Electronic Transactions on Numerical Analysis.
Volume 37, pp. 173-188, 2010.

ISSN 1068-9613.

ETNA
Kent State University
http://etna.math.kent.edu

AN IMPLICIT APPROXIMATE INVERSE PRECONDITIONER FOR SADDLE POINT PROBLEMS*

SABINE LE BORNE† AND CHE NGUFOR‡

Abstract. We present a preconditioner for saddle point problems which is based on an approximation of an implicit representation of the inverse of the saddle point matrix. Whereas this preconditioner does not require an approximation to the Schur complement, its theoretical analysis yields some interesting relationship to some Schur-complement-based preconditioners. Whereas the evaluation of this new preconditioner is slightly more expensive than the evaluation of standard block preconditioners from the literature, it has the advantage that, similar to constraint preconditioners, the iterates of the preconditioned system satisfy the constraint equations exactly. We will demonstrate the performance of the implicit approximate inverse preconditioner in the iterative solution of the discrete two- as well as three-dimensional Oseen equations.

Key words. saddle point problem, preconditioning

AMS subject classifications. 65F05, 65F30, 65F50, 65N22, 65N30

1. Introduction.

The ability to solve large, sparse systems arising from the (linearized) Navier-Stokes equations is critical to the simulation of incompressible fluid flow. Linear systems of equations are typically solved (approximately) by iterative methods that have linear storage and computational complexity (per iteration step) in the number of unknowns. However, the rate of convergence may be unacceptably slow, and one needs to accelerate the convergence by suitable preconditioning techniques. The design of robust and efficient preconditioners for linear systems arising in flow simulations is still a challenge.

Numerous solution techniques have been proposed in the literature for saddle point problems of the type

$$\underbrace{\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix}}_{=:\mathcal{A}} \begin{bmatrix} u \\ p \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix}. \tag{1.1}$$

A comprehensive survey [1] reviews many of the most promising solution methods with an emphasis on the iterative solution of these large, sparse, indefinite problems. Several of these preconditioners are based on block approaches which require approximate solves for auxiliary velocity as well as pressure Schur complement problems [4, 5, 6, 12, 14, 18]. The constraint preconditioner (and its variants) also employ the given block structure and yield iterates that satisfy the constraint of (discretely) divergence free velocity exactly [9, 11, 13].

In this paper, we will develop a preconditioner that is *not* based on a block LU factorization of the saddle point matrix but is a direct approximation to its inverse. The derivation will start from a well-known representation of the exact inverse which requires a matrix Z whose columns form a basis for the kernel of constraints, $ker(B^T)$. While such a matrix Z is typically unavailable, or only available at great expense, the new implicit inverse preconditioner is based on some approximation and reformulation and will no longer require such a matrix Z. Different from preconditioners based on an (approximate) LU factorization of the saddle

*Received August 13, 2009. Accepted for publication December 21, 2009. Published online June 6, 2010. Recommended by M. Benzi. A part of this material is based upon work supported by the National Science Foundation under Grant No. DMS-0913017.

†Department of Mathematics, Box 5054, Tennessee Technological University, Cookeville, TN 38505 (sleborne@tntech.edu).

‡Department of Mathematical Sciences, George Mason University, 4400 University Drive, MS: 3F2, Science & Tech. I, room 203, Fairfax, VA 22030 (cngufor@gmu.edu).

point matrix, the new preconditioner will still be applicable in the case of a singular matrix block A in (1.1).

Similar to constraint preconditioners, the use of the implicit approximate inverse preconditioner ensures that (in exact arithmetic) all of the iterates satisfy the constraints.

The remainder of this paper is structured as follows: In Section 2, we derive the new implicit inverse preconditioner and present some of its theoretical properties. We review some related preconditioners from the literature and discuss similarities, differences, and implementation costs. In Section 3, we introduce the model problem, the Oseen equations, and document numerical tests for problems in two as well as three spatial dimensions which illustrate the performance of the preconditioner. We provide comparative results with the related (BFBt-)Schur-complement preconditioner as well as with the direct solver PARDISO [20, 21].

2. Implicit approximate inverse preconditioner. In this section, we will derive a new block preconditioner for saddle point problems based on a certain representation of the inverse of a saddle point matrix. In particular, we will develop a preconditioner for the saddle point system (1.1), where $A \in \mathbb{R}^{n\times n}, B \in \mathbb{R}^{n\times m}$, $m \leq n$. The preconditioner will exploit the 2×2 block structure but does not require any additional information except for the matrix and right hand side data. We will make the following two assumptions which together guarantee that the saddle point matrix in (1.1) is invertible.

ASSUMPTION 2.1. *$B \in \mathbb{R}^{n\times m}$ has full rank m.*

ASSUMPTION 2.2. *The symmetric part $H := \frac{1}{2}(A^T + A)$ of A is positive semidefinite and $\ker(H) \cap \ker(B^T) = \{0\}$.*

Let $Z \in \mathbb{R}^{n\times(n-m)}$ denote any matrix whose columns form a basis for $ker(B^T)$. Defining

$$W = Z(Z^TAZ)^{-1}Z^T, \qquad V = B^TB,$$

there holds the following representation for the inverse (which does not require A to be invertible but requires B to have full rank),

$$\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix}^{-1} = \begin{bmatrix} W & (I - WA)BV^{-1} \\ V^{-1}B^T(I - AW) & -V^{-1}B^T(A - AWA)BV^{-1} \end{bmatrix}, \tag{2.1}$$

which can easily be proven using the fact that $B(B^TB)^{-1}B^T = I - ZZ^T$. In the past, this representation of the inverse has been of limited practical use, but has primarily been used for theoretical analyses. The reason lies in the need of computing a (well-conditioned) null space matrix Z as well as the product Z^TAZ and its inverse $(Z^TAZ)^{-1}$ as part of the matrix W.

Here, we propose to avoid these disadvantages simultaneously by replacing $(Z^TAZ)^{-1}$ with $Z^TA^{-1}Z$ and making use of the identity $ZZ^T = I - B(B^TB)^{-1}B^T$. This leads to the following approximation $\widetilde{W}$ of W,

$$\widetilde{W} := ZZ^TA^{-1}ZZ^T = (I - BV^{-1}B^T)A^{-1}(I - BV^{-1}B^T), \tag{2.2}$$

and the resulting representation of an approximate inverse,

$$\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix}^{-1} \approx \begin{bmatrix} \widetilde{W} & (I - \widetilde{W}A)BV^{-1} \\ V^{-1}B^T(I - A\widetilde{W}) & -V^{-1}B^TA(I - \widetilde{W}A)BV^{-1} \end{bmatrix} =: P. \tag{2.3}$$

We will refer to P as the *implicit approximate inverse* preconditioner.

THEOREM 2.3. *The evaluation of the matrix-vector product $\begin{bmatrix} v \\ w \end{bmatrix} = P \begin{bmatrix} x \\ y \end{bmatrix}$ requires the following matrix-vector multiplications: One multiplication by A^{-1}, four multiplications*

ETNA
Kent State University
http://etna.math.kent.edu

by $V^{-1} = (B^TB)^{-1}$, two multiplications with the sparse matrix A and six multiplications with the sparse matrix B (or B^T).

Proof. The number of matrix-vector multiplications can be counted from the following implementation,

$$\begin{aligned} d &:= BV^{-1}y;\\ e &:= Ad;\\ f &:= \widetilde{W}(x-e);\\ v &:= d+f;\\ w &:= V^{-1}B^T(x-Av). \end{aligned}$$

Here, the multiplication by $\widetilde{W}$ (2.2) includes one multiplication by A^{-1}, two multiplications by V^{-1}, and four multiplications by B (or B^T). □

In the remainder of this work, we will use the following notation.

DEFINITION 2.4. *Given a saddle point matrix* $\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix}$, *we define*

$$\begin{aligned} Z &: \text{ denotes a matrix whose columns form a basis for } ker(B^T);\\ V &:= B^TB;\\ X &:= BV^{-1}B^T;\\ \widetilde{W} &:= (I-X)A^{-1}(I-X), \qquad (\text{see } (2.2));\\ Y &:= A^{-1}XA. \end{aligned}$$

We use $\sigma(C)$ to denote the spectrum of a matrix C, and I_k to denote a $k \times k$ identity matrix.

LEMMA 2.5. *The matrices X and Y are projections.*

Proof. The proposition follows from

$$\begin{aligned} XX &= BV^{-1}B^TBV^{-1}B^T = BV^{-1}VV^{-1}B^T = X;\\ YY &= A^{-1}XAA^{-1}XA = A^{-1}XXA = Y. \quad \square \end{aligned}$$

The following Lemma will be used in the proofs of subsequent theorems.

LEMMA 2.6. *Let $P \in \mathbb{R}^{n,n}$ denote a projection matrix, i.e., $P^2x = Px$ for all $x \in \mathbb{R}^n$. Let $I \in \mathbb{R}^{n,n}$ denote the identity matrix, and let $A \in \mathbb{R}^{n,n}$ denote an arbitrary matrix. Then there holds*

$$\sigma((I-A)P) \cup \{0,1\} = \{1-\mu \mid \mu \in \sigma(AP)\} \cup \{0,1\}. \tag{2.4}$$

Proof. The following proof makes repeated use of the fact that

$$\sigma(AB) \cup \{0\} = \sigma(BA) \cup \{0\}$$

for any two matrices $A, B \in \mathbb{R}^{n,n}$.

"$\subseteq$": Let $\lambda \in \sigma((I-A)P) \setminus \{0,1\}$. Since

$$\sigma((I-A)P) \cup \{0\} = \sigma(P(I-A)) \cup \{0\},$$

it follows that $\lambda \in \sigma(P(I-A))$, i.e., there exists a corresponding eigenvector v with $P(I-A)v = \lambda v$. Premultiplying this equation by P yields $P(I-A)v = \lambda P v$. A comparison of these two equations yields $Pv = v$ (since $\lambda \neq 0$). Thus, $P(I-A)v = \lambda v$ is equivalent to $PAv = (1-\lambda)v$ and implies $\mu := 1-\lambda \in \sigma(PA)$. It follows that

$$\begin{aligned}\lambda = 1-\mu &\in \{1-\mu \mid \mu \in \sigma(PA)\}\\ &\subseteq \{1-\mu \mid \mu \in \sigma(AP) \cup \{0\}\}\\ &= \{1-\mu \mid \mu \in \sigma(AP)\} \cup \{1\}.\end{aligned}$$

"$\supseteq$": Let $\mu \in \sigma(AP) \setminus \{0,1\}$. Since $\sigma(AP) \cup \{0\} = \sigma(PA) \cup \{0\}$ and $\mu \neq 0$, it follows that $\mu \in \sigma(PA)$. Thus, there exists an associated eigenvector v that satisfies $PAv = \mu v$. Premultiplication by P yields $PAv = \mu P v$ which implies $v = Pv$ (since $\mu \neq 0$). Thus, there holds $P(I-A)v = Pv - PAv = v - \mu v = (1-\mu)v$, so

$$(1-\mu) \in \sigma(P(I-A)) \subseteq \sigma((I-A)P) \cup \{0\}. \quad \square$$

The following theorem lists some of the properties of the implicit approximate inverse preconditioner.

THEOREM 2.7. *Let $P \in \mathbb{R}^{(n+m)\times(n+m)}$ denote the implicit approximate inverse preconditioner as defined in (2.3), and let*

$$M = I_{n+m} - P \begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \tag{2.5}$$

denote the error propagation matrix. Then the following statements hold.
a) If A is symmetric, then P is also symmetric, i.e., $A = A^T$ implies $P = P^T$.
b) The error propagation matrix has the form

$$M = \begin{bmatrix} (I-\widetilde{W}A)(I-BV^{-1}B^T) & 0 \\ -V^{-1}B^TA(I-\widetilde{W}A)(I-BV^{-1}B^T) & 0 \end{bmatrix},$$

i.e., the P-preconditioned iteration is u-dominant (only the velocity error is relevant for the error of the next iterate).
c) $M\begin{bmatrix} B \\ 0 \end{bmatrix} = 0$ and $M\begin{bmatrix} 0 \\ I_m \end{bmatrix} = 0$, i.e., M has $2m$ zero eigenvalues with explicitly known eigenvectors.
d) $rank(I-\widetilde{W}A) \leq 2m$.
e) $rank(M) \leq m$, i.e., M has at least n zero eigenvalues.
f) The m possibly non-zero eigenvalues of M are given by $\lambda = 1-\mu$, where μ are the eigenvalues of the $m \times m$ matrix $V^{-1}B^TA^{-1}BV^{-1}B^TAB$.
g) The approximate solution $(x,y)^T := P(f,g)^T$ satisfies the constraint $B^Tx = g$ in (1.1) exactly.
h) If the nullspace of B^T is an invariant subspace of A, then $M = 0$, i.e., the preconditioner P yields an exact solver.

Proof. The proof uses the previously defined matrices $Z, V, X, \widetilde{W}, Y$; see Definition 2.4.
a) and b) are straightforward.
c) Follows from $(I-BV^{-1}B^T)B = 0$ and b).
d) There holds

$$\begin{aligned}I-\widetilde{W}A &= I-(I-X)A^{-1}(I-X)A\\ &= X+Y-XY\\ &= X+(I-X)Y.\end{aligned} \tag{2.6}$$

ETNA
Kent State University
http://etna.math.kent.edu

The statement now follows from $rank(X) \leq m$ (since $rank(B) = m$) and $rank(Y) \leq m$.
e) In view of b), there holds $rank(M) = rank((I - \widetilde{W}A)(I - BV^{-1}B^T))$. Since

$$(I - \widetilde{W}A)(I - BV^{-1}B^T)B = (I - \widetilde{W}A)0 = 0$$

and, using $XZ = BV^{-1}B^T Z = 0$,

$$\begin{aligned}(I - \widetilde{W}A)(I - BV^{-1}B^T)Z &= (I - \widetilde{W}A)Z \\ &= (X + Y - XY)Z \\ &= (I - X)YZ,\end{aligned}$$

it follows that

$$rank(M) = rank(M \cdot [B\ Z]) = rank((I - X)YZ) \leq rank(Y) \leq m.$$

f) In view of part b), the nonzero eigenvalues of M are the eigenvalues of its first diagonal $(n \times n)$ block $M_{1,1} = (I - \widetilde{W}A)(I - BV^{-1}B^T)$. There holds

$$\begin{aligned}\sigma(M_{1,1}) \cup \{0,1\} &= \sigma((I - \widetilde{W}A)(I - X)) \cup \{0,1\} \\ &= \sigma((I - X)(I - \widetilde{W}A)) \cup \{0,1\} \\ &\overset{(2.6)}{=} \sigma((I - X)Y) \cup \{0,1\} \\ &\overset{(2.4)}{=} \{1 - \mu \mid \mu \in \sigma(XY)\} \cup \{0,1\},\end{aligned} \tag{2.7}$$

where we used $(I - X)X = 0$ and $(I - X)(I - X) = I - X$. The proposition now follows from

$$\begin{aligned}\sigma(V^{-1}B^TA^{-1}BV^{-1}B^TAB) \cup \{0\} &= \sigma(BV^{-1}B^TA^{-1}BV^{-1}B^TA) \cup \{0\} \\ &= \sigma(XA^{-1}XA) \cup \{0\} \\ &= \sigma(XY) \cup \{0\}.\end{aligned}$$

g) The matrix-vector multiplication $(x, y)^T := P(f, g)^T$ yields $x = \widetilde{W}f + (I - \widetilde{W}A)BV^{-1}g$. The statement $B^Tx = g$ now follows from $B^T\widetilde{W} = 0$.
h) Assume that Z is an invariant subspace of A, i.e, there exists a matrix S such that $AZ = ZS$. Then there holds

$$\begin{aligned}(I - \tilde{W}A)(I - BV^{-1}B^T) &= (I - \tilde{W}A)ZZ^T \\ &= ZZ^T - \underbrace{ZZ^TA^{-1}ZZ^T}_{\tilde{W}}\underbrace{ZS}_{AZ}Z^T \\ &= ZZ^T - ZZ^TA^{-1}ZSZ^T \\ &= ZZ^T - ZZ^TA^{-1}AZZ^T = ZZ^T - Z\underbrace{Z^TZ}_{I}Z^T \\ &= 0\end{aligned}$$

which yields $M = 0$. ☐

REMARK 2.8. In the approximate inverse preconditioner P (2.3), the exact inverses A^{-1} and $(B^TB)^{-1}$ may be replaced by approximations (including inner iterations). Multiplication by A^{-1} requires the solution of a (scalar) convection-diffusion problem, whereas the multiplication by $(B^TB)^{-1}$ requires the solution of a symmetric, positive definite system that shows similarities to a Laplace system. For both types of problems there exist highly efficient solution methods in the literature.

ETNA
Kent State University
http://etna.math.kent.edu

2.1. Review of related preconditioners. In the last few years, much work has been devoted to the development of efficient preconditioners for saddle point problems. In this section, we will review some of these techniques. In particular, we will restrict our attention to *blackbox* techniques that are in some sense related to the new implicit approximate inverse preconditioner. By *blackbox* techniques we mean techniques that only require the matrix data (and possibly information about its 2×2 block structure) and right hand side vector, but no information on the underlying system of partial differential equations.

Here, we will consider the following widely applicable classes of preconditioning techniques:

- Schur-complement-based preconditioners, in particular the BFBt preconditioner [5, 7, 18],
- preconditioned nullspace solver [15, 17],
- constraint preconditioners [9, 11, 13].

2.1.1. Schur-complement-based preconditioners (BFBt). The Schur-complement-based preconditioners are derived from a block LU factorization of the saddle point matrix and an approximation to the required Schur complement. The factorization

$$\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} = \begin{bmatrix} I & 0 \\ B^T A^{-1} & I \end{bmatrix} \begin{bmatrix} A & B \\ 0 & S \end{bmatrix},$$

with the Schur complement $S = -B^T A^{-1} B$ leads to the block triangular preconditioner

$$P_{triang}^{-1} := \begin{bmatrix} A & B \\ 0 & \widetilde{S} \end{bmatrix}, \text{i.e.,} \quad P_{triang} := \begin{bmatrix} A^{-1} & -A^{-1}B\widetilde{S}^{-1} \\ 0 & \widetilde{S}^{-1} \end{bmatrix}, \tag{2.8}$$

where $\widetilde{S}$ denotes an approximation to the Schur complement S for which the auxiliary problem $\widetilde{S}v = h$ can be solved efficiently. The error propagation matrix $M_{triang} := I - P_{triang}\mathcal{A}$ has the form

$$M_{triang} = \begin{bmatrix} A^{-1}B\widetilde{S}^{-1}B^T & -A^{-1}B \\ -\widetilde{S}^{-1}B^T & I \end{bmatrix}. \tag{2.9}$$

In the case of $\widetilde{S} = S$, M_{triang} has spectrum $\sigma(M_{triang}) = \{0\}$ since

$$\sigma\left(P_{triang}\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix}\right) = \sigma\left(\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} P_{triang}\right) = \sigma\left(\begin{bmatrix} I & 0 \\ B^T A^{-1} & I \end{bmatrix}\right) = \{1\}.$$

Using a matrix Z as defined in Def. 2.4, the columns of the matrix $\begin{bmatrix} Z & A^{-1}B \\ 0 & -I \end{bmatrix}$ are n (linearly independent) eigenvectors of M_{triang} associated with the eigenvalue $\lambda = 0$.

Here, we will restrict our attention to the BFBt-preconditioner

$$\widetilde{S}_{BFBt}^{-1} = -(B^T B)^{-1} B^T A B (B^T B)^{-1}$$

[5, 7, 18]. The following theorem states a close relationship between this Schur-complement preconditioner and the implicit approximate inverse preconditioner.

THEOREM 2.9. *The error propagation matrices M (2.5) of the implicit approximate inverse preconditioner and M_{triang} (2.9) using the BFBt-Schur-complement preconditioner $\widetilde{S}^{-1} := \widetilde{S}_{BFBt}^{-1}$ have the same set eigenvalues.*

ETNA
Kent State University
http://etna.math.kent.edu

Proof. There holds

$$\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} P_{triang}^{-1} = \begin{bmatrix} I & 0 \\ B^T A^{-1} & S\widetilde{S}_{BFBt}^{-1} \end{bmatrix},$$

thus $\sigma(M_{triang}) = \{0\} \cup \{1 - \mu \mid \mu \in \sigma(S\widetilde{S}_{BFBt}^{-1})\}$. The result now follows from Theorem 2.7f) since $V^{-1}B^T ABV^{-1}B^T A^{-1}B = \widetilde{S}_{BFBt}^{-1}S$. ∎

The block LU factorization of the saddle point matrix yields the following block LU-preconditioner.

$$P_{LU} := \begin{bmatrix} A & B \\ 0 & \widetilde{S} \end{bmatrix}^{-1} \begin{bmatrix} I & 0 \\ B^T A^{-1} & I \end{bmatrix}^{-1} \tag{2.10}$$

$$= \begin{bmatrix} A^{-1} & -A^{-1}B\widetilde{S}^{-1} \\ 0 & \widetilde{S}^{-1} \end{bmatrix} \begin{bmatrix} I & 0 \\ -B^T A^{-1} & I \end{bmatrix} \tag{2.11}$$

where $\widetilde{S}$ approximates the Schur complement $S = -B^T A^{-1} B$.

THEOREM 2.10. *The error propagation matrix $M_{LU} = I - P_{LU}\mathcal{A}$ has the form*

$$M_{LU} = \begin{bmatrix} 0 & -A^{-1}B(I - \widetilde{S}^{-1}S) \\ 0 & (I - \widetilde{S}^{-1}S) \end{bmatrix}.$$

The iteration is p-dominant, and there holds $\sigma(M_{LU}) = \{0\} \cup \sigma(I - \widetilde{S}^{-1}S)$. If we set $\widetilde{S}^{-1} := \widetilde{S}_{BFBt}^{-1}$, then the non-zero eigenvalues of M_{LU} are the same as the non-zero eigenvalues of M_{triang} and M.

2.1.2. Preconditioned nullspace solver. The (preconditioned) nullspace method to solve the linear system (1.1) is based on the following additional assumptions.

ASSUMPTION 2.11. *A particular solution $\hat{x}$ of $B^T x = g$ is available.*

ASSUMPTION 2.12. *A null space basis $Z \in \mathbb{R}^{n\times(n-m)}$ of B^T is available, i.e.,*

$$B^T Z = 0 \quad \textit{and} \quad \textit{rank}(Z) = n - m.$$

The required particular solution $\hat{x}$ may be computed through $\hat{x} = B(B^T B)^{-1}g$. The solution set of $B^T x = g$ is described by $x = Zv + \hat{x}$ as v ranges in $\mathbb{R}^{n-m}$. Substituting $x = Zv + \hat{x}$ into $Ax + By = f$, we obtain $A(Zv + \hat{x}) + By = f$. Premultiplying by the full-rank matrix Z^T yields $Z^T A(Zv + \hat{x}) + Z^T By = Z^T f$, and using $B^T Z = 0$ as well as rearranging the equation yields the reduced, non-singular problem

$$Z^T AZv = Z^T(f - A\hat{x}). \tag{2.12}$$

Once the solution v_* of the reduced problem has been computed, we compute the (velocity) solution $x_* = Zv_* + \hat{x}$.

Finally, the (pressure) solution y_* can be found by solving $B^T By = B^T(f - Ax_*)$ for y, a reduced system of order m with a sparse, symmetric positive definite coefficient matrix $B^T B$. It is easily verified that $(x_*, y_*)^T$ satisfies (1.1).

Preconditioning the reduced system (2.12) imposes difficulties similar to those for preconditioning a Schur complement system: The matrix product $Z^T AZ$ is typically not computed explicitly since matrix-matrix multiplications are to be avoided and the product would result in a fully populated matrix. Here, we will consider the preconditioner $W := Z^T A^{-1} Z$

ETNA
Kent State University
http://etna.math.kent.edu

that has previously been proposed in [15, 17], and we solve the preconditioned reduced nullspace system

$$(Z^T A^{-1} Z)(Z^T AZ)x = (Z^T A^{-1} Z)b.$$

THEOREM 2.13. *The error propagation matrix* $M_{precNull} := I - (Z^T A^{-1} Z)(Z^T AZ)$ *of the preconditioned nullspace method has the same set of eigenvalues as the error propagation matrix* M *(2.5) of the implicit approximate inverse preconditioner.*

Proof. Using the notation introduced in Def. 2.4, there holds

$$\begin{aligned}
\sigma((Z^T A^{-1} Z)(Z^T AZ)) \cup \{0\} &= \sigma((ZZ^T)A^{-1}(ZZ^T)A) \cup \{0\} \\
&= \sigma((I-X)A^{-1}(I-X)A) \cup \{0\} \\
&= \sigma((I-X)(I-A^{-1}XA)) \cup \{0\} \\
&= \sigma((I-X)(I-Y)) \cup \{0\} \\
&\overset{(2.6)}{=} \sigma(\widetilde{W}A) \cup \{0\}. \quad \square
\end{aligned}$$

2.2. Constraint preconditioners. A (non-singular) constraint preconditioner is given in the form

$$\begin{bmatrix} G & B \\ B^T & 0 \end{bmatrix},$$

where $G \in \mathbb{R}^{n,n}$ is some approximation to A. The following Theorem is proven in [13] for the case of symmetric blocks A and G.

THEOREM 2.14. *Assume that* Z *is a basis of* $ker(B^T)$. *The constraint-preconditioned matrix*

$$\begin{bmatrix} G & B \\ B^T & 0 \end{bmatrix}^{-1} \begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix}$$

has the following spectrum:

1. *An eigenvalue at* 1 *with multiplicity* $2m$.
2. $n-m$ *eigenvalues which are defined by the generalized eigenvalue problem* $Z^T AZx = \lambda Z^T GZx$.

COROLLARY 2.15. *The spectrum of the constraint-preconditioned matrix using* $G = I$ *is equal to the spectrum of the reduced matrix in (2.12) (except for the eigenvalue* 1*).*

Proof. Follows from $Z^T GZ = I$ when $G = I$ in Theorem 2.14 (part 2). $\square$

LEMMA 2.16. *The error propagation matrix of the constraint preconditioner is given by*

$$\begin{aligned}
M_{constraint} &:= I - \begin{bmatrix} G & B \\ B^T & 0 \end{bmatrix}^{-1} \begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \\
&= \begin{bmatrix} I - G^{-1}A + G^{-1}BS_G^{-1}B^T(I - G^{-1}A) & 0 \\ -S_G^{-1}B^T(I - G^{-1}A) & 0 \end{bmatrix},
\end{aligned}$$

where $S_G := -B^T G^{-1} B$.

REMARK 2.17. The constraint preconditioner requires the solution of a Schur complement problem $S_G y = b$. Whereas using $G = I$ allows for efficient solvers of this problem, it typically leads to poor convergence of the iterative solution of non-symmetric, strongly indefinite saddle point systems. In [8], the authors suggest to use the symmetric part of A as G, i.e., $G = \omega(A + A^T)$ for some parameter ω. Preconditioning with the resulting constraint preconditioner requires the solution of a Stokes-type (symmetric) problem.

ETNA
Kent State University
http://etna.math.kent.edu

3. Numerical results. In this section, we will provide numerical results to illustrate the performance of the approximate inverse preconditioner (2.3). We will use the technique of (domain-decomposition based) $\mathcal{H}$-matrices [3, 10, 16] to compute an approximate LU factorization $A \approx L_A^{\mathcal{H}} U_A^{\mathcal{H}}$ as well as an approximate Cholesky factorization $B^T B \approx L_{B^TB}^{\mathcal{H}} (L_{B^TB}^{\mathcal{H}})^T$. An $\mathcal{H}$-matrix provides an approximation to a (dense) matrix in which certain off-diagonal blocks are approximated by low-rank matrices. The accuracies of these approximations are controlled by prescribing a maximum relative error δ within each block. As this relative error approaches zero, i.e., $\delta \to 0$, the approximation becomes more accurate at the expense of increased computation times and storage requirements, similar to ILU methods in which a smaller threshhold parameter or increased level of fill leads to better but more expensive approximations.

We replace the exact inverses A^{-1} and $(B^T B)^{-1}$ in $\widetilde{W}$ (2.2) by the approximations $L_A^{\mathcal{H}} U_A^{\mathcal{H}}$ and $L_{B^TB}^{\mathcal{H}} (L_{B^TB}^{\mathcal{H}})^T$, resp. Alternative techniques such as multigrid or incomplete factorizations are also possible to solve these subproblems.

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 *stable* mixed finite element discretization of the Oseen equations leads to a system of equations of the form (1.1). Here, we set up the discrete Oseen equations (1.1) using a Taylor-Hood finite element discretization on a structured mesh on $\Omega = [-1, 1]^d$ ($d = 2, 3$) with Tabata's upwind triangle scheme [19, Chap. III, Sec. 3.1.1]. We perform tests using constant as well as (re-)circulating convection directions

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

Our choice of experiments has resulted from our interest in

- the set-up times and storage requirements of the implicit approximate inverse preconditioner for two and three spatial dimensions; see Tables 3.1, 3.2 ($d = 2$) and Tables 3.5, 3.6 ($d = 3$),
- the dependence of iteration steps on the meshsize h, the convection direction b, and the parameter ϵ, which determines the convection-dominance of the Oseen problem (3.1); see Table 3.3 ($d = 2$) and Table 3.7 ($d = 3$),
- the dependence of the convergence rates on the accuracy of the approximations to $(B^T B)^{-1}$ and A^{-1} in the implicit approximate inverse preconditioner, including a comparison with the BFBt-preconditioner; see Table 3.4 ($d = 2$) and Table 3.8 ($d = 3$).

All numerical tests have been performed on a Dell 690n workstation (2.33GHz, 32GB memory) using the standard $\mathcal{H}$-matrix library HLIB [2]. We choose $x_0 = (0, 0, 0)^T$ as the initial vector in the Bicgstab method. 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”. A breakdown in the bicgstab method is denoted by “br”.

3.1. Two-dimensional Oseen problem. In Tables 3.1 and 3.2, we report the set-up times and storage requirements of the approximate implicit inverse preconditioner. In particular, these numbers result from the $\mathcal{H}$-LU factorization of A and the $\mathcal{H}$-Cholesky factorization of $B^T B$. If these $\mathcal{H}$-factorizations are replaced by alternative methods (e.g., ILU factorization, multigrid, etc.), these numbers will change depending on the chosen method. Therefore, while the results in Tables 3.1 and 3.2 illustrate that $\mathcal{H}$-factorizations are well-suited to solve the required subproblems, they should not be interpreted as set-up costs intrinsic to the preconditioner.

TABLE 3.1
Set-up time (in seconds), $\epsilon = 10^{-2}$, b_{xline}

$n/2$	$\delta_{\mathcal{H}}$	40,401	78,961	160,801	321,489	641,601
	10^{-1}	1	2	6	10	26
$\mathcal{H}$-LU of A	10^{-2}	2	3	7	12	32
	10^{-3}	2	3	8	14	36
m		10,404	20,164	40,804	81,225	161,604
	10^{-2}	1	2	6	14	27
$\mathcal{H}$-Cholesky	10^{-4}	2	4	7	19	38
of $B^T B$	10^{-8}	2	5	10	25	53

TABLE 3.2
Storage (in MB), $\epsilon = 10^{-2}$, b_{xline}

$n/2$	$\delta_{\mathcal{H}}$	40,401	78,961	160,801	321,489	641,601
	10^{-1}	58	106	254	449	1091
$\mathcal{H}$-LU of A	10^{-2}	63	114	273	477	1151
	10^{-3}	66	122	286	507	1213
m		10,404	20,164	40,804	81,225	161,604
	10^{-2}	8	17	36	77	159
$\mathcal{H}$-Cholesky	10^{-4}	10	23	48	107	222
of $B^T B$	10^{-8}	14	30	67	148	319
$n+m$		91,206	178,086	362,406	724,203	1,444,806
PARDISO		183	389	859	1,910	4,100

In Table 3.1, we show the set-up times to compute $\mathcal{H}$-LU factorizations of A and $\mathcal{H}$-Cholesky factorizations of $B^T B$ for varying $\mathcal{H}$-accuracies $\delta_{\mathcal{H}}$. As $\delta_{\mathcal{H}} \to 0$, the factorizations become more accurate but also more expensive to compute. For a fixed accuracy $\delta_{\mathcal{H}}$, the set-up time is (almost) linear in the problem size for both factorizations. We use higher accuracies (i.e., smaller $\delta_{\mathcal{H}}$) for the $\mathcal{H}$-Cholesky factorization since this will result in significantly faster convergence in the subsequent iteration; see Table 3.4.

Similar to the set-up time, the storage requirements are (almost) optimal in the problem size and only increase moderately as we increase the $\mathcal{H}$-accuracy. In Table 3.2, we also provide a comparison with the storage required by the direct solver PARDISO 3.3 [20, 21] which is more than twice the storage required by the $\mathcal{H}$-LU and $\mathcal{H}$-Cholesky factorizations with $\mathcal{H}$-accuracies $\delta_{\mathcal{H}} = 10^{-3}$ and $\delta_{\mathcal{H}} = 10^{-8}$, resp.

In Table 3.3, we list the number of iteration steps and corresponding iteration times (in seconds) for various problem sizes $n+m$ and varying convection dominance as determined by the parameter ϵ in the Oseen problem (3.1). In the case of constant convection b_{xline}, the convergence rate decreases as the problem becomes more convection-dominated, i.e., as

ETNA
Kent State University
http://etna.math.kent.edu

TABLE 3.3
Dependence on meshsize h and convection dominance, table lists iteration steps and time (sec), $\delta_A = 10^{-3}$, $\delta_{B^TB} = 10^{-8}$

$\epsilon/n+m$	91,206	178,086	362,406	724,203	1,444,806
			b_{xline}		
1.0	12/4	12/7	15/20	16/45	19/114
10^{-1}	12/4	15/9	16/22	19/53	20/119
10^{-2}	8/2	9/6	12/17	13/36	17/101
10^{-3}	4/1	4/3	4/6	5/14	5/30
PARDISO	7s	19s	46s	138s	417s
			b_{recirc}		
1.0	10/3	12/7	15/21	16/44	19/114
10^{-1}	14/4	15/9	17/23	18/50	24/142
10^{-2}	13/4	20/12	19/26	21/58	24/142
10^{-3}	26/8	26/16	26/35	29/80	31/183
PARDISO	6s	17s	44s	123s	309s

ϵ decreases. For recirculating convection b_{recirc}, however, the number of required steps increases as ϵ decreases. For a fixed ϵ, the number of required steps increases only slightly as the problem size increases, both in the case of constant and recirculating convection. We also provide a comparison with the solution time required by the direct solver PARDISO 3.3 for the convection-dominated problems using $\epsilon = 10^{-3}$. For a fair comparison, one needs to add the respective set-up time for the approximate inverse preconditioner as reported in Table 3.1 to the solution time. For instance, for problem size $n + m = 1,444,806$, $\epsilon = 10^{-3}$ and $b = b_{xline}$, PARDISO requires $417s$ compared to $36s + 53s + 30s = 120s$ ($\mathcal{H}$-LU of A + $\mathcal{H}$-Cholesky of B^TB + iterative solver) for the approximate inverse preconditioner. For this example, PARDISO required $4100MB$ of memory compared to $1,213MB + 319MB = 1,532MB$ for the $\mathcal{H}$-LU and $\mathcal{H}$-Cholesky factors required in the approximate inverse preconditioner; see Table 3.2. For the same problem size and recirculating convection b_{recirc}, PARDISO requires $309s$ compared to $36s + 53s + 183s = 272s$ for set-up and iterative solution with the approximate inverse preconditioner.

In Table 3.4, we list the number of iteration steps and corresponding iteration times (in seconds) for various problem sizes $n + m$ and varying accuracies for the approximations to A^{-1} and $(B^TB)^{-1}$. The parameters δ_A and δ_{B^TB} denote the adaptive accuracy chosen in the $\mathcal{H}$-LU and $\mathcal{H}$-Cholesky factorizations of A and B^TB, resp. The results show that the number of iteration steps increases significantly when a less accurate approximation to B^TB is used, even leading to divergence in the case of recirculating convection b_{recirc}. The accuracy of the approximation to A^{-1} shows the expected behaviour that fewer iteration steps are required as $\delta_A \to 0$. In view of the only moderate increase in the set-up times (see Table 3.1) as $\delta_A \to 0$, we obtain the fastest overall solution time (set-up time and iteration time combined) for the recirculating convection by choosing $\delta_{B^TB} = 10^{-8}$ and $\delta_A = 10^{-3}$. For the largest problem size $n+m = 1,444,806$, we also list results for the BFBt-preconditioned BiCGStab method. The BFBt-preconditioned iteration converges even for less accurate approximations of $(B^TB)^{-1}$ where the approximate inverse preconditioner fails. Dividing the iteration time by the number of steps, one sees that a single iteration step of the BFBt-method is faster than a step of the approximate inverse method. For highly accurate approximations to the subproblems, both methods show the expected very similar convergence behaviour.

ETNA
Kent State University
http://etna.math.kent.edu

TABLE 3.4
Dependence on $\mathcal{H}$-accuracies δ_A and δ_{B^TB}, table lists iteration steps and time (sec), $\epsilon = 10^{-3}$

n+m	$\delta_A = 10^{-3}$		$\delta_{B^TB} = 10^{-8}$		
	$\delta_{B^TB} = 10^{-2}$	$\delta_{B^TB} = 10^{-4}$	$\delta_A = 10^{-1}$	$\delta_A = 10^{-2}$	$\delta_A = 10^{-3}$
	b_{xline}				
91,206	14/4	5/1	5/2	4/1	4/1
178,086	21/11	5/3	5/3	4/2	4/3
362,406	30/35	7/9	6/8	4/6	4/6
724,203	52/123	7/18	6/16	5/14	5/14
1,444,806	63/312	10/54	7/41	5/30	5/30
BFBt	58/226	12/49	20/84	7/30	6/27
	b_{recirc}				
91,206	118/31	73/21	55/16	28/8	26/8
178,086	div	88/51	48/29	31/19	26/16
362,406	div	128/161	39/51	42/56	26/35
724,203	div	123/309	27/72	39/106	29/80
1,444,806	div	div	51/291	49/285	31/183
BFBt	197/756	145/584	br	72/306	43/186

3.2. Three-dimensional Oseen problem. We repeated the same set of experiments for the Oseen problem in three spatial dimensions. In Table 3.5, we record the set-up times for the approximate inverse preconditioner which are considerably larger compared to the two-dimensional case. This is mainly due to the larger number of non-zero entries per matrix row typical for three-dimensional problems. However, for a fixed $\mathcal{H}$-accuracy, the work complexity is still almost optimal with respect to the increase in problem size.

TABLE 3.5
Set-up time (in seconds), $\epsilon = 10^{-2}$, b_{xline}

$n/3$	$\delta_{\mathcal{H}}$	6,859	15,625	29,791	59,319	132,651	250,047
	10^{-1}	1	3	8	33	62	139
$\mathcal{H}$-LU of A	10^{-2}	2	6	15	63	115	264
	10^{-3}	2	9	22	94	190	441
m		1,331	2,744	4,913	9,261	19,683	35,937
	10^{-2}	1	2	4	10	29	58
$\mathcal{H}$-Cholesky	10^{-4}	1	3	8	20	60	131
of B^TB	10^{-8}	1	5	12	37	134	331

In Table 3.6, we record the storage requirements. Here, we see the same behaviour as observed for the work complexity, i.e., almost linear complexity with respect to the problem size. The direct solver PARDISO requires up to ten times the storage of the approximate $\mathcal{H}$-LU and $\mathcal{H}$-Cholesky factors and runs out of memory ("o.o.m.") for $n + m = 786,078$ unknowns.

In Table 3.7, we show iteration steps and times for varying convection dominance. As for the two-dimensional case, fewer steps are required as the constant convection b_{xline} becomes more dominant. For non-constant convection b_{recirc}, the number of steps only increases when ϵ decreases from 10^{-2} to 10^{-3}. A comparison with the direct solver PARDISO is provided for the convection-dominated case $\epsilon = 10^{-3}$. Even after including the set-up time as reported in Table 3.5, the time required for the iterative solution is significantly smaller than the time

TABLE 3.6
Storage (in MB), $\epsilon = 10^{-2}$, b_{xline}

$n/3$	$\delta_{\mathcal{H}}$	6,859	15,625	29,791	59,319	132,651	250,047
	10^{-1}	18	45	103	232	549	1122
$\mathcal{H}$-LU of A	10^{-2}	22	57	130	297	708	1437
	10^{-3}	25	64	147	342	830	1700
m		1,331	2,744	4,913	9,261	19,683	35,937
	10^{-2}	2	5	10	22	56	117
$\mathcal{H}$-Cholesky	10^{-4}	3	8	17	39	104	219
of B^TB	10^{-8}	4	11	25	65	186	409
$n+m$		21,908	49,618	94,286	187,218	417,636	786,078
PARDISO		150	493	1,292	3,344	10,583	o.o.m

for the direct solver.

TABLE 3.7
Dependence on meshsize h and convection dominance, table lists iteration steps and time (sec), $\delta_A = 10^{-3}$, $\delta_{B^TB} = 10^{-8}$

$\epsilon/n+m$	21,908	49,618	94,286	187,218	417,636	786,078
			b_{xline}			
1.0	10/1	11/3	12/8	15/23	18/73	22/190
10^{-1}	8/1	10/3	11/7	13/21	16/64	20/173
10^{-2}	8/1	9/3	10/7	11/17	12/49	13/112
10^{-3}	9/1	10/3	11/7	11/16	12/45	12/98
PARDISO	11s	64s	287s	1,311s	8,597s	o.o.m.
			b_{recirc}			
1.0	9/1	11/3	12/8	14/21	17/67	22/186
10^{-1}	9/1	12/4	12/8	14/21	17/67	21/178
10^{-2}	8/1	11/3	13/9	14/21	17/67	23/195
10^{-3}	11/1	14/4	19/12	20/30	24/94	33/278
PARDISO	12s	63s	259s	1,371s	8,769s	o.o.m.

Finally, in Table 3.8 we show the dependence of the number of required iteration steps on the accuracies chosen to solve the subproblems A^{-1} and $(B^TB)^{-1}, resp.$ The fastest solution is obtained when $\delta_{B^TB} = 10^{-8}$ and $\delta_A = 10^{-1}$ both for the constant and recirculating convection. The number of required iteration steps increases only moderately as the problem size increases. However, except for the case in which a breakdown occurs for the BFBt-preconditioner, the BFBt-preconditioner outperforms the approximate inverse preconditioner.

Since the use of $\mathcal{H}$-Cholesky and $\mathcal{H}$-LU factorizations is somewhat uncommon, we also performed tests for the three-dimensional Oseen problem using a MATLAB implementation of the approximate inverse preconditioner and the MATLAB routines “luinc” (incomplete LU factorization), “chol” (exact Cholesky factorization), “cholinc” (incomplete Cholesky factorization) and “bicgstab”. In Table 3.9 we record the set-up times for the various factorizations with varying accuracies. Apparently, the exact Cholesky factorization has been optimized to an extent where it outperforms the incomplete Cholesky factorization for the given problem sizes (with respect to time, but not with respect to storage).

Tables 3.10 and 3.11 show iteration steps and times for the MATLAB implementaion solving the three-dimensional Oseen problem with $\epsilon = 10^{-2}$ and convection directions b_{xline}

TABLE 3.8
Dependence on $\mathcal{H}$-accuracies δ_A and δ_{B^TB}, table lists iteration steps and time (sec), $\epsilon = 10^{-2}$

n+m	$\delta_A = 10^{-3}$		$\delta_{B^TB} = 10^{-8}$		
	$\delta_{B^TB} = 10^{-2}$	$\delta_{B^TB} = 10^{-4}$	$\delta_A = 0.5$	$\delta_A = 10^{-1}$	$\delta_A = 10^{-2}$
	b_{xline}				
21,908	14/1	8/1	9/1	8/1	8/1
49,618	15/4	9/2	11/2	9/2	9/3
94,286	17/9	10/6	12/6	9/5	9/6
187,218	20/26	12/16	14/15	11/15	11/16
417,636	24/74	15/51	17/45	12/39	11/41
786,078	29/185	17/118	23/126	14/93	13/100
BFBt	31/176	20/118	27/105	14/71	15/90
	b_{recirc}				
21,908	14/1	8/1	12/1	8/1	8/1
49,618	17/5	11/3	14/3	10/3	11/3
94,286	23/13	13/8	19/8	12/7	12/7
187,218	28/34	16/21	23/23	13/16	14/19
417,636	38/115	26/88	29/75	18/57	18/65
786,078	50/309	40/271	38/202	25/161	22/165
BFBt	39/214	33/192	br	20/97	21/120

TABLE 3.9
MATLAB: Set-up costs

n+m	LUINC(A,δ_A)			CHOLINC(B^TB, δ_{B^TB})	
	$\delta_A = 10^{-1}$	$\delta_A = 10^{-2}$	$\delta_A = 10^{-3}$	exact	$\delta_{B^TB} = 10^{-4}$
49,618	0.04	0.74	1.62	0.10	1.12
94,286	0.09	2.13	4.75	0.27	3.35
187,218	0.29	6.57	15.2	1.02	11.43
417,636	1.16	25.7	56.6	6.05	52.9

and b_{recirc}, resp. Table 3.10 shows the results for the approximate inverse preconditioner whereas Table 3.11 shows the results for the BFBt preconditioner. For less accurate settings of inner solvers, the approximate inverse preconditioner outperforms the BFBt preconditioner. As the accuracies increase, the number of iteration steps become comparable and iteration time is faster for the BFBt preconditioner since each step is cheaper. However, the gain in iteration time comes at the expense of an increase in set-up time for better accuracies of inner solvers. Overall, the approximate inverse preconditioner appears to be less sensitive with respect to the accuracies of inner solvers than the BFBt preconditioner.

REFERENCES

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

[2] S. BÖRM AND L. GRASEDYCK, *HLIB version 1.3*. Available at www.hlib.org.

[3] S. BÖRM, L. GRASEDYCK, AND W. HACKBUSCH, *Hierarchical matrices*, Lecture Notes No. 21, Max-Planck-Institute for Mathematics in the Sciences, Leipzig, Germany, 2003. Available online at www.mis.mpg.de/preprints/ln/, revised version June 2006.

[4] A. C. DE NIET AND F. W. WUBS, *Two preconditioners for saddle point problems in fluid flows*, Internat. J. Numer. Methods Fluids, 54 (2007), pp. 355–377.

TABLE 3.10
MATLAB Approx. Inverse: Dependence on accuracies δ_A and δ_{B^TB}, table lists iteration steps and time (sec)

n+m	δ_{B^TB} exact			$\delta_{B^TB} = 10^{-4}$		
	$\delta_A = 10^{-1}$	$\delta_A = 10^{-2}$	$\delta_A = 10^{-3}$	$\delta_A = 10^{-1}$	$\delta_A = 10^{-2}$	$\delta_A = 10^{-3}$
	b_{xline}					
49,618	15/11	7/5	8/6	20/11	11/6	12/8
94,286	19/31	8/14	8/15	27/34	14/18	15/22
187,218	18/81	10/46	9/47	28/76	18/51	16/51
417,636	24/399	12/197	10/182	46/287	25/162	25/183
	b_{recirc}					
49,618	16/11	9/6	8/6	18/10	16/10	13/9
94,286	18/30	10/18	9/17	26/33	19/25	17/25
187,218	21/95	12/57	10/52	35/94	25/68	28/88
417,636	26/430	16/271	12/215	56/351	36/233	28/205

TABLE 3.11
MATLAB BFBt: Dependence on accuracies δ_A and δ_{B^TB}, table lists iteration steps and time (sec), $\epsilon = 10^{-2}$

n+m	δ_{B^TB} exact			$\delta_{B^TB} = 10^{-4}$		
	$\delta_A = 10^{-1}$	$\delta_A = 10^{-2}$	$\delta_A = 10^{-3}$	$\delta_A = 10^{-1}$	$\delta_A = 10^{-2}$	$\delta_A = 10^{-3}$
	b_{xline}					
49,618	55/25	12/6	7/4	75/29	17/7	10/5
94,286	73/80	14/16	7/10	div	22/20	12/13
187,218	60/178	15/47	8/27	div	31/61	17/39
417,636	div	20/213	9/101	div	49/224	25/133
	b_{recirc}					
49,618	76/35	21/10	8/5	81/32	29/12	12/6
94,286	div	27/31	9/12	div	52/49	16/18
187,218	92/265	38/116	11/39	div	80/157	25/58
417,636	div	39/412	13/151	div	div	44/233

[5] H. C. ELMAN, *Preconditioning of the steady-state Navier-Stokes equations with low viscosity*, SIAM J. Sci. Comp., 20 (1999), pp. 1299–1316.
[6] H. C. ELMAN, V. E. HOWLE, J. SHADID, R. SHUTTLERWORTH, AND R. TUMARINO, *Block preconditioners based on approximate commutators*, SIAM J. Sci. Comput., 27 (2005), pp. 1651–1668.
[7] H. C. ELMAN, D. LOGHIN, AND A. J. WATHEN, *Preconditioning techniques for Newton's method for the incompressible Navier-Stokes equations*, BIT, 43 (2003), pp. 961–974.
[8] H. G. GOLUB AND A. J. WATHEN, *An iteration for indefinite systems and its application to the Navier-Stokes equations*, SIAM J. Sci. Comput., 19 (1998), pp. 530–539.
[9] N. I. M. GOULD, M. E. HRIBAR, AND J. NOCEDAL, *On the solution of equality constrained quadratic programming problems arising in optimization*, SIAM J. Sci. Comput., 23 (2001), pp. 1376–1395.
[10] L. GRASEDYCK AND W. HACKBUSCH, *Construction and arithmetics of $\mathcal{H}$-matrices*, Computing, 70 (2003), pp. 295–334.
[11] J. C. HAWS AND C. D MEYER, *Preconditioning KKT systems*, Technical Report M&CT-Tech-01-021, The Boing Company, 2003.
[12] D. KAY, D. LOGHIN, AND A. WATHEN, *A preconditioner for the steady-state Navier-Stokes equations*, SIAM J. Sci. Comput., 24 (2002), pp. 237–256.
[13] C. KELLER, N. I. M. GOULD, AND A. J. WATHEN, *Constraint preconditioning for indefinite linear systems*, SIAM J. Matrix Anal. Appl., 21 (2000), pp. 1300–1317.
[14] S. LE BORNE, *Hierarchical matrix preconditioners for the Oseen equations*, Comput. Vis. Sci., 11 (2008), pp. 147–158.

[15] S. LE BORNE, *Preconditioned nullspace method for the two-dimensional Oseen problem*, SIAM J. Sci. Comput., 31 (2009), pp. 2492–2509.

[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. B. Widlund and D. E. Keyes, eds., Lect. Notes Comput. Sci. Eng., Vol. 55, Springer, New York, 2006, pp. 661–668.

[17] S. G. NASH AND A. SOFER, *Preconditioning reduced matrices*, SIAM J. Matrix Anal. Appl., 17 (1996), pp. 47–68.

[18] M. A. OLSHANSKII AND Y. V. VASSILEVSKI, *Pressure Schur complement preconditioners for the discrete Oseen problem*, SIAM J. Sci. Comput., 29 (2007), pp. 2686–2704.

[19] H. G. ROOS, M. STYNES, AND L. TOBISKA, *Numerical methods for singularly perturbed differential equations: convection diffusion and flow problems*, Computational Mathematics, Vol. 24, Springer, Berlin, 1996.

[20] 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.

[21] O. SCHENK AND K. GÄRTNER, *On fast factorization pivoting methods for sparse symmetric indefinite systems*, Electron. Trans. Numer. Anal., 23 (2006), pp. 158–179. http://etna.math.kent.edu/vol.23.2006/pp158-179.dir.