Electronic Transactions on Numerical Analysis.
Volume 26, pp. 285-298, 2007.

ISSN 1068-9613.

ETNA
Kent State University
etna@mcs.kent.edu

JOINT DOMAIN-DECOMPOSITION $\mathcal{H}$-LU PRECONDITIONERS FOR SADDLE POINT PROBLEMS*

SABINE LE BORNE† AND SUELY OLIVEIRA‡

Abstract. For saddle point problems in fluid dynamics, several popular preconditioners exploit the block structure of the problem to construct block triangular preconditioners. The performance of such preconditioners depends on whether fast, approximate solvers for the linear systems on the block diagonal (representing convection-diffusion problems) as well as for the Schur complement (in the pressure variables) are available. In this paper, we will introduce a completely different approach in which we ignore this given block structure. We will instead compute an approximate LU-factorization of the complete system matrix using hierarchical matrix techniques. In particular, we will use domain-decomposition clustering with an additional local pivoting strategy to order the complete index set. As a result, we obtain an $\mathcal{H}$-matrix structure in which an $\mathcal{H}$-LU factorization is computed more efficiently and with higher accuracy than for the corresponding block structure based clustering. $\mathcal{H}$-LU preconditioners resulting from the block and joint approaches will be discussed and compared through numerical results.

Key words. hierarchical matrices, data-sparse approximation, Oseen equations, preconditioning, factorization

AMS subject classifications. 65F05, 65F30, 65F50

1. Introduction.

Hierarchical (or $\mathcal{H}$-) matrices have first been introduced in 1998 [9] and since then have entered into a wide range of applications. They provide a format for the data-sparse representation of fully populated matrices. The key idea is to approximate certain subblocks of a matrix by data-sparse low-rank matrices which are represented by a product of two rectangular matrices as follows: Let $A \in \mathbb{R}^{n\times n}$ with rank(A)=k and $k \ll n$. Then there exist matrices $B, C \in \mathbb{R}^{n\times k}$ such that $A = BC^T$. Whereas A has n^2 entries, B and C together have $2kn$ entries which results in significant savings in storage if $k \ll n$. A new $\mathcal{H}$-matrix arithmetic has been developed which allows (approximate) matrix-vector multiplication and matrix-matrix operations such as addition, multiplication, LU-factorization and inversion of matrices in this format. Hierachical matrices are related to fast multipole methods [8, 13] as well as mosaic-skeleton methods [22] in which low-rank representations of off-diagonal blocks are also exploited for the representation and solution of dense systems.

In finite element methods, the stiffness matrix is sparse but its LU factors are no longer sparse and can be approximated by an $\mathcal{H}$-matrix. These approximate $\mathcal{H}$-LU factors may then be used as a preconditioner in iterative methods [2, 7]. Recent developments such as a weak admissibility condition [11], the parallelization of the $\mathcal{H}$-matrix arithmetic [14], and the introduction of an $\mathcal{H}$-matrix format based on domain-decomposition [6, 12, 16] have significantly reduced the set-up times for such preconditioners.

In previous papers, $\mathcal{H}$-matrices have been developed for *scalar* equations, and they have been used to construct preconditioners for linear systems arising from uniformly elliptic differential operators. Recently, the application of $\mathcal{H}$-matrix techniques has been extended to the construction of *block* preconditioners for the discrete Oseen equations [15]. Numerous solution techniques have been proposed in the literature for this type of saddle point problem. A recent comprehensive survey [3] reviews many of the most promising solution methods with an emphasis on the iterative solution of these large, sparse, indefinite problems. Several of these methods are based on the underlying block structure of the system matrix and

*Received January 23, 2007. Accepted for publication March 22, 2007. Recommended by M. Benzi.
†Department of Mathematics, Box 5054, Tennessee Technological University, Cookeville, TN 38505 (sleborne@tntech.edu). The work of this author was supported in part by the US Department of Energy under Grant No. DE-FG02-04ER25649 and in part by the National Science Foundation under grant No. DMS-0408950.
‡Department of Computer Science, The University of Iowa, Iowa City, Iowa 52242 (oliveira@cs.uiowa.edu).

require (an approximation to) an auxiliary Schur complement and its approximate inverse or LU factors to be used as preconditioners. Typically, one avoids the explicit computation of a Schur complement due to complexity constraints and replaces the exact solution to the Schur complement problem by a sufficient number of inner iterations. However, taking advantage of the efficient $\mathcal{H}$-matrix arithmetic, one can compute an explicit approximation to the LU factors of the Schur complement [15]. This $\mathcal{H}$-LU factorization may then be used in block preconditioners which result from a block LU factorization of the saddle point matrix representing the discrete Oseen equations. A drawback of this method is the relatively time consuming computation of the approximate Schur complement. Therefore, in this paper we will develop an alternative approach which we will refer to as a "joint" approach since it does not employ the block structure of the system matrix but computes an $\mathcal{H}$-LU factorization of the complete system matrix after a domain-decomposition based clustering of the unknowns. Such a clustering imitates the nested dissection process and therefore leads to a matrix structure in which large, off-diagonal blocks remain zero in an LU-factorization, therefore reducing the computational complexity of such a computation. In particular, instead of computing a Schur complement in the pressure variables as required in the block approach, we now compute a Schur complement in the variables of the interior boundary of the domain decomposition which typically is of much smaller size. In order to ensure the existence of an LU factorization for the joint approach, we will introduce a local pivoting strategy. This new joint approach will turn out to be clearly superior to the block approach with respect to computational efficiency as well as accuracy.

The remainder of this paper is organized as follows: In Section 2, we provide an introduction to $\mathcal{H}$-matrices with references to further literature on this topic. Section 3 is devoted to the introduction of the model problem, a review of the block $\mathcal{H}$-matrix preconditioner as well as the introduction and analysis of the new joint domain decomposition clustering strategy and resulting preconditioner for saddle point problems. Finally, Section 4 provides numerical results that compare set-up times, storage requirements, and convergence rates of the resulting preconditioned iterative methods for both approaches for a variety of test cases.

2. Introduction to $\mathcal{H}$-matrices. In this section, we introduce the main concepts of $\mathcal{H}$-matrices to the extent of which they are required for the remainder of this paper. For more detailed introductions, we refer the reader to [4, 5, 9, 10] and the references therein.

An $\mathcal{H}$-matrix approximation to a given (dense) matrix is obtained by replacing certain blocks of the matrix by matrices of a low rank k, stored in so-called Rk-format as will be further explained below. Given such an $\mathcal{H}$-matrix, the standard matrix operations such as matrix-vector multiplication, (approximate) matrix-matrix addition and multiplication as well as matrix inversion and LU factorization can be defined for this $\mathcal{H}$-matrix format. Whereas these ($\mathcal{H}$-) matrix operations yield only approximations, they can be performed in almost optimal complexity, i.e., $\mathcal{O}(n \log^\alpha n)$ with moderate parameter α. The construction of $\mathcal{H}$-matrices is reviewed in subsection 2.1 and their arithmetic is reviewed in subsection 2.2.

2.1. Construction of $\mathcal{H}$-matrices. The formal definition of an $\mathcal{H}$-matrix depends on appropriate hierarchical partitionings of the index set I and also of the product index set $I \times I$ which are organized in (block) cluster trees as defined next. Instead of fixed partitionings, these trees will provide hierarchies of partitionings which gives a hierarchical matrix its name.

DEFINITION 2.1 (Cluster tree). *Let I be a finite index set and let $T_I = (V, E)$ be a tree with vertex set V and edge set E. For a vertex $v \in V$, we define the set of successors of v as $S(v) := \{w \in V \mid (v, w) \in E\}$. The tree T_I is called a cluster tree of I if its vertices consist of subsets of I and satisfy the following conditions:*

1. *$I \in V$ is the root of T_I, and $v \subset I$, $v \neq \emptyset$, for all $v \in V$.*
2. *For all $v \in V$, there either holds $S(v) = \emptyset$ or $v = \dot{\bigcup}_{w \in S(v)} w$.*

ETNA
Kent State University
etna@mcs.kent.edu

In the following, we identify V and T_I, i.e., we write $v \in T_I$ instead of $v \in V$. The nodes $v \in V$ are called clusters.

A cluster tree is constructed iteratively by dividing an index cluster v into a number of successors w_i. There exist several alternative clustering strategies. Our new preconditioner will require the so-called domain-decomposition clustering [6, 12, 16] which will be described here.

In preparation, we define the following geometric entities: We associate every index i with a basis function φ_i of the underlying finite element space and define $\Omega_i = supp \varphi_i$. For every index i, we then assign a (fixed) nodal point $x_i = (x_{i,1}, \cdots, x_{i,d}) \in \Omega_i$. For a cluster v, we define $\Omega_v = \cup_{j \in v} \Omega_j$. Since we will later need (upper bounds of) the diameters of these cluster supports Ω_v and distances between two clusters (in the Euclidean norm), we will introduce rectangular bounding boxes for which these quantities can be computed efficiently: For a cluster v, let $B_v = \otimes_{j=1}^{d} [\alpha_{v,j}, \beta_{v,j}]$ such that the bounding box contains the support of the cluster, i.e., $\Omega_v \subset B_v$. We define the direction of maximum extent $k \in \{1, \cdots, d\}$ of a cluster v as the index for which $j \mapsto |\beta_{v,j} - \alpha_{v,j}|$ assumes a maximum. Finally, we define the midpoint of the interval in this direction k as $m_{v,k} := (\alpha_{v,k} + \beta_{v,k})/2$.

We distinguish between subdomain and interface clusters where the root I is set to be a subdomain cluster.

A subdomain cluster v will be divided into three successors

$$w_1 := \{i \in v | \beta_{v,k} < m_{v,k}\},\ w_2 := \{i \in v | \alpha_{v,k} > m_{v,k}\},\ w_3 := v \setminus (w_1 \cup w_2),$$

where w_1, w_2 are set to be subdomain clusters, and w_3 is set as an interface cluster.

An interface cluster v will be subdivided by geometric bisection into successors w_1, w_2 only every other step:

$$S(v) = \begin{cases} \{v\} & : \quad level(v) = 0 \text{ (mod d)} \\ \{w_1, w_2\} & : \quad \text{otherwise.} \end{cases}$$

All successors of an interface cluster are set to be interface clusters themselves. The subdivision of interface clusters is delayed every d'th step in order to calibrate the diameters of interface clusters with those of domain-clusters. A simple example is provided in Fig. 2.1. Here, in the first subdivision step, the index set is divided into three subsets: A domain cluster of 24 vertices that lie to the left of the vertical shaded region, another domain cluster of 24 vertices that lie to the right of the vertical shaded region, and an interface cluster of the remaining 16 vertices within the shaded region. This subdivision process is repeated for the two domain clusters. The interface is subdivided by bisection in the next but one step.

A cluster tree T_I is used in the construction of a hierarchy of block partitionings of the product index set $I \times I$ which is organized in a block cluster tree:

DEFINITION 2.2 (Block cluster tree). *Let T_I be a cluster tree of the index set I. A cluster tree $T_{I\times I}$ is called a block cluster tree (based upon T_I) if for all $v \in T_{I\times I}^{(l)}$ there exist $s, t \in T_I^{(l)}$ such that $v = s \times t$. The nodes $v \in T_{I\times I}$ are called block clusters.*

The objective is to construct a block cluster tree from a given cluster tree such that the leaves (of the block cluster tree) correspond to (preferably large) matrix blocks with "smooth" data that can be approximated by low rank matrices in the following Rk-matrix representation:

DEFINITION 2.3 (Rk-matrix representation). *Let $k, n, m \in \mathbb{N} \cup \{0\}$. Let $M \in \mathbb{R}^{n\times m}$ be a matrix of at most rank k. A representation of M in factorized form*

$$M = AB^T, \qquad A \in \mathbb{R}^{n\times k}, B \in \mathbb{R}^{m\times k},$$

with A and B stored in full matrix representation, is called an Rk-matrix representation of M, or, in short, we call M an Rk-matrix.

ETNA
Kent State University
etna@mcs.kent.edu

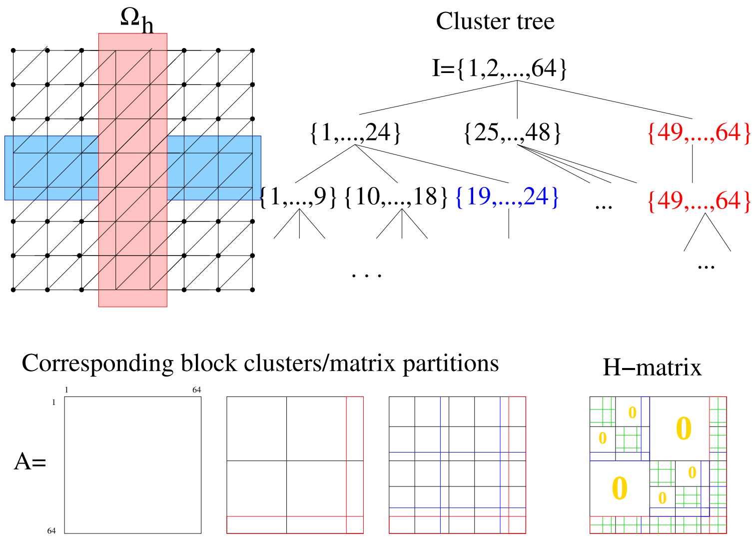

FIG. 2.1. *A mesh Ω_h (left) and its domain-decomposition cluster tree (right).*

If the rank k is small compared to the matrix size given by n and m, we obtain considerable savings in the storage and work complexities of an Rk-matrix compared to a full matrix, i.e., $(m+n)k$ versus mn memory cells (or flops).

In the following construction, we build a block cluster tree iteratively by starting from $I \times I$ and refining the block clusters if they do not satisfy a certain admissibility condition. The choice of the admissibility condition depends on the underlying continuous problem (i.e., the elliptic partial differential equation, in particular its associated Green's function) and shall ensure that all admissible blocks allow a sufficiently accurate Rk-approximation. We will use the following admissibility condition:

$$\begin{aligned} \mathrm{Adm}_{DD}(s \times t) = \text{TRUE} \iff \quad & \min(\mathrm{diam}(s), \mathrm{diam}(t)) \le \eta \, \mathrm{dist}(s,t) \quad \text{or} \\ & (s \neq t, \ \ s, t \text{ are domain clusters}) \end{aligned} \tag{2.1}$$

Here, "diam" and "dist" denote the Euclidean diameter/distance of the (union of the) supports of the basis functions with indices in s, t, resp.. A given cluster tree together with an admissibility condition allows the following canonical construction of a block cluster tree:

Let the cluster tree T_I be given. We define the block cluster tree $T_{I\times I}$ by $\mathrm{root}(T) := I \times I$, and each vertex $s \times t \in T$ has the set of successors

$$S(s \times t) := \begin{cases} \emptyset & \text{if } s \times t \text{ admissible,} \\ \emptyset & \text{if } \min\{\#s, \#t\} \le n_{\min}, \\ \{s' \times t' \mid s' \in S(s), t' \in S(t);\} & \text{otherwise.} \end{cases} \tag{2.2}$$

The parameter n_{min} ensures that blocks do not become too small where the matrix arithmetic of a full matrix is more efficient than any further subdivision. It is typically set such that $10 \le n_{min} \le 100$. The leaves of a block cluster tree obtained through this construction will be used in the definition of an $\mathcal{H}$-matrix:

DEFINITION 2.4 ($\mathcal{H}$-matrix). *Let $k, n_{\min} \in \mathbb{N} \cup \{0\}$, and let $n := \#I$ be the number of indices in an index set I. The set of $\mathcal{H}$-matrices induced by a block cluster tree $T := T_{I\times I}$*

with blockwise rank k and minimum block size $n_{\min}$ is defined by

$$\mathcal{H}(T,k) := \{M \in \mathbb{R}^{n,n} \mid \forall s \times t \in \mathcal{L}(T) : \operatorname{rank}(M|_{s\times t}) \leq k \text{ or } \min\{\#s,\#t\} \leq n_{\min}\}.$$

A matrix $M \in \mathcal{H}(T,k)$ is said to be given in $\mathcal{H}$-matrix representation if the blocks $M|_{s\times t}$ with $\operatorname{rank}(M|_{s\times t}) \leq k$ are in Rk-matrix representation and the remaining blocks with $\min\{\#s,\#t\} \leq n_{\min}$ are stored as full matrices.

Both the accuracy and (storage) complexity of an $\mathcal{H}$-matrix approximation to a given matrix depend on the construction of an appropriate cluster tree, i.e., a hierarchy of index set partitionings. Details regarding approximation errors for blocks that satisfy the admissibility condition as well as storage requirements for full, Rk- and $\mathcal{H}$-matrices are given in [5].

Whereas the classical $\mathcal{H}$-matrix uses a fixed rank for the Rk-blocks, it is possible to replace it by *variable (or adaptive) ranks* in order to enforce a desired accuracy within the individual blocks. In particular, for a given admissible block $s \times t$, we set the rank k of the corresponding matrix block $M|_{s\times t}$ as follows:

$$k(M|_{s\times t}) := \min\{k' \mid \sigma_{k'} \leq \delta\sigma_0\} \tag{2.3}$$

where $\sigma_0 \geq \sigma_1 \geq \cdots$ denote the singular values of $M|_{s\times t}$, and $0 < \delta < 1$ denotes the desired relative accuracy within each block. Numerical tests have shown that adaptive ranks are typically superior to fixed ranks, especially when applied to singularly perturbed problems [7]. A related idea where variable ranks have been assigned depending on the cluster level has been pursued in [22].

2.2. Arithmetic of $\mathcal{H}$-matrices. Given two $\mathcal{H}$-matrices $A, B \in \mathcal{H}(T,k)$ based on the same block cluster tree T, i.e., with the same block structure, the exact sum or product of these two matrices will typically not belong to $\mathcal{H}(T,k)$. In the case of matrix addition, we have $A + B \in \mathcal{H}(T,2k)$; the rank of an exact matrix product is less obvious. We will use a truncation operator $\mathcal{T}^{\mathcal{H}}_{k\leftarrow k'}$ to define the $\mathcal{H}$-matrix addition $C := A \oplus_{\mathcal{H}} B$ and $\mathcal{H}$-matrix multiplication $C := A \otimes_{\mathcal{H}} B$ where again $C \in \mathcal{H}(T,k)$.

A truncation of a rank k' matrix R to rank $k < k'$ is defined as the best approximation with respect to the Frobenius (or spectral) norm in the set of rank k matrices. In the context of $\mathcal{H}$-matrices, we use such truncations for all admissible (rank k) blocks. Using truncated versions of the QR-decomposition and singular value decomposition, the truncation of a rank k' matrix $R \in \mathbb{R}^{n,m}$ (given in the form $R = AB^T$ where $A \in \mathbb{R}^{n,k'}$ and $B \in \mathbb{R}^{m,k'}$) to a lower rank can be computed with complexity $5(k')^2(n+m) + 23(k')^3$; further details are provided in [5].

We then define the $\mathcal{H}$-matrix addition and multiplication as follows:

$$\begin{aligned} A \oplus_{\mathcal{H}} B &= \mathcal{T}^{\mathcal{H}}_{k\leftarrow 2k}(A+B); \\ A \otimes_{\mathcal{H}} B &= \mathcal{T}^{\mathcal{H}}_{k\leftarrow k'}(AB) \end{aligned}$$

where $k' \leq c(p+1)k$ is the rank of the exact matrix product, c denotes some constant (which depends on the block cluster tree T) and p denotes the depth of the tree. Estimates show that the $\mathcal{H}$-matrix addition and multiplication have almost optimal complexity and are provided in [5] along with efficient implementations of these operations.

The approximate $\mathcal{H}$-matrix addition and multiplication permit the explicit computation of an approximate LU factorization in $\mathcal{H}$-matrix format. The $\mathcal{H}$-LU factorization has been described in detail for various clustering techniques [1, 6, 7, 12, 17]. The existence of $\mathcal{H}$-LU factors has been the subject of [2]. The $\mathcal{H}$-LU factorization is defined recursively in the hierarchical block structure of the $\mathcal{H}$-matrix. For a tree of depth 0, we compute the exact LU decomposition in full matrix format.

ETNA
Kent State University
etna@mcs.kent.edu

For domain-decomposition based $\mathcal{H}$-matrices (assuming a problem in two spatial dimensions where the generalization to three dimensions is straight-forward), the structure of the matrix A and its LU factors is

$$\begin{bmatrix} A_{11} & & A_{13} \\ & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \approx \begin{bmatrix} L_{11} & & \\ & L_{22} & \\ L_{31} & L_{32} & L_{33} \end{bmatrix} \otimes \begin{bmatrix} U_{11} & & U_{13} \\ & U_{22} & U_{23} \\ & & U_{33} \end{bmatrix} \tag{2.4}$$

The decomposition is computed in three steps:

1. Determine the $\mathcal{H}$-LU factorizations $A_{ii} \approx L_{ii} \otimes U_{ii}$ for $i \in \{1, 2\}$ (defined recursively).
2. Perform the triangular solves $L_{3j}U_{jj} \approx A_{3j}$ and $L_{jj}U_{j3} \approx A_{j3}$ for $j \in \{1, 2\}$.
3. Compute the Schur complement $\tilde{A}_{33} := A_{33} \ominus (L_{31} \otimes U_{13}) \ominus (L_{32} \otimes U_{23})$ and a subsequent $\mathcal{H}$-LU decomposition $\tilde{A}_{33} \approx L_{33}U_{33}$, where $\tilde{A}_{33}$ is based on a binary tree [7].

The complexity for the $\mathcal{H}$-LU decomposition is estimated in [6] to be almost optimal.

Once we have obtained an approximate $\mathcal{H}$-LU decomposition $A \approx L_{\mathcal{H}}U_{\mathcal{H}}$, we can solve the system $L_{\mathcal{H}}U_{\mathcal{H}}x = b$ by two subsequent triangular solves $L_{\mathcal{H}}y = b$, $U_{\mathcal{H}}x = y$, which both are performed exactly in $\mathcal{O}(n \log nk)$.

3. $\mathcal{H}$-matrix preconditioners in saddle point problems. In this section, we introduce the model problem and its discretization (subsection 3.1), review the construction of the block preconditioner developed in [15] (subsection 3.2), and then develop a new, joint preconditioner for the iterative solution of the discrete model problem (subsection 3.3).

Both the block and joint approaches to construct preconditioners cannot use the standard $\mathcal{H}$-matrix techniques but require careful adjustments to lead to successful preconditioners. The adjustments for the block preconditioner are described in [15], whereas the adjustments for the joint preconditioner, a local pivoting strategy and the enforcement of the pressure uniqueness, are derived in this paper. Without these adjustments, a brute-force application of standard $\mathcal{H}$-matrix techniques would fail to yield successful preconditioners.

3.1. The model problem: Oseen equations. As a model problem, we consider the Oseen equations: Let $\Omega \subset \mathbb{R}^d$, $d \in \{2, 3\}$, denote a bounded, connected domain with a piecewise smooth boundary Γ. Given a force field $f : \Omega \to \mathbb{R}^d$, boundary data $g : \Gamma \to \mathbb{R}^d$, the kinematic viscosity coefficient ϵ, and a given, divergence-free coefficient $b : \Omega \to \mathbb{R}^d$, the problem is to find the velocity field $u : \Omega \to \mathbb{R}^d$ and the pressure $p : \Omega \to \mathbb{R}$ such that the Oseen equations

$$-\epsilon\Delta u + (b \cdot \nabla)u + \nabla p = f \quad \text{in } \Omega, \tag{3.1}$$

$$-\mathrm{div} u = 0 \quad \text{in } \Omega, \tag{3.2}$$

$$\mathcal{B}u = g \quad \text{on } \Gamma, \tag{3.3}$$

are satisfied. Here, $\mathcal{B}$ denotes some type of boundary operator. A (mixed finite element) discretization of the Oseen equations leads to a system of equations of the form

$$\begin{pmatrix} A & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix} \tag{3.4}$$

with $A \in \mathbb{R}^{2n,2n}$ and $B \in \mathbb{R}^{m,2n}$. The pressure p is determined only up to a constant. In order to enforce a unique solution, one typically requires $\int_\Omega p \, d\Omega = 0$ which may be realized through the use of Lagrange multipliers and leads to an augmented, non-singular system

$$\begin{pmatrix} A & B^T & 0 \\ B & 0 & c \\ 0 & c^T & 0 \end{pmatrix} \begin{pmatrix} u \\ p \\ \lambda \end{pmatrix} = \begin{pmatrix} f \\ 0 \\ 0 \end{pmatrix}$$

where $c \in \mathbb{R}^m$. Elimination of the Lagrange multiplier λ leads to the equivalent, non-singular system

$$\begin{pmatrix} A & B^T \\ B & c \cdot c^T \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix}. \tag{3.5}$$

In the following subsections, we will derive preconditioners to be used in the iterative solution of such systems of equations. The block approach of subsection 3.2 builds upon the given two-by-two block structure in (3.5) which results from the distinction of velocity and pressure variables. The joint approach of subsection 3.3 in turn does not employ this block structure.

3.2. Block approach. In the so-called block preconditioners, we employ the given block structure of the matrix. In fact, if the first diagonal block A is invertible, the saddle point matrix in (3.5) admits the following block LDU factorization:

$$\begin{pmatrix} A & B^T \\ B & c \cdot c^T \end{pmatrix} = \begin{pmatrix} I & 0 \\ BA^{-1} & I \end{pmatrix} \begin{pmatrix} A & 0 \\ 0 & S \end{pmatrix} \begin{pmatrix} I & A^{-1}B^T \\ 0 & I \end{pmatrix} \tag{3.6}$$

where $S = c \cdot c^T - BA^{-1}B^T$ denotes the (pressure) Schur complement. In [15], such an LDU factorization (3.6) is used as the basis for the construction of block diagonal, block triangular and block LU preconditioners

$$W_{diag} := \begin{pmatrix} L_A^{\mathcal{H}} U_A^{\mathcal{H}} & 0 \\ 0 & L_S^{\mathcal{H}} U_S^{\mathcal{H}} \end{pmatrix}, \quad W_{triang} := \begin{pmatrix} L_A^{\mathcal{H}} U_A^{\mathcal{H}} & B^T \\ 0 & L_S^{\mathcal{H}} U_S^{\mathcal{H}} \end{pmatrix},$$

$$\begin{aligned} W_{\mathcal{H}LU} &:= \begin{pmatrix} I & 0 \\ B(L_A^{\mathcal{H}} U_A^{\mathcal{H}})^{-1} & I \end{pmatrix} \cdot W_{triang} \\ &= \begin{pmatrix} L_A^{\mathcal{H}} & 0 \\ L_{SA}^{\mathcal{H}} & L_S^{\mathcal{H}} \end{pmatrix} \begin{pmatrix} U_A^{\mathcal{H}} & U_{AS}^{\mathcal{H}} \\ 0 & U_S^{\mathcal{H}} \end{pmatrix} \end{aligned}$$

with $\mathcal{H}$-LU factorizations $L_A^{\mathcal{H}} U_A^{\mathcal{H}} \approx A$ and $L_S^{\mathcal{H}} U_S^{\mathcal{H}} \approx S$, resp.. The off-diagonal blocks $L_{SA}^{\mathcal{H}}$, $U_{AS}^{\mathcal{H}}$ in $W_{\mathcal{H}LU}$ have been computed from $L_{SA}^{\mathcal{H}} U_A^{\mathcal{H}} = B$ and $L_A^{\mathcal{H}} U_{AS}^{\mathcal{H}} = B^T$ using (block) triangular solves in $\mathcal{H}$-arithmetic [6, 7]. Details on how to obtain these $\mathcal{H}$-LU factorizations are given in [15]. In this approach, the set-up time for the preconditioner is dominated by the time to compute the approximate Schur complement (of size $m \times m$ where $m \in \mathcal{O}(n)$ for many mixed finite element discretizations) and its $\mathcal{H}$-LU factorization (both in $\mathcal{H}$-arithmetic). In order to reduce the set-up time, we will propose a so-called joint approach based on domain decomposition clustering in which the computation of a pressure Schur complement of size $\mathcal{O}(n)$ is replaced by an interior boundary Schur complement of size $\mathcal{O}(n^{1-1/d})$ where d denotes the spatial dimension of the problem.

3.3. Joint approach. In the joint approach, we no longer distinguish between velocity and pressure unknowns during the clustering process but apply the domain-decomposition clustering (which has been described in subsection 2.1) to the complete index set. An illustration in the case of a Taylor Hood discretization is given in Fig. 3.1: In the case of two spatial dimensions, every gridpoint (filled or hollow) corresponds to two velocity unknowns whereas every filled grid point corresponds in addition to a pressure unknown. This

ETNA
Kent State University
etna@mcs.kent.edu

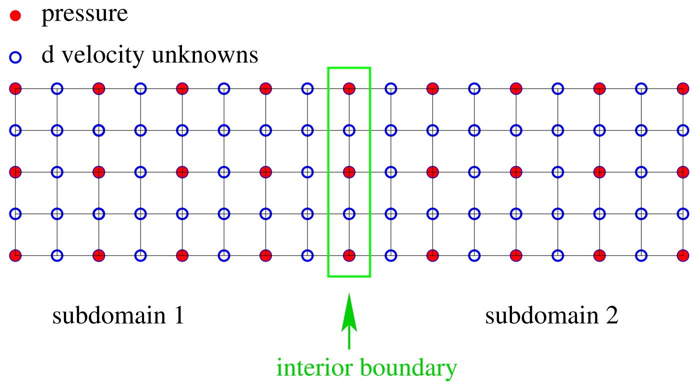

FIG. 3.1. *Domain-decomposition clustering for the Taylor Hood finite elements.*

subdomain cluster is subdivided into three subsets. Two subsets contain the indices of the respective subdomains, and the third subset contains the indices of the interior boundary. All three subsets typically contain velocity as well as pressure variables.

Typical $\mathcal{H}$-matrix structures resulting from the block and joint approaches, resp., are displayed in Fig. 3.1.

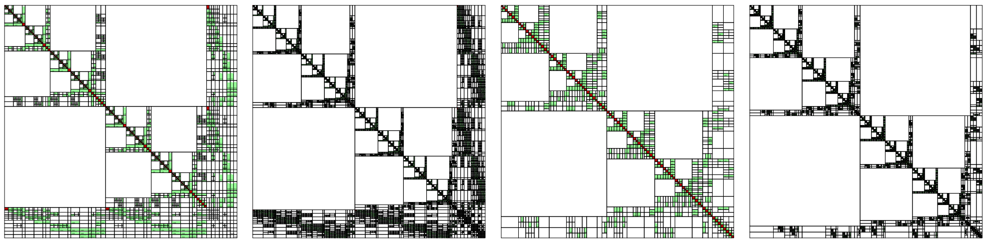

FIG. 3.2. *Typical $\mathcal{H}$-matrix structures: Block approach for* 2211 *(first) and* 8466 *(second) unknowns. Joint approach for* 2211 *(third) and* 8466 *(fourth) unknowns.*

The first two matrix structures result from the block approach for $2n + m = 2211$ and $2n + m = 8466$ unknowns, resp.. We note that the number of pressure variables (size of third block row) in relation to the total number of unknowns remains about the same, i.e., $m \in \mathcal{O}(n)$, when the grid is refined. The third and fourth matrix structures result from the joint approach for the same two matrix sizes as before. Here, the third block row corresponds to the unknowns of the interior boundary in the domain decomposition clustering. We note that its relative size decreases as the problem size increases, namely $m \in \mathcal{O}(n^{1-1/d})$. This fact will have a positive effect on the computation time of the $\mathcal{H}$-LU factorization with respect to the given $\mathcal{H}$-matrix structure.

However, before an $\mathcal{H}$-LU factorization can be computed, we need to address the question of its existence. In particular, we need to

- revisit the use of Lagrange multipliers to regularize the system matrix (3.5),
- address local pivoting strategies to avoid breakdowns through division by zero during the LU factorization.

3.3.1. Uniqueness of pressure. The discrete version of the condition $\int_\Omega p \, dx = 0$ is given by $c^T p = 0$ and enters the discrete system (3.5) in form of the rank 1 block $c \cdot c^T$. These additional matrix entries do not impose any difficulties for the block approach of Sec-

ETNA
Kent State University
etna@mcs.kent.edu

tion 3.2. In fact, their rank 1 representation allows for their efficient addition to an $\mathcal{H}$-matrix approximation of $-BA^{-1}B^T$ in order to obtain an $\mathcal{H}$-approximation to the Schur complement $S = c \cdot c^T - BA^{-1}B^T$.

The joint approach, however, is based on a domain decomposition of the complete index set into two subsets of *pairwise uncoupled* indices and an additional interior boundary. Here, two indices i, j are called *uncoupled* if the corresponding matrix entry a_{ij} is zero. The resulting matrix structure has large, off-diagonal zero blocks which remain zero in a subsequent LU-factorization. In (3.4), any two pressure indices are uncoupled due to the zero diagonal block in the system. However, in the regularized system (3.5), any two pressure indices may now be *coupled* (in case $c_i \neq 0, i = 1, \cdots, m$). As a result, all pressure variables will have to belong to the interior boundary in which case this joint approach would lose its advantage over the block approach.

Therefore, instead of $\int_\Omega p \, dx = 0$, we will require $p_m = 0.0$, i.e., fix the pressure at the geometric location that corresponds to the pressure variable p_m. The corresponding discrete system then has the form

$$\begin{pmatrix} A & B^T \\ B & C \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} \tilde{f} \\ 0 \end{pmatrix}$$

with the (still rank 1) $m \times m$ matrix

$$C = \begin{pmatrix} 0 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & 0 \\ 0 & \cdots & 0 & 1 \end{pmatrix}.$$

As a result, all pressure variables remain uncoupled and may be clustered into separate subdomains in the domain decomposition clustering of this joint approach.

3.4. Local pivoting strategy. According to (2.2), a block cluster $s \times t$ is not further subdivided if the clusters s or t are small, i.e., contain less than n_{min} indices where typically $32 \leq n_{min} \leq 100$. A leaf cluster s typically contains velocity as well as pressure indices, and the corresponding diagonal matrix block $A|_{s \times s}$ is represented as a full matrix. During an $\mathcal{H}$-LU factorization, the entries in this diagonal block will be updated during the elimination of prior diagonal blocks to obtain some matrix $\tilde{A}|_{s \times s}$ for which then an exact LU factorization is computed. Up to now, the ordering of indices *within* leaf clusters has been arbitrary. This, however, may lead to a breakdown in the $\mathcal{H}$-LU factorization if pressure indices are ordered prior to velocity indices (due to the zero entries along the diagonal in the matrix C). We will therefore enforce a *local, apriori* pivoting strategy within leaf clusters by requiring velocity indices to be ordered before pressure indices. Except for pathological, unlikely to occur leaf subdomains (e.g., a leaf consisting of only pressure variables which can be avoided in the preceding clustering process), $A|_{s \times s}$ may be interpreted as a discrete (non-singular) Oseen problem on a subdomain of Ω with appropriate (Dirichlet and Neumann) boundary conditions. If $A|_{s \times s}$ is non-singular, then the local pivoting strategy of ordering velocity variables before pressure variables will ensure that all submatrices $A_k = (a_{ij})_{1 \leq i,j \leq k}$ of $A|_{s \times s}$ are invertible so that there exists a unique LU factorization of $A|_{s \times s}$.

3.5. A recursive evaluation scheme to reduce storage requirements. The off-diagonal blocks in an $\mathcal{H}$-LU factorization (2.4), are typically computed explicitly in $\mathcal{H}$-arithmetic and stored as $\mathcal{H}$-matrices. The (exact) solutions of $L_{\mathcal{H}} x = b$ or $U_{\mathcal{H}} x = b$ for x require matrix-vector multiplications involving these off-diagonal blocks. Recalling that these off-diagonal

ETNA
Kent State University
etna@mcs.kent.edu

blocks have been computed by triangular solves, e.g., L_{31} in (2.4) has been computed from $L_{31}U_{11} = A_{31}$, we may replace the multiplication by L_{21} with a triangular solve involving U_{11}, followed by a multiplication with the sparse matrix block A_{21}, i.e., the computation $z := L_{21}x$ may be replaced by $z = A_{21}U_{11}^{-1}x$. As a result, the matrix block L_{21} is no longer required and need not be stored. Whereas this approach can be applied for both geometric bisection and domain-decomposition based $\mathcal{H}$-LU preconditioners, it is only recommmended if storage is more critical than computational complexity. In general, the indirect evaluation is more expensive than the multiplication by L_{21}, and if this scheme is applied recursively on all levels of the $\mathcal{H}$-matrix block structure, it can be shown that the solution $L_{\mathcal{H}}x = b$ is of $\mathcal{O}(n^2)$ where n denotes the size of $L_{\mathcal{H}}$, i.e., it is no longer (almost) optimal. It should also be noted that even though the off-diagonal blocks are not required for such an indirect solution of the triangular blocks, they are still required for the computation of the (diagonal blocks of the) $\mathcal{H}$-LU factors, i.e., storage for them needs to be initially provided before it can be released again once the $\mathcal{H}$-LU factorization is computed.

4. Numerical results. In this section, we will provide comparative numerical results for the block and joint $\mathcal{H}$-preconditioners in the iterative solution of the discrete (two-dimensional) Oseen equations on $\Omega = [-1,1] \times [-1,1]$. We will provide numerical results for varying values of ϵ (i.e., varying convection dominance) and various convection directions $b \in \{b_{xline}, b_{circ}, b_{recirc}\}$ in the Oseen equations (3.1) where

$$b_{xline}(x,y) = (1,0)^T, \qquad b_{circ}(x,y) = \left(\frac{1}{2} - y,\ x - \frac{1}{2}\right)^T,$$
$$b_{recirc}(x,y) = (4x(x-1)(1-2y),\ -4y(y-1)(1-2x))^T.$$

We use a finite element discretization on a structured mesh with Tabata's upwind triangle scheme [18, Chap. III, Sec. 3.1.1] to obtain the discrete saddle point system of linear equations $\mathcal{A}x = f$.

All numerical tests have been performed on a Dell 690n workstation (2.33GHz, 32GB memory) using the standard $\mathcal{H}$-matrix library HLIB (cf. `http://www.hlib.org`). We choose $x_0 = (0, \cdots, 0)^T$ as the initial vector to solve the discrete system by a preconditioned BiCGstab iteration. We iterate until either the maximum number of 200 iterations has been reached, or until the residual has been reduced by a factor of 10^{-6}. If the residual is not reduced by a factor of at least 10^{-6} within 200 iteration steps, we denote this by "div". We list the average convergence rates which are computed as $\sqrt[n]{r_n/r_0}$ where $r_n = \|b - Ax_n\|_2$ denotes the norm of the n'th residual. Throughout we used the adaptive $\mathcal{H}$-arithmetic (2.3) which is typically superior to fixed ranks, in particular for highly non-symmetric problems [7]. We set $n_{min} = 64$, i.e., any cluster with less than 64 nodes is no further refined, and $\eta = 4$ in the admissibility condition (2.1). The performance of both block and joint $\mathcal{H}$-LU preconditioners in the bicgstab iteration are compared with each other and also compared with the exact factorization solver PARDISO [19, 20, 21].

In the first Table 4.1, we provide the set-up times (in seconds) for the joint and block preconditioners for adaptive $\mathcal{H}$-accuracies $\delta = 0.1, 0.01$, and $\delta = 0.001$, resp.. We note that in the block approach, the first matrix block A in (3.4) is block diagonal with identical diagonal blocks so that only its first diagonal block needs to be stored and factored.

The set-up times increase (almost) linearly in the problem size n, with the joint preconditioner being computed more than twice as fast as the block preconditioner. With increasing problem size, the set-up times for the joint and also block $\mathcal{H}$-LU factorizations become significantly faster than the time for the exact factorization provided by PARDISO.

ETNA
Kent State University
etna@mcs.kent.edu

TABLE 4.1
Set-up times in seconds for the $\mathcal{H}$-LU factorization ($\epsilon = 10^{-2}$, b_{circ})

n	91,206	178,086	362,406	724,203	1,444,806
Joint $\mathcal{H}(0.1)$	9	14	33	51	110
Joint $\mathcal{H}(0.01)$	13	21	50	93	206
Joint $\mathcal{H}(0.001)$	18	30	72	141	326
Block $\mathcal{H}(0.1)$	18	38	95	199	471
Block $\mathcal{H}(0.01)$	25	57	141	321	804
Block $\mathcal{H}(0.001)$	38	90	225	568	1573
PARDISO	17	47	140	406	1172

In Table 4.2, we show the storage (in MB) that is required to compute and store the $\mathcal{H}$-LU factors. As anticipated, the joint $\mathcal{H}$-LU factors require less storage than the block $\mathcal{H}$-LU factors. In both cases, however, storage increases almost linearly in the problem size.

TABLE 4.2
Storage (in MB) for the $\mathcal{H}$-LU factorization ($\epsilon = 10^{-2}$, b_{circ})

n	91,206	178,086	362,406	724,203	1,444,806
Joint $\mathcal{H}(0.1)$	161	285	638	1100	2302
Joint $\mathcal{H}(0.01)$	195	356	790	1500	3185
Joint $\mathcal{H}(0.001)$	228	416	928	1754	3763
Block $\mathcal{H}(0.1)$	163	321	769	1414	3071
Block $\mathcal{H}(0.01)$	191	382	860	1712	3723
Block $\mathcal{H}(0.001)$	236	472	1075	2141	4696
PARDISO	180	385	863	1875	4049

Table 4.3 shows the convergence rates and iteration times (in seconds) that have been achieved by the resulting $\mathcal{H}$-LU preconditioners. Comparing the iteration times with the set-up times in Table 4.1, we note that the set-up still takes longer than the actual iteration. The high set-up times can be justified if the system has to be solved for a higher accuracy or if it has to be solved for several right hand sides. Since PARDISO requires only a backward and forward solve, its solution time is faster than the iterative schemes (however, with greater set-up time, see Table 4.1).

In Table 4.4 we show the convergence rates and times of both the block and the joint approach for varying convection directions. Here, we set $\epsilon = 10^{-2}$ and $n = 724,203$. "b_{xline}" denotes a constant convection in x-direction, resp.. "b_{circ}" and "b_{recirc}" denote two different non-constant, cyclic convection directions as defined at the beginning of this section. The results of the joint preconditioner vary to some degree with the convection directions. The block preconditioner appears to be independent of the convection direction.

In Table 4.5, we test the block and joint preconditioners' dependence on the convection dominance. Interestingly, they behave quite opposite: The block approach yields a convergent iteration for $\epsilon = 1.0$ whereas the joint approach diverges, but as ϵ decreases, i.e., as the convection becomes dominant, the performance of the block approach deteriorates whereas the convergence properties of the joint approach improve.

In Fig. 4.1, we display typical convergence histories for varying convection dominance for both the joint (top) and block (bottom) approach. Here, we plot the residual norm (on a logarithmic scale) versus the number of iteration steps. The convergence histories have been obtained for a problem with $n = 362,406$ unknowns, an adaptive $\mathcal{H}$-accuracy $\delta = 0.01$ and

ETNA
Kent State University
etna@mcs.kent.edu

TABLE 4.3
Convergence rates (top block) and iteration times (in seconds, bottom block) for increasing problem sizes ($\epsilon = 10^{-2}$, b_{circ})

n	91,206	178,086	362,406	724,203	1,444,806
Joint $\mathcal{H}(0.1)$	0.89	0.93	div	div	div
Joint $\mathcal{H}(0.01)$	4.2e-3	0.05	0.50	0.27	0.62
Joint $\mathcal{H}(0.001)$	4.4e-4	3.4e-4	5.5e-4	1.4e-3	3.0e-3
Block $\mathcal{H}(0.1)$	div	div	div	div	div
Block $\mathcal{H}(0.01)$	0.21	0.38	0.78	0.54	0.76
Block $\mathcal{H}(0.001)$	1.9e-2	0.06	0.16	0.42	0.39
Joint $\mathcal{H}(0.1)$	23	70	div	div	div
Joint $\mathcal{H}(0.01)$	0.9	2.6	20	22	114
Joint $\mathcal{H}(0.001)$	0.5	1.5	3	6	16
Block $\mathcal{H}(0.1)$	div	div	div	div	div
Block $\mathcal{H}(0.01)$	2.6	8	20	58	267
Block $\mathcal{H}(0.001)$	0.9	1.8	4	11	23
PARDISO	0.3	0.5	1.2	2.5	5

TABLE 4.4
Set-up (in seconds), iteration rate and time (in seconds), and storage (MB) for various convection directions ($\epsilon = 10^{-2}$, $n = 724,203$)

direction		xline	circ	recirc
Joint $\mathcal{H}(0.01)$	setup	93	93	92
	iteration	0.16/18	0.27/22	0.57/47
	storage	1502	1500	1482
Block$\mathcal{H}(0.01)$	setup	311	321	293
	iteration	0.55/63	0.54/58	0.54/55
	storage	1716	1712	1674
PARDISO	setup	422	406	434
	solve	2.5	2.5	2.5
	storage	1881	1875	1911

TABLE 4.5
Set-up (in seconds), iteration rate and time (in seconds), and storage (MB) for increasing convection dominance ($n = 724,203$, b_{circ})

ϵ		1.0	0.1	1e-2	1e-3	1e-4
Joint $\mathcal{H}(0.01)$	set-up	95	81	93	94	118
	iteration	div	div	0.27/22	0.05/11	0.21/18
	storage	1291	1395	1500	1487	1482
Block $\mathcal{H}(0.01)$	set-up	379	385	321	261	256
	iteration	0.56/59	0.57/70	0.54/58	0.80/147	div
	storage	1749	1755	1712	1665	1696
PARDISO	set-up	405	406	406	406	414
	solve	2.5	2.5	2.5	2.5	2.5
	storage	1875	1875	1875	1875	1887

ETNA
Kent State University
etna@mcs.kent.edu

the convection b_{circ}.

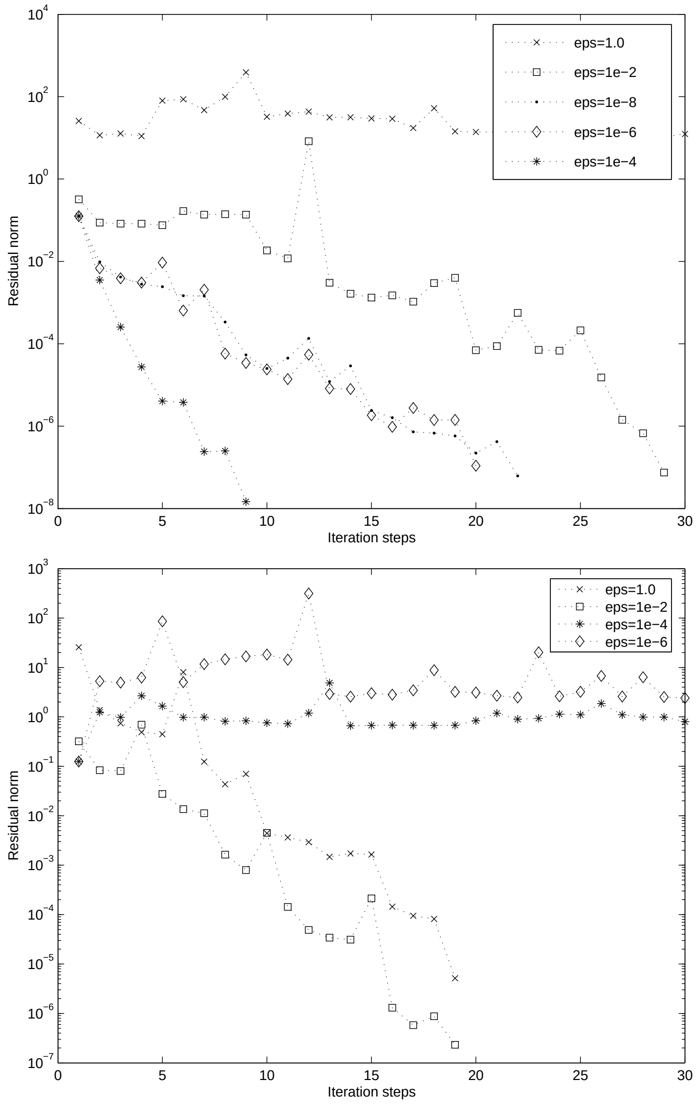

FIG. 4.1. *Convergence histories for* $n = 362406$ *unknowns, convection* b_{circ}*, varying convection dominance* ϵ*. Top: Joint(0.01) preconditioner; Bottom: Block(0.01) preconditioner.*

In summary, the joint approach is significantly more efficient than the block approach with respect to storage, set-up time and resulting convergence rates for the Oseen problem with *dominant* convection. Whereas both approaches lead to almost exact solvers as the adaptive $\mathcal{H}$-accuracy approaches zero, i.e., $\delta \to 0$, the joint approach leads to convergent iterations for much coarser accuracies. As a result, the joint $\mathcal{H}$-LU preconditioners require less set-up time and storage than the block $\mathcal{H}$-LU preconditioners. In the case of the Stokes equations, the block approach performs better than the joint approach. However, for this case there already exist several efficient preconditioners in the literature [3].

REFERENCES

[1] M. BEBENDORF, *Hierarchical LU decomposition based preconditioners for BEM*, Computing, 74 (2005), pp. 225–247.

[2] M. BEBENDORF, *Why approximate LU decompositions of finite element discretizations of elliptic operators can be computed with almost linear complexity*, Tech. Report 8, Max Planck Institute for Mathematics in the Sciences, 2005.

[3] M. BENZI, G. GOLUB, AND J. LIESEN, *Numerical solution of saddle point problems*, Acta Numer., 14 (2005), pp. 1–137.

[4] S. BÖRM, L. GRASEDYCK, AND W. HACKBUSCH, *Hierarchical Matrices*, 2003, Lecture Notes No. 21, Max-Planck-Institute for Mathematics in the Sciences, Leipzig, Germany, available online at www.mis.mpg.de/preprints/ln/.

[5] L. GRASEDYCK AND W. HACKBUSCH, *Construction and arithmetics of $\mathcal{H}$-matrices*, Computing, 70 (2003), pp. 295–334.

[6] L. GRASEDYCK, R. KRIEMANN, AND S. LE BORNE, *Parallel black box domain decomposition based $\mathcal{H}$-LU preconditioning*, Tech. Report 115, Max-Planck-Institute for Mathematics in the Sciences, 2005.

[7] L. GRASEDYCK AND S. LE BORNE, *$\mathcal{H}$-matrix preconditioners in convection-dominated problems*, SIAM J. Math. Anal., 27 (2006), pp. 1172–1183.

[8] L. GREENGARD AND V. ROKHLIN, *A fast adaptive multipole algorithm for particle simulations*, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 669–686.

[9] W. HACKBUSCH, *A sparse matrix arithmetic based on $\mathcal{H}$-matrices. Part I: Introduction to $\mathcal{H}$-matrices*, Computing, 62 (1999), pp. 89–108.

[10] W. HACKBUSCH, L. GRASEDYCK, AND S. BÖRM, *An introduction to hierarchical matrices*, Math. Bohem., 127 (2002), pp. 229–241.

[11] W. HACKBUSCH, B. KHOROMSKIJ, AND R. KRIEMANN, *Hierarchical matrices based on a weak admissibility criterion*, Computing, 73 (2004), pp. 207–243.

[12] I. IBRAGIMOV, S. RJASANOW, AND K. STRAUBE, *Hierarchical Cholesky decomposition of sparse matrices arising from curl-curl-equations*, Tech. Report 154, Universität des Saarlandes, 2005, to appear in J. Numer. Math.

[13] P. JONES, J. MA, AND V. ROKHLIN, *A fast algorithm for the solution of the Laplace equation on regions with fractal boundaries*, J. Comput. Phys., 113 (1994), pp. 35–51.

[14] R. KRIEMANN, *Parallel $\mathcal{H}$-matrix arithmetics on shared memory systems*, Computing, 74 (2005), pp. 273–297.

[15] S. LE BORNE, *Hierarchical matrix preconditioners for the Oseen equations*, Comput. Vis. Sci., (2006), to appear.

[16] S. LE BORNE, L. GRASEDYCK, AND R. KRIEMANN, *Domain-decomposition Based $\mathcal{H}$-LU Preconditioners*, in Domain Decomposition Methods in Science and Engineering XVI, O. Widlund and D. Keyes, eds., Lect. Notes Comput. Sci. Eng., Vol. 55, Springer, Berlin, 2006, pp. 661–668.

[17] M. LINTNER, *The eigenvalue problem for the 2D Laplacian in $\mathcal{H}$-matrix arithmetic and application to the heat and wave equation*, Computing, 72 (2004), pp. 293–323.

[18] H. ROOS, M. STYNES, AND L. TOBISKA, *Numerical Methods for Singularly Perturbed Differential Equations: Convection Diffusion and Flow Problems*, vol. 24, Computational Mathematics, Springer, Berlin, 1996.

[19] O. SCHENK AND K. GÄRTNER, *Solving unsymmetric sparse systems of linear equations with PARDISO*, Journal of Future Generation Computer Systems, 20 (2004), pp. 475–487.

[20] ———, *On fast factorization pivoting methods for symmetric indefinite systems*, Electron. Trans. Numer. Anal., 23 (2006), pp. 158–179. http://etna.math.kent.edu/vol.23.2006/pp158-179.dir/pp158-179.html.

[21] O. SCHENK, K. GÄRTNER, AND W. FICHTNER, *Efficient sparse LU factorization with left-right looking strategy on shared memory multiprocessors*, BIT, 40 (2000), pp. 158–176.

[22] E. E. TYRTYSHNIKOV, *Incomplete cross approximation in the mosaic-skeleton method*, Computing, 64 (2000), pp. 367–380.