Max-Planck-Institut
für Mathematik
in den Naturwissenschaften
Leipzig

Approximate Inverse Preconditioning of FE Systems for Elliptic Operators with non-smooth Coefficients

by

Mario Bebendorf

Preprint no.: 7 2004

Approximate Inverse Preconditioning of FE Systems for Elliptic Operators with non-smooth Coefficients*

Mario Bebendorf†

February 26, 2004

Abstract

A new class of approximate inverse preconditioners for large finite element stiffness matrices arising from elliptic partial differential operators is introduced which guarantees a bounded number of iterations of the conjugate gradients method. This preconditioner can be generated, stored and multiplied by a vector with almost linear complexity. Since the proposed preconditioner is robust with respect to varying coefficients and does not need a grid hierarchy it may be used as a black-box method.

Mathematics Subject Classification (2000): 35C20, 65F05, 65F50, 65N30
Keywords: Approximate inverse, preconditioning, non-smooth coefficients

1 Introduction

In this article a new preconditioning technique for large finite element stiffness matrices of general second order elliptic partial differential operators

$$Lu = -\operatorname{div}\left[A(x)\nabla u + bu\right] + c\cdot\nabla u + du \tag{1.1}$$

on a bounded Lipschitz domain $\Omega \subset \mathbb{R}^n$ with coefficients $a_{ij}, b_i, c_j, d \in L^\infty(\Omega)$, $i,j = 1,\dots,n$, is proposed. The Galerkin matrix of operators (1.1) is sparse but has a bandwidth of order $N^{1-1/n}$, where N is the number of degrees of freedom. Due to fill-in effects direct methods are therefore well suited for small problem sizes, but are not competitive if N is large. In the latter case usually iterative methods such as Krylov subspace methods are more efficient. The advantage of these solution techniques is that the coefficient matrix enters the computation only through the matrix-vector product. The convergence rate usually depends on the condition number of the coefficient matrix. Since L is a second order operator the condition number of the FE stiffness matrix S grows like $N^{2/n}$ for large N, but also depends significantly on the coefficients of L. By preconditioning, i.e., replacing $Sx = b$ by $SCu = b$, $x = Cu$, or $CSx = Cb$, where C is a right or a left preconditioner, respectively, one tries to improve the condition number and hence the convergence properties of the Krylov subspace method.

If L has smooth coefficients there are many developed and well known preconditioning techniques, e.g. multigrid or the Bramble, Pasciak and Xu (BPX) algorithm [8]. If the coefficients

*This work was supported by the DFG priority program SPP 1146 "Modellierung inkrementeller Umformverfahren"

†Fakultät für Mathematik und Informatik, Universität Leipzig, Augustusplatz 10/11, D-04109 Leipzig, Germany, bebendorf@math.uni-leipzig.de

are non-smooth these methods might still work but suffer from poor convergence rates, if the respective method is not adapted to the coefficients. The aim of this article is to present a preconditioner, which adapts itself to the coefficients and, what is even more important for practice, does not need a grid hierarchy.

For preconditioning the best choice for C would obviously be $C = A^{-1}$. However, the computation of A^{-1} is usually more costly than solving $Ax = b$ for x. Hence, not A^{-1} but an approximation of A^{-1} is used for preconditioning. This idea gives rise to the class of approximate inverse preconditioners. Their main advantage is that if many systems with different right hand sides have to be solved computing the preconditioner explicitly will be more efficient than an implicit preconditioner as given for example by the multigrid method. Additionally, the approximate inverse can be easily updated if many systems with coefficient matrices differing in only a small number of entries have to be solved. The inverse of a FE stiffness matrix however is a dense matrix in general, since it corresponds to the inverse operator L^{-1}, which can be represented as a non-local integral operator

$$L^{-1}\varphi(x) = \int_\Omega G(x,y)\varphi(y)\,\mathrm{d}y, \tag{1.2}$$

where G denotes the Green function of L. In order to avoid large dense matrices so called sparse approximate inverse (SAI) preconditioners, see [7, 9, 10], have been introduced. In this approach for a given matrix A the quantity $\|I - AC\|_F$ is usually minimized for matrices C having a certain sparsity pattern. Our aim, in contrast, is to use a dense approximation of the inverse of the FE stiffness matrix. Therefore, the approximant will be enabled to capture more properties of A^{-1} that are important for preconditioning. Nevertheless, it will be possible to generate, store and multiply the approximant by a vector with almost linear complexity.

In the last years fast methods for the treatment of large dense matrices have considerably spread. After the introduction of the fast multipole method [15], numerous methods based on degenerate approximations

$$G(x,y) \approx \sum_{\ell=1}^{k} u_\ell(x)v_\ell(y), \quad x \in D_1,\, y \in D_2 \tag{1.3}$$

on a pair of domains (D_1, D_2) lying in a certain distance relative to their diameters, have been developed. Looking at (1.3) from an algebraic point of view this means nothing but a low-rank approximation of the associated block $A_{s\times t}$ in the rows and columns $s, t \subset \{1, \ldots, N\}$ of A:

$$A_{s\times t} \approx UV^T,$$

where $U \in \mathbb{R}^{s\times k}$, $V \in \mathbb{R}^{t\times k}$ and k is small compared with $|s|$ and $|t|$. The ideas of the fast multipole method originally aiming at an efficient approximate evaluation of matrix-vector products have recently been extended to a structure called hierarchical ($\mathcal{H}$-) matrices [13, 14], see also [18]. Basically, these are matrices that are low-rank on each block of a certain partition stemming from a recursive subdivision of the set of matrix indices. In addition to the efficient matrix-vector multiplication this structure provides approximate operations like matrix addition, matrix-matrix multiplication and matrix inversion of dense matrices with almost linear complexity. Multiplication with $\mathcal{H}$-matrices can easily be parallelized, see [4]. Furthermore, $\mathcal{H}$-matrices can be stored in an almost linear amount of units of memory. The structure of $\mathcal{H}$-matrices has originally been applied to integral equations, see [3, 5]. Although FE stiffness matrices are sparse, it can be seen that they can be represented by $\mathcal{H}$-matrices without approximation. Since the bandwidth of a stiffness matrix is of order $N^{1-1/n}$, the rank of each block of S^{-1} above or below the diagonal

will be of the same order. Since the band has many zeros in it, it may be expected that to approximate each block a much lower rank is sufficient. Recently [6], it was shown that the inverse of a FE stiffness matrix for operators of type (1.1) can be approximated by an $\mathcal{H}$-matrix with blockwise rank of order $\log^{n+3} N$. For this purpose it was proved that the Green function in (1.2) can be approximated in the sense of (1.3). Interestingly, this approximation is very robust with respect to varying coefficients.

In a recent publication [11] a similar approach for preconditioning was investigated. The inverse of A is approximated on a matrix partition similar to the partition of a hierarchical matrix. But in contrast to $\mathcal{H}$-matrices, a blockwise constant approximation of the inverse is used, i.e., only one real number represents all entries of each block. Obviously, this needs less memory than storing a low-rank matrix and for preconditioning the approximant does not have to be too accurate. However, a certain accuracy is needed and the block constant approach is not adaptive, i.e., there is no possibility to guarantee a given accuracy on each block. In our approach the approximate inverse will be calculated with a prescribed accuracy, although this accuracy will not be too high. This will enable us to guarantee a bounded number of iterations of the conjugate gradients method. The procedure in [11] to compute the real number for each block cannot be used if one tries to approximate the block by a matrix of low-rank. For obtaining our approximate inverse preconditioner the $\mathcal{H}$-arithmetic will be employed.

The structure of the rest of this article is as follows: In Section 2 a brief introduction to the structure of $\mathcal{H}$-matrices will be given. All results and notations from the field of $\mathcal{H}$-matrices necessary for this article will be presented. Section 3 is devoted to the theory of preconditioning. A condition on the accuracy of the approximate inverse will be derived which guarantees the existence of an eigenvalue cluster leading to super-linear convergence of the preconditioned conjugate gradient method. In Section 4 this condition is used when generating the approximate inverse by an approximate block inversion procedure known as the $\mathcal{H}$-matrix inversion. In this purely algebraic way, a preconditioner can be obtained from any stiffness matrix. Finally, in Section 5 numerical results for elliptic partial differential operators with non-smooth coefficients will be presented. It will be seen that the proposed preconditioner can be computed, stored and multiplied by a vector with almost linear complexity. Furthermore, it can be seen that with this preconditioner a bounded number of iterations of the conjugate gradients method can be achieved.

2 Hierarchical matrices

This section gives a brief overview over the structure of $\mathcal{H}$-matrices originally introduced by Hackbusch et al. [13, 14]. We will describe the two principles on which the efficiency of $\mathcal{H}$-matrices is based. These are the hierarchical partitioning of the matrix into blocks and the blockwise restriction to low-rank matrices. These principles were also used in the mosaic-skeleton method [18].

In this article we will be interested in matrices $M \in \mathbb{R}^{N\times N}$ with indices

$$m_{ij} = a(\varphi_j, \varphi_i), \quad i, j = 1, \dots, N,$$

where φ_i are basis functions with support $X_i := \operatorname{supp} \varphi_i$, $i \in I := \{1, \dots, N\}$ and a is a bilinear form. Note that this does not imply that M is sparse. Also the variational formulation of non-local operators is of this form. In order to be able to approximate each block $b = s \times t$, $s, t \subset I$, of M by a matrix of low-rank, i.e.,

$$M_b \approx UV^T, \quad U \in \mathbb{R}^{s\times k}, V \in \mathbb{R}^{t\times k},$$

where k is small compared with $|s|$ and $|t|$, b has to satisfy a certain condition, which is caused by the operator L hidden in a. In the field of elliptic partial differential operators the corresponding Green function $G(x, y)$ for L has a singularity for $x = y$. Hence, the following condition on $b = s \times t$ has proved useful:

$$\min\{\operatorname{diam} X_s, \operatorname{diam} X_t\} \leq \eta \operatorname{dist}(X_s, X_t), \tag{2.4}$$

where $\eta > 0$ is a given real number and the support of a cluster t is the union of the supports of the basis functions corresponding to the indices in t:

$$X_t := \bigcup_{i \in t} X_i.$$

Note that condition (2.4) implies that the partition we are looking for has to be refined towards the diagonal of M, since the diagonal entries arise from the interaction of the same basis functions, i.e. $\operatorname{dist}(X_s, X_t) = 0$.

2.1 The cluster tree

Since M cannot be approximated globally by a single low-rank matrix, we have to subdivide M into blocks satisfying (2.4). One possibility is to recursively subdivide a block $b = s \times t$ into four subblocks $s_1 \times t_1$, $s_1 \times t_2$, $s_2 \times t_1$ and $s_2 \times t_2$, where $s = s_1 \cup s_2$ and $t = t_1 \cup t_2$, until its parts satisfy condition (2.4). Another possibility is proposed in [5]. The rule how to subdivide a cluster t is given by the so called cluster tree T_I satisfying the following conditions:

1. I is the root of T_I
2. if $t \in T_I$ is not a leaf, then t has sons $t_1, t_2 \in T_I$, so that $t = t_1 \cup t_2$.

The set of sons of t is denoted by $S(t)$, while $\mathcal{L}(T_I)$ stands for the set of leaves of the tree T_I. A cluster tree is usually generated by recursive subdivision of I so as to minimize the diameter of each part. For practical purposes the recursion should be stopped if a certain cardinality $n_{\min}$ of the clusters is reached, rather than subdividing the clusters until only one index is left. The depth of T_I will be denoted by p. For reasonable cluster trees one would always expect $p \sim \log N$. A strategy based on the *principle component analysis* is used in [2]. The complexity of building the cluster tree in the case of quasi-uniform grids can be estimated as $\mathcal{O}(N \log N)$.

Remark 2.1 *Sometimes, the number of sons of a cluster in the previous definition of a cluster tree is not restricted to two. This generalization has however not proved useful in practice.*

2.2 The block cluster tree

Based on a cluster tree T_I which contains a hierarchy of partitions of I, we are able to construct the so called *block cluster tree* $T_{I \times I}$ describing a hierarchy of partitions of $I \times I$ by the following rule:

```
procedure build_block_cluster_tree(s × t)
begin
    if (s, t) does not satisfy (2.4) and s, t ∉ L(T_I) then
        S(s × t) := {s' × t' : s' ∈ S(s), t' ∈ S(t)}
        for s' × t' ∈ S(s × t) do build_block_cluster_tree(s' × t')
    endif
    else S(s × t) := ∅
end
```

Applying *build_block_cluster_tree* to $I \times I$ we obtain a cluster tree for the index set $I \times I$. Upon completion of the algorithm the set of leaves $P := \mathcal{L}(T_{I\times I})$ is a partition of $I \times I$ with blocks $b = s \times t \in P$ either satisfying (2.4) or consisting of clusters t and s with $\min\{|s|, |t|\} \leq n_{\min}$. The complexity of building the block cluster tree in the case of quasi-uniform grids can be estimated as $\mathcal{O}(\eta^{-n} N \log N)$, cf. [2].

Remark 2.2 *In the case of unstructured grids the computation of the distance in* (2.4) *between two supports X_s and X_t is too costly. Therefore, for practical purposes the supports are enclosed into sets of a simpler structure, e.g. boxes or spheres.*

We are now in a position to define the set of $\mathcal{H}$-matrices for a partition P with blockwise rank k

$$\mathcal{H}(P, k) := \{M \in \mathbb{R}^{I\times I} : \operatorname{rank} M_b \leq k \text{ for all } b \in P\}.$$

Note that $\mathcal{H}(P, k)$ is not a linear space, since in general the sum of two rank-k matrices exceeds rank k.

Remark 2.3 *For a block $B \in \mathbb{R}^{s\times t}$ the low-rank representation $B = UV^T$, $U \in \mathbb{R}^{s\times k}$, $V \in \mathbb{R}^{t\times k}$, is only advantageous compared with the entrywise representation, if $k(|s| + |t|) \leq |s|\,|t|$. For the sake of simplicity in this article we will however assume that each block has the low-rank representation. Employing the entrywise representation for appropriate blocks will accelerate the algorithms.*

2.3 Storage and matrix-vector multiplication

The cost of multiplying an $\mathcal{H}$-matrix $M \in \mathcal{H}(P, k)$ and its transposed M^T by a vector $x \in \mathbb{R}^N$ is inherited from the blockwise matrix-vector multiplication:

$$Mx = \sum_{s\times t\in P} M_{s\times t} x_t \quad \text{and} \quad M^T x = \sum_{s\times t\in P} (M_{s\times t})^T x_s.$$

Since each block $s \times t$ has the representation $M_{s\times t} = UV^T$, $U \in \mathbb{R}^{s\times k}$, $V \in \mathbb{R}^{t\times k}$ (see Remark 2.3), $\mathcal{O}(k(|s|+|t|))$ units of memory are needed to store $M_{s\times t}$ and the matrix-vector products $M_{s\times t}x_t = UV^T x_t$ and $(M_{s\times t})^T x_s = VU^T x_s$ can be done in $\mathcal{O}(k(|s| + |t|))$ operations. Exploiting the hierarchical structure of M it can therefore be shown that both storing M and multiplying M and M^T by a vector has $\mathcal{O}(\eta^{-n} kN \log N)$ complexity. For a rigorous analysis the reader is referred to [2]. Therefore, $\mathcal{H}$-matrices are well suited for iterative schemes such as Krylov subspace methods. The matrix-vector multiplication of an $\mathcal{H}$-matrix by a vector can be computed in parallel. See [4] for an appropriate scheduling algorithm.

2.4 Where can $\mathcal{H}$-matrices be applied ?

The structure of $\mathcal{H}$-matrices was originally designed to accelerate the building process and the matrix-vector multiplication of discrete integral operators with smooth kernels having an algebraic singularity at $x = y$. This kind of integral operator arises for example from the boundary element method. Condition (2.4) was also used to prove convergence of the ACA algorithm [3, 5] for the generation of low-rank approximants based on few of the original matrix entries.

Although $\mathcal{H}$-matrices are primarily aiming at dense matrices, the stiffness matrix S of the differential operator L from (1.1) is in $\mathcal{H}(P, n_{\min})$. If $b \in P$ satisfies (2.4) then the supports of the basis functions are pairwise disjoint. Hence, the matrix entries in this block vanish. In the

remaining case b does not satisfy (2.4). Then the size of one of the clusters is less or equal $n_{\min}$. In both cases the rank of S_b does not exceed $n_{\min}$.

Recently, it was shown [6] that in the case of the Dirichlet problem together with S also its inverse can be approximated by $\mathcal{H}$-matrices on each block of a partition satisfying (2.4). Let L be a uniformly elliptic, i.e., for the coefficient $A(x) \in \mathbb{R}^{n\times n}$ of L it holds that A is symmetric with $a_{ij} \in L^\infty(\Omega)$ and

$$0 < \lambda \le \lambda(x) \le \Lambda$$

for all eigenvalues $\lambda(x)$ of $A(x)$ and almost all $x \in \Omega$. Let $e_h(u) := \|u - P_h u\|_{L^2(\Omega)}$ be the finite element error, where $P_h : H_0^1(\Omega) \to V_h$ is the Ritz projector mapping $u \in H_0^1(\Omega)$ to its finite element solution, i.e., the solution of $a(u_h, v_h) = f(v_h)$ for all $v_h \in V_h$. The weakest form of the finite element convergence is described by

$$e_h(u) \le \varepsilon_h \|f\|_{L^2(\Omega)} \quad \text{for all } u = L^{-1}f,\ f \in L^2(\Omega),$$

where $\varepsilon_h \to 0$ as $h \to 0$.

Theorem 2.4 *Let p be the depth of the cluster tree T_I defined in Section 2.1. Then there is a constant $c > 0$ defining $k := cp^2 \log^{n+1}(p/\varepsilon_h)$ and there is $C_{\mathcal{H}} \in \mathcal{H}(P, k)$ such that*

$$\|S^{-1} - C_{\mathcal{H}}\|_2 \le c\,\varepsilon_h,$$

where c depends on the coefficients of L, the diameter of Ω and η.If $\varepsilon_h = \mathcal{O}(h^\beta)$ for some $\beta > 0$, $k = \mathcal{O}(\log^{n+3} N)$ holds.

To prove the previous theorem the integral representation

$$L^{-1}\varphi(x) = \int_\Omega G(x, y)\varphi(y)\,\mathrm{d}y$$

with the Green function G of L and Ω was used. In contrast to operators arising from the boundary element method the kernel function G is locally only in H^1 with respect to each variable. It is nevertheless shown in [6] that G and hence the inverse of the finite element stiffness matrix can be approximated.

3 Spectral equivalence and eigenvalue clusters

When speaking about the complexity of an algorithm for the solution of a linear system we are investigating a sequence of systems

$$A_N x_N = b_N, \quad N \to \infty$$

where $A_N \in \mathbb{R}^{N\times N}$ is invertible. However, for the sake of readability the index N will be dropped in the rest of this article whenever this dependency is obvious from the context. A right preconditioner $C \in \mathbb{R}^{N\times N}$ for A is an invertible matrix which reduces the condition number of A, i.e., for the condition number of the coefficient matrix AC of the equivalent linear system

$$ACu = b,$$

where $x = Cu$, it holds that

$$\mathrm{cond}_2(AC) \le \mathrm{cond}_2(A).$$

Usually, C is expected to be multiplied by a vector much easier than A^{-1}. Therefore, the choice $C = A^{-1}$, which is optimal with respect to the condition number, is not valid for preconditioning.

In this article a preconditioner for the finite element stiffness matrices $S \in \mathbb{R}^{N\times N}$ is proposed. Since the spectral condition number of S grows with N, we will show how to generate matrices C such that the behavior of the eigenvalues of the product AC is improved.

Definition 3.1 *Let $\{A_N\}_{N\in\mathbb{N}}$ and $\{C_N\}_{N\in\mathbb{N}}$ be sequences of Hermitian matrices. Assume that there are two constants $c_1, c_2 > 0$ independent of N such that all eigenvalues λ of $A_N C_N$ satisfy $c_1 \le \lambda \le c_2$. Then $\{A_N\}_{N\in\mathbb{N}}$ and $\{C_N^{-1}\}_{N\in\mathbb{N}}$ are said to be* spectrally equivalent.

In this case

$$\mathrm{cond}_2(A_N C_N) \le c_2/c_1$$

holds and the *preconditioned conjugate gradients method* for the solution of $Ax = b$:

Let $x_0 \in \mathbb{R}^N$ be an arbitrary vector.
Calculate $r_0 = Ax_0 - b$, $p_0 = v_0 = Cr_0$ and $\rho_0 = (v_0, r_0)$.
for $k = 0, 1, 2, \ldots, N-1$ **do**
$q_k = Ap_k$
$x_{k+1} = x_k - \alpha_k p_k$, where $\alpha_k = \frac{\rho_k}{(p_k, q_k)}$
$r_{k+1} = r_k - \alpha_k q_k$
$v_{k+1} = Cr_{k+1}$
$\rho_{k+1} = (v_{k+1}, r_{k+1})$
$p_{k+1} = v_{k+1} + \beta_k p_k$, where $\beta_k = \frac{\rho_{k+1}}{\rho_k}$

---

is known to converge linearly, as can also be seen from the following error estimate, cf. [1]. Let the spectrum of AC be decomposed in the following way:

$$\sigma(AC) = \{\lambda'_i,\ i = 1, \ldots, p\} \cup M \cup \{\lambda''_i,\ i = 1, \ldots, q\}, \quad M \subset [a, b].$$

Theorem 3.2 *For the error $x_k - x$ of the PCG it holds that*

$$\|x_k - x\|_A \le 2(\mathrm{cond}_2(AC) + 1)^p \left(\frac{\sqrt{b/a} - 1}{\sqrt{b/a} + 1}\right)^{k-p-q} \|x_0 - x\|_A, \quad k = 0, \ldots, N-1.$$

Therefore, if A and C^{-1} are spectrally equivalent we may take $M = \sigma(AC)$. Then the PCG converges linearly, i.e., the number of iterations depends only on the condition number of AC and not upon N. From the last theorem it can also be seen that few small or large eigenvalues do not affect the rate of convergence. Therefore, even more important for the rate of convergence is the distribution of eigenvalues within the spectrum. Spectral equivalence gives a lower and an upper bound for the eigenvalues of AC. There is no condition on the distribution of the eigenvalues within this interval. A faster convergence of the PCG can be obtained by a condition on this distribution neglecting the size of the interval.

Definition 3.3 *By $\gamma_N(\varepsilon)$ we denote the number of eigenvalues of A_N lying outside a circle with radius ε around the origin. The eigenvalues of a sequence of matrices $\{A_N\}_{N\in\mathbb{N}}$ is said to have a* cluster *at 0, if*

$$\gamma_N(\varepsilon) \le c(\varepsilon)$$

for all $N \in \mathbb{N}$ and all $\varepsilon > 0$, where $c(\varepsilon)$ does not depend on N. $\{A_N\}_{N\in\mathbb{N}}$ is said to have a cluster *at $z \in \mathbb{C}$, if $\{A_N - zI_N\}_{N\in\mathbb{N}}$ has a cluster at 0.*

If A_N and C_N are Hermitian, and $\{A_N C_N\}_{N\in\mathbb{N}}$ has a cluster at 1 then the PCG converges super-linearly, i.e., for all $0 < q < 1$ and for all sufficiently large N the residual at the k-th iteration is bounded by cq^k. Why this super-linear convergence happens is explained in [1]. If A_N and C_N are not Hermitian, we can apply the PCG to the normal equations $A_N^H A_N x = A_N^H b$. In this case instead of the eigenvalues the singular values have to be inspected.

Since in practice we do not want to investigate the spectrum of AC for an eigenvalue cluster, we need a criterion which is easy to check. It is known that if $\|I - AC\|_2 =: \delta < 1$ then A and C are spectrally equivalent, since by the triangle inequality

$$\|AC\|_2 \le \|I\|_2 + \|I - AC\|_2 = 1 + \delta$$

and from the Neumann series one has

$$\|(AC)^{-1}\|_2 \le \sum_{k=0}^{\infty} \|I - AC\|_2^k = \frac{1}{1-\delta}.$$

Hence, $\mathrm{cond}_2(AC) \le (1+\delta)/(1-\delta)$. The following theorem states that for the existence of eigenvalue clusters the approximation C of A does not have to be too accurate.

Theorem 3.4 *Let $\{A_N\}_{N\in\mathbb{N}} \subset \mathbb{R}^{N\times N}$ be a bounded sequence, i.e. $\|A_N\|_F \le c$, where c does not depend on N. Then both the singular values and the eigenvalues of $\{A_N\}_{N\in\mathbb{N}}$ cluster at 0.*

Proof. By $\mu_N(\varepsilon)$ we denote the number of non-increasingly ordered singular values $\sigma_i(A)$ of A lying outside a circle with radius ε around the origin. Let $\varepsilon > 0$ and assume that $\mu_N(\varepsilon) > c^2/\varepsilon^2$. Then

$$c^2 < \mu_N(\varepsilon)\varepsilon^2 \le \sum_{i=1}^{N} |\sigma_i(A)|^2 = \|A\|_F^2 \le c^2,$$

which gives the contradiction. Hence, $\mu_N(\varepsilon) \le c^2/\varepsilon^2$. Let $\delta = e\varepsilon$ and $\lambda_i(A)$ be the eigenvalues of A such that $|\lambda_1(A)| \ge |\lambda_2(A)| \ge \ldots \ge |\lambda_N(A)|$. If $\gamma_N(\delta) < \mu_N(\varepsilon)$, we are done. In the other case, i.e. $\gamma_N(\delta) \ge \mu_N(\varepsilon)$, we use Weyl's inequality

$$\prod_{i=1}^{\gamma_N(\delta)} |\lambda_i(A)| \le \prod_{i=1}^{\mu_N(\varepsilon)} \sigma_i(A) \prod_{i=\mu_N(\varepsilon)+1}^{\gamma_N(\delta)} \sigma_i(A) \le \|A\|_2^{\mu_N(\varepsilon)} \varepsilon^{\gamma_N(\delta)-\mu_N(\varepsilon)}.$$

On the other hand

$$\prod_{i=1}^{\gamma_N(\delta)} |\lambda_i(A)| \ge \delta^{\gamma_N(\delta)}$$

and the last two inequalities give

$$\left(\frac{\delta}{\varepsilon}\right)^{\gamma_N(\delta)} \le \left(\frac{c}{\varepsilon}\right)^{\mu_N(\varepsilon)} \quad \Longleftrightarrow \quad \gamma_N(e\varepsilon) \le \mu_N(\varepsilon) \ln \frac{c}{\varepsilon},$$

which proves the assertion. ∎

Therefore, if $\{A_N\}_{N\in\mathbb{N}}$ and $\{C_N\}_{N\in\mathbb{N}}$ are two sequences of matrices such that

$$\|I_N - A_N C_N\|_F \le c \tag{3.5}$$

with a constant $c > 0$ which does not depend on N, the eigenvalues of the sequence $\{A_N C_N\}_{N\in\mathbb{N}}$ cluster at 1 and PCG converges super-linearly. Especially, the number of iterations of PCG is then bounded by a constant.

Theorem 3.5 *Assume that* (3.5) *holds, then the preconditioned conjugate gradients method applied to the system* $Ax = b$ *converges to any prescribed accuracy in a bounded number of iterations.*

Remark 3.6 *Instead of using the Frobenius norm it is sometimes more convenient to use the spectral norm* $\|\cdot\|_2$. *Since* $\|\cdot\|_F \leq \sqrt{N}\|\cdot\|_2$ *for* $N \times N$ *matrices,* (3.5) *then has to be replaced by*

$$\|I - AC\|_2 \leq \frac{c}{\sqrt{N}}.$$

4 Algorithms

In this section it is described how the preconditioner is generated from the stiffness matrix S. From Theorem 2.4 we know that an approximation of S^{-1} exists in the set of $\mathcal{H}$-matrices. Therefore, our aim is to find it in this set using the $\mathcal{H}$-arithmetic. Since S is in $\mathcal{H}(P, n_{\min})$, see Section 2.4, the generation of the preconditioner can be completely done in the format of $\mathcal{H}$-matrices.

Since $\mathcal{H}(P, k)$ is not a linear space, we have to replace the usual matrix operations by truncated ones. Starting with the $\mathcal{H}$-matrix addition, an $\mathcal{H}$-matrix multiplication is defined. Using these modified operations it is possible to define an $\mathcal{H}$-matrix inversion based on the Frobenius formulas. These ideas already appeared in the early papers on $\mathcal{H}$-matrices, cf. [13, 14]. Due to the various representations of matrix blocks for the implementation many cases have to be distinguished. Therefore only the main ideas of the $\mathcal{H}$-arithmetic are presented. According to the remark after Definition 3.3 we may restrict ourselves to symmetric positive definite matrices. Hence, the usual $\mathcal{H}$-arithmetic can be simplified to a symmetric one leading to approximately half the amount of operations and storage.

4.1 Truncated addition

In order to make the sum of two $\mathcal{H}(P, k)$-matrices be in $\mathcal{H}(P, k)$, we have to add them blockwise and truncate each sum UV^T, $U = (U_1, U_2) \in \mathbb{R}^{s\times 2k}$, $V = (V_1, V_2) \in \mathbb{R}^{t\times 2k}$, of two rank-$k$ blocks $U_1V_1^T$ and $U_2V_2^T$ to a rank-k matrix. This truncated addition will be denoted by $\oplus_k$ and we define the addition of two submatrices A, B in the entries $\hat{b} \in T_{I\times I}$, the cluster tree from Section 2.2, by

$$A \oplus B = \{A_b \oplus_k B_b \text{ for all } b \in P,\ b \text{ is a descendant of } \hat{b} \text{ in } T_{I\times I}\}.$$

The truncation can be done by the following algorithm, which was already presented in [2] to find the approximant with lowest rank to a prescribed accuracy:

---

procedure truncate$(U, V, k, \texttt{var}\,\tilde{U}, \texttt{var}\,\tilde{V})$
begin
  Compute the QR-decompositions $U = Q_U R_U$ and $V = Q_V R_V$.
  Compute $M := R_U R_V^T \in \mathbb{R}^{2k\times 2k}$.
  Compute the singular value decomposition $M = XSY^T$.
  Let S_ℓ and Y_ℓ be the first ℓ columns of S and Y, respectively.
  Compute $\tilde{U} := Q_U X S_\ell$ and $\tilde{V} := Q_V Y_\ell$.
end

Obviously, $\tilde{U}\tilde{V}^T$ has rank ℓ and the error in spectral norm can be controlled using

$$\|UV^T - \tilde{U}\tilde{V}^T\|_2 = \frac{s_{\ell+1}}{s_1}\|UV^T\|_2,$$

where s_i is the i-th largest entry of the diagonal matrix S. Therefore, by the following criterion

$$s_{\ell+1} \leq \varepsilon s_1$$

the rank ℓ of a block may be adaptively chosen to guarantee a relative error ε. The actual rank of a block within an $\mathcal{H}$-matrix may therefore be less than k. The previous truncation algorithm needs $\mathcal{O}(k^2(|s|+|t|))$ operations if $b = s \times t$. Hence, exploiting the block-hierarchy the complexity for the $\mathcal{H}$-matrix addition of two matrices from $\mathcal{H}(P,k)$ can be shown to be of order $k^2 N \log N$.

4.2 Truncated matrix-matrix multiplication

Since the partition P consists of the leaves of the block cluster tree $T_{I\times I}$, it is possible to recursively define a modified matrix-matrix multiplication $C \oplus= A \odot B$, $A \in \mathcal{H}(P,k)$, $B \in \mathcal{H}(P,k)$, making use of the partitioned matrix-matrix multiplication. Let $r \times s$, $s \times t$, $r \times t \in T_{I\times I}$ be block clusters. In order to define what is meant with $C_{r\times t} \oplus= A_{r\times s} \odot B_{s\times t}$, we focus on the 2×2 partition of the blocks

$$\begin{bmatrix} C_{r_1\times t_1} & C_{r_1\times t_2} \\ C_{r_2\times t_1} & C_{r_2\times t_2} \end{bmatrix} \oplus= \begin{bmatrix} A_{r_1\times s_1} & A_{r_1\times s_2} \\ A_{r_2\times s_1} & A_{r_2\times s_2} \end{bmatrix} \odot \begin{bmatrix} B_{s_1\times t_1} & B_{s_1\times t_2} \\ B_{s_2\times t_1} & B_{s_2\times t_2} \end{bmatrix}$$

with the sons of the clusters s, t and r. Hence, we define

$$\begin{aligned}
C_{r_1\times t_1} &\oplus= A_{r_1\times s_1} \odot B_{s_1\times t_1}, & C_{r_1\times t_1} &\oplus= A_{r_1\times s_2} \odot B_{s_2\times t_1}, \\
C_{r_1\times t_2} &\oplus= A_{r_1\times s_1} \odot B_{s_1\times t_2}, & C_{r_1\times t_2} &\oplus= A_{r_1\times s_2} \odot B_{s_2\times t_2}, \\
C_{r_2\times t_1} &\oplus= A_{r_2\times s_1} \odot B_{s_1\times t_1}, & C_{r_2\times t_1} &\oplus= A_{r_2\times s_2} \odot B_{s_2\times t_1}, \\
C_{r_2\times t_2} &\oplus= A_{r_2\times s_1} \odot B_{s_1\times t_2}, & C_{r_2\times t_2} &\oplus= A_{r_2\times s_2} \odot B_{s_2\times t_2}.
\end{aligned}$$

Obviously, the result C is an $\mathcal{H}(P,k)$-matrix. If the truncation would not produce any approximation error, then $C \oplus= A \odot B$ would coincide with $C := C + AB$. The complexity of the truncated matrix-matrix multiplication can be estimated as $\mathcal{O}(k^2 N \log^2 N)$, cf. [12].

4.3 Inversion

We assume that each block $A_{s\times s}$, $s \in T_I$, of $A \in \mathcal{H}(P,k)$ is invertible. This is for example the case if A is symmetric positive definite. Let $s \in T_I \setminus \mathcal{L}(T_I)$. The corresponding matrix block $A_{s\times s}$ is subdivided into the sons of $s \times s$:

$$A_{s\times s} = \begin{bmatrix} A_{s_1\times s_1} & A_{s_1\times s_2} \\ A_{s_2\times s_1} & A_{s_2\times s_2} \end{bmatrix}.$$

According to the Frobenius formulas for the inverse of A it holds that:

$$A^{-1}_{s\times s} = \begin{bmatrix} A^{-1}_{s_1\times s_1} + A^{-1}_{s_1\times s_1} A_{s_1\times s_2} S^{-1} A_{s_2\times s_1} A^{-1}_{s_1\times s_1} & -A^{-1}_{s_1\times s_1} A_{s_1\times s_2} S^{-1} \\ -S^{-1} A_{s_2\times s_1} A^{-1}_{s_1\times s_1} & S^{-1} \end{bmatrix},$$

where S is the Schur complement $S = A_{s_2\times s_2} - A_{s_2\times s_1} A^{-1}_{s_1\times s_1} A_{s_1\times s_2}$. The $\mathcal{H}$-matrix inverse $C_{s\times s}$ of $A_{s\times s}$ is defined by replacing the matrix-matrix multiplication and the addition by the $\mathcal{H}$-versions. We need a temporary matrix $T \in \mathcal{H}(P,k)$, which together with C is initialized to zero.

```
procedure invertH(s, A, var C)
begin
  if s ∈ L(T_I) then C_{s×s} := A^{-1}_{s×s} is the usual inverse.
  else begin
    invertH(s_1, A, C).
    T_{s_1×s_2} ⊖= C_{s_1×s_1} ⊙ A_{s_1×s_2}.
    T_{s_2×s_1} ⊖= A_{s_2×s_1} ⊙ C_{s_1×s_1}.
    A_{s_2×s_2} ⊕= A_{s_2×s_1} ⊙ T_{s_1×s_2}.
    invertH(s_2, A, C).
    C_{s_1×s_2} ⊕= T_{s_1×s_2} ⊙ C_{s_2×s_2}.
    C_{s_2×s_1} ⊕= C_{s_2×s_2} ⊙ T_{s_2×s_1}.
    C_{s_1×s_1} ⊕= T_{s_1×s_2} ⊙ C_{s_2×s_1}.
  end
end
```

Then $C \in \mathcal{H}(P,k)$ is an approximation of A^{-1}. The matrix A is destroyed during the previous algorithm. The cost for the calculation of the $\mathcal{H}$-inverse is mainly determined by the cost for the $\mathcal{H}$-multiplication. Therefore, an approximation to the inverse of A can be obtained with complexity $\mathcal{O}(k^2 N \log^2 N)$.

The same algorithm can also be used for preconditioning boundary element (BE) Galerkin matrices. We already mentioned in Section 2.4 that BE Galerkin matrices can be efficiently treated by $\mathcal{H}$-matrices. Since elliptic boundary integral operators are pseudo-differential operators and its inverses have the same property, we are also able to approximate the inverses by $\mathcal{H}$-matrices.

4.4 Updating the preconditioner

If many linear systems with coefficient matrices differing in only a small number of entries have to be solved, after a small update the preconditioner generated for one matrix can be used for preconditioning the other. Assume that A and

$$\tilde{A} = A + \alpha e_i e_j^T$$

with the canonical vectors $e_i, e_j \in \mathbb{R}^n$ are invertible. Then due to the Sherman-Morisson formula for the inverse of $\tilde{A}$ it holds that

$$\tilde{A}^{-1} = (A + \alpha e_i e_j^T)^{-1} = A^{-1} - \frac{\alpha}{1 + \alpha e_j^T A^{-1} e_i} A^{-1} e_i e_j^T A^{-1}.$$

Hence, $\tilde{A}^{-1}$ and A^{-1} differ only by matrix of rank 1. In the case that A and $\tilde{A}$ have p different entries, the update will be of rank at most p. Using the $\mathcal{H}$-matrix addition we are able to compute an approximation $C \in \mathcal{H}(P,k)$ of $\tilde{A}^{-1}$. Obviously, this much faster than building a new preconditioner for $\tilde{A}$.

Since S is in $\mathcal{H}(P, n_{\min})$ the $\mathcal{H}$-inversion procedure can be directly applied to S. Due to Remark 3.6 for preconditioning the blockwise accuracy has to be chosen such that globally

$$\|I - SC\|_2 \leq \frac{c}{\sqrt{N}}$$

holds. We need the following lemma [12] by which the spectral norm of an $\mathcal{H}$-matrix can be estimated by its blockwise norms. P is again assumed to be generated as in Section 2.

Lemma 4.1 *There is a constant c such that for any matrix $M \in \mathcal{H}(P,k)$ the following inequality holds:*

$$\|M\|_2 \leq c\,p \max_{b \in P} \|M_b\|_2.$$

Therefore, a blockwise rank $k \sim \log N$ is sufficient. Hence, C can be computed, stored and multiplied by a vector with complexity $N \log^{n+3} N$.

5 Numerical Results

We consider the Dirichlet boundary value problem

$$\begin{aligned} Lu &= 0 \quad \text{in } \Omega \\ u &= f \quad \text{on } \partial\Omega \end{aligned}$$

with $L = -\text{div}\, A(x)\nabla$. The coefficients A of L are chosen to be of the following form

$$A(x) = \begin{bmatrix} 1 & 0 \\ 0 & \alpha(x) \end{bmatrix}, \quad x \in \Omega,$$

where $\alpha(x) = 1$ in the lower region of Figure 1 and a random number from the interval $[0, a]$ in the remaining part of the unit square. In the following tables we compare the conjugate

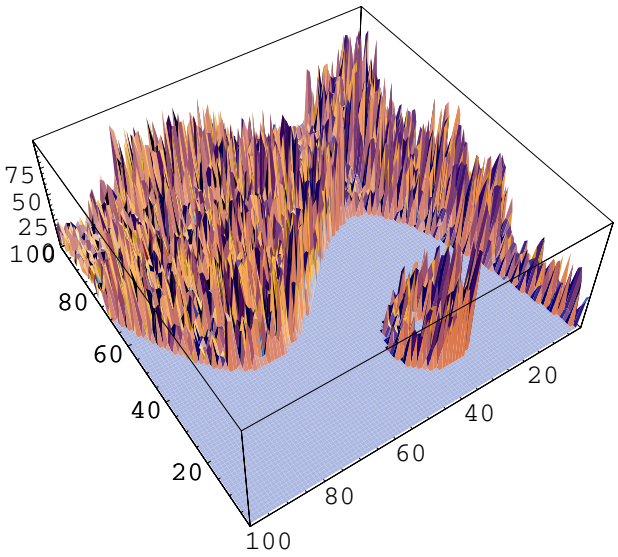

Figure 1: The coefficient $\alpha(x)$

gradients method (CG) with a diagonally preconditioned CG (DPCG) and the $\mathcal{H}$-approximate inverse preconditioned CG (HPCG). The relative accuracy of the residual is $1e-8$. The tests were carried out on an Intel Pentium IV (2.666GHz) with 2GB of core memory.

In Table 1 the amplitude a of the coefficient $A(x)$ is chosen to be 1. For increasing problem sizes N in the third column the norm $\|I - SC\|_2$ and in the forth column the time to compute the approximate inverse C is shown. The truncation accuracy we have used for C can be found in the second column. In the fifth and sixth column the number of iterations and the CPU-time for unpreconditioned CG can be found. Column seven holds the number of iterations when using a diagonal preconditioner. Finally, in the last two columns the number of iterations and

				CG		DPCG	HPCG	
N	ε	$\|I - SC\|_2$	T_{appr}	#iter	T	#iter	#iter	T
38025	$7_{10}-4$	1.6	13.5s	864	19.7s	747	10	0.8s
50176	$5_{10}-4$	1.9	21.9s	978	35.4s	877	9	1.0s
65025	$4_{10}-4$	1.7	35.5s	1116	62.1s	985	10	1.6s
82944	$3_{10}-4$	1.7	48.2s	1281	70.1s	1107	8	1.6s
159201	$2_{10}-4$	2.0	118.5s	1776	178.1s	1586	9	3.9s
193600	$1_{10}-4$	1.1	173.6s	1998	269.1s	1696	8	4.8s
233289	$1_{10}-4$	1.3	230.7s	2181	424.6s	1927	9	6.9s

Table 1: Amplitude $a = 1$

				CG		DPCG	HPCG	
N	ε	$\|I - SC\|_2$	T_{appr}	#iter	T	#iter	#iter	T
38025	$7_{10}-4$	3.4	13.3s	1245	28.4s	791	10	0.8s
50176	$5_{10}-4$	2.7	21.3s	1444	51.8s	926	10	1.1s
65025	$4_{10}-4$	2.6	34.8s	1653	91.2s	1037	10	1.6s
82944	$3_{10}-4$	4.6	46.1s	1845	101.4s	1176	10	1.9s
159201	$2_{10}-4$	2.4	114.0s	2631	263.2s	1661	10	4.2s
193600	$1_{10}-4$	1.3	169.6s	2855	399.9s	1848	9	5.0s
233289	$1_{10}-4$	1.6	228.5s	3237	599.4s	1992	9	7.0s

Table 2: Amplitude $a = 10$

the time for the solution of the linear systems by PCG using the proposed approximate inverse preconditioner from this article are presented. In Table 2 and 3 the same results are shown for the amplitudes $a = 10$ and $a = 100$, respectively.

These numerical results show that the conjugate gradients method without preconditioning cannot be used in practice for the solution of FE systems. Even with a diagonal preconditioner the situation does not improve much. With the proposed approximate inverse preconditioner it is however possible to solve such systems in a bounded number of iterations. In contrast to CG or DPCG for the new preconditioner the number of iterations does not depend much on the amplitude a. Since we used the same truncation accuracy for all three amplitudes, the proposed preconditioner is able to adapt itself to jumping coefficients. This preconditioner is particularly

				CG		DPCG	HPCG	
N	ε	$\|I - SC\|_2$	T_{appr}	#iter	T	#iter	#iter	T
38025	$7_{10}-4$	9.9	13.5s	3108	71.0s	1353	13	1.0s
50176	$5_{10}-4$	10.6	22.2s	3616	126.6s	1493	13	1.4s
65025	$4_{10}-4$	14.0	35.8s	4127	226.5s	1659	14	2.4s
82944	$3_{10}-4$	17.9	46.2s	4745	256.4s	1908	14	2.7s
159201	$2_{10}-4$	23.1	118.1s	6636	659.4s	2693	13	5.5s
193600	$1_{10}-4$	18.6	172.5s	7356	1000.6s	3011	12	7.1s
233289	$1_{10}-4$	23.6	232.6s	8230	1516.7s	3297	12	9.0s

Table 3: Amplitude $a = 100$

efficient if the same system with many right hand sides has to be solved.

References

[1] O. Axelsson and G. Lindskog: *On the rate of convergence of the preconditioned conjugate gradient method.* Numer. Math. 48, 499–523, 1986.

[2] M. Bebendorf: *Effiziente numerische Lösung von Randintegralgleichungen unter Verwendung von Niedrigrang-Matrizen.* dissertation.de, Verlag im Internet, 2001. ISBN 3-89825-183-7.

[3] M. Bebendorf: *Approximation of boundary element matrices.* Numer. Math. 86, 565–589, 2000.

[4] M. Bebendorf and R. Kriemann: *Fast Parallel Solution of Boundary Integral Equations and Related Problems.* Preprint, 2003.

[5] M. Bebendorf and S. Rjasanow: *Adaptive Low-Rank Approximation of Collocation Matrices.* Computing 70, 1–24, 2003.

[6] M. Bebendorf and W. Hackbusch: *Existence of $\mathcal{H}$-Matrix Approximants to the Inverse FE-Matrix of Elliptic Operators with L^∞-Coefficients.* Numer. Math. 95, 1–28, 2003.

[7] M. W. Benson and P. O. Frederickson: *Iterative solution of large sparse linear systems arising in certain multidimensional approximation problems*, Utilitas Math. 22, 127–140, 1982.

[8] J. H. Bramble, J. E. Pasciak, and J. Xu: *Parallel multilevel preconditioners*, Math. Comp. 55, 1–22, 1990.

[9] E. Chow and Y. Saad: *Approximate inverse preconditioners via sparse-sparse iterations*, SIAM J. Sci. Comput. 19, 995–1023, 1998.

[10] J. D. F. Cosgrove, J. C. Diaz, and A. Griewank: *Approximate inverse preconditioning for sparse linear systems*, Internat. J. Comput. Math. 44, 91–110, 1992.

[11] Ph. Guillaume, A. Huard and C. Le Calvez: *A Block Constant Approximate Inverse for Preconditioning large Linear Systems*, SIAM J. Matrix Anal. Appl. 24, 822–851, 2003.

[12] L. Grasedyck: *Theorie und Anwendungen Hierarchischer Matrizen.* Dissertation, Universität Kiel, 2001.

[13] W. Hackbusch: *A sparse matrix arithmetic based on $\mathcal{H}$-matrices. I. Introduction to $\mathcal{H}$-matrices.* Computing 62, 89–108, 1999.

[14] W. Hackbusch and B. N. Khoromskij: *A sparse $\mathcal{H}$-matrix arithmetic. II. Application to multi-dimensional problems.* Computing 64, 21–47, 2000.

[15] V. Rokhlin: *Rapid solution of integral equations of classical potential theory.* J. Comput. Phys. 60:187–207, 1985.

[16] E. Tyrtyshnikov: *A unifying approach to some old and new theorems on distribution and clustering.* Linear Algebra Appl. 232, 1–43, 1996.

[17] E. Tyrtyshnikov: *Clusters, Preconditioners, Convergence.* Linear Algebra Appl. 263, 25–48, 1997.

[18] E. Tyrtyshnikov: *Mosaic-skeleton approximations.* Calcolo 33, 47–57 (1998), 1996.