Konrad-Zuse-Zentrum
für Informationstechnik Berlin

Takustraße 7
D-14195 Berlin-Dahlem
Germany

RUDOLF BECK

Graph-Based Algebraic Multigrid for Lagrange-Type Finite Elements on Simplicial Meshes

Preprint SC 99-22 (July 1999)

Graph-Based Algebraic Multigrid for Lagrange-Type Finite Elements on Simplicial Meshes

Rudolf Beck

Abstract

We present an algebraic multigrid preconditioner which uses only the graphs of system matrices. Some elementary coarsening rules are stated, from which an advancing front algorithm for the selection of coarse grid nodes is derived. This technique can be applied to linear Lagrange-type finite element discretizations; for higher-order elements an extension of the multigrid algorithm is provided. Both two- and three-dimensional second order elliptic problems can be handled. Numerical experiments show that the resulting convergence acceleration is comparable to classical geometric multigrid.

Key words: Algebraic multigrid, mesh coarsening, preconditioning

AMS(MOS) subject classifications: 65N55, 65F10

1 Introduction

During the past decades multigrid methods have come up as efficient tools for the numerical solution of partial differential equations. They mainly rely on sequences of nested grids endowed with appropriate transfer or interpolation operators. The grids are often obtained by the successive refinement of an initial mesh. As long as the system matrix belonging to the latter one can be factorized, e.g. by a direct sparse matrix solver, multigrid methods may be vastly superior to any other solution technique.

However, in many situations the initial mesh may be quite large due to the complicated structure of the domain to be discretized. So it is desirable to construct grid families by taking the opposite direction, i.e. coarsening the initial mesh in subsequent steps until a basic grid with a small number of unknowns is obtained. As such procedures mainly rely on the matrix of the linear equation system, the name algebraic multigrid has been assigned to them.

Meanwhile a considerable amount of research has been done in this field [5, 6] and a large variety of numerical problems has been tackled. The resulting algorithms are often very sophisticated and based on quite different approaches, like matrix-weighted interpolation methods [7, 9, 14], incomplete LU-factorization [1, 2], or aggregation [3, 10, 11]. In most schemes the values of the matrix entries are taken into account for mesh coarsening and grid transfer.

In our approach we simply use the matrix graphs for coarsening. In the context of linear nodal finite elements it is quite obvious that the graph defined by the system matrix of a problem resembles the geometric structure of the respective finite element discretization. The diagonal entries of the system matrix are the nodes of the graph, representing the vertices of the mesh; off-diagonal matrix entries contribute the edges.

The basic idea of our coarsening algorithm is to select a favourably distributed subset of the grid (or matrix) nodes, which are to become the nodes of the coarse grid. A simple, quasi-linear interpolation will be employed for the grid transfer. The scheme works both in two and three space dimensions. On regular grids with homogeneous materials the algebraic preconditioner operates in a way very similar to classical geometric multigrid.

The extension to applications where several degrees of freedom are located at each mesh vertex is quite straightforward if the different species are approximated by the same shape functions. However, it is evident that the proposed algorithm cannot cope efficiently with strongly anisotropic media, as no matrix information with regard to directed coupling strength is exploited.

2 Creation of Nested Finite Element Spaces

Suppose that we have a grid, say on level l, and a coarser one on level $l-1$. If the associated finite element spaces are nested, i.e. $V^{l-1} \subset V^l$, then we can set up an interpolation or prolongation operator P^l which leaves any function $u^{l-1} \in V^{l-1}$ invariant after the transfer $P^l : V^{l-1} \to V^l$. In matrix-vector notation we write

$$u_i^l = P_{ij}^l \, u_j^{l-1} \tag{1}$$

with the subscripts i and j denoting the coefficients of the vectors.

Equivalently we may say that each finite element basis function φ_k^{l-1} of the coarse space is defined by a linear combination of fine grid basis functions:

$$\varphi_k^{l-1} \;=\; \left(P_{km}^l\right)^T \varphi_m^l \; . \tag{2}$$

The superscript T denotes the transpose. On claiming invariance for any linear or bilinear form after a coarse-to-fine transfer of vectors, some basic algebra reveals that the coefficients r_m^l of a linear form defined in V^l (i.e. of a right-hand-side vector in algebraic notation) are restricted to V^{l-1} by

$$r_k^{l-1} \;=\; \left(P_{km}^l\right)^T r_m^l \; . \tag{3}$$

The coefficients of bilinear forms are related by the so-called Galerkin product

$$A_{ij}^{l-1} \;=\; \left(P_{ik}^l\right)^T A_{km}^l \; P_{mj}^l \; . \tag{4}$$

In the context of classical multigrid the operator P^T is called the canonical restriction [8].

Thus in principle a nested, purely algebraic coarsening scheme can be devised by defining suitable coarse grid functions via eq. (2) and applying this procedure recursively until a space of sufficiently low dimension is created, where an exact solution can be obtained at low numerical cost. By utilizing eq. (1), (3), and (4) a multigrid algorithm working on a sequence of nested spaces is obtained. However, the basic question remains: how can we construct transfer operators P^l that provide a good numerical performance?

3 The Coarsening Algorithm for Linear Elements

In a first step we will present a scheme for linear nodal elements. The case of higher-order elements will be discussed in the following section.

Like in a classical multigrid algorithm, we rely on the concept of smoothing (which will be done in our case by Gauss-Seidel relaxation). The motivation for our coarsening algorithm stems from the following demands:

(a) The number of unknowns should be reduced substantially in every coarsening step.

(b) The coarse grid matrices should preserve the sparsity pattern of the fine grid matrix, that is, coarse grid basis functions should retain a local support and a restricted overlap with neighbouring ones.

(c) If we assume that we had an exact coarse grid solution, then, after the grid transfer, a relaxation sweep on the finer grid should efficiently reduce the high-frequency components of the error.

(d) Coarse grid solutions should remain smooth after a coarse-to-fine grid transfer.

It is obvious that (a) and (b) are essential requirements for keeping the number of entries small in coarse grid matrices. Otherwise each smoothing step might become unbearably costly on coarse grids. With requirements (c) and (d) we try to guarantee an efficient action of a smoother, which typically tackles only high-frequency components of the error. Furthermore, (d) implicitly claims that the smoothness of basis functions should increase on coarsening, thus yielding equation systems with smaller condition numbers.

Guided by the above demands, we have devised the following coarsening strategy:

On any given grid, the nodes will be divided into two disjoint sets: the *master nodes*, which are to become the nodes of the coarse grid, and the *slave nodes*, which will be dropped. There are three rules for the selection of master nodes and the definition of the coarse-to-fine transfer operator P:

(1) No master node may be directly connected to another master node (two nodes i and j are connected, if there exists a matrix entry a_{ij}).

(2) There should be as many master nodes as possible.

(3) The values of all master nodes are transferred with weight 1. The value for a slave node s is interpolated from the n_s master nodes it is connected to, where each master node contributes with weight $\frac{1}{n_s}$ (examples are given below).

Remark: Of course, all nodes not declared to be masters will be slave nodes. An exception appears on the finest grid, where the nodes on Dirichlet boundaries are not taken into consideration.

Rules (1) and (2) are aimed at satisfying the requirements (a) to (c). The master nodes are dispersed in a quasi-uniform manner on the grid, thus trying to imitate geometric coarsening (at least on regular meshes). Each slave node is coupled to at least one master node, which is essential with regard to (c). With rule (3) we try to maintain the smoothness claimed in requirement (d); in particular on regions with constant solution the transfer is exact. Thus the kernel of the gradient is preserved.

We do not aim at an algorithm that satisfies rule (2) exactly; this might become a too costly procedure. Instead, we will be content with a scheme that achieves a nearly optimal distribution of the master nodes. So we present an "advancing front" algorithm, which proceeds by examining the immediate neighbours of the nodes at the present front:

The coarsening algorithm for selecting master and slave nodes on a grid:

We denote by N the set of all nodes of the grid, i.e. of the matrix graph. Dirichlet nodes are not taken into account. M is the set of master nodes and S the set of slave nodes. Nodes neither in M nor in S form the set of remaining nodes R.

(1) Pick one node i of the grid, which will become the first master node: $M = \{\, i \,\}$. Set $S = \{\ \}$ and $R = N \setminus \{\, i \,\}$.

(2) Pick all those nodes of R which are coupled to a node in M or S.
These nodes form a temporary set F called the active front:
$F = \{f_1, \ldots, f_n\},\ R \leftarrow R \setminus F$.

(3) Subsequently determine the status for the nodes $f_i \in F$:

- if f_i is connected to any node in M, move it to the set of slave nodes: $S \leftarrow S \cup \{f_i\}\,,\ F \leftarrow F \setminus \{f_i\}\,.$
- otherwise f_i is to become a master node: $M \leftarrow M \cup \{f_i\}\,,\ F \leftarrow F \setminus \{f_i\}\,.$

(4) Repeat steps (2) and (3) until R is empty.

Two examples are shown in figure 1. For the regular grid on the left hand side the algorithm has produced a master node distribution which is equivalent to regular geometric coarsening. Here the interpolation weights for all slave nodes are $\frac{1}{2}$.

If we retain the numbering of the master nodes (as shown in figure 1) on the coarse grid, the prolongation matrices for our examples read

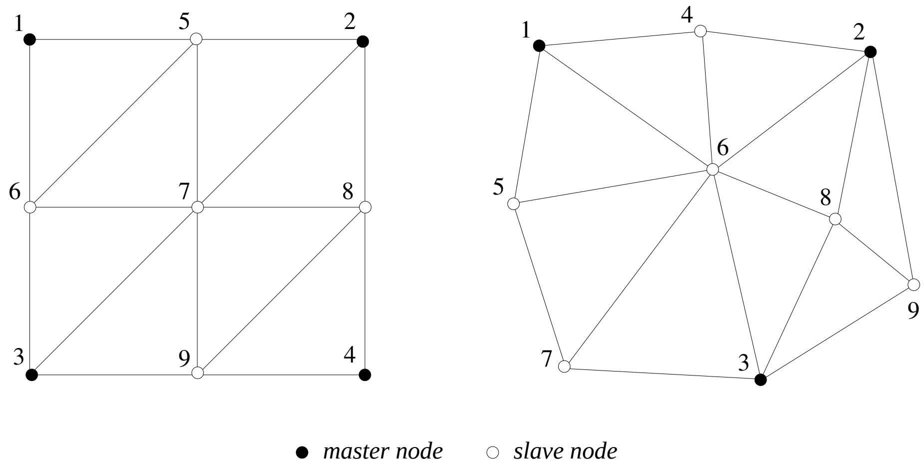

Figure 1: *Distribution of master and slave nodes on a regular two-dimensional grid (left) and an unstructured one. For practical reasons, the lowest numbers have been assigned to the master nodes, which will be retained on the coarse grid (observe that in principle the numbering is arbitrary).*

$$
P_{reg} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \end{pmatrix}, \qquad P_{unstruct} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ \frac{1}{2} & \frac{1}{2} & 0 \\ 1 & 0 & 0 \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ 0 & 0 & 1 \\ 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & \frac{1}{2} & \frac{1}{2} \end{pmatrix}.
$$

The extension to systems of equations, where several degrees of freedom (henceforth called species) are located at each node, is straightforward. In this case only one species is chosen for the selection of master and slave nodes; all entries of the matrix which connect it to another species are neglected. The resulting transfer operator is then applied to all species in the same manner. This technique keeps the storage requirements for the transfer operators small, too.

We remark that for systems it may be advisable to employ Block-Gauss-Seidel smoothing, where each block is formed by all the species residing at a particular grid vertex.

4 Algebraic Multigrid

Now we are able to formulate a recursive algebraic multigrid preconditioner for linear elements. It is constructed in a symmetric manner, using a V-cycle with varying numbers of smoothing steps on the different levels. Numerical experiments show that increasing the number of smoothing steps n^l on coarser levels l may improve the performance. On the fine grid, we smooth just once and increase n^l by one on every subsequent coarser grid, i.e. if L denotes the finest level, n^l is given by $n^l = L - l + 1$.

We stop coarsening if the number of unknowns is below 500. The coarsest level is called level 1; on this level the system matrix is factorized by sparse LU-decomposition. All coarse grid matrices are computed by the Galerkin product (4).

$\mathbf{AMG}(A^l, x^l, r^l)$: V-cycle for an approximate solution of $A^l x^l = r^l$
(linear elements)

(1) *Pre-smoothing: relax n^l times with forward Gauss-Seidel on $A^l x^l = r^l$*
(2) *Restrict new residual:* $r^{l-1} \leftarrow \left(P^l\right)^T (r^l - A^l x^l)$
(3) *If* $l-1 = 1$: *solve* $A^1 x^1 = r^1$ *by forward-backward substitution*
else : $x^{l-1} \leftarrow 0$, *call* $\mathbf{AMG}(A^{l-1}, x^{l-1}, r^{l-1})$
(4) *Prolongate coarse grid solution:* $x^l \leftarrow x^l + P^l x^{l-1}$
(5) *Post-smoothing: relax n^l times with backward Gauss-Seidel on $A^l x^l = r^l$*

Extension to elements of higher order:

In this case the fine grid matrix becomes denser and the associated graph more intricate. Appropriate transfer operators now should interpolate with higher-order polynomials, otherwise local information may be lost. As this appears to be a difficult task, we simply propose an initial restriction to the space of linear elements, where the above procedures can be applied without any alteration. At least for quadratic elements, such a procedure should not suffer much deterioration.

If hierarchical elements are employed, the additional overhead becomes minimal. Here the adequate transfer operation between linear and higher-order space is mere injection; i.e. for (1), (3), and (4) only the vector and matrix entries belonging to the linear space have to be copied.

In the following we use the superscripts H for the entities of the higher-order space and L for the space of linear elements, which both “live” on the finest grid. The associated transfer operator is denoted by $P^H : V^L \rightarrow V^H$. The complete preconditioner now reads:

AMG_H (A^H, x^H, r^H) : V-cycle for $A^H x^H = r^H$ (higher-order elements)

(1) Pre-smoothing: relax n^H *times with forward Gauss-Seidel on* $A^H x^H = r^H$
(2) Restrict new residual to linear space: $r^L \leftarrow \left(P^H\right)^T (r^H - A^H x^H)$
(3) $x^L \leftarrow 0$, *call* $\mathbf{AMG}(A^L, x^L, r^L)$
(4) Prolongate linear solution: $x^H \leftarrow x^H + P^H x^L$
(5) Post-smoothing: relax n^H *times with backward Gauss-Seidel on* $A^H x^H = r^H$

The number of smoothing steps n^H should be chosen with regard to the order of the space. In all our numerical experiments with quadratic elements $n^H = 1$ gave the best results.

5 Numerical Examples

In this section we present some numerical examples for problems in two and three space dimensions. The initial triangulations of the domains are refined either uniformly or adaptively in order to demonstrate the performance of the algebraic multigrid preconditioners both on regular and unstructured grids. We use a conjugate gradient algorithm for the basic solver, which is endowed with three different preconditioners:

- SGS : single-level symmetric Gauss-Seidel relaxation
- MG : geometric multigrid
  (symmetric V-cycle with one pre- and one post-smoothing step)
- AMG: algebraic multigrid (AMG and AMG_H, respectively)

In all computations the guess for the initial solution is $u = 0$. The iteration is terminated if the euclidian norm of the residual has dropped by ten orders of magnitude, i.e.

$$\frac{|\,\mathbf{r}\,|}{|\,\mathbf{r}_0\,|} < 10^{-10} .$$

In the first two columns of our tables below we list the refinement levels of the grids and the corresponding numbers of nodes (level 1 is assigned to the initial grids). Furthermore, the total iteration counts and the processor times are given. For the AMG-preconditioners also the numbers of levels created by algebraic coarsening and the respective setup times are listed in brackets. In the setup phases most time is spent in the Galerkin procedures required for computing the coarse grid matrices.

In exp. 1 and exp. 4 calculations with hierarchical quadratic shape functions are included. Here the extended version AMG_H is employed, the smoothing parameter n_H is set to 1. The very like restriction to the linear space is used for the geometric multigrid preconditioner, too.

Exp. 1: Heat conduction on a domain with large jumps in the material coefficients

We consider the domain $\Omega = [0,1]^2$, where we solve the equation

$$\nabla \left(\kappa(x)\,\nabla u\right) \;=\; 1$$

with homogeneous Dirichlet boundary conditions. We split the domain into two disjoint sub-regions Ω_1 and Ω_2; Ω_1 is defined by $\Omega_1 = [0.25, 0.75]^2 \setminus [0.375, 0.625]^2$. The conductivity κ is given by

$$\kappa(x) \;=\; \begin{cases} 1^{-6} & \text{on } \Omega_1 \\ 1.0 & \text{on } \Omega_2 \end{cases} \quad ,$$

thus Ω_1 forms a rectangular ring with very low conductivity.

We solve the problem thrice: in the first run, grids with different mesh spacings are created by uniform refinement. Next we use adaptive refinement; the initial and an adaptively refined grid are shown in figure 2. Finally, elements with quadratic shape functions are employed. The results are listed in the tables 1 to 3.

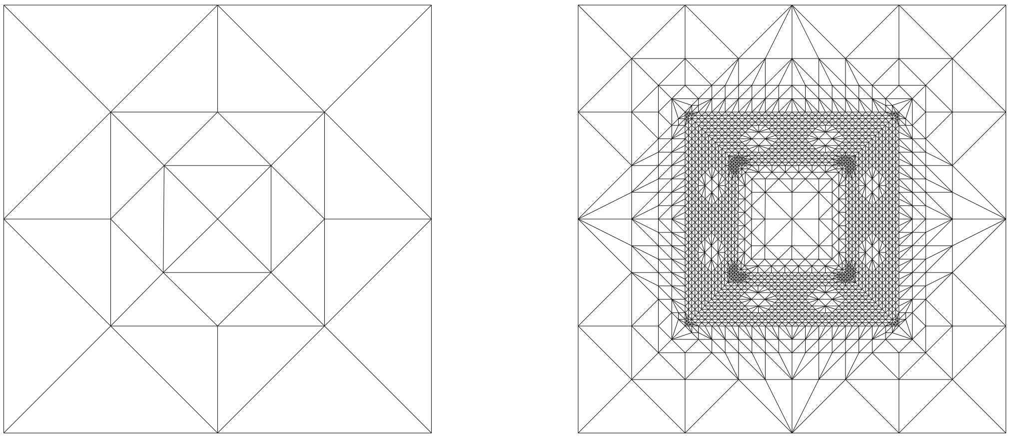

Figure 2: *Initial and adaptively refined grid of exp. 1.*

Level	Nodes	#Iter				CPU [sec]			
		SGS	MG	AMG	(Levels)	SGS	MG	AMG	(Setup)
5	4161	155	13	16	(3)	1.0	0.2	0.3	(0.1)
6	16513	391	14	18	(4)	13.6	1.2	1.7	(0.3)
7	65793	781	14	21	(5)	130	5.2	8.8	(1.2)
8	262657	1558	15	24	(6)	1040	26.9	42.8	(4.8)

Table 1: *Iteration counts and CPU-times for exp. 1 on quasi-uniform triangulations.*

Level	Nodes	#Iter				CPU [sec]			
		SGS	MG	AMG	(Levels)	SGS	MG	AMG	(Setup)
6	1925	101	10	19	(2)	0.3	0.1	0.1	(0.1)
8	7595	174	11	18	(3)	2.5	0.8	0.7	(0.3)
10	29686	318	11	22	(4)	23	4.5	4.3	(0.8)
12	116153	628	12	25	(5)	190	24	21	(2.8)

Table 2: *Results for exp. 1 on triangulations created by local mesh refinement.*

Level	Nodes	#Iter				CPU [sec]			
		SGS	MG	AMG	(Levels)	SGS	MG	AMG_H	(Setup)
5	2081	62	28	20	(2)	0.3	0.3	0.2	(0.03)
7	8797	108	31	22	(3)	2.7	1.9	1.1	(0.07)
9	36609	186	29	22	(4)	23	9	5.2	(0.25)
11	149633	364	29	22	(4)	189	43	23	(1.1)

Table 3: *Exp. 1 with quadratic elements (triangulations created by local mesh refinement).*

As to be expected, the solver with plain SGS-preconditioning deteriorates in all test cases. The geometric multigrid preconditioner remains practically stable, whereas AMG exhibits a slight increase of iteration counts in the first two runs.

As table 2 reveals, the algebraic scheme matches the geometric multigrid in the adaptive case with regard to cpu-time, although it requires more iterations. This is due to the fact that only few coarse grid levels are created, whereas MG has to cope with a substantially higher number of geometric levels. The applied technique of local smoothing cannot alleviate this drawback of MG.

This behaviour is even more apparent for hierarchical quadratic elements (table 3): the algebraic multigrid preconditioner is clearly more efficient. Looking at the iteration counts, we may conclude that the restriction to the linear space affects the convergence behaviour of MG worse than AMG. We have no explanation for this effect. Note that the setup time for AMG is practically negligible here, as there are substantially fewer numbers of unknowns in the linear spaces where the algebraic coarsening is carried out.

Exp. 2: Skin effect at high frequency

In our second experiment we solve the stationary skin-effect equation, which models the penetration of magnetic fields in a conductor at high frequencies. The problem is formulated in terms of the complex magnetic vector potential A. We consider the two-dimensional case, where only the z-component of A has to be taken into account [12]:

$$\nabla A_z(\frac{1}{\mu}\nabla A_z) \; - \; i\,2\pi\nu\sigma\, A_z \; = \; j_z \; .$$

Here μ denotes the magnetic permeability, i the imaginary unit, ν the frequency, and j_z an impressed current density. We use the domain of the first experiment. The region Ω_1 defines the conducting material, where we assume an electric conductivity $\sigma = 10^3$ and to which we assign a current density $j_z = 1$. The remaining area is filled by some dielectric material; on the total domain Ω the permeability μ is set to 1. The applied frequency is $\nu = 1$. Contrary to the first experiment, we now encounter a situation with a jump of material coefficients in the mass term.

For the given parameters, the penetration depth is very small compared to the thickness of the conductor (region Ω_1). As the grids are constructed adaptively in our computations, we obtain triangulations with strongly local refinement along the outer conductor edges. Plots of a refined mesh and the contours of A_z are shown in figure 3.

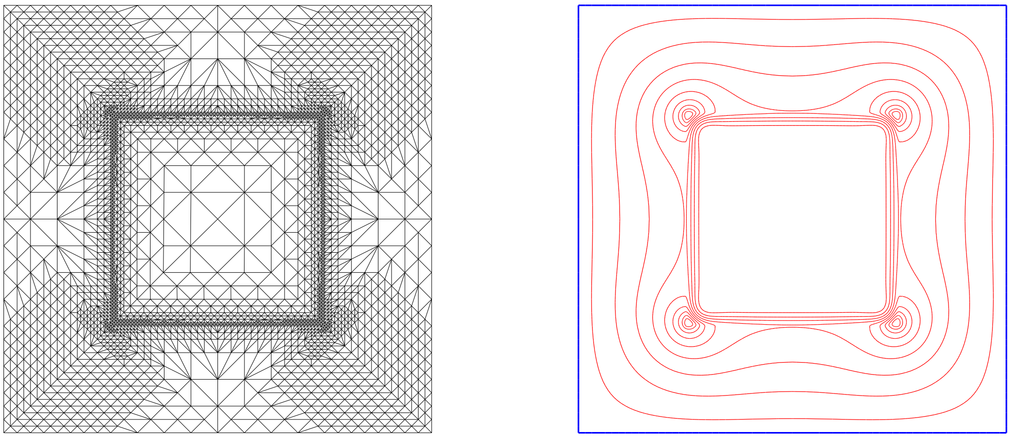

Figure 3: *Adaptively refined grid of Exp. 2 (left) and contour lines of the vector potential A_z.*

Level	Nodes	#Iter SGS	MG	AMG	(Levels)	CPU [sec] SGS	MG	AMG	(Setup)
6	1125	47	8	12	(2)	0.22	0.14	0.12	(0.18)
8	7233	128	10	19	(3)	4.5	1.5	1.7	(0.37)
10	37345	279	11	18	(4)	60	9.4	9.7	(1.1)
12	166674	612	11	23	(5)	590	55	59	(4.2)

Table 4: *Iteration counts and CPU-times for exp. 2 (adaptively refined triangulations).*

Looking at the results of table 4, we encounter a situation similar to the one documented in table 2 of the previous experiment. Although AMG needs more iterations, its overall performance is comparable to MG. The explanation given above applies here as well.

Exp 3: Helmholtz equation

In this experiment we consider the stationary scalar wave equation

$$\Delta u + k^2 u = f . \qquad (5)$$

Here $k = \frac{2\pi}{c}\nu$ is the wave number, c denotes the phase velocity, and ν the frequency.

It is well-known that the arising indefinite equation systems are also a challenge for classical multigrid methods. As modes with negative eigenvalues cannot be tackled by smoothing iterations, it is essential to resolve them on the coarsest grid [4]. If this condition cannot be fulfilled, a smoother which only works with the elliptic part of the equation (the Laplacian in eq. (5)) may be employed which is able to cope with the positive spectral modes [13].

We solve (5) on the unit square with homogeneous Dirichlet boundary conditions. The values for the above coefficients are $c = 1$ and $f = 1$ on the whole domain; the frequency ν is set to 3. Plots of the initial grid and the solution are displayed in figure 4. By uniform refinement we produce a sequence of regular grids; adaptive refinement does not make much sense for this problem.

In a first run we factorize the system matrices on the coarsest grids both for geometric and algebraic multigrid. These coarse grids comprise approximately 500 nodes (observe that on regular meshes algebraic coarsening is very similar to the geometric case). As the results in table 5 reveal, the negative modes are resolved quite well. Both for MG and AMG the convergence rates are stable; the plain SGS scheme deteriorates with each additional refinement level.

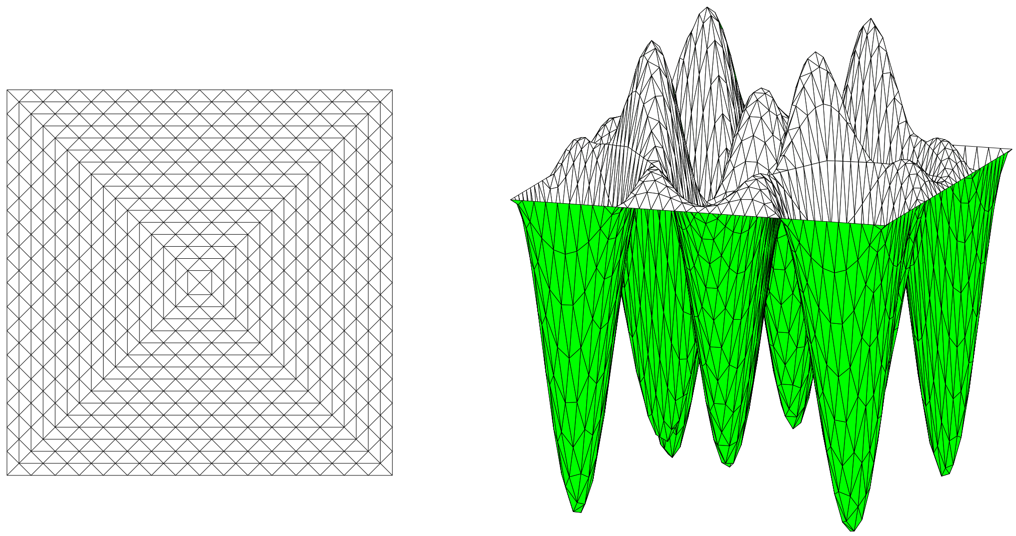

Figure 4: *Exp. 3: Grid and solution for the Helmholtz equation.*

In order to simulate a situation where the coarse grids do not resolve the negative eigenmodes (or are too large for factorization), we carried out a second run. Now all preconditioners rely only the elliptic part of equation (5). Table 6 shows the

Level	Nodes	#Iter				CPU [sec]			
		SGS	MG	AMG	(Levels)	SGS	MG	AMG	(Setup)
2	2113	218	19	26	(2)	0.7	0.16	0.18	(0.1)
3	8321	405	21	31	(3)	5.9	0.75	1.27	(0.2)
4	33025	752	22	31	(4)	57	3.9	5.9	(0.6)
5	131585	1385	20	31	(5)	460	15	27	(2.3)

Table 5: *Iteration counts and CPU-times for exp. 3 (Helmholtz equation). The coarse grid matrices are factorized (MG and AMG).*

Level	Nodes	#Iter				CPU [sec]			
		SGS	MG	AMG	(Levels)	SGS	MG	AMG	(Setup)
2	2113	226	100	108	(2)	0.75	0.86	0.88	(0.1)
3	8321	407	104	109	(3)	6.2	4.4	4.5	(0.2)
4	33025	752	106	107	(4)	59	20	21	(0.6)
5	131585	1409	106	106	(5)	465	86	90	(2.3)

Table 6: *Exp. 3 revisited. Here only the elliptic part of the system matrix is employed for smoothing.*

effects: SGS is nearly unaffected, whereas both multigrid preconditioners yield substantially worse convergence rates. However, they remain fairly stable and are still superior to SGS for large node numbers.

Exp. 4: Three-dimensional region with a crack

In this example we solve Laplace's equation $\Delta u = 0$ on a domain with a crack. Dirichlet boundary conditions are prescribed at the upper and lower fronts above the crack as indicated in figure 5. The remaining boundaries are subject to homogeneous Neumann conditions. The angle of nearly $360°$ along the crack is accountable for a solution with singular behaviour.

Like in our first experiment we solve the problem both with uniform and adaptive mesh refinement. A third run employs hierarchical quadratic elements. The results are shown in the tables 7 to 9.

For the sequence of regularly refined grids, the results are comparable to those of exp. 1. However, as it is typical for the three-dimensional case, substantially more unknowns must be encountered until multigrid preconditioners show up considerably faster than single-level smoothers.

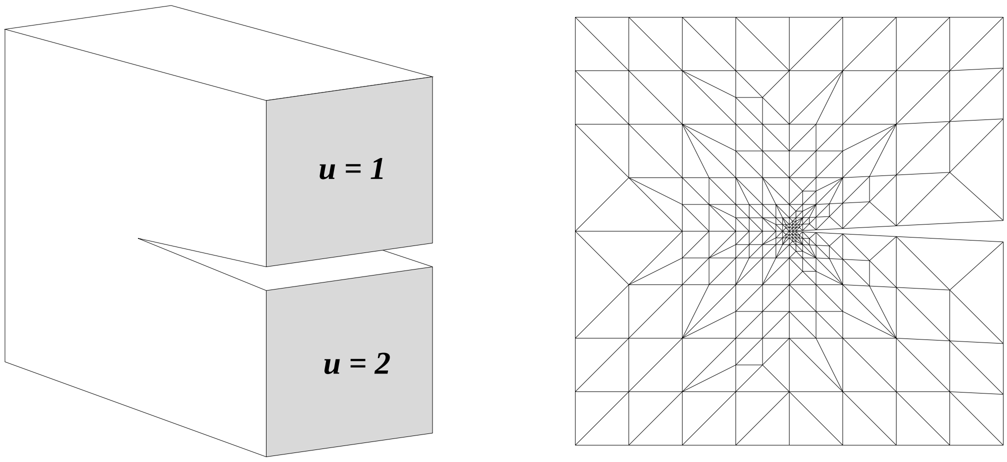

Figure 5: *3D-region with crack (Exp. 4): structure with applied boundary conditions (left) and cross-section of an adaptively refined grid.*

Level	Nodes	#Iter				CPU [sec]			
		SGS	MG	AMG	(Levels)	SGS	MG	AMG	(Setup)
5	9537	80	14	15	(3)	2.3	0.84	1.1	(0.5)
6	70785	150	15	19	(4)	40	8.0	12.6	(4.1)
7	545025	298	15	22	(5)	943	91	172	(37)

Table 7: *Exp. 4 on uniform triangulations (3D-domain with a crack).*

Level	Nodes	#Iter				CPU [sec]			
		SGS	MG	AMG	(Levels)	SGS	MG	AMG	(Setup)
7	9100	71	15	17	(3)	2.0	1.5	1.1	(0.5)
8	27311	102	16	17	(3)	10	6.0	4.0	(2.1)
9	82522	137	16	19	(4)	44	21	15	(4.8)
10	246172	193	17	20	(4)	195	77	51	(16)

Table 8: *Exp. 4 on adaptively created triangulations (3D-domain with a crack).*

Level	Nodes	#Iter				CPU [sec]			
		SGS	MG	AMG	(Levels)	SGS	MG	AMG_H	(Setup)
5	6857	45	26	26	(2)	1.6	1.7	1.7	(0.1)
6	30397	65	25	26	(3)	12.2	8.7	8.6	(0.3)
7	154360	106	26	26	(3)	111	52	49	(1.9)
8	763870	174	25	26	(4)	951	266	260	(8.5)

Table 9: *Exp. 4 with hierarchical quadratic elements (adaptively created triangulations are used).*

A difference with regard to exp. 1 can be observed in the adaptive cases: for linear and quadratic elements MG and AMG now exhibit similar convergence rates. Without presenting detailed results, we remark that there is no significant change if jumps in the material coefficients are introduced. Thus we assume that the singular solution along the crack is responsible for the specific behaviour of MG and AMG in this experiment.

References

[1] R.E. Bank and C. Wagner. Multilevel ILU decomposition. *Numer. Math.*, 82, pp. 543-576, 1999.

[2] R.E. Bank and J. Xu. The hierarchical basis multigrid method and incomplete LU decomposition. In D. Keyes and J. Xu, editors, *Seventh International Symposium on Domain Decomposition Methods for Partial Differential Equations*, pp. 163-173, AMS, Providence, Rhode Island, 1994.

[3] D. Braess. Towards algebraic multigrid for elliptic problems of second order. *Computing*, 55, pp. 379-393, 1995.

[4] J.H. Bramble, D.Y. Kwak, and J.E. Pasciak. Uniform convergence of multigrid V-cycle iterations for indefinite and nonsymmetric problems. *SIAM J. Numer. Anal.*, 31(6, pp. 1746-1763), 1994.

[5] A. Brandt. Algebraic multigrid theory: the symmetric case. *Appl. Math. Comput.*, 19, pp. 23-56, 1986.

[6] A. Brandt, S.F. McCormick, and J.W. Ruge. Algebraic multigrid (AMG) for sparse matrix equations. In D.J. Evans, editor, *Sparsity and its applications*, pp. 257-284, Cambridge University Press, Cambridge, 1985.

[7] Qianshun Chang, Yau Shu Wong, and Hanging Fu. On the algebraic multigrid method. *J. Comp. Phys.*, 125, pp. 279-292, 1996.

[8] Wolfgang Hackbusch. *Iterative Solution of Large Sparse Linear Systems of Equations.* Springer, Berlin, 1993.

[9] J.W. Ruge and K. Stüben. Algebraic multigrid. In S.F. McCormick, editor, *Multigrid Methods*, Vol.4, SIAM, Philadelphia, 1987.

[10] P. Vaněk, M. Brezina, and J. Mandel. Convergence of algebraic multigrid based on smoothed aggregation. Technical report, Center for Computational Mathematics, University of Colorado, Denver, 1998.

[11] P. Vaněk, J. Mandel, and M. Brezina. Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems. *Computing*, 56, pp. 179-196, 1996.

[12] P. Waldow and I. Wolff. The skin-effect at high frequencies. *IEEE Trans. Microw. Theory Tech.*, 33, pp. 1076-1982, 1985.

[13] H. Yserentant. Preconditioning indefinite discretization matrices. *Num. Math.*, 54, pp. 719-734, 1988.

[14] de Zeeuw. Matrix-dependent prolongations and restrictions in a black-box multigrid solver. *J. Comp. Appl. Math*, 33, pp. 1-27, 1990.