, in enlarging the set of possible interpolation points, we cannot disregard the concept of strong connections without affecting the ability of interpolation to approximate algebraically smooth functions. For this reason, we introduce the idea of *long-range strong connections*. We say that i is strongly connected to j along a path of length l if there exists a sequence of points $i_0, i_1, i_2, \cdots, i_l$ with $i = i_0$ and $j = i_l$ such that $i_k \in S_{i_{k-1}}$ for $k = 1, 2, \cdots, l$. Let S_i^l denote the set of all points to which i is strongly connected along a path of length less than or equal to l.

The algorithm can be generalized to allow interpolation from points in S_i^l for some specified l. Generally, not all points in S_i^l are suitable for interpolation. We can see this easily by considering simple geometric cases: in geometric multigrid methods, if the mesh size is increased by more than a factor of 2 from one grid to the next, the work required for relaxation increases, and overall efficiency decreases even though less coarse grid work is required. For this reason, we may take only the "strongest" connections in S_i^l, where the strength of the connection from i to some point $j \in S_i^l$ is measured, for example, in terms of the number of paths of length less than or equal to l from i to j.

Once this restricted set of points, denoted by $\tilde{S}_i^l$, is determined, the algorithm can proceed as in § 4.6.2, with some minor changes, if we choose C and F so that, for each $i \in F$, each point $j \in S_i$ is either in C or is *directly* strongly connected to some point in $C \cap \tilde{S}_i^l$, which we take as C_i. In this case, interpolation can be defined as before. If this is not the case, one of the methods discussed in § 4.6.6.2 may be used.

Convex interpolation. As stated in § 4.6.1, in problems with an underlying geometric structure, the coarsening process tends to produce grids in which each F-point is "surrounded" by its interpolation points. This situation, where each F-point lies (at least approximately) in the convex hull

of its interpolation points, is referred to as *convex interpolation.* Such an arrangement allows for $O(h^2)$ interpolation and is one of the main reasons for the efficiency of AMG. The coarsening algorithm of § 4.6.2, however, does not guarantee convex interpolation where strong connections are naturally one-sided, such as along boundaries. This does not essentially affect the behavior of AMG for the problems we have considered so far, but it must be corrected for some of the applications given in § 4.7. In those cases, it is sometimes useful to choose C and F so that we have convex interpolation even at the boundaries (by explicitly using the coordinates of the points at which the variables are defined). Of course, interpolation must still be from strong connections (either direct or long-range); otherwise the correction is meaningless, and convergence will deteriorate.

4.6.6.2. Modifications to the interpolation formula. The accuracy of interpolation can be only indirectly controlled in AMG. When interpolation is defined as in § 4.6.1, convex interpolation usually ensures good results. However, for some of the problems considered in § 4.7, for example, even convex interpolation is not sufficient for good convergence. This generally occurs when the algebraically smoothest components (i.e., those least affected by relaxation) are not well enough approximated by the range of interpolation. These components can be determined in several ways, either through use of the matrix itself or through more extensive use of geometrical information. Once these components are known, or approximated, it is possible to improve interpolation:

Suppose we have chosen C, F, and C_i for each $i \in F$ given, and we have q functions $z^1, z^2, \cdots, z^q$ which we want to interpolate exactly, if possible. (Note that here, superscripts indicate the function number, *not* the grid.) We restrict our discussion to a particular point $i \in F$. Let $C_i = \{k_1, k_2, \cdots, k_p\}$, and denote the value of the function z^μ at point k_ν by z_ν^μ, and the value of z^μ at the point i by z_0^μ. Assume that we have previously computed some interpolation weights $w_{ik_1}, w_{ik_2}, \cdots, w_{ik_p}$, which we would like to modify so that $z^1, z^2, \cdots, z^q$ can be approximated as closely as possible by the range of interpolation. For this, corrections $v_1, v_2, \cdots, v_p$ are computed that minimize the functional

$$\sum_{\mu=1}^{q} \left(z_0^\mu - \sum_{\nu=1}^{p} (w_{ik_\nu} + v_\nu) z_\nu^\mu \right)^2,$$

with the added constraint that $\sum_\nu v_\nu^2$ is minimal in the event that there is more than one solution. The weights are then updated by $w_{ik_\nu} \leftarrow w_{ik_\nu} + v_\nu$ for $\nu = 1, 2, \cdots, p$. This ensures that, if the original interpolation weights already interpolate the desired functions, they are not changed. Otherwise, the new weights will differ from the old ones in a minimal way. Below, we discuss two ways to obtain functions of interest that may be used in this approach.

"Linear" interpolation. For many physically based problems, functions that are algebraically smooth are also smooth in the usual geometric sense. This is not always the case. For example, discontinuities in the system or structure being analyzed can result in a problem where algebraically smooth functions vary greatly from one point to another, even if the points are physically close to each other. However, connections between such points, as defined by the matrix entries, will be weak. Thus, if the coarse grids are chosen as described before, we can expect the smoothest functions in the neighborhood of a point i to be linear on C_i. So, for a 2-D problem, for example, we can improve the interpolation using the above method by taking $z^1 = 1$, $z^2 = x$, and $z^3 = y$. When we define the interpolation weights in this way, it is natural to enforce convex interpolation.

Note that this use of linear interpolation does not introduce the same kind of disadvantages as linear interpolation in geometric multigrid methods. There, the assumption that smooth functions are locally linear, even in the direction of weak connections, requires a careful choice of the relaxation method in order to smooth, since the interpolation points are fixed independently of the operator.

Interpolation of eigenvectors. The algebraically smoothest functions of a problem are generally the eigenvectors corresponding to the smallest eigenvalues. The smaller the eigenvalue, the better the corresponding eigenvector must be approximated by the range of interpolation. The interpolation definition given in § 4.6.1 is usually sufficient for this purpose. However, in some problems (cf. § 4.5.4.3), this must be specifically enforced in order to maintain satisfactory convergence. This approach requires the approximate calculation of the eigenvectors corresponding to the smallest eigenvalues (generally there are very few small enough to cause problems in convergence). Such accuracy is rarely necessary, since the linear interpolation described above usually ensures accurate enough interpolation of the needed eigenvectors when standard interpolation does not. Therefore we do not report on our experiences here, and give only a rough outline of the method for completeness.

One (inefficient) method for finding the eigenvectors is to first construct all the coarse grid components in the usual way, then use a modified cycle to obtain an approximation to the desired eigenvector. The Gauss–Seidel relaxation used in AMG is actually a functional minimization of the energy norm of the error. Instead, a relaxation process can be formulated to minimize the Rayleigh quotient of the grid function (this requires carrying some additional corrections to all levels). An AMG cycle using this relaxation produces an approximation to the eigenvector that corresponds to the smallest eigenvalue. This eigenvector approximation can be used to modify interpolation and the coarse grid matrices on all levels. The cycling and interpolation improvement can be repeated until the desired accuracy is reached.

The computation of the eigenvector and updating of the interpolation can also be integrated so that, before the coarse grid correction on each level during the Rayleigh quotient minimization cycle, the interpolation and the coarser grid operator are updated with the current eigenvector approximation *before* the operation proceeds to the next coarser grid. This can be done in a quite efficient way and, if necessary, several eigenvector approximations can be computed simultaneously in a similar way if they are kept (approximately) orthogonal on all levels. The eigenvector approximations are used only as a tool for the improvement of interpolation. Once they have been calculated to the desired accuracy (thereby producing the desired interpolation), the AMG components can be used in standard cycling to solve the given problem.

4.7. AMG for systems problems. The algorithm described in the previous section was developed under the assumption that the fine grid matrix was a symmetric M-matrix. This assumption is often strongly violated when a number of different quantities are being approximated. For this reason, that method described in § 4.6 can fail when applied directly to discretized systems of partial differential equations and other similar matrix problems.

This class of problems is simply too large to allow any one algebraic multigrid approach to work efficiently for all cases, but methods can be developed that apply to limited classes. For these approaches, the user must be required to supply some information concerning the nature of the problem. Here, for ease of motivation, we concentrate on discretized systems of PDEs, and assume that some of the details of the discretization can be easily supplied. However, as in AMG for scalar problems, uniform discretizations and regular domains are not required, so such approaches should still prove quite useful, even though much of the "black box" nature is lost.

Consider a system of partial differential equations in p unknown functions:

$$\begin{aligned} &\mathcal{L}_{[11]}u_{[1]} + \mathcal{L}_{[12]}u_{[2]} + \cdots + \mathcal{L}_{[1p]}u_{[p]} = f_{[1]}, \\ &\mathcal{L}_{[21]}u_{[1]} + \mathcal{L}_{[22]}u_{[2]} + \cdots + \mathcal{L}_{[2p]}u_{[p]} = f_{[2]} \\ &\qquad\qquad\vdots \\ &\mathcal{L}_{[p1]}u_{[1]} + \mathcal{L}_{[p2]}u_{[2]} + \cdots + \mathcal{L}_{[pp]}u_{[p]} = f_{[p]} \end{aligned} \quad \text{on } \Omega, \tag{7.1}$$

where $\mathcal{L}_{[ij]}$ is a linear differential operator and $u_{[i]}$ is a scalar function defined on Ω. We consider two approaches, obtained from (7.1), to discrete problems; both are closely related to AMG for scalar problems. The first, which is similar to the simplest usual multigrid approach, is to define interpolation separately for each of the unknown functions, and is called the "unknown" approach. The second, which can be applied when all unknown

functions of the discrete problem are defined on the same set of points (i.e., on *a nonstaggered* grid), treats the problem in a point-oriented block fashion and is therefore called the "point" approach.

4.7.1. The "unknown" approach. A simple extension of the scalar algorithm allows the method to be used for a number of problems in which more than one unknown is involved. We consider the system (7.1). To discretize this problem, it is necessary to define p grids $\Omega_{[1]}, \Omega_{[2]}, \cdots, \Omega_{[p]}$, each a discretization of Ω. Let $u_{[i]}$, defined on $\Omega_{[i]}$, be the discrete approximation to the ith unknown function. Then, let $A_{[ij]}$ be a finite difference or finite element approximation to $\mathcal{L}_{[ij]}$. The discrete problem can be written as follows:

$$\begin{bmatrix} A_{[11]} & A_{[12]} & \cdots & A_{[1p]} \\ A_{[21]} & A_{[22]} & \cdots & A_{[2p]} \\ \vdots & \vdots & & \vdots \\ A_{[p1]} & A_{[p2]} & \cdots & A_{[pp]} \end{bmatrix} \begin{bmatrix} \mathbf{u}_{[1]} \\ \mathbf{u}_{[2]} \\ \vdots \\ \mathbf{u}_{[p]} \end{bmatrix} = \begin{bmatrix} b_{[1]} \\ b_{[2]} \\ \vdots \\ b_{[p]} \end{bmatrix}. \tag{7.2}$$

In order to avoid confusion in parts of our discussion, it is important to distinguish between what we call *points, unknowns,* and *variables.* An *unknown* is one of the functions being approximated (e.g., pressure, temperature, or a component of displacement), and each unknown is identified by a number. A *point* is a physical point of one of the grids $\Omega_{[1]}, \Omega_{[2]}, \cdots \Omega_{[p]}$. The value representing the unknown $u_{[i]}$ at a particular point of $\Omega_{[i]}$ is called a *variable.* Thus each variable is associated with both a point and an unknown. Here, the user must explicitly provide the correspondence between the variables and the unknowns.

The method we call the "unknown" approach can be simply described in terms of the scalar algorithm of § 4.6. In choosing the coarse grids and defining interpolation, the method of § 4.6.2 is used, but all off-diagonal blocks are ignored. This results in a block-diagonal interpolation operator, with each F-variable interpolating only from other variables corresponding to the same unknown. The coarse grid operator is constructed as before, using the *full* fine grid operator, so that connections between the different functions are not lost on coarser grids. In the cycling process, simple Gauss–Seidel is used, and the variable/unknown correspondence is no longer needed.

For this approach to be formally applicable, there are several restrictions on system (7.1) and the discretization chosen. The ith equation in (7.1) must be naturally associated with the ith unknown function, $u_{[i]}$. (This is not the case, for example, with the Cauchy–Riemann equations.) The problem should be discretized so that there is a natural one-to-one correspondence

between variables and equations, since this is required for Gauss–Seidel relaxation. Finally, the resulting discrete problem must have diagonal blocks $A_{[ii]}$ to which scalar AMG can be applied.

This method works well in problems for which variable-wise relaxation produces algebraically smooth error separately in each unknown (i.e., if $e_{[i]}$ is the error in the ith unknown, $A_{[ii]}e_{[i]}$, properly scaled, is small compared to $e_{[i]}$). The character of the off-diagonal blocks determines whether or not there is such smoothing: it is sufficient that the off-diagonal blocks in each row are "small" compared to the corresponding diagonal block. However, the entries in the off-diagonal blocks do not necessarily have to be weak connections in the usual sense. For example, consider the case of the plane-stress problem of elasticity given in (7.3) below. When Poisson's ratio ν is not too large (say $\nu \leq .5$), the present approach works well even though the connections between unknowns in the discrete operator are relatively large.

When this method can be used, it is quite effective. Below, we give some results of the application of this approach to elasticity problems and a discrete problem in VLSI design.

Elasticity problems. The plane-stress elasticity problem can be written in terms of the displacements u and v as

$$\begin{aligned} u_{xx} + \frac{1-\nu}{2} u_{yy} + \frac{1+\nu}{2} v_{xy} &= f, \\ \frac{1+\nu}{2} u_{xy} + \frac{1-\nu}{2} v_{xx} + v_{yy} &= g, \end{aligned} \tag{7.3}$$

where ν is Poisson's ratio, chosen to be $\frac{1}{3}$ (which is a realistic value for many applications). The domain used is the unit square. A finite element discretization of the problem on a uniform grid with bilinear test functions is used, with $h = 1/16$, $1/32$, and $1/48$.

We first applied the "unknown" approach without any of the modifications described in § 4.6.6. Each side of the domain was specified as either fixed (Dirichlet boundary condition) or free. Table 4.5 shows the V-cycle convergence factors and CPU times per cycle obtained as a function of h

TABLE 4.5

	$l=0$		$l=1$		$l=2$		$l=3$	
h	ρ	t_V	ρ	t_V	ρ	t_V	ρ	t_V
1/16	0.13	0.07	0.20	0.08	0.36	0.09	0.46	0.10
1/32	0.18	0.34	0.30	0.36	0.52	0.38	0.65	0.39
1/48	0.18	0.82	0.31	0.84	0.49	0.87	0.75	0.88

and l, the number of free sides, where we take $l = 0$, 1, 2, and 3 (for $l = 2$ the free sides are adjacent).

For the full Dirichlet problem, convergence is good for all h tested. However, the behavior becomes worse as free boundaries are introduced, with the convergence clearly dependent on problem size for the case of 3 free boundaries. This indicates that the quality of interpolation is not good along the free boundaries. We then used two methods to improve the interpolation. These are as follows:

Method 1. Convex interpolation, as described in § 4.6.6.1, was forced, so that each F-point was in the convex hull of its set of interpolation points.

Method 2. In addition to enforcing convex interpolation, the interpolation weights were modified to interpolate linear functions exactly according to § 4.6.6.2.

The results for these approaches on the problem with 3 free boundaries are given in Table 4.6. Although there may still be a slight dependence on problem size when Method 1 is used, the convergence factors obtained are much better than those for the original method. With Method 2, the factors approach those for the original method on the full Dirichlet problem, and do not depend on h.

Remark 7.1. When convex interpolation is enforced, the times per cycle increase. This is because the use of convex interpolation generally increases the number of C-points somewhat. Although no additional C-points are forced in order to define the linear interpolation, the coarser grid operators produced are different from those when only convex interpolation is used, which explains the difference in cycle times between Method 1 and Method 2.

We also performed some preliminary experiments using a 3-D elasticity problem. The domain was the unit cube. We took $h = 1/8$. Although the problem sounds small, there are 2187 variables and around 150,000 matrix entries stored. The 4 bottom corners are fixed, and the body is free elsewhere. In this case, Method 2 from above is used. Furthermore, long-range interpolation is used with path length 2 in order to lower the storage requirements. The results were quite promising. The asymptotic

TABLE 4.6

	Method 1		Method 2	
h	ρ	t_V	ρ	t_V
1/16	0.25	0.15	0.21	0.15
1/32	0.30	0.64	0.22	0.69
1/48	0.33	1.46	0.22	1.82

convergence factor obtained was .276, with an operator complexity of 1.50 for a (1, 1) V-cycle.

A VLSI design problem. The following problem occurs as part of a CAD-system for VLSI-design that is under development at the University of Saarbrücken (see [Be1]). As a first step in developing the physical layout of a chip, the chip components are regarded as nodes of a planar graph, some connected not only to certain neighbors but also to some geometrically fixed points at the boundary of the chip (*graph with fixed boundary*). Starting from this graph, which describes the logical connections of the chip components, we compute a first guess for a layout by computing geometrical locations of the nodes, minimizing the functional

$$\sum_l w_l f(\|s_l\|),$$

where s_l denote the edges of the graph (including those to the fixed boundary points), w_l are certain nonnegative weights, and $f(\xi)$ is some "cost" function.

In the simplest case, one considers $w_l = 1$ and $f(\xi) = \xi^p$ with some $p \geq 2$, the most interesting cases being $p = 2$ (which corresponds to requiring that the overall signal transmission time be minimum) and $p \to \infty$ (which corresponds to requiring the maximum signal transmission time to be minimum). The above minimization corresponds to solving the following system of equations:

$$\sum_{j \in N_i} \|\mathbf{P}_i - \mathbf{P}_j\|^{p-2}(\mathbf{x}_i - \mathbf{x}_j) = 0 \qquad (i = 1, \cdots, n),$$

$$\sum_{j \in N_i} \|\mathbf{P}_i - \mathbf{P}_j\|^{p-2}(\mathbf{y}_i - \mathbf{y}_j) = 0 \qquad (i = 1, \cdots, n),$$

where the $\mathbf{P}_i = (\mathbf{x}_i, \mathbf{y}_i)$ $(i = 1, \cdots, n)$ denote the nodes of the graph (including the fixed boundary nodes). N_i denotes the neighborhood of i, that is, the set of indices j such that $\mathbf{P}_i$ and $\mathbf{P}_j$ are connected by an edge of the graph. The unknowns are the coordinates of the *inner* nodes.

If $p = 2$, this system is linear and similar to two decoupled discrete Laplace equations, and we can apply the algorithm described in § 4.6 without any modification. For $p > 2$, the system is nonlinear, and the Jacobian matrix is symmetric and has the form

$$\begin{pmatrix} M_x & B \\ B & M_y \end{pmatrix},$$

with M_x and M_y being weakly diagonally dominant M-matrices. If p is not too large ($p \leq 10$, say), the "unknown" approach allows for a rapid solution if combined with Newton's method in a straightforward way. We found that, in order to keep the linearization work small, it was reasonable to

TABLE 4.7

Number of nodes	Convergence factor			
	$p=2$	$p=4$	$p=6$	$p=8$
250	0.16	0.15	0.17	0.20
500	0.17	0.15	0.18	0.22
1000	0.19	0.17	0.20	0.29
2000	0.21	0.20	0.23	0.32
4000	0.23	0.19	0.23	0.32
8000	0.24	0.19	0.24	0.33

make a simple "continuation" in p. Furthermore, instead of recomputing all coarse grid components in each Newton step, we can reduce the setup work of AMG by up to 50% (without affecting convergence essentially) by recomputing only the coarse grid operators, keeping the grids and the interpolation operators fixed.

In Table 4.7 we give numerically observed convergence factors per cycle, for different p, that typically are around 0.2, with a tendency to increase slightly as p becomes large. We mention that $p > 10$ is usually not needed in practice; solutions then change very little as p is increased further. Figures 4.5 and 4.6 show two explicit examples for demonstration, where the figures on the left and right show the given and optimized graphs, respectively (where $p = 4$ for Fig. 4.5 and $p = 8$ for Fig. 4.6).

4.7.2. The "point" approach. In many cases, systems of the form (7.1) are discretized on *nonstaggered grids* (i.e., $\Omega_{[1]} = \Omega_{[2]} = \cdots = \Omega_{[p]}$). The scalar algorithm of § 4.6 can, in principle, be extended to such problems in a straightforward way. The main idea is to apply the algorithm in a "block" manner, with all variables that correspond to the same point relaxed, interpolated, and coarsened together. However, a practical application can be quite expensive and difficult. We have not developed an efficient implementation of the ideas described below, but a preliminary test, presented below, shows that the method can be effective.

Clearly, the correspondence between variables and points must be provided before we can use this approach. We can then rewrite (7.2) so that all variables and equations associated with the same point are grouped together:

$$\begin{bmatrix} A_{(11)} & A_{(12)} & \cdots & A_{(1n)} \\ A_{(21)} & A_{(22)} & \cdots & A_{(2n)} \\ \vdots & \vdots & & \vdots \\ A_{(n1)} & A_{(n2)} & \cdots & A_{(nn)} \end{bmatrix} \begin{bmatrix} \underline{u}_{(1)} \\ \underline{u}_{(2)} \\ \vdots \\ \underline{u}_{(n)} \end{bmatrix} = \begin{bmatrix} \underline{b}_{(1)} \\ \underline{b}_{(2)} \\ \vdots \\ \underline{b}_{(n)} \end{bmatrix}, \tag{7.4}$$

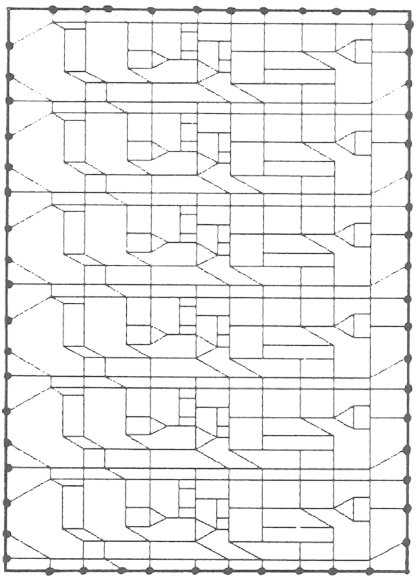


FIG. 4.5(a)

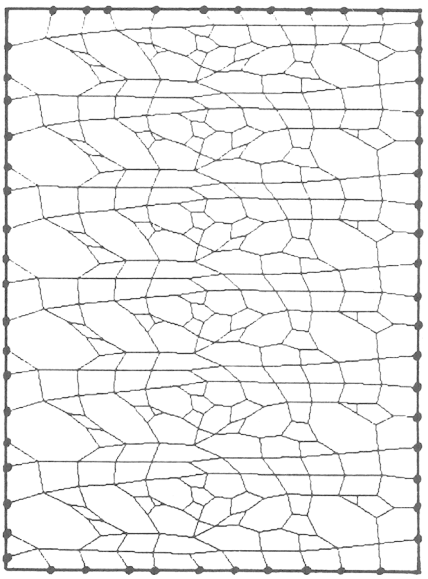


FIG. 4.5(b)

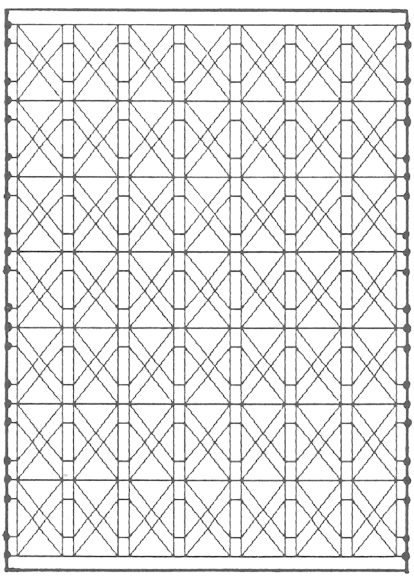


FIG. 4.6(a)

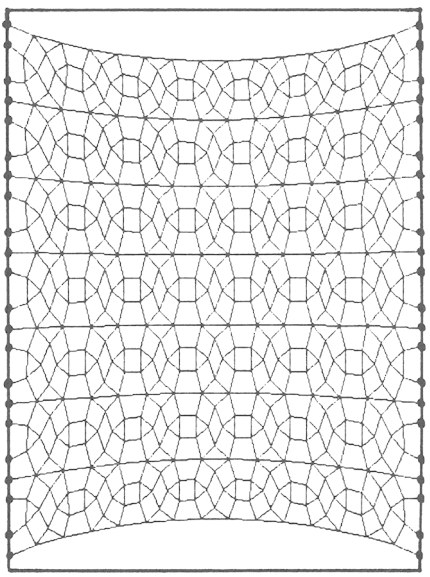


FIG. 4.6(b)

where $\underline{u}_{(i)}$ is the vector of variables defined at point i, $\underline{b}_{(i)}$ is the appropriate vector of values for the right-hand side, and n is the number of points on the grid. Here, the obvious analogue to the scalar relaxation is a point-oriented block relaxation in which all the variables associated with a particular point are changed simultaneously to satisfy the equations defined at that point. The grid can then be coarsened pointwise, so that all the variables at each point are either C-variables or F-variables. We will call such points C-points and F-points, respectively. Interpolation is then defined so that all variables at a given F-point interpolate from all variables at the surrounding C-points.

As in the scalar case, the coarsening process should be defined in such a way that smooth error lies in the range of interpolation. For problems that exhibit the same type of algebraic smoothing behavior as M-matrices with usual Gauss–Seidel relaxation, the properly-scaled residual becomes small relative to the error. This property can be used to express the error at a point in terms of the error at surrounding C-points, in a way analogous to that in the scalar algorithm. Let i be an F-point. Then, letting $\underline{e}_{(j)}$ denote the error vector at a point j, the error at point i after relaxation must satisfy

$$\underline{A}_{(ii)}\underline{e}_{(i)} \approx -\sum_{j \neq i} \underline{A}_{(ij)}\underline{e}_{(j)},$$

which is analogous to (6.2). We again wish to use this formula to construct the interpolation operator, and the coarse grid should be chosen in such a way that this is possible. We will say that a point i is strongly connected to j if $\underline{A}_{(ij)}$ is large (in some chosen norm) compared to the other off-diagonal matrices in row i. As in the scalar case, we define S_i to be the set of strong connections of i, and given C and F, set $C_i = S_i \cap C$. We can now restate the goal in choosing the coarse grid (cf. Criterion (C1) in § 4.6.1): for each point $i \in F$, each point $j \in S_i$ must either be in C_i, or must strongly depend on C_i *in the proper way*.

This is where a difficulty arises. Simply requiring $\underline{A}_{(jk)}$ to be large for some $k \in C_i$ is not sufficient. (For example, consider two unrelated scalar problems defined on the same grid. Such a criterion would allow the use of completely irrelevant information in choosing the coarse grid.) Instead, we note that $\underline{e}_{(i)}$ is only affected by $\underline{e}_{(j)}$ if $\underline{A}_{(ij)}\underline{e}_{(j)}$ is large. Then it is clear that we only need that part of $\underline{e}_{(j)}$ corresponding to large eigenvalues of $\underline{A}_{(ij)}$ to be well determined by C_i. Suppose that $C_i = \{k_1, k_2, \cdots, k_l\}$. Then we should require that these eigenvectors be well approximated by the range of the matrix $[\underline{A}_{(jk_1)}\underline{A}_{(jk_2)} \cdots \underline{A}_{(jk_l)}]$. Once this is ensured, an approximation to $\underline{e}_{(j)}$ can be derived in terms of the e at points in C_i, and interpolation can be defined in a manner similar to that in the scalar algorithm.

This method requires that, in the discretized system, the set of equations defined at each point be associated with the set of variables defined there. A

strict variable/equation correspondence is not necessary, since this method performs relaxation inverting the diagonal blocks of (7.4).

The approach outlined above would clearly be more expensive than the "unknown" approach of § 4.7.1, although some simplifications are possible. However, this seems the most natural method for a number of problems with a true point-oriented character, such as those arising in the analysis of frame structures. An example of such a problem is the following.

A structural problem. One structural problem, a rigid frame, is illustrated in Fig. 4.7. The actual structure is twenty levels high, rather than the eight shown in the figure. The nodes are the locations where members (beams) are joined, and two nodes are connected whenever there is a member between them. (Although the members cross on the faces of the structure, there are no nodes there since the members are not actually joined.) The unknowns in this problem are the displacements of the nodes in the x, y, and z directions. The equations to be solved can be obtained by considering each member as a spring, with the end displacements related to the forces at the nodes by Hooke's Law; linearizing around the initial node locations; and setting the sum of all internal and external forces at each node to zero. In rigid frame problems, rotations of the joints are usually included as unknowns, but they have been eliminated here for simplicity. Here, the four points along the bottom are fixed, and the remainder of the structure is free. (For a more detailed discussion of such problems, see [Ch1].)

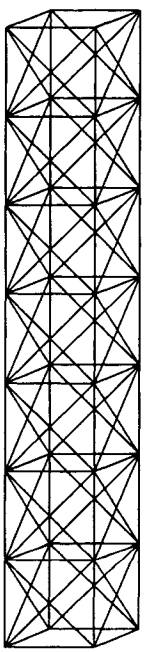


FIG. 4.7

There are difficulties in applying the "unknown" approach to this problem. Even with convex linear interpolation, asymptotic convergence factors of .8 and above were obtained. As a test, we then applied the point-oriented approach described above, with the coarse grids chosen to satisfy the given criterion. The V-cycle convergence factor obtained was .18. This clearly shows that the point approach may be essential for such problems.

4.8. Conclusions and further research. Algebraic multigrid has proven to be a robust and efficient black box solution method for several types of matrix equations. Although many of the numerical experiments reported here have been for problems defined on uniform discretizations of simple domains, we have tried to show that the method is certainly not restricted to these cases. For such "nice" problems, properly designed geometric multigrid algorithms can be more efficient, although the ease of use of AMG makes it an attractive tool even here. However, there are many

problems that could benefit from multigrid ideas, but to which the application of usual multigrid methods is difficult or impossible. For example, many applications programs generate irregular grids tailored to particular domains or operators. Because AMG is insensitive to complexities in the domains and grids chosen, it is ideal for such problems. Another class of problems to which usual multigrid methods cannot be applied are those that are discrete in nature, since the necessary multigrid components cannot be defined. These are often similar to the matrix equations obtained from the discretization of PDEs on irregular grids, and AMG can be easily applied.

Current and planned work in AMG lies in two main directions. The primary goal is to extend the application range of the method and to develop robust, efficient algorithms for new classes of problems. The second and related goal is to develop methods for the efficient solution of chains of related problems, where there are smooth or local changes to the system, in either the right-hand sides or the fine grid operators, from one problem to the next.

New problem areas. There are a number of areas in which the application of AMG seems promising, both in terms of the algorithm's success and of its potential benefit. These include problems in fluid flows and structural mechanics. The ability of AMG to deal with complex domains makes it attractive in both of these areas. This is particularly true in structural problems. Here, there is also a need to solve large discrete problems, for which AMG is well suited.

One change in the algorithm that can make it applicable to a wider range of problems is the use of different relaxation methods. The use of block Gauss–Seidel in the method of § 4.7.2 is one example. Another would be the use of distributed relaxation methods, such as Kaczmarz iteration. This would allow the use of AMG for problems, such as the Cauchy–Riemann and Stokes equations, that lack the natural variable/equation correspondence necessary for Gauss–Seidel relaxation.

In the development of AMG algorithms for larger classes of problems, it is necessary to balance convenience and efficiency. Currently, it seems clear that different approaches are necessary for different types of problems. An algorithm capable of automatically determining the proper approach for a particular problem may spend too much time trying to find information that could be easily specified by the user. Another option is simply to develop, for certain types of problems, specialized algorithms that could be used over a wide range of changes in the domain or coefficients of the operator.

Chains of problems. The second area of research—developing efficient solutions to chains of related problems—is important in a number of applications, including nonlinear and time-dependent problems, as well as reliability analysis and design problems. A large amount of work in the

AMG algorithm is invested in the setup phase, and for the problems mentioned, the setup work for later problems can sometimes be reduced or avoided altogether, effectively reducing the time required for the solution of each problem. The extent of the changes in the matrix equation determines the amount of work necessary to resolve the problem. Local changes may require only a small amount of work in both the setup and the solution phase.

The simplest change is in the right-hand side. This is often done in time-dependent problems or in structural problems in order to study the effect of different loads on a structure. Then the setup phase need not be repeated at all. If the new right-hand side is a perturbation of the previous one, few cycles may be necessary. In addition, if the perturbation is only local, then relaxation on the finest level is only necessary in the neighborhood of the change, since the error introduced will normally be smooth far away. The area of relaxation increases on coarser grids, with global relaxation occurring only on the coarsest. In this way, the smooth changes in the solution are obtained by interpolation.

When the matrix itself changes, as in the solution of nonlinear problems by Newton's method, the entire setup process may not need to be repeated. If the change is small enough, all coarse grid components may be retained. It is more likely that some changes will be necessary. If the character of the connections in the matrix does not change drastically, it is possible to keep the coarse grids and the interpolation operators, while simply recomputing the coarse grid operators. This costs much less than repeating the entire setup phase, since a large amount of work is used in choosing the coarse grids. Another possibility is to recompute interpolation while again keeping the coarse grid structure.

It is also possible that the changes introduced in the matrix are only local. This is the case when local changes are made to determine their effect on the behavior of the entire structure under a load. In this case, the changes in the coarsening may also be only local. If necessary, new coarse grids or interpolation may be needed in the neighborhood of the change. The coarse grid matrices need to be only locally recomputed, since most of the matrix entries will be unaffected by the changes. The effect will spread on coarser grids, but the overall work necessary is small compared to the initial setup. Once the setup is completed, the solution may be quickly computed, again starting from the previous solution and using local relaxation.

It should be possible to automatically determine the amount of relaxation required and the extent of the changes needed in the coarse grid matrices. For example, during local relaxation, the area of relaxation can be increased until changes in the solution approximation are seen to be small. Deciding the changes necessary in the coarse grids, the interpolation, and the coarse grid matrices is more difficult, but feedback is available in the form of the resulting convergence factors.

Acknowledgments. This work was supported by the Air Force Office of Scientific Research under grant AFOSR-86-0126, the Department of Energy under grant DE-AC03-84ER80155, the NASA Langley Research Center under grant NAG-1-453, and the Gesellschaft für Mathematik und Datenverarbeitung.

REFERENCES

[Be1] B. BECKER, G. HOTZ, R. KOLLA AND P. MOLITOR, *Ein CAD-System zum Entwurf integrierter Schaltungen,* Report no. 16, Universität Saarbrücken, Fachbereich 10, 1984.

[Ch1] W. CHU-KIA AND C. G. SALMON, *Introductory Structural Analysis,* Prentice–Hall, Englewood Cliffs, NJ, 1984.

[Ei1] S. C. EISENSTAT, M. C. GURSKY, M. H. SCHULTZ AND A. H. SHERMAN, *Yale sparse matrix package.* I: *The symmetric codes,* Internat. J. Numer. Methods Engrg., 18 (1982), pp. 1145–1151.

[Po1] A. POPE, *Geodetic computation and sparsity,* Proc. Sparse Matrix Symposium, Fairfield Glade, TN, October 1982.

[Sc1] J. SCHRÖDER, *Operator Inequalities,* Math. Sci. Engrg., 147, Academic Press, New York, 1980.