A Modular Algebraic Multilevel Method

Jürgen Fuhrmann *
Weierstrass Institute
for Applied Analysis and Stochastics
Mohrenstr. 39
10117 Berlin
fuhrmann@wias-berlin.de

AMS (MOS) Subject classification: 65N20, 65N55.
Key words: algebraic multigrid, preconditioning, unstructured meshes.

*This work has been supported by the SFB 359 of the Deutsche Forschungsgemeinschaft.

Abstract

In this paper, we propose a modular algebraic multilevel method on unstructured meshes which is intended to generalize known methods on rectangular meshes. To define the transfer operators, we utilize the block structure of the matrix which is induced by the fine-coarse partitioning of the matrix graph and approximate the block of fine-fine couplings by a modification of its lower, respectively upper triangular part. Numerical experiments show that this approach yields a working preconditioner. Its efficiency in the current implementation depends on the amount of structure information given on input.

1 Introduction

Let $\mathbb{M}$ be an Euclidean vector space, and let

$$A : \mathbb{M} \to \mathbb{M}$$

be a symmetric, positive definite matrix[1].
For a given right hand side $f \in \mathbb{M}$, we want to solve the problem

$$Au = f. \tag{1.1}$$

To define a multigrid method to solve (1.1), we need the following ingredients:

- *coarse grid spaces* - Euclidean vector spaces $\mathbb{M}_0, \dots \mathbb{M}_j = \mathbb{M}$ with scalar products
$$(\cdot, \cdot)_k : \mathbb{M}_k \times \mathbb{M}_k \to \mathbb{R}$$
- symmetric, positive definite with respect to $(\cdot, \cdot)_k$ *coarse grid operators*
$$A_k : \mathbb{M}_k \to \mathbb{M}_k,$$
- *interpolations*
$$P_k : \mathbb{M}_{k-1} \to \mathbb{M}_k$$
- *restrictions*
$$R_k : \mathbb{M}_k \to \mathbb{M}_{k-1}.$$
- *smoothers*
$$G_k : \mathbb{M}_k \to \mathbb{M}_k$$

The well known multigrid algorithm then looks a follows:

procedure mgcycle $(k \in \mathbb{N}, u_k, f_k \in \mathbb{M}_k)$
begin
    if $k = 0$ then

[1] In practice, the symmetry assumption can often be weakened. For example, we may require that A is selfadjoint with respect to a weighted scalar product as described in [Fuh94]. Methods of this kind can converge even under still weaker assumptions - see [Reu95].

```
        u_k := A_k^{-1} f_k
    else
        let i ∈ ℕ
        let u_{k-1}, f_{k-1} ∈ 𝕄_{k-1}
        let d_k ∈ 𝕄_k
        for i := 1 until m do u_k := u_k - G_k(A_k u_k - f_k)
        d_k := A_k u_k - f_k
        u_{k-1} := 0
        f_{k-1} := R_{k-1} d_k
        for i = 1 until γ do mgcycle (k - 1, u_{k-1}, f_{k-1})
        u_k := u_k - P_k u_{k-1}
        for i := 1 until m do u_k := u_k - G_k(A_k u_k - f_k)
    endif
end
```

If A arises from a finite element discretization of a selfadjoint boundary value problem with constant coefficients or coefficient jumps aligned with coarse mesh element boundaries, in the case of a hierarchical structure of the finite element space, all components but the smoother are canonically defined. Under certain assumptions, a rigorous theory assures the convergence of the multigrid method with a rate independent or nearly independent on the number of grid points [Hac85, Xu92, Yse93, Osw94].
If some of these assumptions do not hold, e.g. when

- coefficient jumps are not aligned to coarse grid lines
- the approximation of complicated geometries with standalone mesh generators like [Sch93] does not result in a hierarchical structure
- A is the linearization of a nonlinear operator
- A is nonselfadjoint and comes e.g. from an upwind discretizations of a convection - diffusion equation,

the multigrid components defined by the usual FEM based scheme do not work properly (in the sense that the convergence speed slows down with increasing problem size), or do not work at all.
A promising way out of this dilemma is the usage of algebraic multigrid methods. One can trace two main approaches in the development of algebraic multilevel methods:

- methods using grid structure information [ABDJP81, DJ87, Kuz89, AV89, dZ90, HK91, FG91, Fuh94, Reu95, WKW95] and others
- methods using only the matrix on input [RS87, VMB94, Bra95]

In this paper, a modular algebraic multigrid method is proposed. This method is intended to be able to use grid structure information if such information can be provided, and otherwise, to create it's own coarse grid sequences. It should consist of three main steps:

I. Coarsening, symbolic calculation of coarse grid and transfer operators.

II. Calculation of coarse grid and transfer operators.

III. Multilevel preconditioning.

Step I (which might be not an O(n) operation) needs to be performed only once, when a grid is created. At the same time, the method should be able to use various kinds of information which may be known a priori - refinement history, strong couplings, anisotropies etc.
Step II (which should be O(n)) needs to be performed when a new matrix from a series of problems is created on a given grid.
Step III (hopefully O(n), too) is the actual preconditioner which can be efficiently implemented using the information created during steps I and II.
This idea of an algorithm is inspired by the AMG method of Ruge and Stüben [RS87]. Their algorithm combines steps I and II into one. As this coarsening algorithm is very complex, one could imagine that a splitting into the two parts mentioned could improve the overall performance when one has to solve series of problems on the same grid, but with different matrices.
In this paper, we will focus mainly on the following question:

> Given the matrix A and the subset of variables corresponding to the coarse grid, is it possible to define transfer and coarse grid operators for a multigrid method using only this information ?
> Does the method defined in this way show the desired multigrid efficiency ?

The transfer operators in the case of constant coefficients and rectangular or hierarchical finite element meshes should coincide with those yielded by the usual theory. In the case of rectangular meshes and jumping coefficients, one would like to obtain transfer operators similar to those defined in [ABDJP81, Fuh94] which are known to work well.
In the sequel, C will denote the subset of variables corresponding to the coarse grid points (C-points), and F will denote the subset of variables corresponding to the fine grid points not belonging to the coarse grid (F-points). We assume that C is given.
In this case, we can sort the unknowns in such a way that the matrix A has the block structure

$$A = \begin{pmatrix} A_{FF} & A_{FC} \\ A_{CF} & A_{CC} \end{pmatrix}.$$

In the following sections, we describe a framework which we would like to use for the description of our multigrid algorithm.

2 An Exact Multigrid Method

The starting point of our considerations will be the observation that in certain cases we are able to define an exact multigrid method, where "exact" means that it yields the exact solution within one iteration step.
First, we observe that A can be transformed into a block diagonal form. Define

$$\hat{T} = \begin{pmatrix} I & A_{FF}^{-1}A_{FC} \\ 0 & I \end{pmatrix}.$$

Then

$$A = \hat{T}^T \begin{pmatrix} A_{FF} & 0 \\ 0 & \hat{S} \end{pmatrix} \hat{T}$$

where

$$\hat{S} = A_{CC} - A_{CF} A_{FF}^{-1} A_{FC}$$

is the Schur complement.
Let

$$\begin{aligned} \hat{P} &= \begin{pmatrix} -A_{FF}^{-1} A_{FC} \\ I \end{pmatrix} \\ \hat{R} &= \begin{pmatrix} -A_{CF} A_{FF}^{-1} & I \end{pmatrix} \\ \hat{G} &= \begin{pmatrix} A_{FF}^{-1} & 0 \\ 0 & 0 \end{pmatrix}. \end{aligned} \tag{2.1}$$

We can see $\hat{P}$ as an interpolation, $\hat{R}$ as a restriction and $\hat{G}$ as a smoother.

Lemma 2.1 *The Schur complement $\hat{S}$ is the Galerkin coarse grid operator corresponding to $\hat{P}$ and $\hat{R}$.*

Proof. Indeed,

$$\begin{aligned} &\hat{R} A \hat{P} \\ &= \begin{pmatrix} -A_{CF} A_{FF}^{-1} & I \end{pmatrix} \begin{pmatrix} A_{FF} & A_{FC} \\ A_{CF} & A_{CC} \end{pmatrix} \hat{P} \\ &= \begin{pmatrix} 0 & A_{CC} - A_{CF} A_{FF}^{-1} A_{FC} \end{pmatrix} \begin{pmatrix} -A_{FF}^{-1} A_{FC} \\ I \end{pmatrix} \\ &= \hat{S} \end{aligned}$$

□

Lemma 2.2 *The two grid method consisting of the coarse grid correction defined by $\hat{P}, \hat{R}, \hat{S}$ and one post-smoothing step using $\hat{G}$ is exact.*

Proof. First, it is easy to verify, that

$$\begin{aligned} \hat{G} &= \hat{T}^{-1} \begin{pmatrix} A_{FF}^{-1} & 0 \\ 0 & 0 \end{pmatrix} \hat{T}^{-T} \\ \hat{P} \hat{S}^{-1} \hat{R} &= \hat{T}^{-1} \begin{pmatrix} 0 & 0 \\ 0 & \hat{S}^{-1} \end{pmatrix} \hat{T}^{-T} \\ A^{-1} &= \hat{T}^{-1} \begin{pmatrix} A_{FF}^{-1} & 0 \\ 0 & \hat{S}^{-1} \end{pmatrix} \hat{T}^{-T}. \end{aligned}$$

It is sufficient to show that the error propagation operator describing the method is zero. One has

$$\begin{aligned} &(I - \hat{G} A)(I - \hat{P} \hat{S}^{-1} \hat{R} A) = \\ &= \left(I - \hat{T}^{-1} \begin{pmatrix} A_{FF}^{-1} & 0 \\ 0 & 0 \end{pmatrix} \hat{T}^{-T} A \right) \left(I - \hat{T}^{-1} \begin{pmatrix} 0 & 0 \\ 0 & \hat{S}^{-1} \end{pmatrix} \hat{T}^{-T} A \right) \\ &= I - \hat{T}^{-1} \left(\begin{pmatrix} A_{FF}^{-1} & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & \hat{S}^{-1} \end{pmatrix} \right) \hat{T}^{-T} A + \end{aligned}$$

$$
\begin{aligned}
& +\hat{T}^{-1}\begin{pmatrix} A_{FF}^{-1} & 0 \\ 0 & 0 \end{pmatrix}\hat{T}^{-T}A\hat{T}^{-1}\begin{pmatrix} 0 & 0 \\ 0 & \hat{S}^{-1} \end{pmatrix}\hat{T}^{-T}A \\
= \; & I-\hat{T}^{-1}\begin{pmatrix} A_{FF}^{-1} & 0 \\ 0 & \hat{S}^{-1} \end{pmatrix}\hat{T}^{-T}A+ \\
& +\hat{T}^{-1}\begin{pmatrix} A_{FF}^{-1} & 0 \\ 0 & 0 \end{pmatrix}\begin{pmatrix} A_{FF}^{-1} & 0 \\ 0 & \hat{S}^{-1} \end{pmatrix}\begin{pmatrix} 0 & 0 \\ 0 & \hat{S}^{-1} \end{pmatrix}\hat{T}^{-T}A \\
= \; & 0.
\end{aligned}
$$

□

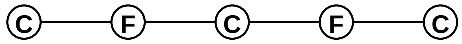

Figure 1: *Structure of an one-dimensional grid*

In the one-dimensional case, we can choose the set of C-points as in fig. 2. Here, A_{FF} is diagonal, so that this method can be efficiently implemented. In fact, this method is well known as the 1D case of the method of *cyclic reduction* [BGN70].
In the general case, it is not possible to invert A_{FF} in an efficient way. In the literature, various approaches to this problem can be found. So, in [Vas92], a Čebyšev polynomial in A_{FF} is used to obtain an approximation to its inverse. In this case it is hard to define the Galerkin coarse grid operator, so the discretization on the coarser level is used. In our case, this discretization is not available, so we try to use a scheme with a different structure.

3 A Framework for the Definition of AMG Components

The basic idea is to replace A_{FF} by its upper, respectively lower triangular part with a modified main diagonal. Thus, we replace the solution of the system

$$A_{FF}u_F = f_F$$

by the result of a Gauss-Seidel like step with an appropriate ordering.
Assume that

$$A_{FF} = l + b + u,$$

where l, u are strictly lower, respectively upper triangular matrices, and b is a block diagonal matrix. Further, let d be some "modification" of the main diagonal of A_{FF} in the sense of a partial lumping.
The order of the F-points (which actually defines how to understand "triangular") and the choice of the matrix d are the main parameters in the proposed scheme.
Let

$$
\begin{aligned}
L &= l + d \\
U &= u + d \\
B &= b - 2d.
\end{aligned}
\tag{3.1}
$$

Then

$$A_{FF} = L + B + U,$$

where L and B are invertible lower, respectively upper triangular matrices.
Define

$$\begin{aligned} P &= \begin{pmatrix} -U^{-1}A_{FC} \\ I \end{pmatrix}, \\ R &= \begin{pmatrix} -A_{CF}L^{-1} & I \end{pmatrix} \end{aligned}$$

by formally replacing in (2.1) A_{FF} by U and L, respectively.
Then defining

$$\begin{aligned} U_\delta &= I - A_{FF}U^{-1}, \\ L_\delta &= I - L^{-1}A_{FF}, \end{aligned}$$

the Galerkin coarse grid operator

$$\begin{aligned} S &= R \begin{pmatrix} L+B+U & A_{FC} \\ A_{CF} & A_{CC} \end{pmatrix} P \\ &= A_{CC} + A_{CF}L^{-1}BU^{-1}A_{FC} \\ &= \hat{S} + A_{CF}L_\delta A_{FF}^{-1} U_\delta A_{FC} \end{aligned}$$

can be seen as a perturbation of the Schur complement [HLM91a, HLM91b].
If this operator appears to be too complex, one might replace it by a simpler operator $\tilde{S} \approx S$ which should be spectrally equivalent to S.
The smoother $\hat{G}$ can be replaced by some "classical" smoother.
In [Fuh94] it has been shown that the convergence of a method using these components depends on $||U_\delta||_{A_{FF}}$, the spectral equivalence between D and A_{FF} and the strengthened Cauchy inequality between the F-point space and the C-point space. This hardly reflects the real nature of the diagonal modification explained below, as shown in numerical experiments in [Fuh94][2].
In [Fuh94, FG94] it has been shown how to set up converging, effectively implementable methods fitting to this scheme in the case of one- two- and three-dimensional logically orthogonal meshes. These methods are in some sense semi-algebraic, because they use grid structure information given from outside. The do not use any information on the kind of the discretization, on mesh spacing or point locations. This is true also for the available code *mg2537*.
We describe the two-dimensional case of this method in the next section. Then, a generalization to the case of unstructured meshes is proposed.

[2] A method using $D = d$ could be defined correctly, however it converges *very* slowly

4 An Algebraic Multigrid Method on Two-Dimensional, Logically Rectangular Meshes

On a two-dimensional, logically rectangular mesh, the set of F-points of an operator with a five point discretization stencil can be subdivided further into the sets of coarse grid cell edge midpoints (E-points) and coarse grid cell midpoints (M-points) - see fig. 2. The term "midpoint" should be understood topologically, not geometrically. After reordering nodes, we have the following matrix

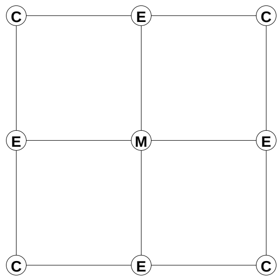

Figure 2: *Coarse grid cell structure for a two-dimensional grid*

partition:

$$A = \begin{pmatrix} \begin{pmatrix} A_{MM} & A_{ME} \\ A_{EM} & A_{EE} \end{pmatrix} & \begin{pmatrix} 0 \\ A_{EC} \end{pmatrix} \\ \begin{pmatrix} 0 & A_{CE} \end{pmatrix} & A_{CC} \end{pmatrix}$$

If A is a weakly diagonally dominant M-matrix, the diagonal blocks

$$\begin{aligned} A_{MM} &= D_{ME} + M_{MM} \\ A_{EE} &= D_{EM} + D_{EC} + M_{EE} \\ A_{CC} &= D_{CE} + M_{NN} \end{aligned}$$

are positive diagonal matrices which consist of the sum of the sign reversed off diagonal row entries and a nonnegative "mass" term. The off diagonal blocks have only nonpositive entries. Let

$$b = \begin{pmatrix} D_{ME} + M_{MM} & 0 \\ 0 & D_{EM} + D_{EC} + M_{EE} \end{pmatrix} = \begin{pmatrix} A_{MM} & 0 \\ 0 & A_{EE} \end{pmatrix}$$

and

$$d = \begin{pmatrix} D_{ME} + M_{MM} & 0 \\ 0 & D_{EC} + M_{EE} \end{pmatrix} = \begin{pmatrix} d_{MM} & 0 \\ 0 & d_{EE} \end{pmatrix}$$

Then

$$\begin{aligned} L &= \begin{pmatrix} D_{ME} + M_{MM} & A_{ME} \\ 0 & D_{EC} + M_{EE} \end{pmatrix} \\ U &= \begin{pmatrix} D_{ME} + M_{MM} & 0 \\ A_{EM} & D_{EC} + M_{EE} \end{pmatrix} \\ B &= \begin{pmatrix} -D_{ME} - M_{MM} & 0 \\ 0 & D_{EM} - D_{EC} - M_{EE} \end{pmatrix} \end{aligned}$$

and

$$
\begin{aligned}
P &= \left(\begin{pmatrix} d_{MM}^{-1} A_{ME} d_{EE}^{-1} A_{EC} \\ -d_{EE}^{-1} A_{EC} \end{pmatrix} \atop I \right) \\
R &= \left(\begin{pmatrix} A_{CE} d_{EE}^{-1} A_{EM} d_{MM}^{-1} & -A_{CE} d_{EE}^{-1} \end{pmatrix} \quad I \right)
\end{aligned}
$$

The interpolation P can be implemented in three stages: First perform a straight injection of the coarse grid node values, then by means of $-d_{EE}^{-1} A_{EC}$ interpolate the edge midpoint values (by a formula which is similar to the one-dimensional case), and at last, use $-d_{MM}^{-1} A_{ME}$ to calculate the cell midpoint values from the edge midpoint values.
Similar transfer operators have been defined in [ABDJP81, Hac85, DJ87] The Galerkin coarse grid operator is

$$
\begin{aligned}
S &= A_{CC} - 2A_{CE} d_{EE}^{-1} A_{EC} + A_{CE} d_{EE}^{-1} (A_{EE} - A_{EM} d_{MM}^{-1} A_{ME}) d_{EE}^{-1} A_{EC} \\
&= A_{CC} - A_{CE} d_{EE}^{-1} A_{EC} + A_{CE} d_{EE}^{-1} (D_{EM} - A_{EM} d_{MM}^{-1} A_{ME}) d_{EE}^{-1} A_{EC}.
\end{aligned}
$$

This coarse grid operator has a nine-diagonal structure, and we replace it by the five-diagonal operator

$$
\tilde{S} = 2(A_{CC} - A_{CE} d_{EE}^{-1} A_{EC})
$$

which is twice the Schur complement of the matrix

$$
A^0 = \begin{pmatrix} d_{EE} & A_{EC} \\ A_{CE} & A_{CC} \end{pmatrix}.
$$

A^0 as well as its Schur complement $\tilde{S}$ inherit from A symmetry, positive definiteness, nonpositivity of the off diagonal entries and weak diagonal dominance, if at least one entry of M_{EE} or M_{CC} is positive [Axe94]. Thus, if the last condition is fulfilled, $\tilde{S}$ inherits the M-property from A. The inheritance of the M-property cannot be shown for S.
Another advantage of $\tilde{S}$ is the possibility of a numerically stable generation of the transfer and coarse grid operators, provided, we use a matrix storage scheme where we store

$$
\begin{pmatrix} M_{MM} & 0 & 0 \\ 0 & M_{EE} & 0 \\ 0 & 0 & M_{CC} \end{pmatrix}
$$

in the locations for the main diagonal entries of A. A more detailed discussion of this issue together with numerical experiments one finds in [Fuh94, FG94].
The numerical experiments [Fuh94], see also section 6, show a good convergence behaviour of this algorithm for problems with strongly varying coefficients.
In a very similar fashion, one is able to describe the algorithm for three-dimensional problems with a seven diagonal operator.

5 An Algebraic Multigrid Method on Unstructured Meshes

Here, we propose an answer to the question from section 1.1: Given the matrix A and the subset of variables corresponding to the coarse grid, is it possible to define transfer and coarse grid operators for a multigrid method using only this information ?
The basic idea is taken from the tensor product case. In the two-dimensional as well as in the three-dimensional case, we utilized a natural hierarchical block structure of the set of F-points. We could try to create such a block structure for the unstructured mesh case, too.
So, assuming, the coarse grid nodes are known, we sort the remaining F-points in a hierarchical fashion. This means that we partition the set of grid vertices $V(A)$ into

$$V = V(A) = C \cup F_1 \cup \ldots \cup F_L.$$

Let for $x \in V$, $V_x := \{y \in V(A) | (x, y) \in E(A)\}$ denote the neighbourhood of x in the matrix graph $\Gamma(A) = (V(A), E(A))$ of A. Consider the following algorithm:

$l := 1$
$V_l := C$
while $V_l \neq V$ do
    $F_l := \{x \in V \setminus V_l \mid \mathrm{card}\,(V_x \cap V_l) \geq 2\}$
    if $F_l = \emptyset$ then
        $F_l := \{x \in V \setminus V_l \mid \mathrm{card}\,(V_x \cap V_l) \geq 1\}$
    endif
    $V_{l+1} := V_l \cup F_l$
    $l := l + 1$
done

For each node on a new level, in the generic case we get a set of two or more nodes to interpolate from. For tensor product meshes, this produces the same partition as used in the method described in section 4. So, we get a matrix partition

$$A = \begin{pmatrix} \begin{pmatrix} A_{F_L F_L} & \ldots & A_{F_L F_1} \\ \vdots & \ddots & \vdots \\ A_{F_1 F_L} & \ldots & A_{F_1 F_1} \end{pmatrix} & \begin{pmatrix} A_{F_L C} \\ \vdots \\ A_{F_1 C} \end{pmatrix} \\ \begin{pmatrix} A_{C F_L} & \ldots & A_{C F_1} \end{pmatrix} & A_{CC} \end{pmatrix}$$

Note that here, the diagonal blocks $A_{F_l F_l}$ are not necessarily diagonal matrices. Now, we try to formalize the structure of the diagonals $D_{F_l F_l}$ of these blocks. Assume that

$$D_{F_l F_l} = M_{F_l F_l} + D^{+}_{F_l, F_l} + D^{-}_{F_l, F_L} + D^{0}_{F_l, F_l}$$

is the sum of

- a "mass" term $M_{F_l F_l}$,
- the sign-reversed column entries connecting to higher level nodes $D^{+}_{F_l, F_l}$

- the sign-reversed column entries connecting to lower level nodes $D^{-}_{F_l,F_l}$
- and the sign-reversed column entries connecting level l nodes $D^{0}_{F_l,F_l}$.

This partitioning is inspired from the splitting of the diagonals introduced in section 4 for the M-matrix case, where we can assume $M_{F_lF_l} \geq 0$. As, in general, the M-property is not inherited by the Galerkin projection, on the coarse meshes it has only a formal sense.
Let

$$d_{F_lF_l} = M_{F_lF_l} + D^{-}_{F_lF_l}$$

Define

$$\begin{aligned}
u &= \begin{pmatrix} 0 & \dots & A_{F_LF_1} \\ \vdots & \ddots & \vdots \\ 0 & \dots & 0 \end{pmatrix}, \\
l &= \begin{pmatrix} 0 & \dots & 0 \\ \vdots & \ddots & \vdots \\ A_{F_1F_L} & \dots & 0 \end{pmatrix}, \\
b &= \begin{pmatrix} A_{F_lF_l} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & A_{F_1F_1} \end{pmatrix}, \\
d &= \begin{pmatrix} d_{F_lF_l} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & d_{F_1F_1} \end{pmatrix}.
\end{aligned}$$

Let as in (3.1)

$$\begin{aligned}
L &= l + d \\
U &= u + d.
\end{aligned}$$

Then for

$$B = \begin{pmatrix} A_{F_LF_L} - 2d_{F_LF_L} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & A_{F_1F_1} - 2d_{F_1F_1} \end{pmatrix},$$

we have

$$A_{FF} = L + B + U,$$

and we can define the transfer operators

$$\begin{aligned}
P &= \begin{pmatrix} -U^{-1}A_{FC} \\ I \end{pmatrix} \\
R &= \begin{pmatrix} -A_{FC}L^{-1} & I \end{pmatrix}.
\end{aligned}$$

In the logically orthogonal case, we get the same transfer operators as introduced there, provided, we had defined the same set of C-points. The Galerkin coarse grid operator

$$S = A_{CC} + A_{CF}L^{-1}BU^{-1}A_{FC}$$

inherits symmetry and positive definiteness from A, but not the nonpositivity of the off diagonal entries.

6 Numerical Examples

6.1 The problems solved

On the series of meshes described below, we solve the *constant coefficient problem*

$$-\Delta u = 0$$

with the boundary conditions introduced below.
Further, we solve *jumping coefficient problems* of the type

$$-\nabla \cdot (a(x)\nabla u) = 0$$

with coefficient jumps of the magnitude 10^5 according to the material distributions shown in fig. 3, 4 and 5. The boundary conditions remain the same.

6.2 The Meshes

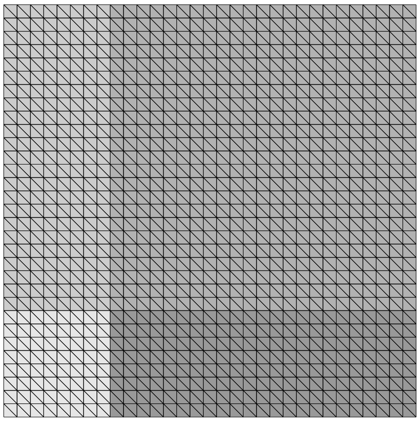

Figure 3: *Material distribution and triangulation for mesh R (1024 nodes)*

The mesh scale R. We consider the domain $\Omega = [0,1] \times [0,1]$. We assume that it is subdivided into four subdomains with different materials according to fig. 3. We apply the Dirichlet boundary conditions

$$\begin{array}{lll} \Gamma_0 & \text{on} & [0,1] \times 1 \subset \partial\Omega \\ \Gamma_1 & \text{on} & [0,1] \times 0 \subset \partial\Omega \end{array}$$

It is subdivided into triangles in the standard manner. The meshes have a clear regular structure.

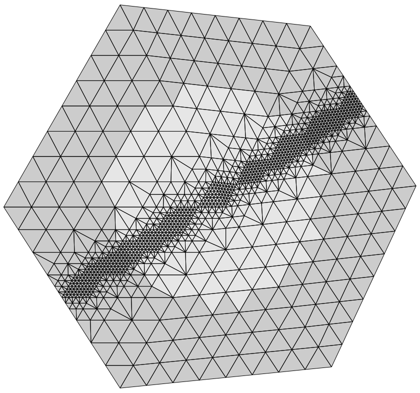

Figure 4: *Material distribution and triangulation of mesh H (1294 nodes)*

The mesh scale H. The meshes of this scale are examples for adaptively refined meshes, where the adaptivity criterion is defined not by the coefficient jump, but e.g. by a front to be followed. Further, the coefficient jumps are *not* aligned to coarse grid cells. Within this mesh scale, the meshes are generated by KASKADE 3.0 [BER94] with different levels of local refinement along a given line. We apply Dirichlet boundary conditions with value 0 respectively 1 at two disjoint lines of the domain boundary. These meshes are *not unstructured* but possess a clear *hierarchical* structure which will be used and honored by the proposed modular AMG method.

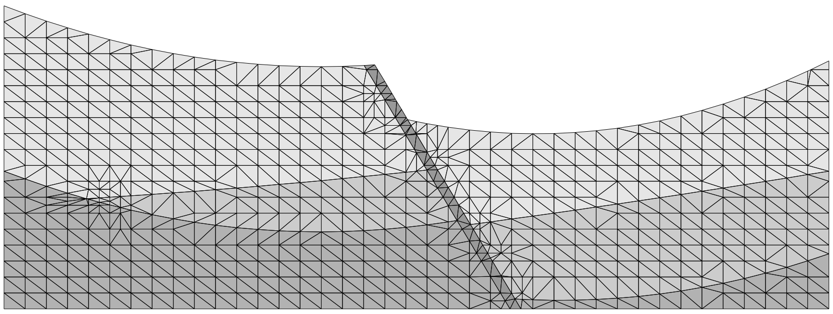

Figure 5: *Material distribution and triangulation of mesh U (696 nodes)*

The mesh scale U. The meshes of this scale correspond to applications in groundwater flow simulation. They describe a layered soil structure which is perturbed in the middle part as it can be seen in fig. 5. We apply Dirichlet boundary conditions with value 0 at the left and 1 at the right boundary lines of the domain. Within this mesh scale, the meshes are generated by the package IBG [Sch93] with the same geometry, but with different node numbers on the coarsest grid frame.

6.3 A Simple Coarsening Algorithm

On mesh scale R in the nonstandard case, and on mesh scale U, we need a coarsening algorithm to be able to use the proposed modular algebraic multigrid method. We proceed as follows: We pick some mesh point which is still unmarked and mark it as a C-point. Then we mark all its unmarked neighbours as F-points and repeat this process until all points have been marked:

$F := \emptyset$
$C := \emptyset$
$U := V(A)$
while $U \neq \emptyset$ do
    pick $x \in U$
    $C := C \cup \{x\}$
    $F := F \cup (V_x \cap U)$
    $U := U \setminus ((V_x \cap U) \cup \{x\})$
done

This choice of the coarsening algorithm ensures the desired modularity of the algorithm, but as we will see, it seems necessary to put more information into the coarsening process when one wants to obtain condition numbers independent on the mesh size for general unstructured meshes.

6.4 The methods compared

ilu: This is a conjugate gradient method preconditioned by standard ILU with zero fill-in pattern.

m-amg: We use the multigrid components described in subsection 5 together with an ILU smoother and a Čebyšev polynomial on the coarsest grid to build a V-cycle preconditioner for conjugated gradients.
On mesh scale R we use the coarsening algorithm from subsection 6.3. On the finest grid, we ignore the “weak” connections corresponding to the diagonal edges in the Courant mesh. On "standard" meshes with $(2^n + 1) \times (2^n + 1)$ nodes, this produces the same coarsening sequence as standard multigrid, on more general rectangular meshes, we get a method which is slightly less efficient but still shows multigrid efficiency (see the saw-teeth in fig. 6).
On mesh scale H, we use the refinement history to generate the coarse grid information. So we get the same coarse grid points as for standard multigrid on these meshes.

On mesh scale U, coarsening is performed using the algorithm from subsection 6.3, again ignoring zero connections on the finest grid.

rs-amg: This is the AMG method of Ruge and Stüben in their original FORTRAN code [3] used as preconditioner. All numerical parameters of the method are defined by the defaults given in the code.

rect-amg: On the rectangular mesh, for appropriate point numbers, we apply he rectangular mesh preconditioner described in section 4, the code used is the *mg2537* package written by the author.

6.5 Results

We do no comparison of CPU times, because the implementation of the proposed method is at a test stage.
In the figures, on the y-axis we show the average residual contraction per iteration step, after an overall residual contraction of 10^{-10} has been reached. The x-axis (logarithmically) shows the number of grid points.

Performance on mesh scale R. (fig. 6.) The best performance shows the

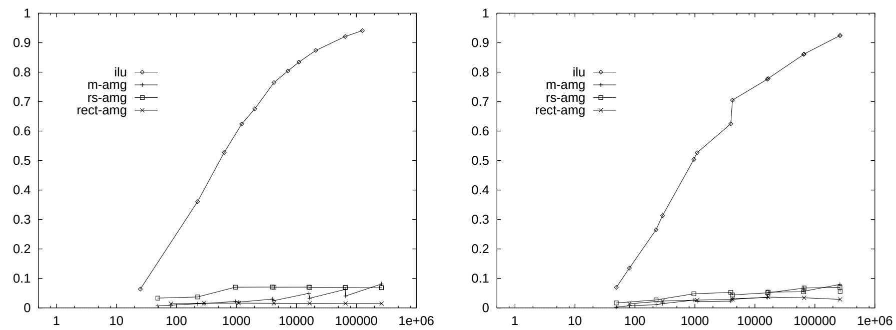

Figure 6: *Performance on mesh scale R for constant (left) and jumping (right) coefficients*

rectangular mesh multigrid method which in the case of constant coefficients is equivalent to the standard multigrid method.
The proposed AMG method on the rectangular meshes shows slightly decreasing performance with increasing mesh size. The saw-teeth in the curve show a better convergence behaviour in the case when the mesh is rectangular of the size $(2^n + 1) \times (2^n + 1)$.
The AMG method of Ruge and Stüben shows convergence rates nearly independent of the mesh scale.

Performance on mesh scale *H*. (fig. 7.) On the hierarchically structured

[3]AMG1R5, Oct. 1990 release

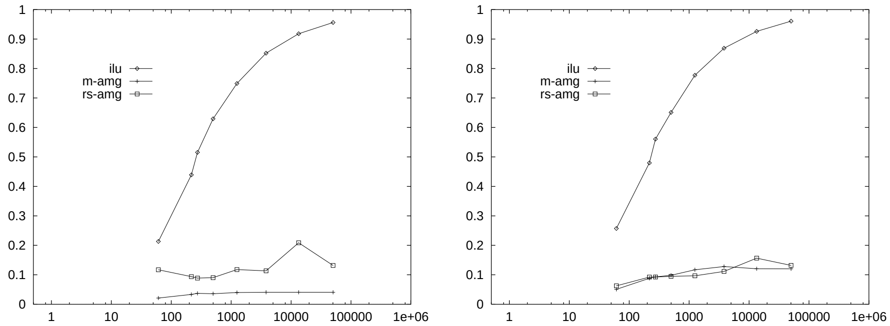

Figure 7: *Performance on mesh scale H for constant (left) and jumping (right) coefficients*

meshes of the scale H, the proposed method shows a convergence behaviour nearly independent of the mesh size. In the constant coefficient case, its performance is very good, in the jumping coefficient case, its performance is satisfactory and not worse than that of the Ruge-Stüben code[4].

Performance on mesh scale U. (See fig. 8.) The convergence rate of the

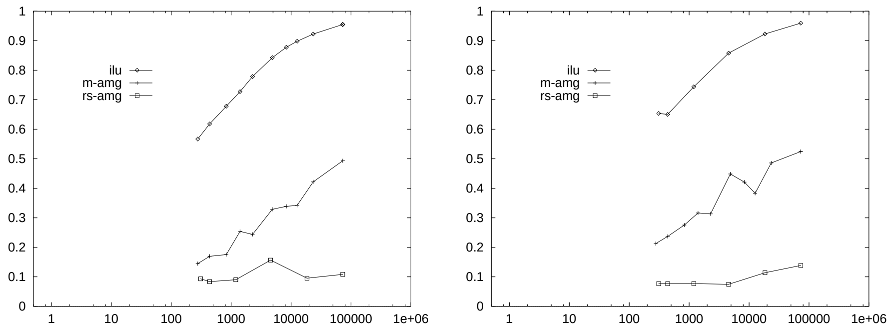

Figure 8: *Performance on mesh scale U for constant (left) and jumping (right) coefficients*

proposed AMG method is significantly less than that of the ILU method, but increases with the mesh size. It is in question if the overhead introduced in comparison to the ILU method can be compensated by the faster convergence. The main reason seems to be the "trivial" coarsening algorithm of subsection 6.3 which in this case does not use enough information about the problem.

[4]The performance of rs-amg in the constant coefficient case might be due to the fact that it has been treated as a black box, i.e. that the default numerical parameters of the code haven't been changed.

The AMG method of Ruge and Stüben converges with very good contraction rates nearly independently of the mesh size.

7 Conclusions

The numerical experiments indicate that the proposed modular AMG method works. It shows the desired multigrid behaviour

- for problems with jumping coefficients on rectangular meshes with $(2^n + 1) \times (2^n + 1)$ nodes in it's rectangular mesh implementation
- for problems with jumping coefficients on rectangular meshes with general node numbers
- for problems on hierarchically structured meshes with jumping coefficients not resolved by the coarsest mesh

On general unstructured meshes, it does not show the multigrid efficiency one would like to obtain. To reach the desired efficiency, or, at least, near mesh independence, it is necessary to put more information about the problem and/or grid structure into the coarsening process than just the matrix graph. If such information exists and is provided to the algorithm, this is honored by the method as shown by the rectangular and hierarchical mesh cases.

References

[ABDJP81] R. Alcouffe, A. Brandt, J. Dendy Jr, and J. Painter, *The multi-grid method for the diffusion equation with strongly discontinuous coefficients*, SIAM Journal on Scientific and Statistical Computing, 1 (1981), 430–454.

[AV89] O. Axelsson and P. S. Vassilevski, *Algebraic multilevel preconditioning methods, I*, Numerische Mathematik, 56 (1989), 157–177.

[Axe94] O. Axelsson, *Iterative solution methods*, Cambridge Unversity Press, Cambridge, 1994.

[BER94] R. Beck, B. Erdmann, and R. Roitzsch, *KASKADE 3.0 - an object oriented finite element code*, technical report, Konrad-Zuse-Zentrum für Informationstechnik, Berlin, 1994. TR 95-4.

[BGN70] B. Buzbee, G. G.H., and C. Nielson, SIAM Journal on Numerical Analysis, 7 (1970), 753–763.

[Bra95] D. Braess, *Towards algebraic multigrid for elliptic problems of second order*, Computing, (1995). to appear.

[DJ87] J. Dendy Jr., *Two multigrid methods for three-dimensional problems with discontinuous and anisotropic coefficients*, SIAM Journal on Scientific and Statistical Computing, 8 (1987) 2, 673–685.

[dZ90] P. de Zeeuw, *Matrix-dependent prolongations and restrictions in a black box multigrid solver*, J. Comp. Appl. Math, 33 (1990), 1–27.

[FG91] J. Fuhrmann and K. Gärtner, *A multigrid method for the solution of a convection - diffusion equation with rapidly varying coefficients*, in Proceedings of the Third European Multigrid Conference, October 1 - 4,1990, Bonn, Germany, W. Hackbusch and U. Trottenberg, eds., vol. 98 of ISNM, Basel, 1991, Birkhäuser Verlag.

[FG94] ——, *On matrix data structures and the stability of multigrid algorithms*, in Contributions to multigrid. A selection of contributions to the Fourth European Multigrid Conference, Amsterdam, July 6-9, 1993, P. Hemker and P. Wesseling, eds., Amsterdam, 1994, CWI. CWI Tract 103.

[Fuh94] J. Fuhrmann, *Zur Verwendung von Mehrgitterverfahren bei der numerischen Behandlung elliptischer partieller Differentialgleichungen zweiter Ordnung mit variablen Koeffizienten*, PhD thesis, Technische Universität Chemnitz-Zwickau, 1994. Verlag Shaker, Aachen 1995, ISBN 3-8256-0522-0.

[Hac85] W. Hackbusch, *Multi-grid methods and applications*, vol. 4 of Ser. Comp. Math., Springer Verlag, 1985.

[HK91] Y. Hakopian and Y. Kuznetsov, *Algebraic multigrid/substructuring preconditioners for triangular grids*, Soviet Journal on Numerical Analysis and Mathematical Modelling, 6 (1991) 6, 453–483.

[HLM91a] G. Haase, U. Langer, and A. Meyer, *The approximate Dirichlet domain decomposition method. part I: An algebraic approach*, Computing, 47 (1991), 137–151.

[HLM91b] ——, *The approximate Dirichlet domain decomposition method. part II: Applications to 2nd-order elliptic B.V.P.s*, Computing, 47 (1991), 153–167.

[Kuz89] Y. Kuznetsov, *Multigrid domain decomposition methods for elliptic problems*, Computer Methods in Applied Mechanics and Engineering, 75 (1989), 185–193.

[Osw94] P. Oswald, *Multilevel Finite Element Approximation*, Teubner, Stuttgart, 1994.

[Reu95] A. Reusken, *Fourier analysis of a robust multigrid method for convection-diffusion equations*, Numerische Mathematik, 71 (1995), 365–398.

[RS87] J. Ruge and K. Stüben, *Algebraic multigrid*, in Multigrid methods, S. McCormick, ed., vol. 4 of Frontiers in Applied Mathematics, SIAM, Philadelphia, 1987, ch. 4, 73–130.

[Sch93] I. Schmelzer, *3D anisotropic grid generation with intersection-based geometry interface*, preprint, IMA, Minneapolis, 1993. no. 1180.

[Vas92] P. Vassilevski, *Hybrid V-cycle algebraic multilevel preconditioners*, Mathematics of Computation, 58 (1992) 198, 489–512.

[VMB94] P. Vaněk, J. Mandel, and M. Brezina, *Algebraic multigrid on unstructured meshes*, manuscript, University of Colorado, Denver, CO., 1994. `ftp://tiger.denver.colorado.edu/pub/reports`.

[WKW95] C. Wagner, W. Kinzelbach, and G. Wittum, *Schur-complement multigrid - a robust method for groundwater flow and transport problems*, tech. rep., ICA, Stuttgart, 1995. ICA-Preprint 95/1.

[Xu92] J. Xu, *Iterative methods by space decomposition and subspace correction*, SIAM Review, 34 (1992) 4, 581–613.

[Yse93] H. Yserentant, *Old and new convergence proofs for multigrid methods*, Acta Numerica, (1993), 285–326.