Sparse Algebraic Multigrid Methods for Large Scale Boundary Element Equations *

U. Langer D. Pusch

19th December 2003

Institute of Computational Mathematics
Johannes Kepler University Linz
{ulanger,pusch}@numa.uni-linz.ac.at

Abstract

This paper presents new algebraic multigrid (AMG) preconditioners for sparse boundary element matrices arising from the adaptive cross approximation (ACA) to dense boundary element matrices. In the following we consider the single layer potential integral equation resulting from the direct boundary integral formulation of the interior Dirichlet boundary value problem for the Laplace equation in two, or three spatial dimensions. The standard realization of collocation, or Galerkin boundary element discretizations lead to fully populated system matrices which require $O(N_h^2)$ of storage units and cause the same complexity for a single matrix-by-vector multiplication, where N_h denotes the number of boundary unknowns. Sparse matrix approximations schemes such as ACA reduce this complexity to an almost linear behavior in N_h. Since the single layer potential operator is a pseudo-differential operator of the order -1, the resulting boundary element matrices are ill-conditioned. Iterative solvers dramatically suffer from this property for growing N_h. Our AMG-preconditioners avoid the increase of the iteration numbers and result in almost optimal solvers with respect to the total complexity. This behavior is confirmed by our numerical experiments. Let us mention that our AMG-preconditioners use only single grid information provided by the usual mesh data and by the ACA system matrix on the finest level.

Keywords integral equations of first kind, boundary element method, adaptive cross approximation, algebraic multigrid, preconditioners, iterative solvers

1 Introduction

In this paper we are concerned with the fast solution of sparse boundary element equations by algebraic multigrid methods. The most common technique

*This work has been supported by the Austrian Science Fund 'Fonds zur Förderung der wissenschaftlichen Forschung (FWF)'.

for discretizing elliptic boundary value problems for second order partial differential equations is certainly the finite element method (FEM). Nevertheless, in many applications it will be advantageous to use alternative approaches. The boundary element method (BEM) is certainly a preferable discretization technique for some specific problem classes. For instance, one typical application area for BEM is the treatment of unbounded domains. Since only the boundary $\Gamma = \partial\Omega$ of the computational domain $\Omega \subset \mathbb{R}^d$, $d = 2, 3$ has to be discretized, the dimension of the arising matrices will be reduced essentially. Once the complete Cauchy data (Dirichlet data u and Neumann data $v = \partial u/\partial n$) are available on Γ, the solution in the total computational domain Ω can easily be computed by the representation formulae.

If we are going to use the standard BEM, we are faced with at least one essential drawback. The system matrices are fully populated. More precisely, for a growing number of (boundary) unknowns N_h, every iterative solving algorithm will result in a complexity of $O(N_h^2)$ with respect to the arithmetical cost and memory demand. Furthermore let us notice that using the Galerkin method for constructing the BE-matrices leads to symmetric and positive (semi) definite system matrices, which can be solved by the conjugate gradient (CG) method. However, since we have to evaluate two integrals for each matrix entry, most of the commercial software packages gladly prefer collocation methods for constructing BE matrices. Unfortunately, this approach generally yields non-symmetric matrices. For this reason it is not possible to apply the CG algorithm, and thus, we need some appropriate Krylov-subspace methods. The most familiar algorithms are the GMRES (generalized minimal residual) and the BiCGStab (bi-conjugate gradients stabilized) methods, see e.g. [22]. We refer the reader to [11] or [29] for a detailed treatment of boundary element methods.

Therefore, in the case of boundary element methods iterative solvers are only efficient, if the cost for a single matrix-by-vector multiplication can be reduced essentially (especially in 3D). During the last two decades different sparse approximation techniques for boundary element matrices have been developed. The multipole method [24, 10], the panel clustering method [14], the $\mathcal{H}$-matrix approach [13] and wavelet techniques [18] are certainly now the most popular ones. In our paper we will consider the adaptive cross approximation (ACA) method recently suggested by M. Bebendorf and S. Rjasanow [3, 2, 4]. The basic idea is to decompose the boundary Γ into clusters and approximate the corresponding admissible submatrices by low-rank matrices. If some $n \times m$ matrix A is approximated by a r-rank matrix $\widetilde{A}$, then the effort for multiplication and storage will decrease from $O(n * m)$ to $O(r * (n + m))$. In conclusion, the application of a sparse representation algorithm allows us to realize the matrix-by-vector multiplication in almost $O(N_h)$ operations. A detailed explanation and a rigorous analysis of the ACA-method can be found in [3, 2, 4].

It is well-known that standard iterative solvers like the Krylov-subspace solvers heavily suffer from the bad conditioning of the system matrices. The condition number $\kappa(K_h)$ of system matrices K_h arising from the standard FE-discretization of boundary value problems for self-adjoint second order partial differential equations (PDEs) behaves like $O(h^{-2})$, where h is the typical mesh size. Boundary element matrices originating from the discretization of the single layer potential or the hypersingular operator also lead to badly conditioned system matrices K_h with a condition number $\kappa(K_h)$ of $O(h^{-1})$. Thus, it is obvious that we need appropriate preconditioning techniques in order to avoid the steady rise of the number of iterations for finer and finer discretization. In [19, 20] we introduced algebraic multigrid preconditioners for dense boundary element matrices arising from standard collocation or Galerkin discretizations of the single layer potential and the hypersingular operator. In this paper we generalize these results to the ACA-approximation of the single layer potential operator. Let us mention that the single layer potential operator is the more interesting case since it represents a pseudo-differential operator of order minus one. Therefore, the corresponding boundary element matrices reveal quite different spectral properties compared to standard FE stiffness matrices or BE-matrices arising from the hypersingular operator. Due to the reverse action of eigenvalues and eigenvectors of the (discrete) single layer potential operator standard multigrid smoothers like Gauss-Seidel or damped Jacobi will fail inevitably. For the geometric multigrid method, an appropriate smoother was suggested by J. Bramble, Z. Leyk and J. Pasciak in [6], abbreviated as BLP-smoother in the remaining of this paper. Moreover, we have to take into account that the system matrix is available in ACA-format only. Thus, the construction of the coarser levels, especially setting up proper transfer operators, require some modifications. It is our aim to construct an AMG-preconditioned iterative solver for ACA boundary element equations that shows linear, or at least almost linear behavior with respect to the complexity in time and memory demand.

The paper is organized as follows: Section 2 gives a brief overview on the integral operators considered and their properties. In addition, the ACA-method is briefly described. In Section 3, we develop and analyze the AMG technique for ACA-matrices. Some results of our numerical studies are presented in Section 4. Finally, we draw some conclusion and discuss some further applications in Section 5.

2 Problem Formulation and the ACA-Method

2.1 Boundary Integral Operators and their Properties

Let $\Omega \subset \mathbb{R}^d$, $d = 2, 3$ be a bounded, simply connected domain with one closed boundary piece $\Gamma = \partial\Omega$ that is supposed to be sufficiently smooth, and let us consider a continuous, linear, self-adjoint pseudo-differential operator

$$\mathcal{A} : H^s(\Gamma) \mapsto H^{-s}(\Gamma), \tag{1}$$

where $H^s(\Gamma)$ denotes the usual Sobolev space for some real $s \in \mathbb{R}$, see e.g. [1]. In the remaining of this paper we consider the single layer potential operator

$$(Vv)(y) = \int_\Gamma E(x,y)v(x)ds_x \tag{2}$$

and the hypersingular operator

$$(Du)(y) = -\partial_{n_y} \int_\Gamma \partial_{n_x} E(x,y)u(x)ds_x \tag{3}$$

arising from the Dirichlet and Neumann boundary value problems for the Laplace equation, respectively. The pseudo-differential operators V and D are special mappings $\mathcal{A} \in \mathcal{L}(H^\alpha(\Gamma), H^{-\alpha}(\Gamma))$ with $\alpha = -1/2$ and $\alpha = +1/2$, respectively. In addition, n_ξ is the unit outward normal vector to Γ at some point $\xi \in \Gamma$, and $E(x,y)$ denotes the fundamental solution of the differential operator under consideration, i.e. the Laplace operator in our paper.

Let us assume that the boundary $\Gamma \in C^{0,1}$ is Lipschitz. Then several properties for boundary integral operators can be observed (see, e.g., [29]):

1. The operators V and D are self-adjoint in the $L_2(\Gamma)$ inner product, i.e.

$$\begin{aligned} \langle v, Vu\rangle_0 &= \langle Vv, u\rangle_0 \quad \forall u,v \in H^{-1/2}(\Gamma), \\ \langle v, Du\rangle_0 &= \langle Dv, u\rangle_0 \quad \forall u,v \in H^{1/2}(\Gamma). \end{aligned}$$

2. The hypersingular operator D is positive semidefinite, i.e.

$$\begin{array}{rcl} & \langle Du, u\rangle_0 \geq 0 & \forall u \in H^{1/2}(\Gamma), \\ \exists \mu_D > 0: & \langle Du, u\rangle_0 \geq \mu_D \|u\|^2_{H^{1/2}(\Gamma)} & \forall u \in H^{1/2}(\Gamma)|_{\ker D}, \\ & \ker D = \operatorname{span}\{1\}. & \end{array}$$

3. In the case of $\Omega \subset \mathbb{R}^3$ the single layer potential is positive definite, i.e.

$$\exists \mu_V > 0 \quad \langle v, Vv\rangle_0 \geq \mu_V \|v\|^2_{H^{-1/2}(\Gamma)} \quad \forall v \in H^{-1/2}(\Gamma).$$

   If we consider $\Omega \subset \mathbb{R}^2$ then V will not be positive definite in general. Nevertheless, ellipticity can be obtained if the condition diam $\Omega < 1$ is satisfied. But this condition can easily be fulfilled by an appropriate scaling of the domain Ω.

If we apply the collocation method we first have to define a set of collocation points $\{y_i | i = 1, \ldots, N_h\}$. In the case of $\Omega \subset \mathbb{R}^2$ an appropriate choice could be $y_i = x_i + \frac{1}{2}(x_{i+1} - x_i)$ where x_i are the discretization nodes on the boundary Γ. Thus, the approximated solution v_h can be represented in the form

$$v_h(x) = \sum_{i=1}^{N_h} v_i \phi_i(x),$$

where the trial functions ϕ_j define the finite dimensional subspace

$$X_h := \text{span } [\phi_1, \ldots, \phi_{N_h}] \subset H^{-1/2}(\Gamma).$$

This leads to the collocation equations

$$Vv_h(y_j) = f(y_j) \qquad j = 1, \ldots, N_h. \tag{4}$$

If we are using piecewise constant trial functions ϕ_i for the Neumann data, the left-hand side of the collocation equations (4) can be rewritten in the form

$$\begin{aligned}(V_h v_h)(y_j) &= \int_{\Gamma_h} E(x,y_j) v_h(x) ds_x \\ &= \int_{\Gamma_h} E(x,y_j) \sum_{i=1}^{N_h} v_i \ \phi_i(x) ds_x \\ &= \sum_{i=1}^{N_h} v_i \int_{\Gamma_i} E(x,y_j) ds_x, \qquad j=1,\ldots,N_h,\end{aligned}$$

where Γ_i denotes the straight boundary element with the vertices x_{i-1} and x_i (see also Figure 2). Finally we arrive at the system of boundary element equations

$$K_h \underline{v}_h = \underline{f}_h \quad \text{in} \quad \mathbb{R}^{N_h} \tag{5}$$

with $(K_h)_{ji} = \int_{\Gamma_i} E(x,y_j) ds_x$, $\underline{v}_h = (v_i)_{i=1,\ldots,N_h}$ and $\underline{f}_h = (f(y_j))_{j=1,\ldots,N_h}$. Therefore, the collocation matrix V_h is fully populated and in general non-symmetric. Hence, it is a quite difficult task to solve (5) efficiently. Most software packages are using some direct solver or Krylov-subspace methods like GMRES and BiCGStab.

The Galerkin method

$$\langle V v_h, \phi_j \rangle_0 = \langle f, \phi_j \rangle_0 \qquad j=1,\ldots,N_h$$

again gives a system of the form (5), but now with a symmetric and positive definite (SPD) dense system matrix V_h. Although these SPD systems are rather easy to handle numerically (CG-algorithm), most commercial software packages are using collocation methods in order to avoid the evaluation of two integrals for a single matrix entry.

2.2 Adaptive Cross Approximation (ACA)

The adaptive cross approximation is a very elegant method to approximate matrices originating from Galerkin or collocation boundary element discretization. On the contrary to other matrix approximation techniques, an explicit description of the integral kernel is not necessary. More precisely, only a procedure for evaluating selected matrix entries has to be available, the rest are simple algebraic operations.

The basic idea is to decompose the computational domain into smaller clusters and classify them into a near-field domain and a far-field domain, respectively. Furthermore, the matrix blocks from the interaction of the far-field clusters can be approximated by low-rank matrices, i.e. for a matrix block $A \in \mathbb{R}^n_m$ the cost of a matrix-by-vector multiplication will decrease from $O(m*n)$ down to $O(r*(m+n))$, with r denotes the rank of the low-rank matrix.

Thus, we first take a closer look to the domain decomposition and the partitioning of the resulting system matrix. The notations connected with the ACA are directly adopted from [3].

Let us consider a set $\mathcal{D}_h \subset \mathbb{R}^d$, $d = 2, 3$, with

$$\mathcal{D}_h = \bigcup_{i=1}^{N_h} X_i \tag{6}$$

where X_i denotes the support of the trial functions ϕ_i. Then we have for the

1. Collocation Method

$$(K_h)_{ij} = \int_{X_j} E(x, y_i)\phi_j(x)ds_x, \tag{7}$$

   where y_i are the collocation points of subset X_i, and for the

2. Galerkin Method

$$(K_h)_{ij} = \int_{X_i}\int_{X_j} E(x, y)\phi_i(x)\phi_j(y)ds_x ds_y. \tag{8}$$

Based on geometrical information we split the index set $I = \{1, ..., N_h\}$ into index clusters $t_i \subset I$ with $I = \bigcup_i t_i$. This decomposition is done recursively and yields a hierarchical structured set of index clusters. The corresponding set of clusters $\mathcal{D}_i$ is called cluster tree and enables a subdivision of the system matrix into block matrices $A_i \in \mathbb{R}^{m_i \times n_i}$. In order to select the blocks that can be approximated by low-rank matrices, we give an admissibility condition that classifies clusters into a near-field part and a far-field part. The essential assumption is, that a cluster pair $(\mathcal{D}_1, \mathcal{D}_2)$ has a certain distance and therefore, we need not take into account the singularity of the boundary integral kernel.

Definition 2.1. *Let $(\mathcal{D}_1, \mathcal{D}_2)$ be a cluster pair with $\mathcal{D}_1, \mathcal{D}_2 \subset \mathbb{R}^d$, then $(\mathcal{D}_1, \mathcal{D}_2)$ is called η - admissible if*

$$\operatorname{diam} \mathcal{D}_2 \leq \eta \operatorname{dist}(\mathcal{D}_1, \mathcal{D}_2). \tag{9}$$

In addition, we define η-admissibility for index cluster pairs (t_1, t_2)

$$\operatorname{diam} \mathcal{D}_h^{t_1} \leq \eta \operatorname{dist}(\mathcal{D}_h^{t_1}, \mathcal{D}_h^{t_2}). \tag{10}$$

As customary $\operatorname{dist}(X, Y) = \inf\{|x - y|, x \in X, y \in Y\}$.

Now we want to give a brief overview of the algebraic matrix approximation. Hence, it is our aim to find a low-rank description of the system matrix K_h. In particular, for every block matrix arising from the interaction of two clusters $(\mathcal{D}_1, \mathcal{D}_2)$ we have to decide whether it is η-admissible or not. Assuming that a block matrix satisfies the condition (9) then only few matrix entries have to be calculated and thus the cost of storage and CPU-time for matrix-by-vector multiplication will decrease essentially. If a cluster pair does not fulfill η-admissibility the according matrix will be calculated directly.

Thus, in [4] we find the following Algorithm 1 with $\|.\|_F$ denoting the Frobe-

Algorithm 1 Fully Pivoted ACA

Fully Pivoted ACA
  let $A \in \mathbb{R}^{n\times m}$ be a given matrix
  set $R_0 = A$
  for all $k = 1, \dots$ **do**
    $(R_k)_{i_{k+1},j_{k+1}} = max_{i,j}|(R_k)_{i,j}|$
    $u_{k+1} = R_k e_{j_{k+1}}$
    $v_{k+1} = R_k^\top e_{i_{k+1}}$
    $\gamma_{k+1} = (R_k)^{-1}_{i_{k+1},j_{k+1}}$
    $R_{k+1} = R_k \; - \; \gamma_{k+1} u_{k+1} v_{k+1}^\top$
    if $\|R_k\|_F \leq \epsilon \|A\|_F \rightarrow$ **stop**
  end for

nius norm. The matrix S_r approximating A can be described by a sum of dyadic products of the vectors u_i, v_i

$$S_r = \sum_{i=1}^{r} u_i v_i^\top . \tag{11}$$

This variant of the ACA algorithm is called fully pivoted ACA. It is obvious that each entry of the given matrix A is necessary for the construction of S_r. Hence we need $O(n*m)$ arithmetical operations and that is clearly not efficient. To avoid this drawback it is possible to modify the fully pivoted ACA-algorithm in such a manner, that only the pivot element of a single row or column has to be found and in addition only the corresponding column or row has to be evaluated. This algorithm is denoted partially pivoted ACA.

The following paragraph gives results of the complexity analysis for the considered adaptive cross approximation technique. For more detailed proofs we again refer to [3, 2, 4]. The total complexity of the ACA-algorithm is mainly determined by the cost of the approximation procedure. In order to gain a desired accuracy $\|S_r - A\|_F \leq \epsilon$ for every admissible matrix block A the numerical effort is of order $O(\epsilon^{-\alpha} N_h^{1+\alpha})$ with an arbitrarily small positive α. In conclusion the memory demand is also of the same order as the arithmetical cost of constructing an approximation of the system matrix K_h.

Based on the splitting of the system matrix K_h into a near-field matrix K_h^{near} and a far-field matrix K_h^{far} we are able to construct an approximated system matrix $\widetilde{K}_h$. Since the proposed adaptive cross approximation technique provides a low-rank approximation of K_h^{far} consisting of submatrices which are η-admissible we finally obtain the result

$$\widetilde{K}_h = K_h^{near} + \widetilde{K}_h^{far}. \tag{12}$$

Starting from this representation we are able to present an appropriate construction of an algebraic multigrid method in the next section.

3 An Algebraic Multigrid Method

In the following we consider the interior Dirichlet problem for the Laplace equation. Therefore, the system matrix K_h originates from the single layer potential operator V, which is the most interesting case concerning our multigrid approach. Hence, we have to solve

$$K_h \underline{v}_h = \underline{f}_h \quad \text{in } \mathbb{R}^{N_h} \tag{13}$$

with $\underline{v}_h$ are the unknown Neumann data and $\underline{f}_h$ the corresponding right-hand-side. In the following we are discussing the AMG components by means of a twogrid algorithm. The indices h and H denote the fine grid and coarse grid quantities, respectively.

3.1 General Approach

One of the most efficient methods to solve (13) are multigrid methods. For this purpose geometric multigrid methods need a grid hierarchy, on the contrary the algebraic multigrid approach only needs the matrix information corresponding to the finest grid. In order to obtain a 'virtual' grid hierarchy there exist several different coarsening strategies [17, 26, 30]. Once the prolongation operators are appropriately defined, the properties of the fine grid system hand over to the coarser levels by projection.

In fact, the efficiency of AMG-methods depends on a clever interaction of smoothing sweeps on the fine level and coarse grid correction on the coarsest level. We have to recognize that multigrid methods are not able to solve (13) for an arbitrary symmetric positive definite matrix K_h. The components have to be adapted properly according to the underlying physical problem and variational formulation, respectively. In many cases, we will obtain better results (i.e. smaller number of iterations, smaller setup time, etc.) if we are using additional geometrical information. Thus, the setup procedure is performed by using an auxiliary matrix B_h, which contains the geometrical information in a certain sense. Preferably, we are using M-matrices, which allow the construction of a prolongation operator with some important properties [7, 9, 26, 25]. The numerical realization of the AMG-approach allows a proper treatment of anisotropies, polynomial trial functions of higher order and other applications.

3.2 Coarsening Strategies

A crucial point of AMG is the coarsening strategy in order to get a matrix hierarchy. Most coarsening techniques are based on the matrix graph, see [5, 8, 26, 30]. Standard coarsening strategies, which have been developed for sparse FE-matrices, will obviously fail for dense BE-matrices as well as for ACA BE-matrices. Hence, we are using the general approach proposed in [12, 23] and specify an auxiliary matrix $B_h \in \mathbb{R}^{N_h \times N_h}$ which represents the discretized geometry. In order to obtain a coarse auxiliary matrix B_H we have to define a corresponding prolongation operator $P_h^B : \mathbb{R}^{N_H} \mapsto \mathbb{R}^{N_h}$, where N_h and N_H denote the number of unknowns on the fine and coarse level, respectively. In addition, a prolongation operator P_h^K is constructed according to the system matrix K_h. Finally, appropriate restriction operators have to be formulated for both matrices. Furthermore, we assume B_h to be a sparse M-matrix which represents the underlying nodal mesh. An appropriate definition of the auxiliary matrix B_h is the so-called nodal distance matrix (14), which reflects the underlying mesh of the boundary element discretization. And since we are using piecewise constant trial functions in our 2D examples, the number of unknowns in B_h is related to the number of grid nodes.

$$(B_h)_{ij} = \begin{cases} -\frac{1}{\|e_{ij}\|_0} & \text{if } (i,j) \text{ isconnected,} \\ \delta_{ii} + \sum\limits_{k \neq i} \frac{1}{\|e_{ik}\|_0} & \text{if } i = j, \\ 0 & \text{otherwise.} \end{cases} \tag{14}$$

The vector e_{ij} denotes the distance vector between two grid-nodes x_i and x_j, i.e. $e_{ij} = x_j - x_i$ and δ_{ii} is some positive number or comes from the lumped mass matrix. In addition, the length of the vector e_{ij} is measured in the Euclidean norm $\|\cdot\|_0$. Once an auxiliary matrix B_h is available, we are able to introduce the following index sets on a pure algebraic level. Let ω_h be the set of unknowns on level h, then

$$\begin{aligned} N_h^i &= \{j \in \omega_h : |(B_h)_{ij}| \neq 0, i \neq j\}, \\ S_h^i &= \{j \in N_h^i : |(B_h)_{ij}| > coarse(B_h, i, j), i \neq j\}, \\ S_h^{i,T} &= \{j \in N_h^i : i \in S_h^j\}, \end{aligned}$$

where N_h^i is the set of neighbors around a node $i \in \omega_h$, further S_h^i denotes the set of strong connections, and finally $S_h^{i,T}$ is the set of nodes with a strong connection to node i. The decision whether a node i represents a coarse node or not will be done by the cut-off function $coarse(B_h, i, j)$, for details see [23] and references therein. Now we are able to apply a standard coarsening procedure on the auxiliary matrix B_h. This matrix will be interpreted as a description of a 'virtual' mesh (Figure 1) which can be split into two disjoint sets of grid nodes, i.e.

$$\omega_h = \omega_C \cup \omega_F, \qquad \omega_C \cap \omega_F = \emptyset,$$

where ω_C denotes the set of coarse grid nodes and ω_F the set of fine grid nodes. Usually the set of coarse grid nodes fulfills two properties,

1. no coarse grid nodes are connected directly,

2. the number of coarse grid nodes is as large as possible.

Once the set ω_C was built up, the grid on the coarse level H is defined by $\omega_H = \omega_C$.

In order to construct the coarse auxiliary matrix we use the Galerkin projection method. Therefore, we need a transfer operator $P_h^B : \mathbb{R}_H^N \mapsto \mathbb{R}_h^N$ which is constructed by giving several rules on the sets ω_h and ω_H. Let us apply this prolongation operator to the system matrix K_h as well (i.e. $P_h^K \equiv P_h^B \equiv P_h$). Furthermore, let us mention that the Galerkin method assumes the restriction

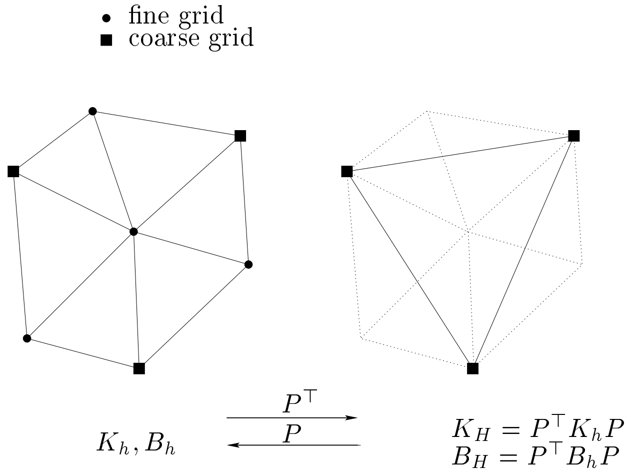

Figure 1: Coarsening

operator to be transposed to the prolongation operator $R_h \equiv P_h^\top$. Then we have

$$K_H = P_h^\top K_h P_h \quad \text{and} \quad B_H = P_h^\top B_h P_h. \tag{15}$$

Taking into account the index sets mentioned above, we are able to give an explicit description of the prolongation operator:

$$(P_h)_{ij} = \begin{cases} 1 & i = j \in \omega_C, \\ \frac{1}{|\omega_C \cap S_h^{i,T}|} & i \in \omega_F,\, j \in \omega_C, \\ 0 & \text{otherwise.} \end{cases} \tag{16}$$

Now we have all ingredients to perform a setup phase in order to build up the matrix hierarchy and the corresponding transfer operators. A recursive application results in the well-known V-cycle, which is presented in Algorithm 2. The coarsest level is denoted by the variable COARSELEVEL.

Algorithm 2 MG($\underline{u}_\ell, \underline{f}_\ell, \ell$)

MG($\underline{u}_\ell, \underline{f}_\ell, \ell$)
  if $\ell =$ COARSELEVEL **then**
    define $\underline{u}_\ell = (K_\ell)^{-1}\underline{f}_\ell$ by some coarse grid solver
  else
    smooth ν_F times on $K_\ell \underline{u}_\ell = \underline{f}_\ell$
    calculate the defect $\underline{d}_\ell = \underline{f}_\ell - K_\ell \underline{u}_\ell$
    restrict the defect to the next coarser level $\ell + 1$: $\underline{d}_{\ell+1} = P_\ell^\top \underline{d}_\ell$
    set $\underline{u}_{\ell+1} \equiv 0$
    call MG($\underline{u}_{\ell+1}, \underline{d}_{\ell+1}, \ell + 1$)
    prolongate the correction $\underline{s}_\ell = P_\ell \underline{u}_{\ell+1}$
    update the solution $\underline{u}_\ell = \underline{u}_\ell + \underline{s}_\ell$
    smooth ν_B times on $K_\ell \underline{u}_\ell = \underline{f}_\ell$
  end if

If we apply Galerkin's prolongation technique on ACA-matrices, their specific representation have to be taken into account. According to (12) in Section 2.2 an ACA-compressed boundary element matrix $\widetilde{K}_h$ consists of a sparse near-field matrix K_h^{near} and the approximated far-field matrix

$$\widetilde{K}_h^{far} = \sum_{i=1}^{N_B} \sum_{j=1}^{r_i} u_j^i \; (v_j^i)^\top, \tag{17}$$

Therefore, the Galerkin method immediately leads us to matrices corresponding to the coarse level

$$\begin{aligned} K_H^{near} &= P_h^\top K_h^{near} P_h, & (18)\\ \widetilde{K}_H^{far} &= \sum_{i=1}^{N_B} \sum_{j=1}^{r_i} P_h^\top u_j^i \; (P_h^\top v_j^i)^\top. & (19) \end{aligned}$$

Thus, the approximated coarse system matrix $\widetilde{K}_H$ has also a near-field contribution K_H^{near} and a low-rank far-field matrix $\widetilde{K}_H^{far}$. Due to the exact preserving of representation (12) on the coarser grid, we are able to use the same ACA-datastructures in our numerical realization.
Let us mention, that applying direct solvers on the coarsest level requires an explicit evaluation of our ACA-matrix.

3.3 Smoothing Operators

The smoothing process strongly depends on the type of the boundary integral operator. Since the hypersingular operator $D : H^{1/2}(\Gamma) \mapsto H^{-1/2}(\Gamma)$ defined by (3) is a pseudo-differential operator of the order plus one, the eigenvectors and eigenvalues behave as elliptic differential operators. Small eigenvalues correspond to low-frequency eigenfunctions, whereas large eigenvalues correspond to high-frequency eigenfunctions. Therefore, standard smoothers like the damped Jacobi method or Gauss-Seidel sweeps are appropriate.

In order to construct a smoothing procedure for the single layer potential operator $V : H^{-1/2}(\Gamma) \mapsto H^{1/2}(\Gamma)$ defined by(2) we have to take into account that eigenfunctions of high frequencies correspond to small eigenvalues. In particular, we have to introduce a smoother for pseudo-differential operators of order minus one. The following idea was suggested by J. Bramble, Z. Leyk and J. Pasciak, for details see [6]. In order to reduce the highly oscillating components of the error we introduce a matrix $A_h \in \mathbb{R}^{N_h \times N_h}$ being some discretization of the Laplace-Beltrami operator on the boundary Γ. Performing a smoothing iteration of the form

$$\underline{u}_h \leftarrow \underline{u}_h + \tau_h \cdot A_h(\underline{f}_h - \widetilde{K}_h \underline{u}_h) \tag{20}$$

leads us to an appropriate smoothing procedure, provided that the damping parameter τ_h satisfies the inequality $0 < \tau_h \leq 1/\bar{\lambda}$ where $\overline{\lambda}$ denotes the largest eigenvalue of the generalized eigenvalue problem

$$\widetilde{K}_h \underline{\phi} = \lambda A_h^{-1} \underline{\phi}. \tag{21}$$

We presented a proper method for an easy construction of an estimate for the upper eigenvalue $\overline{\lambda}$ in [19] based on [21].

Let us remark, that the difference operator A_h in (20) can be set identically equal to the auxiliary matrix B_h defined by (14). This procedure can be realized at each multigrid level.

3.4 Complexity analysis

Our aim in this section is to analyze the complexity for realizing the proposed AMG-preconditioner, provided that the BE-matrices originated from the ACA approximation. It turns out, that the cost of the preconditioning procedure is exactly bounded by the cost of an ACA matrix-by-vector multiplication up to some constant factor. In order to show this it is necessary to study the performance of a single projected ACA matrix-by-vector multiplication on each level ℓ.

As shown in [4] the cost of an ACA matrix-by-vector multiplication on the finest grid behaves like $O(\epsilon^{-\alpha} N_h^{1+\alpha})$. Since the near-field part of the matrix is sparse, we can assume that Galerkin projection yields a sparse matrix again on the coarser level.

In the case of the approximated far-field matrix $\widetilde{K}_h^{far} = \sum_i^{N_B} \sum_{j=1}^{r_i} u_j^i (v_j^i)^\top$ it is easy to see that $\sum_{i=1}^{N_B} r_i(m_i + n_i)$ operations are necessary for a single multiplication. Here, m_i and n_i denote the number of non-zero entries (NNE) of the vectors u_j^i and $(v_j^i)^\top$, which approximate a submatrix of dimension $\mathbb{R}^{m_i \times n_i}$. Thus, after the restriction process considered in [19] the non-zero entries of a single vector u_j^i will be halved: $\mathrm{NNE}(P^\top u_j^i) = \mathrm{NNE}(u_j^i)/2$. This is true in the 2D case (in 3D the vector entries will be reduced by some parameter $\tau < 1$, e.g. $\tau = 1/4$). In conclusion, the effort for a multiplication with $\widetilde{K}_h^{far}$ reduces on each coarser level by the factor 2.

Now we are able to analyze the number of arithmetical operations on each AMG-level $\ell \in \{1, \ldots, L\}$ and in conclusion for the total complexity of our AMG-preconditioner. Considering the AMG components on each level leads to the following results:

1. Smoother: The essential operations are one multiplication with the corresponding matrices K_ℓ and B_ℓ. This leads to $\frac{\nu}{2^{\ell-1}} O(\epsilon^{-\alpha} N_h^{1+\alpha})$ arithmetical operations, with ν the number of smoothing steps.

2. Prolongation (Restriction): The transfer operators are defined locally and therefore the application of them is of order $\frac{1}{2^{\ell-1}} O(N_h)$.

3. Galerkin projection: Since the effort of a matrix-by-vector multiplication reduces each level and since the prolongation (restriction) operators are sparse, the computation of the coarser matrix can be done at least in $\frac{1}{2^{\ell-1}} O(\epsilon^{-\alpha} N_h^{1+\alpha})$. Let us remark, that the Galerkin projection method performed for ACA matrices contains some potential for speeding-up the construction of the coarse system matrices. For instance, one could think about the reduction of the number of blocks on the coarser levels. But in this paper, we only implemented a straightforward coarsening strategy,

which preserves the number of admissible blocks on each multigrid level that probably would not be necessary.

4. Coarse grid correction: Since the dimension of the coarsest level is kept small, the solution of the coarse grid system takes $c_C = O(1)$ operations.

Finally, we determine the number of arithmetical operations $Q(M_L)$ for one single algebraic multigrid step (V-cycle) on the finest grid L:

$$\begin{aligned}
Q(M_L) &\leq (\nu c_1 \epsilon^{-\alpha} N_h^{1+\alpha} + c_2 N_h) + \frac{1}{2}(\nu c_1 \epsilon^{-\alpha} N_h^{1+\alpha} + c_2 N_h) + \\
&\quad + \frac{1}{2^2}(\nu c_1 \epsilon^{-\alpha} N_h^{1+\alpha} + c_2 N_h) + \dots \\
&\quad \dots + \frac{1}{2^{L-2}}(\nu c_1 \epsilon^{-\alpha} N_h^{1+\alpha} + c_2 N_h) + c_C \\
&= (\nu c_1 \epsilon^{-\alpha} N_h^{1+\alpha} + c_2 N_h) \sum_{j=0}^{L-2} (\frac{1}{2})^j + c_C \\
&= (\nu c_1 \epsilon^{-\alpha} N_h^{1+\alpha} + c_2 N_h) \cdot (2 - (\frac{1}{2})^{L-2}) + c_C \\
&= O(\epsilon^{-\alpha} N_h^{1+\alpha}).
\end{aligned}$$

Therefore, we have shown that performing one algebraic multigrid cycle is of the same order as the effort needed for a single ACA matrix-by-vector multiplication.

4 Numerical Studies

In order to show the efficiency of the suggested AMG approach we present some results in 2D. The collocation boundary element matrices are generated by the software package OSTBEM developed by O. Steinbach [28] and the AMG-preconditioner is realized within the software package PEBBLES [15]. The proposed technique is used as a preconditioner C_h in the CG-algorithm and in the BiCGStab-algorithm, respectively (see [16, 22]). In particular, we use the symmetric $V(1,1)$-cycle, i.e. one pre-smoothing and one post-smoothing step with the BLP-smoother per iterative cycle. The iteration error is measured in the $K_h C_h^{-1} K_h$ energy norm for the CG-solver, and the defect-test is used in the BiCGStab-solver. All calculations are done on a PC with 1800 MHz AMD Athlon(tm) processor.

4.1 Disk and L-Shape Domain (Comparisons)

First we consider $\Omega \subset \mathbb{R}^2$ to be a disk with radius 0.1 and thus, the boundary $\Gamma = \partial\Omega$ is the circle with the same radius (Figure 2). Such a diameter induces a positive definite single layer potential operator (actually diam $\Omega < 1$ would be sufficient). Assuming an uniform discretization we additionally obtain symmetric BE collocation matrices. Thus, the CG-solver can be used. The collocation points are taken as the middle of the direct connection of two neighbor grid-nodes.

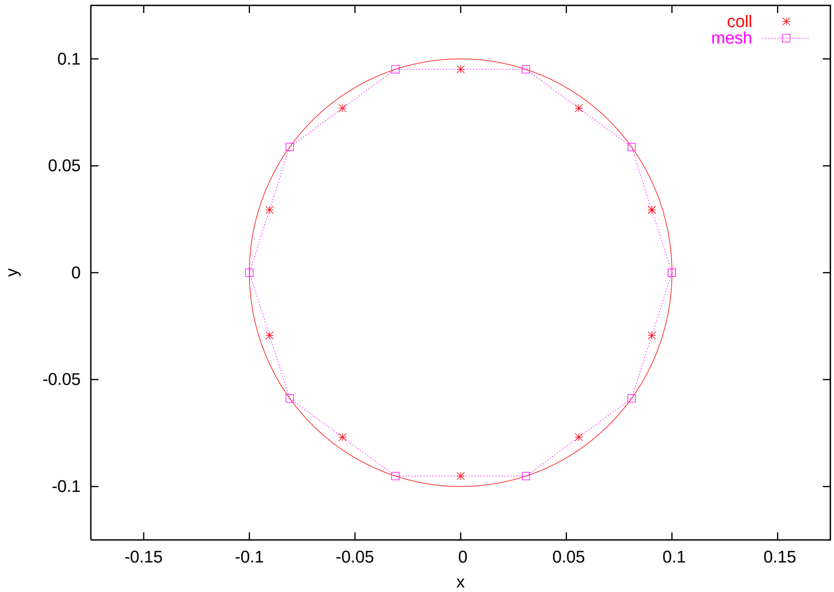

Figure 2: Circle

First of all, we compare the ACA results with the results obtained from the case where the corresponding dense BE matrices were used. The superiority of the ACA approach is impressively reflected in Table 1. One can clearly

Table 1: Comparison: sparse approximated BE-matrix

Number of Unknowns	Setup (sec) full BEM	setup (sec) ACA BEM	CG-Cycle (sec) full BEM	CG-Cycle (sec) ACA BEM
1024	0.60	0.41	0.09	0.03
2048	2.46	0.62	0.36	0.07
4096	10.04	1.06	1.42	0.16
8192	45.98	2.01	5.78	0.37

observe the expected behavior of the CPU-times for one single CG-cycle and for constructing the matrix hierarchy, respectively. In the case of dense BE-matrices a growth of order $O(N_h^2)$ can be noticed, whereas the algorithm shows linear increase $O(N_h)$ in the case of ACA BE-matrices.

Of course, the disk is a very simple geometry. Therefore, in the next example, we are faced with an L-shaped domain $\Omega \subset \mathbb{R}^2$, see Figure 3. Once again we assume diam $\Omega < 1$ in order to obtain a positive definite single layer potential operator. In this example we are using the Galerkin discretization method which provides symmetric and positive definite matrices that allows us to use the CG-algorithm again. Moreover, we now give some comparison results concerning different preconditioning techniques. The reference values are generated by the BEM-software package OSTBEM again. The preconditioner used therein is a slightly modified hypersingular operator. In the following cal-

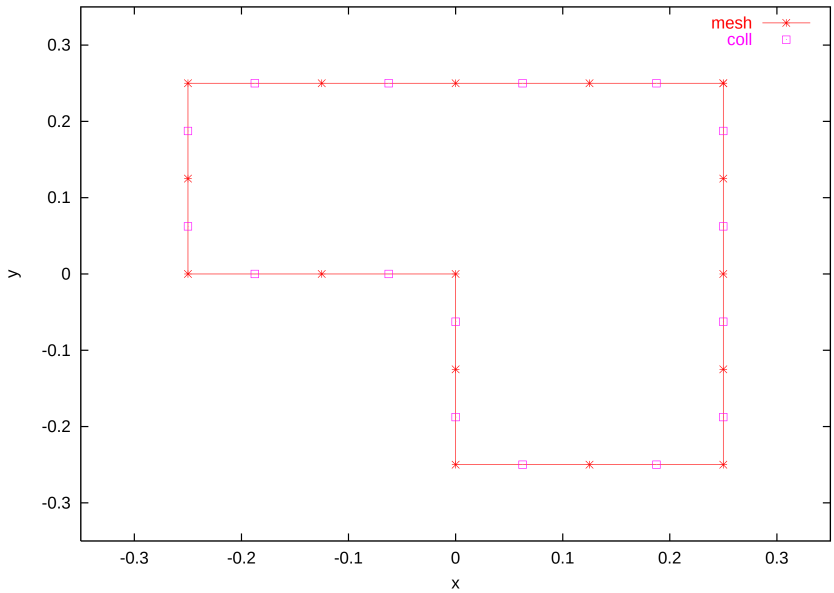

Figure 3: L-Shape

culations the relative accuracy is set to $\epsilon_{rel} = 10^{-8}$ and the evaluation is done at the coordinate $x^* = (0.125, 0.125)$.

Firstly, we discuss the case of a linear Dirichlet boundary condition $g(x) = 4(x_1 - x_2)$. Since the resulting Neumann data are included in the trial space, we may expect results independent of the characteristic mesh size h. Indeed, as we can see in Table 2 and Table 3, the results at $x^* = (0.125, 0.125)$ remain almost constant.

Taking a look at the number of iterations, we clearly observe a smaller number of iterations in the case of the AMG-preconditioner. Compared with the hypersingular operator preconditioner, the number of iterations was obviously reduced. Moreover, the setup of the matrix hierarchy is faster than computing the hypersingular preconditioner. Considering the Neumann data we notice constant L_2-errors as expected (up to the accuracy of the CG-algorithm).

The second numerical example deals with the Dirichlet condition $g(x) = \log|x - y|$, $y \notin \overline{\Omega}$. In our example we choose $y = (-0.1, -0.1)$. In Table 4 and Table 5 one can notice, that the number of iterations for the AMG-based solver is clearly smaller once again. The setup times confirm the previous results as well. Considering the L_2-error of the Neumann data, we can observe a decrease of order $O(h)$ for the Galerkin approach. The convergence rate for u_h in x^* is of order $O(h^2)$. More convergence estimates and their theoretical background can be found in [11, 27].

Table 2: Solver data, C_h : AMG

Number of Unknowns	Iterations	Setup (sec)	Solution (sec)	L_2-error of $\frac{\partial u}{\partial n}$ on Γ	$\lvert u(x^*) - u_h(x^*)\rvert$
128	4	0.00	0.01	2.16e-7	2.03e-11
256	5	0.02	0.05	1.87e-9	3.78e-13
512	5	0.06	0.18	3.57e-9	1.47e-13
1024	5	0.26	0.73	4.27e-8	4.54e-13

Table 3: Solver data, C_h : Hypersingular Operator

Number of Unknowns	Iterations	Matrix (sec)	Solution (sec)	L_2-error of $\frac{\partial u}{\partial n}$ on Γ	$\lvert u(x^*) - u_h(x^*)\rvert$
128	23	0.02	0.03	2.92e-7	1.78e-16
256	22	0.11	0.15	5.82e-7	2.59e-17
512	22	0.43	0.60	6.51e-7	2.65e-16
1024	21	1.74	2.36	1.26e-6	9.16e-16

Table 4: Solver data, C_h : AMG

Number of Unknowns	Iterations	Setup (sec)	Solution (sec)	L2-error of $\frac{\partial u}{\partial n}$ on Γ	$\lvert u(x^*) - u_h(x^*)\rvert$
128	4	0.00	0.01	1.21e-1	1.83e-4
256	5	0.02	0.05	6.02e-2	4.66e-5
512	5	0.06	0.18	3.00e-2	1.18e-5
1024	5	0.26	0.73	1.50e-2	2.96e-6

Table 5: Solver data, C_h : Hypersingular Operator

Number of Unknowns	Iterations	Matrix (sec)	Solution (sec)	L2-error of $\frac{\partial u}{\partial n}$ on Γ	$\lvert u(x^*) - u_h(x^*)\rvert$
128	23	0.02	0.03	1.71e-1	1.87e-4
256	23	0.11	0.15	8.52e-2	4.73e-5
512	22	0.43	0.60	4.25e-2	1.19e-5
1024	22	1.74	2.36	2.12e-2	2.98e-6

4.2 Disk and L-Shape Domain (large-scale)

In the previous paragraph the number of unknowns were bounded by using the classical boundary element method which provides dense matrices. Now, we are going to perform experiments only with pure ACA-matrices. Therefore, we are able to refine the mesh essentially. In the remaining, the iterative solution process starts with a random initial guess $\underline{u}_h^0$ and the right-hand-side $\underline{f}_h = 0$, the system matrix is generated by the collocation discretization method. Table 6 gives an impression about the quality of the AMG-preconditioner. We

Table 6: ACA: Quality of AMG Preconditioner

Number of Unknowns	Disk: CG $\kappa(C_h^{-1}K_h)$	L-Shape: BiCGStab Iterations
10000	1.04	3
20000	1.04	3
40000	1.04	3
80000	1.04	3

can observe almost constant condition numbers close to one for the disk and constant iteration numbers for the L-shaped domain, respectively. This observation leads us to the conclusion that our AMG-preconditioner in connection with the ACA-technique yields a very efficient solver. This is finally confirmed by the CPU-time needed. Several key data for large numbers of unknowns are presented in Table 7.

Table 7: ACA: CPU Times for Disk and L-Shape

Disk N_h	Setup Coll / Gal	CG Cycle Coll / Gal	Iterations Coll / Gal
20000	4.9/4.9	1.16/1.15	5/4
40000	10.5/10.4	2.59/2.55	5/4
80000	22.7/22.7	5.64/5.64	5/4
160000	50.0/50.6	13.9/14.7	5/4

L-Shape N_h	Setup Coll / Gal	BiCGStab/CG Cycle Coll / Gal	Iterations Coll / Gal
20000	5.5/5.5	1.94/1.26	4/5
40000	11.5/11.5	4.25/2.77	4/5
80000	25.0/25.0	8.27/6.02	3/5
160000	56.1/56.1	21.1/15.2	3/5

As expected for a sparse BE-matrix, the calculation time for one single iteration grows in almost $O(N_h)$. One can observe that the computing time for the matrix hierarchy (setup-time) is of same order than total iteration time.

5 Conclusions and Further Remarks

In this paper we presented an algebraic multigrid approach for the solution of large-scale boundary element equations. For that purpose an approximation of the boundary element matrices is absolutely essential. Our numerical experiments has been realized by the adaptive cross approximation technique which guarantees, that the effort for storing the matrices and for a single matrix-by-vector multiplication can be reduced to almost $O(N_h)$. If we are using collocation methods for discretizing the single layer potential, we will obtain non-symmetric BE-matrices in general. Solving the corresponding linear equation system requires some Krylov-subspace methods.

Due to the sparse representation of our matrices, we had to adapt each component of our AMG-algorithm properly. In order to set up the matrix hierarchy and the according transfer operators an auxiliary matrix was constructed. Moreover, it turns out that the same matrix can be used in the BLP-smoother for the single layer potential.

The overall algorithm provides interesting numerical results. One can notice that the small iteration numbers for the CG and BiCGStab-solver confirm the quality of our AMG-preconditioner. In addition, the CPU time for a single iterative step almost grows like $O(N_h)$ because we are concerned with sparse approximated BE-matrices.

Finally, we mention that the presented algebraic multigrid technique can be enhanced in order to solve boundary element equations arising from 3D boundary value problems. In the case of the single layer potential operator, only an appropriate realization of the Laplace-Beltrami operator on the corresponding surface of a 3D domain will ensure an optimal smoothing process which is necessary for a fast convergence of the AMG.

References

[1] R. A. Adams, *Sobolev Spaces*, Pure and Applied Mathematics, Academic Press, New York, 1975.

[2] M. Bebendorf, *Approximation of boundary element matrices*, Numerische Mathematik **86** (2000), 565–589.

[3] ———, *Effiziente numerische Lösung von Randintegralgleichungen unter Verwendung von Niedrigrang-Matrizen*, Ph.D. thesis, Unviversität Saarbrücken, 2000.

[4] M. Bebendorf and S. Rjasanov, *Adaptive low-rank approximation of collocation matrices*, Computing **70** (2003), no. 1, 1–24.

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

[6] J. H. Bramble, Z. Leyk, and J. E. Pasciak, *The Analysis of Multigrid Algorithms for Pseudo-Differential Operators of Order Minus One*, Math. Comp. **63** (1994), no. 208, 461–478.

[7] A. Brandt, *Algebraic multigrid theory: The symmetric case*, Appl. Math. Comput. **19** (1986), 23–56.

[8] ———, *Generally highly accurate algebraic coarsening*, Elec. Trans. Num. Anal. **10** (2000), 1–20.

[9] A. Brandt, S. McCormick, and J. W. Ruge, *Algebraic multigrid (AMG) for automatic multigrid solution with application to geodetic computations*, Report, Inst. Comp. Studies Colorado State Univ., 1982.

[10] J. Carrier, L. Greengard, and V. Rokhlin, *A fast adaptive multipole algorithm for particle simulations.*, SIAM J. Sci. Statist. Comput. **9(4)** (1988), 669–686.

[11] G. Chen and J. Zhou, *Boundary element methods*, Computational Mathematics and Applications, Academic Press, 1992.

[12] G. Haase, U. Langer, S. Reitzinger, and J. Schöberl, *A general approach to algebraic multigrid*, SFB-Report 00-33, Johannes Kepler University Linz, SFB "Numerical and Symbolic Scientific Computing", 2000, (can be downloaded from http://sfb013.uni-linz.ac.at).

[13] W. Hackbusch, *A sparse matrix arithmetic based on $\mathcal{H}$-matrices*, Computing **62** (1999), no. 2, 89–108.

[14] W. Hackbusch and Z. P. Nowak, *On the fast matrix multiplication in the boundary element method by panel clustering*, Numer. Math. **54** (1989), no. 4, 463–491.

[15] Johannes Kepler University Linz, *PEBBLES - User's Guide*, SFB "Numerical and Symbolic Scientific Computing", 1999, http://www.numa.uni-linz.ac.at/Research/Projects/pebbles.html.

[16] M. Jung and U. Langer, *Applications of multilevel methods to practical problems*, Surveys Math. Indust. **1** (1991), 217–257.

[17] F. Kickinger, *Algebraic multigrid for discrete elliptic second-order problems*, Multigrid Methods V. Proceedings of the 5th European Multigrid conference (W. Hackbusch, ed.), Springer Lecture Notes in Computational Science and Engineering, vol. 3, 1998, pp. 157–172.

[18] C. Lage and C. Schwab, *Wavelet galerkin algorithms for boundary integral equations*, SIAM J. Sci. Comput. **20** (1999), 2195–2222.

[19] U. Langer, D. Pusch, and S. Reitzinger, *An efficient preconditioner for boundary element matrices based on algebraic multigrid methods*, SFB-Report 01-30, University of Linz, December 2001.

[20] ———, *Efficient Preconditioners for Boundary Element Matrices Based on Grey-Box Algebraic Multigrid Methods*, International Journal for Numerical Methods in Engineering **58** (2003), no. 13, 1937–1953.

[21] U. Langer and W. Queck, *Preconditioned Uzawa-type iterative methods for solving mixed finite element equations*, 3/1987, Wissenschaftliche Schriftenreihe der TU Karl-Marx-Stadt, Karl-Marx-Stadt, 1987.

[22] G. Meurant, *Computer solution of large linear systems*, Studies in Mathematics and its Applications, vol. 28, Elsevier, 1999.

[23] S. Reitzinger, *Algebraic Multigrid Methods for Large Scale Finite Element Equations*, Reihe C - Technik und Naturwissenschaften, Universitätsverlag Rudolf Trauner, Linz, 2001.

[24] V. Rokhlin, *Rapid solution of integral equations of classical potential theory.*, J. Comput. Phys. **60(2)** (1985), 187–207.

[25] J. W. Ruge and K. Stüben, *Efficient solution of finite difference and finite element equations*, Multigrid Methods for integral and differential equations (D. Paddon and H. Holstein, eds.), 3, Clarendon Press, Oxford, 1985, pp. 169–212.

[26] ______, *Algebraic multigrid (AMG)*, Multigrid Methods (S. McCormick, ed.), Frontiers in Applied Mathematics, vol. 5, SIAM, Philadelphia, 1986, pp. 73–130.

[27] O. Steinbach, *Gebietszerlegungsmethoden mit Randintegralgleichungen und effiziente numerische Lösungsverfahren für gemischte Randwertprobleme*, Ph.D. thesis, University of Stuttgart, 1996.

[28] ______, *OSTBEM - A boundary element software package*, University of Stuttgart, 2000.

[29] ______, *Numerische Näherungsverfahren für elliptische Randwertprobleme: Finite Elemente und Randelemente*, Teubner-Verlag, Darmstadt, 2003.

[30] P. Vanek, J. Mandel, and M. Brezina, *Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems*, Computing **56** (1996), 179–196.