Computing 77, 57–75 (2006)
Digital Object Identifier (DOI) 10.1007/s00607-005-0147-x

Computing
Printed in Austria

Algebraic Multigrid Based on Computational Molecules, 1: Scalar Elliptic Problems

J. K. Kraus and **J. Schicho,** Linz

Received April 1, 2005; revised July 22, 2005
Published online: January 4, 2006

Abstract

We consider the problem of splitting a symmetric positive definite (SPD) stiffness matrix A arising from finite element discretization into a sum of edge matrices thereby assuming that A is given as the sum of symmetric positive semidefinite (SPSD) element matrices. We give necessary and sufficient conditions for the existence of an exact splitting into SPSD edge matrices and address the problem of best positive (nonnegative) approximation.

Based on this disassembling process we present a new concept of "strong" and "weak" connections (edges), which provides a basis for selecting the coarse-grid nodes in algebraic multigrid methods. Furthermore, we examine the utilization of computational molecules (small collections of edge matrices) for deriving interpolation rules. The reproduction of edge matrices on coarse levels offers the opportunity to combine classical coarsening algorithms with effective (energy minimizing) interpolation principles yielding a flexible and robust new variant of AMG.

AMS Subject Classifications: 65F10, 65N20, 65N30.

Keywords: Edge matrices, algebraic multigrid, interpolation weights, coarse-grid selection.

1. Introduction

We are concerned with the solution of large-scale systems of linear equations

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

arising from finite element (FE) discretization of second-order self-adjoint elliptic boundary-value problems. In this situation, the matrix A in (1) is typically sparse and symmetric positive definite (SPD).

In many instances (of this huge class of problems) Algebraic MultiGrid (AMG) methods [2]–[5] can be used to build highly efficient and robust linear solvers [10], [12], [16], [18], [19], [21]. AMG using element interpolation (AMGe) [6], [13], [14], so-called spectral AMGe [9], and AMG based on smoothed aggregation [17], [22], [24], [25], [26] have even broadened the range of applicability of the classical AMG algorithm [18]. These more recent developments are based on techniques of energy-minimizing interpolation (or prolongation), which can be achieved by different means.

The computation of edge matrices, we are suggesting in the present paper, is motivated by the fact that they provide a good starting point for building efficient

AMG components, while keeping the set-up costs low. The main emphasis of this paper is on the algebraic construction of edge matrices and their utilization in the framework of algebraic multigrid: We discuss how to alter the concept of "strong" and "weak" connections, as it is used in the process of coarse-grid selection (and interpolation) with classical AMG. The interpolation component in our approach is very similar to the element interpolation used in so-called AMGe methods. However, the *computational molecules* involved in the arising local min-max problem are assembled from edge matrices in our case.

The resulting method lies in-between classical AMG (*strong* and *weak* edges affect the coarsening and the formation of interpolation molecules) and AMGe based on element agglomeration (small-sized neighborhood matrices serve for the computation of the actual interpolation coefficients). Numerical tests indicate the robustness of the considered method to which we refer as AMGm (Algebraic MultiGrid based on computational molecules) with respect to anisotropy as well as perturbations of the M-matrix property.

2. Edge Matrices

Let A_T be an $(nd)\times(nd)$ symmetric and semipositive element matrix. Here, n denotes the number of nodes in the element T, and d denotes the number of degrees of freedom in each node. For i, j, $1 \le i < j \le n$, let E_{ij} be an $(nd) \times (nd)$ symmetric matrix whose entries are zero except for the $(2d) \times (2d)$ entries corresponding to the nodes i, j. Such a matrix will also be called an *edge matrix*. We say that A_T has a positive splitting iff we can write it as a sum of positive semidefinite edge matrices. Note that positivity implies that $E_{ij}\mathbf{v} = \mathbf{0}$ for all $\mathbf{v} \in \ker(A_T)$, because for any sum of positive semidefinite matrices, the kernel of the summands contains the kernel of the sum.

As the edge matrices correspond to edges connecting nodes, we sometimes write $E_{ij} := E_{ji}$ for $i > j$; if $i = j$, then E_{ij} is not defined.

The main goal in this section is to give a necessary and sufficient criterion for the existence of a positive splitting in the case $d = 1$. We also specify the construction for the case when this criterion is fulfilled.

Using the terminology from [23], we say that a matrix is *irreducible* iff the graph with nodes $1, \ldots, n$ and edges representing nonzero matrix entries is connected. If A_T is reducible, then there is a numbering of nodes for which A_T has the shape of a block diagonal matrix. One can show that A_T has a positive splitting iff every diagonal block has a positive splitting. Therefore, we will focus our attention to irreducible matrices.

We say that $A_T = (a_{ij})_{i,j}$ is an L-matrix iff $a_{ii} > 0$ and $a_{ij} \le 0$ for $1 \le i \ne j \le n$. If A_T is not an L-matrix, then there is a unique L-matrix $B := (b_{ij})_{i,j}$ such that $|a_{ij}| = |b_{ij}|$ for all i, j. We say that B is the L-ation of A_T.

Lemma 2.1: *A symmetric matrix A_T has a positive splitting iff its L-ation has a positive splitting.*

Proof: Let $A_T = \sum_{i,j} E_{ij}$ be a positive splitting into edge matrices. Then the L-ation of A_T is the sum of the L-ations of the edge matrices E_{ij}. Because positivity of a symmetric 2×2-matrix does not depend on the sign of the off-diagonal entry, this decomposition is a positive splitting of the L-ation.

Conversely, assume that $\sum_{i,j} E_{ij}$ is a positive splitting of the L-ation. We multiply the off-diagonal entries of the matrices E_{ij} by ± 1, according to the sign of the entry $(A_T)_{i,j}$. The 2×2-matrices remain positive semidefinite, and sum up to A_T. □

Lemma 2.2: *If A_T is a symmetric irreducible singular L-matrix, then its kernel has dimension* 1*, and it is generated by a positive vector.*

Proof: This is well-known. We include a proof for the sake of self-containedness.

Let $\mathbf{v} = (v_1, \ldots, v_n)^t$ be an element of the kernel. By simultanuous permutation of rows and columns, and maybe replacing $\mathbf{v}$ by $-\mathbf{v}$, we may assume that $v_1, \ldots, v_l > 0$, $v_{l+1}, \ldots, v_m < 0$, and $v_{m+1} = \ldots = v_n = 0$ for indices l, m such that $1 \leq l \leq m \leq n$. Then A_T can be written as block matrix

$$A_T = \begin{pmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{pmatrix}$$

with positive semidefinite A_{ii}, and $A_{ij} \leq 0$ elementwise for $i \neq j$.

For $i = l + 1, \ldots, m$, we have

$$a_{i1}v_1 + \cdots + a_{il}v_l + \sum_{j=l+1}^{m} a_{ij}v_j = 0,$$

hence

$$\sum_{j=l+1}^{m} a_{ij}v_i v_j = v_i \left(\sum_{j=1}^{l} a_{ij}v_j \right) \leq 0.$$

Summing up, we get $\sum_{i,j=l+1}^{m} a_{ij}v_i v_j \leq 0$. But A_{22} is positive semidefinite, which shows that we have equality everywhere. Hence $A_{21} = 0$ and the vectors $\mathbf{w} := (v_1, \ldots, v_l, 0, \ldots, 0)^t$ and $\mathbf{v} - \mathbf{w}$ are both in the kernel of A. But $A\mathbf{w} = 0$ implies $A_{31} = 0$, and $A(\mathbf{v} - \mathbf{w}) = 0$ imples $A_{32} = 0$. Since A is assumed to be connected, we have $l = m = n$. □

Lemma 2.3: *If A_T is a symmetric irreducible singular L-matrix, then it has a unique splitting into edge matrices. Moreover, this splitting is positive.*

Proof: By Lemma 2.2, there is a positive vector $\mathbf{v} = (v_1, \ldots, v_n)^t$ generating the kernel. For any i, j, $1 \leq i < j \leq n$, there is a unique edge matrix E_{ij} annihilating $\mathbf{v}$ and with off-diagonal entry a_{ij}, namely the matrix with nonzero entries

$$\begin{pmatrix} -a_{ij}v_j/v_i & a_{ij} \\ a_{ij} & -a_{ij}v_i/v_j \end{pmatrix}$$

(except when $a_{ij} = 0$, which yields a zero edge matrix). Direct computation shows that the sum of these E_{ij} equals A_T. This shows that we have a unique splitting. The positivity is a consequence of $a_{ij} \leq 0$ and $v_i v_j > 0$. □

Lemma 2.4: *Any positive semidefinite matrix can be written as a sum of a singular positive semidefinite matrix and a positive semidefinite diagonal matrix.*

Proof: Let B be a positive semidefinite matrix. Let D be a nonzero positive diagonal matrix. The set of all real numbers λ such that $B - \lambda D$ is positive semidefinite is closed and it contains zero. Moreover, it is bounded because $-D$ is certainly not positive semidefinite. If λ_0 is the maximum of this set, then $B - \lambda_0 D$ is singular and positive semidefinite and $\lambda_0 D$ is a positive semidefinite diagonal matrix. □

Theorem 2.1: *A symmetric matrix A_T has a positive splitting iff its L-ation is positive semidefinite.*

Proof: Without loss of generality, we may assume that A_T is irreducible. By Lemma 2.1, we may also assume that A_T is an L-matrix. Then one direction is obvious: if A_T has a positive splitting, then it is positive semidefinite.

Assume that A_T is positive semidefinite. If A_T is singular, then it has a positive decomposition by Lemma 2.3. Otherwise, we use Lemma 2.4 and write $A_T = A' + D$, where A' is a singular L-matrix and D is a positive diagonal matrix. By Lemma 2.3, A' has a positive splitting. Clearly, D has a positive splitting. By summing the edge matrices for A' and for D, we get a positive splitting for A_T. □

We turn to the question how to compute a positive splitting, in case we know there exists one? If A_T is a singular positive semidefinite L-matrix, then we can use the explicit construction in the proof of Lemma 2.3. If A_T is a nonsingular positive semidefinite L-matrix, then we compute the smallest eigenvalue $\lambda_{\min}$, construct a positive splitting of $A_T - \lambda_{\min} I$, and add suitable diagonal edge matrices. If A_T is not an L-matrix but its L-ation is positive semidefinite, then we L-ate A_T, compute a positive splitting of the L-ation, and de-L-ate the edge matrices again by multiplying the off diagonal elements with ± 1.

The next question is what to do when there does not exist a positive splitting? In such a case, we want to come as close as possible to a positive splitting. More precisely, we want to compute a symmetric matrix A_+ with a positive splitting, such that there exist positive numbers λ, μ with

$$\lambda \langle \mathbf{x}, A_+ \mathbf{x} \rangle \leq \langle \mathbf{x}, A_T \mathbf{x} \rangle \leq \mu \langle \mathbf{x}, A_+ \mathbf{x} \rangle \ \forall \mathbf{x}, \tag{2}$$

and the quotient μ/λ is minimal. Such an A_+ will be called a best positive approximation.

Such a best positive approximation does not always exist: if $\operatorname{rank}(A_T) \leq n-2$, and the kernel is not in a special position with respect to the basis, then there is no nonzero edge matrix that annihilates the kernel. It follows that for any nonzero matrix A_+ with a positive splitting, there is a vector $\mathbf{v} \in \ker(A_T)$ which is not in the kernel of any of its edge matrix summands. It follows that $\langle \mathbf{v}, A_+\mathbf{v}\rangle > 0$, and there is no positive λ such that (2) holds.

Assume now that $\operatorname{rank}(A_T) \leq n-1$, and that the vector $\mathbf{v}$ generating the kernel has nonzero entries. By multiplying A_T from both sides with a suitable diagonal matrix, we can reduce to the case $\mathbf{v} = (1, \ldots, 1)^t$. If A_+ is a sum of positive semidefinite edge matrices E_{ij} such that (2) holds for some positive λ, μ, then $E_{ij}\mathbf{v} = 0$ for all edge matrices E_{ij}. This determines E_{ij} up to a constant factor, as in the proof of Lemma 2.3.

We define an SPM matrix to be a symmetric positive definite matrix with off diagonal entries negative or zero, and with row sums (or, equivalently, column sums) positive or zero. Our problem of computing a best approximation to matrices with kernel as above can be reduced to best SPM approximation, in the following way.

Lemma 2.5: *Let A_T be a symmetric semipositive matrix with kernel K generated by $\mathbf{v} = (1, \ldots, 1)^t$. Let A_{n-1} be the first minor of A_T (obtained by removing the last row and column). Let B be a symmetric matrix with kernel K, such that its first minor B_{n-1} is an SPM matrix. Then B is a best positive approximation of A_T iff B_{n-1} is a best SPM approximation of A_{n-1}.*

Proof: Note that B is uniquely determined by B_{n-1} and the requirements to be symmetric and to have $\mathbf{v}$ in the kernel. In fact, there is a unique way of adding a new row and column such that all row sums are zero and the new matrix is symmetric. As B_{n-1} is an SPM matrix, the so constructed matrix has no positive off diagonal elements. For any negative off diagonal entry, there is exactly one positive semidefinite edge matrix annihilating $\mathbf{v}$ with the same off diagonal entry. Obviously, B is the sum of all these edge matrices. This shows in particular that B is a sum of positive definite edge matrices.

As B_{n-1} is a best SPM approximation of A_{n-1}, there exist $\lambda, \mu > 0$ such that

$$\lambda\langle \mathbf{x}, B_{n-1}\mathbf{x}\rangle \leq \langle \mathbf{x}, A_{n-1}\mathbf{x}\rangle \leq \mu\langle \mathbf{x}, B_{n-1}\mathbf{x}\rangle \ \forall \mathbf{x}$$

holds, and such that the quotient μ/λ is minimal. Because the construction described above (adding a row and column such that row and column sums are zero) is linear and preserves positivity, we conclude that (2) holds with the same $\lambda, \mu > 0$. The quotient μ/λ is minimal with respect to the validity of (2), because otherwise we could find a better SPM approximation of A_{n-1} by cutting off the last row and column. □

The problem of computing the best SPM approximation has been considered in [16]. They gave a complete solution for $n = 3$ and $n = 4$ (i.e., for matrices of size 2×2 and 3×3). We write down the solution for $n = 3$, which will be used later.

As $(1, 1, 1)^t$ is in the kernel of A_T, we can write the matrix uniquely as

$$A_T = \begin{pmatrix} a+b & -a & -b \\ -a & a+c & -c \\ -b & -c & b+c \end{pmatrix}$$

with $a, b, c \in \mathbb{R}$. Because A_T is positive semidefinite, there is at most one negative number among a, b, c (otherwise we have a negative diagonal entry). We distinguish two cases.

Case 1: All numbers a, b, c are positive or zero. Then A_T has itself a positive splitting:

$$A_T = \begin{pmatrix} a & -a & 0 \\ -a & a & 0 \\ 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} b & 0 & -b \\ 0 & 0 & 0 \\ -b & 0 & b \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 \\ 0 & c & -c \\ 0 & -c & c \end{pmatrix}.$$

Case 2: One number, say a, is negative. Then the best positive approximation is

$$A_+ = \begin{pmatrix} a+b & 0 & -a-b \\ 0 & 0 & 0 \\ -a-b & 0 & a+b \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 \\ 0 & a+c & -a-c \\ 0 & -a-c & a+c \end{pmatrix}.$$

If A_T has full rank n, then there exists also a best positive approximation, but we could not compute a solution formula.

3. Coarse Grid Selection

3.1. "Strong" and "Weak" Edges

In contrast to Geometric MultiGrid (GMG) the relaxation in AMG is fixed [18]. Thus the coarsening process and the interpolation rule have to be chosen in a way such that the range of interpolation approximates those errors not efficiently reduced by relaxation.[1] These algebraically smooth error components $\mathbf{e}$ are characterized by

$$\|S\mathbf{e}\|_A \approx \|\mathbf{e}\|_A, \tag{3}$$

wherein S denotes the smoother. For (most of) the common smoothers, e.g., Gauß-Seidel or Jacobi, error that is slow to converge in energy norm equivalently fulfills the condition

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

In particular, for M-matrices this means that for each node i the error component e_i is essentially determined by those e_j for which $-a_{ij}$ is large. This leads to the following definition of strong connections used in classical AMG [18]:

[1] In GMG the hierarchy of grid equations is given, i.e., a fixed coarsening is used, and the smoother is adjusted in order to obtain efficient multigrid cycles.

Definition 3.1 (*Strong connections in Classical AMG*): *Node i is strongly connected to node j (strongly depends on j) if $-a_{ij} \geq \theta \cdot \max_{k \neq i}\{-a_{ik}\}$ with some $0 < \theta \leq 1$ (e.g., $\theta = 0.25$).*

Here we want to base the concept of strong connectivity on edge matrices. In [8] a reliable evaluation of strong connections based on element stiffness matrices has been presented. It uses the simple formula

$$s_{ij} := \frac{|a_{ij}|}{\sqrt{a_{ii}a_{jj}}}, \tag{5}$$

where $A_T = (a_{ij})_{i,j}$ is a local stiffness matrix corresponding to some element $T \in \mathcal{T}$ and $\mathcal{T}$ is a triangulation of the computational domain. Note that (5) defines the energy cosine of the abstract angle between the i-th and j-th (nodal) basis function in this case. However, the reproduction of local element stiffness matrices on coarser levels increases the set-up costs of an AMG method significantly and – what is even more serious – is subject to strong geometrical restrictions. That is why we suggest to construct edge matrices (from element matrices) and reproduce those on coarser levels.

Definition 3.2 (Direct connections): *Any two nodes i and j are said to be directly connected iff there is an edge $\{i, j\}$ connecting i and j; let E_{ij} denote the corresponding edge matrix.*

Now for every loop of length three (triangle) in the *algebraic grid* with direct connections (edges) $\{i, j\}$, $\{j, k\}$, and $\{k, i\}$ we consider the molecule

$$M^{(i,j,k)} := E_{ij} + E_{jk} + E_{ki}. \tag{6}$$

Furthermore, let

$$\mathcal{M}^{\triangle} := \{M^{(i,j,k)} = (c_{pq})_{p,q} : c_{pp} \neq 0 \quad \forall p = 1, 2, 3\} \tag{7}$$

be the set of all such local matrices given as the sum of three edge contributions (for edges that form a triangle) having non vanishing diagonal entries. Then the following definition provides an altered concept of strong connections ("strong" edges).

Definition 3.3 (Strong connections via edge matrices): *The strength of a (direct) connection $\{i, j\}$ is defined by*

$$s_{ij} := \min\left\{1, \min_{M^{(i,j,k)} \in \mathcal{M}^{\triangle}} \{|c_{p_i p_j}|/(\sqrt{c_{p_i p_i} c_{p_j p_j}})\}\right\} \tag{8}$$

where connections with $s_{ij} \geq \theta$ are said to be strong, $0 < \theta < 1$ (e.g., $\theta = 0.25$). Here p_i and p_j denote the local indices associated with nodes i and j, respectively, i.e., $1 \leq p_i \equiv p(i)$, $p_j \equiv p(j) \leq 3$.

Remark 3.1: *Note that the strength of a connection $\{i, j\}$ computed via either of the formulas (5) or (8) will be nonnegative and bounded above by 1 for general symmetric positive semidefinite stiffness matrices. A value close to one (an abstract angle close to zero) indicates a strong connection. Extending the definition (8) by* $\min_{\emptyset}\{\dots\} := \infty$ *we have* $s_{ij} = 1$ *whenever nodes i and j are directly connected but there is no path of length two connecting i and j via a third node k.*

3.2. Coarsening Algorithm

There are several reasonable ways of selecting the coarse grid nodes in AMG. Our approach is similar to the one used in classical AMG [18] in that it is based on a concept of strong connections (here "strong" edges). However, since the precise coarsening algorithm we use in detail differs from the one proposed in [18] it will be presented in this section. Following [18] a *good* coarse grid $\mathcal{D}_c$ should satisfy the following two criterions:

C1: *$\mathcal{D}_c$ should be a maximum independent set, which means that no strong connections within $\mathcal{D}_c$ are allowed.*

C2: *Each node j being strongly connected to an f-node i is either contained in $\mathcal{D}_c$ or it strongly depends on at least one c-node k that itself is strongly connected to node i.*

Similar to the procedure proposed in [18] we select the coarse grid in a two-stage process: First, a quick c-node choice attempts to enforce criterion (C1). Then, at a second stage, all f-nodes resulting from the first stage are tested to ensure that criterion (C2) holds, adding new c-nodes if necessary.

The main difference to the methodology in [18] is that our relation of strong connectivity, as defined in Definition 3.3, is symmetric whereas this is not the common practice in classical AMG, even not in the SPD case. That means, whenever a node i is strongly connected to a node j the reverse (j being strongly connected to i) is also true. However, this even simplifies the selection of an initial coarse grid that takes into account criterion (C1). A greedy algorithm serving this purpose is given by Algorithm 3.1. The testing of the initial coarse grid, and its adjustment with respect to criterion (C2), can be performed according to Algorithm 3.2. Note that we slightly simplified the corresponding procedure from [18] by avoiding multiple testing of f-nodes in the course of adding c-nodes (doing without an intermediate choice of "tentative" c-nodes). Instead, we prefer to decide immediately which f-nodes are going to be changed into c-nodes: if two f-nodes are strongly connected to each other with no c-node that strongly depends on both of them the necessity arises to change either of them to a c-node; we take the one having fewer strongly dependent c-nodes. For a formal description of this two stage process we define the node sets:

$\mathcal{D}_f$	...	fine nodes (f-nodes)
$\mathcal{D}_c$	...	coarse nodes (c-nodes)
$\mathcal{D} := \mathcal{D}_f \cup \mathcal{D}_c$	...	all nodes
$\mathcal{N}_i$	...	direct neighbors of node i
$\mathcal{N}_i^f := \mathcal{N}_i \cap \mathcal{D}_f$	...	fine direct neighbors

$\mathcal{S}_i$ ... strongly connected direct neighbors of node i
$\mathcal{S}_i^c := \mathcal{S}_i \cap \mathcal{D}_c$ ... strongly connected coarse direct neighbors
$|\mathcal{D}|$ or $|\mathcal{S}_m|$ ... cardinality of $\mathcal{D}$ respectively $\mathcal{S}_m$

Algorithm 3.1: (*Selection of initial coarse grid*)

$\mathcal{D}_c := \emptyset$; $\mathcal{D}_f := \emptyset$; $U := \mathcal{D}$;
$\lambda_m = |\mathcal{S}_m| \quad \forall m = 1, 2, \ldots, |\mathcal{D}|$; $n = 0$;
while $(n < |\mathcal{D}|)$
  find i *such that* $\lambda_i = \max_{m \in U} \lambda_m$
  $\mathcal{D}_c := \mathcal{D}_c \cup \{i\}$
  $U := U \setminus \{i\}$
  $n := n + 1$
  for all $j \in \mathcal{S}_i \cap U$
    $\mathcal{D}_f := \mathcal{D}_f \cup \{j\}$
    $U := U \setminus \{j\}$
    $n := n + 1$
    for all $k \in \mathcal{S}_j \cap U$
      $\lambda_k := \lambda_k + 1$

Algorithm 3.2: (*Adjustment of initial coarse grid*)

$\lambda_m = 0 \quad \forall m = 1, 2, \ldots, |\mathcal{D}|$; $i = 0$;
while $(i < |\mathcal{D}|)$
  $i := i + 1$
  if $(i \in \mathcal{D}_f)$
    $n_1 = 0$
    for all $k \in \mathcal{S}_i \cap \mathcal{D}_c$
      $n_1 := n_1 + 1$
      $\lambda_k = 1$
    for all $j \in \mathcal{S}_i \cap \mathcal{D}_f$
      $n_2 = 0$; $n_3 = 0$;
      for all $k \in \mathcal{S}_j \cap \mathcal{D}_c$
        $n_2 := n_2 + 1$
        if $(\lambda_k = 1)$
          $n_3 := n_3 + 1$
      if $(n_3 < 1)$
        if $(n_1 < n_2)$
          $\mathcal{D}_c := \mathcal{D}_c \cup \{i\}$
          $\mathcal{D}_f := \mathcal{D}_f \setminus \{i\}$
        else
          $\mathcal{D}_c := \mathcal{D}_c \cup \{j\}$
          $\mathcal{D}_f := \mathcal{D}_f \setminus \{j\}$
          $n_1 := n_1 + 1$
          $\lambda_j = 1$
    for all $k \in \mathcal{N}_i \quad \lambda_k = 0$

4. Interpolation

In this section, we want to figure out how to benefit from edge matrices when constructing the interpolation component of our AMGm method. The basic idea is to construct suitable small-sized *computational molecules* from edge matrices and to choose the interpolation coefficients in such a way that they provide a local minimum energy extension with respect to the considered *interpolation molecule*.

4.1. Interpolation Molecules

We will say that M is a computational molecule if M is a small-sized irreducible matrix that can be assembled from edge matrices. Let $\mathcal{E}_M$ be a small subset of the set of all edges $\mathcal{E}$, $\mathcal{E}_M \subset \mathcal{E}$. Then, for notational convenience we represent the molecule associated with the edge set $\mathcal{E}_M$ by

$$M := \sum_{\{i,j\}\in\mathcal{E}_M} E_{ij}. \tag{9}$$

Note that M is a small-sized $n_M \times n_M$ matrix where n_M denotes the number of distinct nodes k belonging to any of the edges $\{i, j\} \in \mathcal{E}_M$. To be precise, this matrix is obtained from the full-sized $N \times N$ matrix

$$C := \sum_{\{i,j\}\in\mathcal{E}_M} R_{ij}^T E_{ij} R_{ij} \tag{10}$$

by deleting all its zero rows and columns; the $2 \times N$ permutation matrices R_{ij} in (10) provide the mapping to the global ordering of nodes.

Consider now the set $\{A_T\}$ of individual element matrices all of which are split (disassembled) into edge matrices, i.e.,

$$A_T \approx \sum_{\{i,j\}\subset T} E_{ij} \quad \forall T. \tag{11}$$

We note that if the splittings (11) are positive throughout, i.e., all edge matrices E_{ij} are SPSD, then every computational molecule locally preserves the kernel of the global stiffness matrix:

Lemma 4.1: *Let* $B = \sum_{T\in\mathcal{T}_B} A_T$ *and* $\mathbf{v}_B \in \ker(B)$, *i.e.,* $B\mathbf{v}_B = \mathbf{0}$. *Further, let* $M = \sum_{\{i,j\}\in\mathcal{E}_M} E_{ij}$ *be any computational molecule such that every edge* $\{i, j\} \in \mathcal{E}_M$ *belongs to some element* $T \in \mathcal{T}_B$. *Moreover, let* $\mathbf{v}_M$ *denote the restriction of* $\mathbf{v}_B$ *to the edges in* $\mathcal{E}_M$, *i.e.,* $\mathbf{v}_M := \mathbf{v}_B|\mathcal{E}_M$. *If the splitting* (11) *is positive for all elements* $T \in \mathcal{T}_B$, *it follows that*

$$M\mathbf{v}_M = \mathbf{0}. \tag{12}$$

Proof: For an SPSD matrix B the condition $B\mathbf{v}_B = \mathbf{0}$ is equivalent to $\mathbf{v}_B^T B\mathbf{v}_B = 0$. Thus,

$$0 = \mathbf{v}_B^T B\mathbf{v}_B = \mathbf{v}_B^T \left(\sum_{\{i,j\}\in\mathcal{E}_M} E_{ij} + \sum_{\{i,j\}\in\mathcal{E}_B\setminus\mathcal{E}_M} E_{ij} \right) \mathbf{v}_B$$

$$= \mathbf{v}_M^T \left(\sum_{\{i,j\}\in\mathcal{E}_M} E_{ij} \right) \mathbf{v}_M + \underline{\mathbf{v}}^T F \underline{\mathbf{v}}$$

for an SPSD matrix F and an adequate restriction $\underline{\mathbf{v}}$ of the vector $\mathbf{v}_B$. This proves $\mathbf{v}_M^T M\mathbf{v}_M = 0$ and thus (12). □

The task is now to define suitable computational molecules for building interpolation. Assume that "weak" and "strong" edges have been identified, the coarse grid has been selected, and a set of edge matrices is available. Then for any f-node i (to which interpolation is desired) we define a so-called interpolation molecule

$$M(i) := \sum_{k\in\mathcal{S}_i^c} E_{ik} + \sum_{j\in\mathcal{N}_i^f:\, \exists k\in\mathcal{S}_i^c\cap\mathcal{N}_j} E_{ij} + \sum_{k\in\mathcal{S}_i^c\cap\mathcal{N}_j:\, j\in\mathcal{N}_i^f} E_{jk}. \tag{13}$$

This molecule arises from assembling all edge matrices associated with three types of edges: The first sum corresponds to the strong edges connecting node i to some coarse direct neighbor k (*interpolatory edges*). The second sum represents edges connecting the considered f-node i to any of its fine direct neighbors j being directly connected to at least one c-node k that is strongly connected to node i. Finally, the last sum in (13) corresponds to these latter mentioned connections (edges) between fine direct neighbors j and strongly connected coarse direct neighbors k of node i. The formation of interpolation molecules is illustrated in Fig. 1.

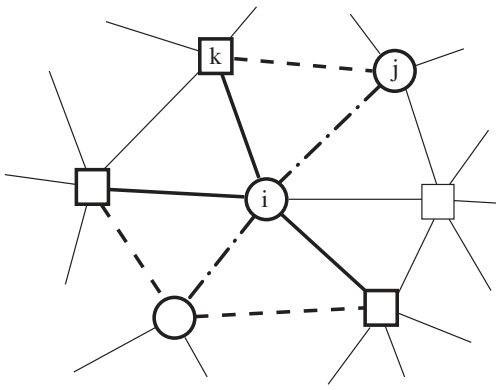

Fig. 1. Formation of interpolation molecule

4.2. Interpolation Rule

Element interpolation has been established in connection with so-called AMGe methods [6], [14]. This technique is based on a heuristic for SPD matrices that takes into account the nature of algebraically smooth error, cf. (3): Provided that a standard smoother is used, error that is slow to converge in energy norm corresponds to the lower part of the spectrum. Hence, one tries to fit interpolation to these low-energy modes, in particular, to the (near) null space components. The key idea is to construct local neighborhood matrices that represent the correct coupling between any given fine node and its interpolatory coarse (neighbor) nodes. In AMGe these neighborhood matrices are local versions of the stiffness matrix, i.e., small collections of element matrices. We propose the usage of the interpolation molecule (13), instead.

For a given f-node i let

$$M(i) = M = \begin{pmatrix} M_{ff} & M_{fc} \\ M_{cf} & M_{cc} \end{pmatrix} \tag{14}$$

be the interpolation molecule where the 2×2 block structure in (14) corresponds to the n_M^f f-nodes and the n_M^c c-nodes the molecule is based on. Then there is a bijection between the local and the global ordering of these nodes, which maps the global number i to some local number i', $1 \leq i' \leq n_M^f$. Consider now the small-sized (local) interpolation matrix

$$P_M = P = \begin{pmatrix} P_{fc} \\ I_{cc} \end{pmatrix} \tag{15}$$

associated with (14). The $n_M^f \times n_M^c$ submatrix P_{fc} produces interpolation in the f-nodes; for the c-nodes P equals the identity. Under the assumption that M is SPSD the AMGe interpolation concept can be applied directly [6], [11]:

For any vector $\mathbf{e}^T = (\mathbf{e}_f^T, \mathbf{e}_c^T) \perp \ker(M)$ we denote by

$$\mathbf{d}_f := \mathbf{e}_f - P_{fc}\mathbf{e}_c \tag{16}$$

the defect of (local) interpolation. With the objective of realizing the AMGe heuristic we choose P_{fc} to be the argument that minimizes

$$\max_{\mathbf{e} \perp \ker(M)} \frac{(\mathbf{e}_f - P_{fc}\mathbf{e}_c)^T(\mathbf{e}_f - P_{fc}\mathbf{e}_c)}{\mathbf{e}^T M \mathbf{e}}. \tag{17}$$

Using the substitutions (16) and $G := P_{fc}^T M_{ff} P_{fc} + P_{fc}^T M_{fc} + M_{cf} P_{fc} + M_{cc}$ we derive the follwing equivalence for (17):

$$\max_{\mathbf{d}_f,\mathbf{e}_c} \frac{\mathbf{d}_f^T\mathbf{d}_f}{\begin{pmatrix}\mathbf{d}_f + P_{fc}\mathbf{e}_c\\ \mathbf{e}_c\end{pmatrix}^T \begin{pmatrix}M_{ff} & M_{fc}\\ M_{cf} & M_{cc}\end{pmatrix}\begin{pmatrix}\mathbf{d}_f + P_{fc}\mathbf{e}_c\\ \mathbf{e}_c\end{pmatrix}}$$

$$= \max_{\mathbf{d}_f,\mathbf{e}_c} \frac{\mathbf{d}_f^T\mathbf{d}_f}{\langle M_{ff}(\mathbf{d}_f + P_{fc}\mathbf{e}_c), \mathbf{d}_f + P_{fc}\mathbf{e}_c\rangle + 2\langle M_{fc}\mathbf{e}_c, \mathbf{d}_f + P_{fc}\mathbf{e}_c\rangle + \langle M_{cc}\mathbf{e}_c, \mathbf{e}_c\rangle}$$

$$= \max_{\mathbf{d}_f,\mathbf{e}_c} \frac{\mathbf{d}_f^T\mathbf{d}_f}{\begin{pmatrix}\mathbf{d}_f\\ \mathbf{e}_c\end{pmatrix}^T B\begin{pmatrix}\mathbf{d}_f\\ \mathbf{e}_c\end{pmatrix}}, \tag{18}$$

where

$$B = \begin{pmatrix} M_{ff} & M_{ff}P_{fc} + M_{fc}\\ P_{fc}^T M_{ff} + M_{cf} & G\end{pmatrix} \tag{19}$$

is SPSD. Hence,

$$\min_{P_{fc}} \max_{\mathbf{d}_f,\mathbf{e}_c} \frac{\mathbf{d}_f^T\mathbf{d}_f}{\begin{pmatrix}\mathbf{d}_f\\ \mathbf{e}_c\end{pmatrix}^T B\begin{pmatrix}\mathbf{d}_f\\ \mathbf{e}_c\end{pmatrix}} = \min_{P_{fc}} \max_{\mathbf{d}_f} \frac{\mathbf{d}_f^T\mathbf{d}_f}{\min_{\mathbf{e}_c}\begin{pmatrix}\mathbf{d}_f\\ \mathbf{e}_c\end{pmatrix}^T B\begin{pmatrix}\mathbf{d}_f\\ \mathbf{e}_c\end{pmatrix}}$$

$$= \min_{P_{fc}} \max_{\mathbf{d}_f} \frac{\mathbf{d}_f^T\mathbf{d}_f}{\mathbf{d}_f^T\left[M_{ff} - (M_{ff}P_{fc} + M_{fc})G^{-1}(P_{fc}^T M_{ff} + M_{cf})\right]\mathbf{d}_f}. \tag{20}$$

Assuming that M_{ff} and G both are SPD the denominator of (20) for an arbitrary vector $\mathbf{d}_f$ is maximized and thus the minimum is attained for

$$P_{fc} \coloneqq -M_{ff}^{-1}M_{fc} \tag{21}$$

which results in $1/(\lambda_{\min}(M_{ff}))$. This motivates to choose the interpolation coefficients for node i to equal the i'-th row of (21). For a more general framework of AMG (including convergence analysis) we refer to [11].

5. Multilevel Algorithm

In this section we will describe briefly a multilevel procedure for AMGm (Algebraic MultiGrid based on computational molecules). So far we discussed how to disassemble element matrices into edge matrices that can be utilized in the coarse-grid selection process as well as in the interpolation set-up, resulting in a new two-level method. However, assuming that the individual element matrices are given for the initial (fine) grid only, we need some technique for generating coarse-edge matrices in order to enable recursion and finally define a multilevel algorithm on this basis.

5.1. Generation of Coarse-edge Matrices

The construction of edge matrices as described in Sect. 2 is such that their total contributions add up to an auxiliary matrix B_0, which agrees with the fine-grid stiffness matrix A_0 before imposing essential boundary conditions if all element matrices have only nonpositive off-diagonal entries.

If we stick to the coarse-grid operator $A_k = P_{k-1}^T A_{k-1} P_{k-1}$, obtained from the usual Galerkin approach, the only way to control the amount of fill (number of nonzero entries) in A_k is via the nonzero pattern of P_{k-1}. Since interpolation rests on "strong" edges, the goal is to reduce the number of edges on coarser levels. Therefore, in the first instance we determine which coarse-grid nodes to connect via coarse edges:

Every pair (i, j) of c-nodes is linked by a coarse edge iff there is a path of at most three successive strong fine edges connecting nodes i and j via at most two f-nodes, i.e., $\{i, j\}$, or $\{i, k_1\}$ and $\{k_1, j\}$, or $\{i, k_1\}$ and $\{k_1, k_2\}$ and $\{k_2, j\}$ are strong fine edges where k_1, k_2 are f-nodes.[2]

Note that the adjacency matrix associated with coarse edges can easily be computed by evaluating the product of three Boolean sparse matrices.

Given the set of coarse edges, we assemble the auxiliary matrix B_{k-1} from all fine-grid edge matrices and evaluate the tripple matrix product $P_{k-1}^T B_{k-1} P_{k-1}$ in those off-diagonal positions determined by adjacency along coarse-edges. This yields the off-diagonal entries for the corresponding two-by-two coarse-grid edge matrices.[3]

5.2. AMGm

Regarding the multilevel algorithm, we notice that AMGm agrees with classical AMG, except for the coarse-grid selection and the interpolation component, which are controlled by edge matrices in case of AMGm. One can also view this as involving an auxiliary problem – the one determined by the edge matrices – in the process of coarsening and interpolation. The coarse-grid operators, however, are computed via the usual Galerkin tripple matrix product, i.e., $A_k = P_{k-1}^T A_{k-1} P_{k-1}$ for all coarse levels $k = 1, \ldots, l$.

6. Numerical Experiments

For the numerical experiments presented in this section we considered the boundary-value problem

[2] The number of coarse edges can further be reduced by cutting down to paths of length at most two.

[3] For scalar elliptic PDEs every edge matrix has the form

$$E = \begin{pmatrix} a & -a \\ -a & a \end{pmatrix}$$

and thus we store only one value per edge.

$$-\nabla \cdot [C\nabla u] = f \quad \text{in } \Omega \subset \mathrm{R}^2 \tag{22}$$

$$u = g \quad \text{on } \Gamma_D \subset \partial\Omega \tag{23}$$

$$\frac{\partial u}{\partial n} = 0 \quad \text{on } \Gamma_N = \partial\Omega \backslash \Gamma_D. \tag{24}$$

Two different specifications of the matrix C, the right-hand side f, the domain Ω, and of the boundary conditions yield the considered test problems:

Problem 1:

$$C = \begin{pmatrix} 1 & 0 \\ 0 & \epsilon \end{pmatrix}, \quad f = 0, \quad \Omega = (-3, 3) \times (-3, 3) \setminus (\Omega_1 \cup \Omega_2),$$

where $\Omega_1 = (0.2, 0.3) \times (-0.5, 0.5)$, $\Omega_2 = (-0.3, -0.2) \times (-0.5, 0.5)$,

$\Gamma_N = \partial((-3, 3) \times (-3, 3))$, $\Gamma_D = \partial\Omega_1 \cup \partial\Omega_2$, *and*

$$g = \begin{cases} 1 & \text{on } \partial\Omega_1 \\ -1 & \text{on } \partial\Omega_2 \end{cases}.$$

For discretization we used a finite element space of piecewise linear functions with Lagrangian basis, where the underlying (locally refined) triangular mesh was generated using the NETGEN[4] mesh generator [20] used in NGSolve[5], see Fig. 2.

Problem 2:

$$C = \begin{pmatrix} \epsilon + (\cos\Phi)^2 & \sin\Phi\cos\Phi \\ \sin\Phi\cos\Phi & \epsilon + (\sin\Phi)^2 \end{pmatrix}, \quad f = 1, \quad \Omega = (0, 2) \times (0, 1),$$

$\Gamma_N = \{(x, y) : 0 \le x \le 2 \text{ and } y \in \{0, 1\}\}$, $\Gamma_D = \partial\Omega \backslash \Gamma_N$, *and* $g = 0$.

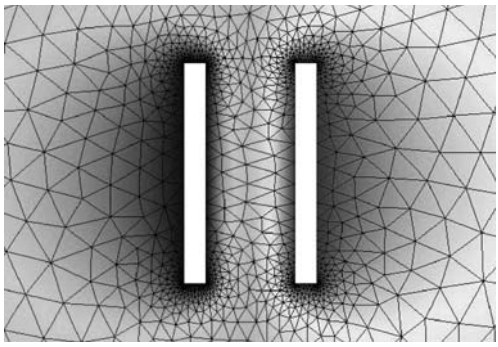

Fig. 2. Problem 1: Locally refined unstructured mesh

[4] http://www.hpfem.jku.at/netgen/index.html
[5] http://www.hpfem.jku.at/ngsolve/index.html

Again we used linear shape-functions, in order to compute a numerical solution by the finite element method. A value of $\pi/12$ was chosen for the angle Φ causing the direction of anisotropy to be not aligned with the mesh.

Note that both problems result in stiffness matrices that are not contained in the class of M-matrices. Moreover, the variation from an M-matrix increases when ϵ tends to zero, i.e., the positive off-diagonal entries gain weight in this case. This usually makes the problem harder to solve for (algebraic) multigrid methods.

In the first experiment, regarding Problem 1, we studied the convergence of the preconditioned conjugate gradient method (PCG) utilizing either a single V(1,1), V(2,2), or a W(1,1) cycle of AMGm with symmetric Gauß-Seidel pre- and post-smoothing. Table 1 contains the number of PCG iterations that reduced the residual norm by a factor 10^{-8}, the average convergence factor, as well as the *grid complexity* σ^{Ω} and the *operator complexity* σ^{A}.[6] The results for a three-dimensional analog of Problem 1 can be found in [15].

Table 1. AMGm convergence results for Problem 1

#elements		4062		16248		64992		259968	
#levels		2		4		5		7	
$\epsilon = 1$:	V(1,1)	6	0.04	9	0.12	10	0.15	11	0.18
	V(2,2)	4	0.02	7	0.06	7	0.07	8	0.09
	W(1,1)	-	-	8	0.09	8	0.09	8	0.09
	σ^{Ω}		1.40		1.61		1.64		1.64
	σ^{A}		1.70		2.31		2.48		2.52
$\epsilon = 0.5$:	V(1,1)	6	0.06	9	0.13	10	0.14	11	0.18
	V(2,2)	5	0.03	7	0.06	7	0.06	8	0.09
	W(1,1)	-	-	8	0.09	8	0.09	8	0.09
	σ^{Ω}		1.44		1.68		1.70		1.69
	σ^{A}		1.80		2.50		2.64		2.65
$\epsilon = 0.1$:	V(1,1)	7	0.06	10	0.14	10	0.15	12	0.20
	V(2,2)	5	0.03	8	0.08	8	0.08	9	0.11
	W(1,1)	-	-	9	0.11	9	0.11	8	0.09
	σ^{Ω}		1.55		1.82		1.85		1.84
	σ^{A}		1.97		2.76		2.94		2.95
$\epsilon = 0.05$:	V(1,1)	7	0.07	11	0.16	10	0.14	12	0.20
	V(2,2)	6	0.04	8	0.08	8	0.08	9	0.12
	W(1,1)	-	-	9	0.12	7	0.06	8	0.09
	σ^{Ω}		1.54		1.83		1.86		1.86
	σ^{A}		1.94		2.75		2.91		2.95
$\epsilon = 0.01$:	V(1,1)	8	0.08	13	0.23	14	0.26	15	0.29
	V(2,2)	6	0.05	10	0.14	11	0.18	12	0.22
	W(1,1)	-	-	11	0.16	10	0.15	8	0.09
	σ^{Ω}		1.53		1.63		1.88		1.88
	σ^{A}		1.89		2.68		2.85		2.91

[6] σ^{Ω} is the ratio of the total number of points on all grids to that on the fine grid; σ^{A} is the ratio of the total number of nonzeros in all matrices to that in the fine-grid matrix.

For the second experiment, regarding Problem 2, using a quasiuniform mesh with decreasing mesh size, i.e., 49152, 196608, and 786432 elements, we decided to compare the performance of AMGm to that of classical AMG (both used as a preconditioner for conjugate gradients). For comparison (in terms of computing time), we used the commercial software package FEMLAB version 3.1.[7], which implements the classical AMG algorithm.

The iteration was initialized with the zero start vector in all experiments. In Table 2, we list the number of PCG iterations that reduced the norm of the initial residual by a factor 10^{-6}. The number of levels was chosen such that the number of nodes on the coarsest grid was less or equal to one thousand. We used the same value for the threshold parameter θ, i.e., $\theta = 1/4$, for classical AMG and AMGm. The solution time provided in the respective right column of Table 2 in each case includes the set-up time for the AMG(m) components. All computations were performed on a 2.4 GHz Linux-PC.

7. Concluding Remarks

The application of any AMG method splits into a set-up phase and a solution phase. Hence, solving a linear system (1) for a single right-hand side, the computational costs of these two phases have to be balanced properly.

Table 2. Performance comparison for Problem 2

49152 elements (24833 degrees of freedom)												
	AMG						AMGm					
ϵ	V(1,1)		V(2,2)		W(1,1)		V(1,1)		V(2,2)		W(1,1)	
1.0	15	1.38	13	1.48	10	1.49	12	1.31	9	1.30	10	1.43
0.5	15	1.57	13	1.67	11	1.83	9	1.35	8	1.42	7	1.49
0.1	20	1.70	18	2.11	14	1.92	11	1.43	9	1.48	9	1.55
0.05	23	1.81	21	1.93	16	2.03	12	1.49	10	1.51	9	1.57
0.01	34	2.10	32	2.32	24	2.43	15	1.54	13	1.62	10	1.56
196608 elements (98817 degrees of freedom)												
	AMG						AMGm					
ϵ	V(1,1)		V(2,2)		W(1,1)		V(1,1)		V(2,2)		W(1,1)	
1.0	36	12.9	34	14.5	19	13.1	12	6.2	10	6.6	9	6.4
0.5	20	12.8	19	15.0	12	14.0	11	7.0	9	7.4	8	7.4
0.1	28	13.6	26	15.1	17	15.7	12	7.3	10	7.7	9	7.7
0.05	31	13.8	29	15.3	20	15.8	13	7.5	11	8.0	9	7.6
0.01	44	15.4	41	16.6	30	20.8	17	7.7	14	8.3	10	7.4
786432 elements (394241 degrees of freedom)												
	AMG						AMGm					
ϵ	V(1,1)		V(2,2)		W(1,1)		V(1,1)		V(2,2)		W(1,1)	
1.0	79	189	72	202	35	197	13	29	11	31	9	29
0.5	56	251	51	253	20	245	11	32	9	34	8	33
0.1	52	245	50	252	21	246	12	33	10	35	9	36
0.05	60	244	56	255	27	250	14	35	12	38	9	35
0.01	60	244	56	256	36	264	21	38	16	40	10	33

[7] http://www.comsol.com

In fact, the set-up of AMGm took approximately half the (total) time reported in Table 2. The improved robustness is achieved by controlling the coarse-grid selection and the interpolation component via edge-matrices. Using AMGm instead of classical AMG preconditioning, a faster convergence and thence a shorter solution time was obtained. In particular, as one would expect, the iteration count for the W(1,1) cycle is (almost) independent of the mesh size (for both problems). The grid- and operator complexity incurred by AMGm are comparable to those of classical AMG, cf. [18].

Future investigations will deal with the generalization of the presented AMGm methodology to cover also systems of PDEs.

Acknowledgements

The authors would like to thank Josef Synka who carried out the computations with FEMLAB, which served for the presented performance comparison. This work was supported by the Austrian Academy of Sciences and by the Austrian Science Fund (FWF) in the frame of special research area F013.

References

[1] Axelsson, O.: Iterative solution methods. Cambridge University Press 1994.
[2] Brandt, A.: Algebraic multigrid theory: the symmetric case. Preliminary Proc. Int. Multigrid Conf., Copper Mountain, CO, 1983.
[3] Brandt, A.: Algebraic multigrid theory: the symmetric case. Appl. Math. Comput. *19*, 23–56 (1986).
[4] Brandt, A., McCormick, S. F., Ruge, J. W.: Algebraic multigrid (AMG) for automatic multigrid solutions with applications to geodetic computations. Report, Inst. for Computational Studies, Fort Collins, CO, 1982.
[5] Brandt, A., McCormick, S. F., Ruge, J. W.: Algebraic multigrid (AMG) for sparse matrix equations. In: Sparsity and its applications (D. J. Evans, ed.). Cambridge: Cambridge University Press 1985, pp. 257–284.
[6] Brezina, M., Cleary, A. J., Falgout, R. D., Henson, V. E., Jones, J. E., Manteuffel, T. A., McCormick, S. F., Ruge, J. W.: Algebraic multigrid based on element interpolation (AMGe). SIAM J. Sci. Comput. *22*, 1570–1592 (2000).
[7] Briggs, W. L., Henson, V. E., McCormick, S. F.: A multigrid tutorial, 2nd edition. Philadelphia: SIAM Books 2001.
[8] Chan, T. F., Vanek, P.: Detection of strong coupling in algebraic multigrid solvers. In: Multigrid methods VI, vol. 14. Berlin: Springer 2000, pp. 11–23.
[9] Chartier, T., Falgout, R. D., Henson, V. E., Jones, J. Manteuffel, T., McCormick, S., Ruge, J., Vassilevski, P. S.: Spectral AMGe (ρAMGe). SIAM J. Sci. Comput. *25*, 1–26 (2004).
[10] Cleary, A. J., Falgout, R. D., Henson, V. E., Jones, J. E., Manteuffel, T. A., McCormick, S. F., Miranda, G. N., Ruge, J. W.: Robustness and scalability of algebraic multigrid. SIAM J. Sci. Stat. Comput. *21*, 1886–1908 (2000).
[11] Falgout, R. D., Vassilevski, P. S.: On generalizing the algebraic multigrid framework. SIAM J. Numer. Anal. *42*, 1669–1693 (2004).
[12] Haase, G., Langer, U., Reitzinger, S., Schöberl, J.: Algebraic multigrid methods based on element preconditioning. Int. J. Comp. Math. *78*, 575–598 (2004).
[13] Henson,V. E., Vassilevski, P.: Element-free AMGe: general algorithms for computing the interpolation weights in AMG. SIAM J. Sci. Comput. *23*, 629–650 (2001).
[14] Jones, J. E., Vassilevski, P.: AMGe based on element agglomeration. SIAM J. Sci. Comput. *23*, 109–133 (2001).
[15] Kraus, J. K.: On the utilization of edge matrices in algebraic multigrid. Lecture Notes in Computer Science. Proc. 5th Int. Conf. on "Large-Scale Scientific Computations", Sozopol, Bulgaria, June 6–10, 2005 (to appear).
[16] Langer, U., Reitzinger, S., Schicho, J.: Symbolic methods for the element preconditioning technique. In: Proc. SNSC Hagenberg (Langer, U., Winkler, F., eds.). Springer 2002.

[17] Mandel, J., Brezina, M., Vaněk, P.: Energy optimization of algebraic multigrid bases. Computing *62*, 205–228 (1999).
[18] Ruge, J. W., Stüben, K.: Efficient solution of finite difference and finite element equations by algebraic multigrid (AMG). In: Multigrid methods for integral and differential equations (Paddon, D.J., Holstein, H., eds.). The Institute of Mathematics and Its Applications Conference Series. Oxford: Clarendon Press 1985, pp. 169–212.
[19] Ruge, J. W., Stüben, K.: Algebraic multigrid (AMG). In: Multigrid methods. Frontiers in Applied Mathematics, Vol. 3 (McCormick, S.F., ed.). Philadelphia: SIAM 1987, pp. 73–130.
[20] Schöberl, J.: NETGEN – an advancing front 2D/3D-mesh generator based on abstract rules. Comput. Visual. Sci. 41–52 (1997).
[21] Stüben, K.: Algebraic multigrid (AMG): experiences and comparisons. Appl. Math. Comput. *13*, 419–452 (1983).
[22] Tuminaro, R. S., Tong, C.: Parallel smoothed aggregation multigrid: aggregation strategies on massively parallel machines. Report, Sandia National Laboratories, 2000.
[23] Varga, R. S.: Matrix iterative analysis. Prentice-Hall 1962.
[24] Vaněk, P., Brezina, M., Mandel, J.: Convergence of algebraic multigrid based on smoothed aggregation. Numer. Math. *88*, 559–579 (2001).
[25] Vaněk, P., Mandel, J., Brezina, M.: Algebraic multigrid based on smoothed aggregation for second- and fourth-order problems. Computing *56*, 179–196 (1996).
[26] Wan, W. L., Chan, T. F., Smith, B.: An energy-minimizing interpolation for robust multigrid methods. Siam J. Sci. Comput. *21*, 1632–1649 (2000).

J. K. Kraus
Johann Radon Institute for Computational and Applied Mathematics
Altenbergerstraße 69
4040 Linz
Austria
e-mail: johannes.kraus@oeaw.ac.at

J. Schicho
Johann Radon Institute for Computational and Applied Mathematics
Altenbergerstraße 69
4040 Linz
Austria
e-mail: josef.schicho@oeaw.ac.at