Numer. Math. Theor. Meth. Appl.
doi: 10.4208/nmtma.2010.33.1

Vol. **3**, No. 3, pp. 245-275
August 2010

REVIEW ARTICLE

Preconditioners for Incompressible Navier-Stokes Solvers[†]

A. Segal, M. ur Rehman and C. Vuik[*]

Delft University of Technology, Delft Institute of Applied Mathematics and J.M. Burgerscentrum, Mekelweg 4, 2628 CD, Delft, The Netherlands.

Received 30 September 2009; Accepted (in revised version) 27 January 2010

Available online 2 July 2010

Abstract. In this paper we give an overview of the present state of fast solvers for the solution of the incompressible Navier-Stokes equations discretized by the finite element method and linearized by Newton or Picard's method. It is shown that block preconditioners form an excellent approach for the solution, however if the grids are not to fine preconditioning with a Saddle point ILU matrix (SILU) may be an attractive alternative. The applicability of all methods to stabilized elements is investigated. In case of the stand-alone Stokes equations special preconditioners increase the efficiency considerably.

AMS subject classifications: 65F10, 65N30, 76D05

Key words: Navier-Stokes equations, finite element method, block preconditioners, SIMPLE-type schemes, iterative methods, incompressible fluids.

1. Introduction

The numerical solution of the incompressible Navier-Stokes equations has been a challenge for over 50 years. Was the attention in the early days focused on the discretization, nowadays efficient solution is a hot topic.

In this paper we deal with efficient solution of the stationary, laminar incompressible Navier-Stokes equations, discretized by the finite element method. Due to the absence of the pressure in the continuity equation, discretization of these equations requires special care. Finite element approximations can not be chosen at random, but the elements must satisfy the *Ladyshenskaya-Brezzi-Babuška* (LBB) condition in order to guarantee stability of the discretization [1, 2]. This condition usually requires an approximation of the velocity

[†]This paper is presented at the International Conference on Preconditioning Techniques for Scientific and Industrial Applications, August 24-26, 2009, Hong Kong.
[*]Corresponding author. *Email addresses:* `a.segal@tudelft.nl` (A. Segal), `mehfoozrehman@gmail.com` (M. ur Rehman), `c.vuik@tudelft.nl` (C. Vuik)

that is one degree higher than that of the pressure. Most elements satisfying the LBB condition have a quadratic velocity approximation [3].

The alternative is to stabilize the elements by adapting the continuity equation in some way, thus allowing for example equal order interpolation [4]. Although such an approximation makes the implementation more easy, in general the price to be paid is a less accurate pressure.

Solution of the non-linear Navier-Stokes equations requires a suitable linearization technique like Newton or Picard (successive substitution) [3]. The result is a system of linear equations of saddle-point type, containing a zero block on the main diagonal corresponding to the continuity equation. Direct solution of such a system requires a suitable renumbering technique to avoid zero pivots [5]. A possible way of avoiding this problem is to segregate pressure and velocity computation. An approach in this direction is the so-called penalty function approach [3], which has been successfully applied to medium-sized 2d problems. Due to the presence of the penalty parameter, the condition of the linear system is very high and iterative solution is impossible. The alternative is to use Uzawa-type iteration schemes, which unfortunately converge very slowly [6].

During the last decade various iterative solution techniques to solve saddle-point type equations, have been the subject of research. Some methods are focused on clever renumbering schemes in combination with a classical iterative approach, like for example the SILU scheme [5] and ILU schemes proposed by Wille et al. [7–9].

Other methods are based on segregation. The system of equations is split into a velocity and a pressure part. The complete system is solved by an iterative method, but the necessary preconditioner is based on the splitting. We distinguish between block preconditioners and preconditioners based on the classical SIMPLE method of Patankar [10]. A number of block-preconditioners have been devised, for example the Pressure-Convection Diffusion commutator (PCD) of Kay, Logan and Wathen [11, 12], the Least Squares Commutator (LSC) by Elman, Howle, Shadid, Shuttleworth and Tuminaro [13], the Augmented Lagrangian Approach (AL) of Benzi and Olshanskii [14], the Artificial Compressibility (AC) preconditioner [15] and the Grad-Div (GD) preconditioner [15]. For an overview of block preconditioners, we refer to [16–18].

SIMPLE-type preconditioners form a different class of preconditioners, although they can also be considered as a block preconditioner. Besides the standard SIMPLE preconditioner, also improvements like SIMPLER and variants have been developed.

In this paper we present a survey of the most popular solvers for the incompressible Navier-Stokes equations. In the case of the incompressible Stokes equations, we derive some special methods, that outperform the generally applicable solvers. New in this paper is that we investigate the possibility to extend all solvers to stabilized elements. Furthermore we discuss termination criteria and propose a method to improve these criteria. Solvers are compared on the basis of implementation issues, dependence on tuning parameters and numerical experiments.

The remaining part of this paper is as follows. In Section 2 the discretization of the incompressible Navier-Stokes by the Finite element discretization is considered. Section 3 deals with block preconditioners and in Section 4 SIMPLE and its variants are discussed. In

Section 5 we study the modified ILU preconditioner SILU, which is an alternative for block preconditioners. Section 6 treats special methods for the Stokes equations and Section 7 handles the important issue of termination criteria. In Section 8 some of these preconditioners are compared based on the solution of some standard benchmark problems. Finally in Section 9 we give our conclusions.

2. The incompressible Navier-Stokes equations

We consider the steady state Navier-Stokes equations for the incompressible flow of a Newtonian, viscous fluid, with constant viscosity:

$$-\nu\nabla^2\mathbf{u}+\mathbf{u}\cdot\nabla\mathbf{u}+\nabla p=\mathbf{f}\quad \text{in } \Omega, \tag{2.1}$$

$$\nabla\cdot\mathbf{u}=0\quad \text{in } \Omega, \tag{2.2}$$

where Ω is a 2 or 3 dimensional bounded domain with a piecewise smooth boundary $\partial\Omega$, $\mathbf{u}$ is the fluid velocity, p is the pressure field, $\nu > 0$ is the kinematic viscosity coefficient (inversely proportional to the Reynolds number Re), ∇ the gradient and $\nabla\cdot$ is the divergence operator. In Section 6 we shall also consider the case of variable viscosity.

Eq. (2.1) represents conservation of momentum, while Eq. (2.2) represents mass conservation (incompressibility condition). The boundary value problem we consider, is the system (2.1) and (2.2) posed on Ω, together with boundary conditions on $\partial\Omega=\partial\Omega_E\cup\partial\Omega_N$ given by

$$\mathbf{u}=\mathbf{w}\quad \text{on } \partial\Omega_E,\qquad \nu\frac{\partial\mathbf{u}}{\partial\mathbf{n}}-\mathbf{n}p=0\quad \text{on } \partial\Omega_N,$$

or a combination of Dirichlet and Neumann boundary conditions.

The discretization of the Navier-Stokes equations is done through the finite element method. The weak formulation of the Navier-Stokes equations is given by: find $\mathbf{u}\in\mathbf{H}^1_E$ and $p\in L_2(\Omega)$ such that

$$\begin{aligned}\nu\int_\Omega \nabla\mathbf{u}:\nabla\mathbf{v}d\Omega+\int_\Omega(\mathbf{u}\cdot\nabla\mathbf{u})\cdot\mathbf{v}d\Omega+\int_\Omega p\cdot\nabla\mathbf{v}d\Omega\\ =\int_\Omega \mathbf{f}\cdot\mathbf{v}d\Omega,\quad \forall\mathbf{v}\in\mathbf{H}^1_{E_0},\end{aligned} \tag{2.3}$$

$$\int_\Omega(\nabla\cdot\mathbf{u})qd\Omega=0,\quad \forall q\in L_2(\Omega), \tag{2.4}$$

where $\mathbf{H}^1_E$ is the Sobolev space of functions satisfying the essential boundary conditions, $\mathbf{H}^1_{E_0}$ the Sobolev space of functions that satisfy homogeneous essential boundary conditions, and : denotes the dyadic product.

For a discrete weak formulation we define finite dimensional subspaces $\mathbf{X}^h_0\subset\mathbf{H}^1_{E_0}$, $M^h\subset L_2(\Omega)$, and $\mathbf{X}^h_E\subset\mathbf{H}^1_E$. The discrete version of (2.3) and (2.4) is: find $u_h\in\mathbf{X}^h_E$ and

$p_h \in M^h$ such that

$$\begin{aligned}\nu \int_\Omega \nabla \mathbf{u_h} : \nabla \mathbf{v_h} d\Omega + \int_\Omega (\mathbf{u_h} \cdot \nabla \mathbf{u_h}) \cdot \mathbf{v_h} d\Omega - \int_\Omega p_h (\nabla \cdot \mathbf{v_h}) d\Omega \\ = \int_\Omega \mathbf{f} \cdot \mathbf{v_h} d\Omega, \quad \forall \mathbf{v_h} \in \mathbf{X}_0^h,\end{aligned} \tag{2.5}$$

$$\int_\Omega (\nabla \cdot \mathbf{u_h}) q_h d\Omega = 0, \quad \forall q_h \in M^h. \tag{2.6}$$

Formally, the system of non-linear equations can be written as

$$A\mathbf{u} + N(\mathbf{u}) + B^T p = \mathbf{f}, \tag{2.7}$$

$$B\mathbf{u} = \mathbf{g}, \tag{2.8}$$

where $A\mathbf{u}$ is the discretization of the viscous term, $N(\mathbf{u})$ the discretization of the nonlinear convective term, $B\mathbf{u}$ denotes the discretization of the negative divergence of $\mathbf{u}$, and $B^T p$ is the discretization of the gradient of p. The right-hand side vectors $\mathbf{f}$ and $\mathbf{g}$ contain all the contributions of the source term, the boundary integral, as well as the contribution of the prescribed boundary conditions.

Systems of the form (2.7)-(2.8) are called saddle-point problems. Due to the absence of the pressure term in Eq. (2.8), the system of equations may be under-determined for an arbitrary combination of pressure and velocity unknowns. In order to guarantee a unique solution of Eqs. (2.7)-(2.8), the finite element discretization should satisfy the *Ladyshenskaya-Brezzi-Babuška* (LBB) condition formally defined by

$$\inf_{q_h \in M^h} \sup_{v_h \in X_0^h} \frac{(\nabla \cdot v_h, q_h)}{\|v_h\|_{X_0^h} \|q_h\|_{M^h}} \geq \gamma > 0. \tag{2.9}$$

In practice Eq. (2.9) is hard to verify, but Fortin [2] has given a simple method to check the LBB condition. The practical consequence of this condition is that most stable elements have a quadratic velocity and a linear pressure approximation.

Finite elements for the incompressible Navier-Stokes equations are distinguished in a group with continuous pressure, Taylor-Hood elements [19], and the set of discontinuous pressure, Crouzeix Raviart [20] elements.

If a finite element does not satisfy the LBB condition, solution of (2.7), (2.8) is only possible by applying stabilization techniques [18]. A typical example of a stabilized element is the so-called mini-element consisting of a linear pressure approximation and a velocity that is approximated by a linear polynomial extended with a bubble function [21]. The internal point in this Taylor-Hood element is usually eliminated, so that this element has the appearance of an equal order element.

The resulting non-linear equations are linearized as part of an iteration process (2.7), (2.8). Common linearization methods are Picard and Newton or variants of these. After linearization the system can be written as

$$\begin{bmatrix} F & B^T \\ B & C \end{bmatrix} \begin{bmatrix} u \\ p \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix}, \tag{2.10}$$

where F contains the contributions from the viscous part as well as the linearized convection term. The system matrix of (2.10) is indefinite and non-symmetric. The matrix C corresponds to the stabilization term and is zero in case of elements satisfying the LBB condition. The eigenvalues of F have a positive real part, but the eigenvalues of C are either zero or have a negative real part. Therefore classical iterative methods usually have convergence problems for those systems.

In the remainder of this paper we shall mostly be concerned with stable elements ($C = 0$). Elements denoted by $Q2 - Q1$ are Taylor-Hood elements with quadratic velocity and linear pressure approximation for either quadrilaterals (2D) or hexahedrons (3D). $Q2 - P1$ elements belong to the Crouzeix Raviart family also with quadratic velocity and linear pressure.

3. Block preconditioners

Block preconditioners are based on an LDU decomposition of the coefficient matrix (2.10):

$$\mathscr{A} = \mathscr{L}_b \mathscr{D}_b \mathscr{U}_b = \begin{bmatrix} F & B^T \\ B & C \end{bmatrix} = \begin{bmatrix} I & 0 \\ BF^{-1} & I \end{bmatrix} \begin{bmatrix} F & 0 \\ 0 & S \end{bmatrix} \begin{bmatrix} I & F^{-1}B^T \\ 0 & I \end{bmatrix}, \tag{3.1}$$

where $S = C - BF^{-1}B^T$ is the Schur complement matrix. Most preconditioners are based on a combination of these blocks and a suitable approximation of the Schur complement matrix. In this section we consider preconditioners based on $\mathscr{D}_b \mathscr{U}_b$ factors known as block triangular preconditioners:

$$P_t = \begin{bmatrix} F & B^T \\ 0 & S \end{bmatrix}. \tag{3.2}$$

It is an easy exercise to show that the eigenvalues of the system (2.10), premultiplied by the inverse of the preconditioner (3.2) are all equal to 1 and as a consequence GMRES converges in two iterations provided exact arithmetic is used [22]. Of course computing F^{-1} and S^{-1} is impractical due to storage and CPU requirements. Therefore all variants of block triangular preconditioners contain a cheap approximation of S. The system $Fu = f$ is solved approximately, usually by a small number of iterations with an iterative method. Application of the preconditioner (3.2) involves solving the system $P_t z = r$, where $z = [z_1; z_2]$ and $r = [r_1; r_2]$, implies the steps in Algorithm 3.1.

The Schur complement S is not formed, but approximated by a simple matrix. How this approximation is done defines the various block preconditioners. Below we consider some of the more popular methods.

Algorithm 3.1: Perform $P_t z = r$

```
1. Solve $Sz_2 = r_2$
2. update $r_1 = r_1 - B^T z_2$
3. Solve $Fz_1 = r_1$
```

3.1. Pressure-convection diffusion preconditioner

The convection diffusion operator [11] defined on the velocity space can be written as

$$\mathscr{L} = -\nu \nabla^2 + \mathbf{w_h} \cdot \nabla, \tag{3.3}$$

where $\mathbf{w_h}$ is the approximation to the discrete velocity, computed in the most recent Picard iteration. Suppose, that the commutator of this operator, ε, multiplied by the gradient operator, on the velocity space, and the gradient operator, acting on the convection diffusion operator in the pressure space ($\mathscr{L}_p$), is small.

$$\varepsilon = \mathscr{L}\nabla - \nabla \mathscr{L}_p. \tag{3.4}$$

Then the discrete commutator in terms of finite element matrices defined by

$$\varepsilon_h = (Q_v^{-1}F)(Q_v^{-1}B^T) - (Q_v^{-1}B^T)(Q_p^{-1}F_p), \tag{3.5}$$

might also be small. F_p is a discrete convection-diffusion operator on pressure space. Q_v denotes the velocity mass matrix and Q_p the pressure mass matrix. The multiplication by Q_v^{-1} and Q_p^{-1}, transforms quantities from integrated values to nodal values. Premultiplication of (3.5) by $BF^{-1}Q_v$, post-multiplication by $F_p^{-1}Q_p$ and assuming that the commutator is small, leads to the Schur approximation

$$BF^{-1}B^T \approx BQ_v^{-1}B^T F_p^{-1} Q_p. \tag{3.6}$$

The expensive part $BQ_v^{-1}B^T$ in (3.6) is replaced by its spectral equivalent matrix A_p known as the pressure Laplacian matrix, so

$$S = C - BF^{-1}B^T \approx C - A_p F_p^{-1} Q_p. \tag{3.7}$$

The preconditioner (3.2), (3.7) is known as the Pressure Convection Diffusion (PCD) preconditioner. The convergence of this preconditioner combined with a Krylov method is very good for enclosed flows if the equations are linearized by the Picard method. The preconditioner leads to many iterations for inflow/outflow problems. The reason might be that the approximation of $BQ^{-1}B^T$ with A_p is only well-defined for enclosed flow problems [18]. Boundary conditions are treated such that A_p and F_p are computed with Neumann boundary conditions for an enclosed flow problem. However, in inflow/outflow problems, rows and columns of A_p and F_p corresponding to pressure nodes on an inflow boundary are treated as though they are associated with Dirichlet boundary conditions [18]. One of the main disadvantages of PCD is the necessity to construct the matrices A_p and F_p and the definition of boundary conditions for the pressure matrix. This makes implementation in standard finite element codes less obvious.

3.2. Least Squares Commutator preconditioner

A method based on the same principle as the PCD preconditioner is the Least Squares Commutator (LSC) preconditioner of Elman et al. [13]. This method is based on the matrices in (2.10) as well as the simple velocity mass matrix and is therefore more suitable for standard FEM codes. Unfortunately this is only true for stable elements. The extension to stabilized elements is far from trivial [23], since it requires the estimation of two tuning parameters. So we limit ourselves to the stable case only.

The idea is to approximate the matrix operator, F_p, in (3.6), such that the commutator (3.4) becomes small. This is done by solving a least squares problem. For the jth column of matrix F_p, the least squares problem has the form:

$$\min \|[Q_v^{-1}FQ_v^{-1}B^T]_j - Q_v^{-1}B^TQ_p^{-1}[F_p]_j\|_{Q_v}, \tag{3.8}$$

where $\|.\|_{Q_v}$ is the $\sqrt{\underline{x}^TQ_v\underline{x}}$ norm. The normal equations associated with this problem are:

$$Q_p^{-1}BQ_v^{-1}B^TQ_p^{-1}[F_p]_j = [Q_p^{-1}BQ_v^{-1}FQ_v^{-1}B^T]_j,$$

which leads to the following definition of F_p:

$$F_p = Q_p(BQ_v^{-1}B^T)^{-1}(BQ_v^{-1}FQ_v^{-1}B^T).$$

Substituting this expression into (3.6) gives an approximation to the Schur complement matrix:

$$BF^{-1}B^T \approx (BQ_v^{-1}B^T)(BQ_v^{-1}FQ_v^{-1}B^T)^{-1}(BQ_v^{-1}B^T). \tag{3.9}$$

To avoid the dense inverse Q_v^{-1}, the mass matrix is approximated by its diagonal. This matrix acts as a scaling matrix, which improves the convergence rate of the method.

The algorithm for the LSC preconditioner reads:

Algorithm 3.2: LSC preconditioner

```
1. Solve $S_f z_2 = r_2$, where $S_f = B\hat{Q}_v^{-1}B^T$
2. update $r_2 = B\hat{Q}_v^{-1}F\hat{Q}_v^{-1}B^T z_2$
3. Solve $S_f z_2 = -r_2$
4. update $r_1 = r_1 - B^T z_2$
5. Solve $Fz_1 = r_1$
```

The above Algorithm 3.2 involves two Poisson-type solves for the pressure subsystem and one velocity solve. The LSC Preconditioner is build from readily available matrices and no extra boundary conditions are required.

3.3. Augmented Lagrangian approach (AL)

A complete different approach has been published by Benzi and Olshanskii [14]. This method is developed for stable elements. Recently Benzi et al. [24] have extended their method to stabilized elements. Necessary for their method is to augment the velocity matrix in the original equation by a penalty-like term $\gamma B^T W^{-1} B$ with γ relatively small and W a scaling matrix, usually the diagonal of the pressure mass matrix.

The system of equations (2.10) is replaced by

$$\begin{bmatrix} F_\gamma & B_\gamma^T \\ B_\gamma & C_\gamma \end{bmatrix} \begin{bmatrix} u \\ p \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix}, \tag{3.10}$$

with $F_\gamma = F + \gamma B^T W^{-1} B$, $B_\gamma^T = B^T + \gamma B^T W^{-1} C$, $C_\gamma = C - \gamma C W^{-1} C$. Note that Equations (2.10) and (3.10) have the same solution, due to the continuity equation (2.8).

Eq. (3.10) is preconditioned with the matrix:

$$P_{AL} = \begin{bmatrix} F_\gamma & B \\ 0 & \hat{S} \end{bmatrix}, \tag{3.11}$$

where the inverse of the Schur complement is approximated by

$$\hat{S}^{-1} = -(\nu \hat{Q}_p^{-1} + \gamma W^{-1}). \tag{3.12}$$

$\hat{Q}_p$ denotes the approximate pressure mass matrix, ν is the viscosity. Usually, W and $\hat{Q}_p$ are the same.

A good choice of the parameter γ is essential. A large value reduces the number of outer iterations, but makes the solution of the inner equations expensive. In the limit for γ going to infinity, only one outer iteration is necessary and the method reduces to the penalty function formulation. If γ is too small the number of outer iterations increases. In the stable case $\gamma = 1$ is a practical choice, however, to get an optimal value, γ depends on the viscosity. Tuning this parameter may be necessary.

Benzi [14] has used multigrid (MG) with a new smoothing technique to solve the adapted velocity subsystem. In [24] the extension to AMG is discussed.

Although convergence of AL is independent of the Reynolds number and mesh size it has a number of drawbacks. First of all the matrix F_γ contains cross derivatives and is therefore twice (2D) or three (3D) times the size of the original matrix in case of Picard iteration and constant viscosity if the reduced form of the Navier-Stokes equations is used. Since most FEM packages use the non-reduced form, suitable for general boundary conditions and non-constant viscosity this is not a real problem. However, if Taylor-Hood elements are used instead of Crouzeix-Raviart elements, the size of F_γ is much larger than that of F even in the non-reduced case. The solution to this problem is not to build the matrix, but to perform matrix-vector multiplication in the solver using the submatrices. Of course this makes the multiplication more expensive.

Another problem is that extension of the method for non-constant viscosity is not simple at all. Numerical experiments performed by us, have shown that the convergence speed drastically decreases in case of strong varying viscosity. Besides that, fast convergence can only be achieved in combination with MG. If a Krylov subspace solver is used for the subsystems, the convergence rate becomes much smaller.

4. Simple-type preconditioners

In this section, we discuss block preconditioners that are based on the SIMPLE method formulation. SIMPLE (Semi-Implicit Pressure Linked Equation) has been introduced by Patankar as an iterative method to solve the finite volume discretized incompressible Navier-Stokes equations, using a staggered grid arrangement of the unknowns [10]. The staggering assures a stable discretization, hence $C = 0$. The scheme belongs to the class of classical iterative methods. Its convergence depends on relaxation parameters for the velocity and pressure, but is usually very slow. Still the scheme is very popular in the CFD community and has been used in many commercial packages like for example FLUENT[§].

A much faster convergence can be achieved if the SIMPLE method is used to accelerate a Krylov method. Variants of SIMPLE (SIMPLER, SIMPLEC) are also used as preconditioner to solve the incompressible Navier-Stokes problem.

In this section we shall discuss both SIMPLE and SIMPLER as preconditioners and we present a new variant, MSIMPLER (Modified SIMPLER), that improves the convergence considerably.

4.1. SIMPLE preconditioner

Originally SIMPLE has been developed for finite volume and finite difference discretizations [10,25]. The algorithm is based on the following steps. First the pressure is assumed to be known from the prior iteration. Then the velocity is solved from the momentum equations. The newly obtained velocities do not satisfy the continuity equation since the pressure is only a guess. In the next substeps the velocities and pressures are corrected in order to satisfy the discrete continuity equation. The Patankar formulation for FVM can be written in the form of a distributive iterative method (block matrices) see Wesseling [25].

In this section we apply SIMPLE-type preconditioners for the (Navier-) Stokes equations discretized by the finite element method.

Combination of the matrices $\mathscr{L}_b$ and $\mathscr{D}_b$ in Eq. (3.1) gives a new matrix, L_{bt}, defined by

$$\mathscr{L}_{bt} = \mathscr{L}_b \mathscr{D}_b = \begin{bmatrix} F & 0 \\ B & S \end{bmatrix}. \tag{4.1}$$

Our preconditioner will be based on the splitting $A = L_{bt} U_b$. An approximation of this matrix will be used as preconditioner.

If we solve $\mathscr{L}_{bt} z = r$; $z = [u; p]$; $r = [r_u; r_p]$, with this splitting we get Algorithm 4.1.

Algorithm 4.1: Basic SIMPLE algorithm

```
1. Solve $Fu^* = r_u$.
2. Solve $S\delta p = r_p - Bu^*$.
3. update $u = u^* - F^{-1}B^T \delta p$.
4. update $p = \delta p$.
```

§http://www.fluent.com/

To get the actual algorithm, F in the update step is approximated by the matrix D, which is the diagonal of F and the matrix S by $\hat{S} = C - BD^{-1}B^T$. Vuik et al. [26], used SIMPLE and its variants as a preconditioner to solve the incompressible Navier-Stokes problem. One iteration of the SIMPLE algorithm is used as a preconditioner. The preconditioner consists of one velocity solve and one pressure solve. Since the systems of equations in Algorithm 4.1 are solved to a certain accuracy, the preconditioner can not be considered constant in subsequent iterations. For that reason we use GCR, which allows variable preconditioners, as outer iteration.

It can be proven that the SIMPLE preconditioner improves the overall spectrum of the preconditioned system. Some of the eigenvalues are clustered around 1. The other ones depend on the approximation of the Schur complement matrix. More details on eigenvalue analysis are given in [27].

4.2. SIMPLER

A variant of SIMPLE, known as SIMPLER, is supposed to give Reynolds independent convergence. Algorithm 4.1 is extended by a first estimate of the pressure p^* as solution of the subsystem:

$$\hat{S}p^* = r_p - BD^{-1}r_u. \tag{4.2}$$

The right-hand side in the first step in Algorithm 4.1 is extended with $-B^T p^*$.

Consistency requires that the right-hand side in the next step is updated with $-Cp^*$.

The complete SIMPLER algorithm is given in Algorithm 4.2.

Algorithm 4.2: SIMPLER algorithm

```
1. **Solve** $\hat{S}p^* = r_p - BD^{-1}r_u$.
2. **Solve** $Fu^* = r_u - B^T p^*$.
3. **Solve** $\hat{S}\delta p = r_p - Bu^* - Cp^*$.
4. **update** $u = u^* - F^{-1}B^T\delta p$.
5. **update** $p = p^* + \delta p$.
```

This SIMPLER algorithm proposed by Patankar consists of two pressure solves and one velocity solve. Unfortunately if SIMPLER preconditioned GCR is used for finite element discretizations, the convergence may be poor or even divergence may occur, especially in case of low accuracy for the inner systems and in case of fine grids.

4.3. MSIMPLER

An improvement of the SIMPLER method is inspired by work of Elman et al. [13,18]. Elman et al. discussed relations between SIMPLE and commutator preconditioners. The more general form of (3.9) is given by:

$$(BF^{-1}B^T)^{-1} \approx F_p(BM_1^{-1}B^T)^{-1}, \tag{4.3}$$

with

$$F_p = (BM_2^{-1}B^T)^{-1}(BM_2^{-1}FM_1^{-1}B^T),$$

where M_1 and M_2 are scaling matrices. Consider a new block factorization preconditioner in which the Schur complement is based on a commutator approximation but built on SIMPLE's approximate block factorization written as:

$$P = \mathscr{L}_{bt}U_b \begin{bmatrix} I & 0 \\ 0 & F_p^{-1} \end{bmatrix}. \tag{4.4}$$

When $\hat{S} = C - BD^{-1}B^T$, $M_u = D$ and F_p is the identity matrix, then the preconditioner formulation (4.4) corresponds to SIMPLE. The formulation given in (4.4) is equivalent to the SIMPLE algorithm if the subsystem for the pressure part in step 2 in the SIMPLE algorithm is solved with the approximation given in (4.3)

$$\hat{S}\delta p = r_p - Bu^*, \quad \text{where } \hat{S} = C - (BM_1^{-1}B^T)F_p^{-1}.$$

When FD^{-1} is close to identity, F_p will also be close to identity. This is true in a time dependent problem with small time steps where the diagonal of F has larger entries than the off-diagonal entries [13].

Here we utilize the observation of Elman regarding the time dependent problem. We know that in time dependent problems,

$$F_t = \frac{1}{\Delta t}Q_u + F, \tag{4.5}$$

where F_t represents the velocity matrix for the time dependent problem and Δt represents the time step. For small time step $F_t \approx \frac{1}{\Delta t}Q_u$. This kind of approximation has been used in fractional step methods for solving the unsteady Navier-Stokes problem [28–30]. We use this idea in solving the steady Navier-Stokes problem. Therefore, we choose $M_1 = M_2 = \hat{Q}_v$ in (4.3) resulting in:

$$F_p = (B\hat{Q}_v^{-1}B^T)^{-1}(B\hat{Q}_v^{-1}F\hat{Q}_v^{-1}B^T).$$

If we assume that the factor $F\hat{Q}_v^{-1}$ in F_p is close to identity, then

$$F_p = (B\hat{Q}_v^{-1}B^T)^{-1}(B\hat{Q}_v^{-1}B^T) \approx I,$$

and the approximation (4.3) becomes

$$BF^{-1}B^T \approx (B\hat{Q}_v^{-1}B^T). \tag{4.6}$$

Based on this result we replace D^{-1} in the SIMPLER algorithm by $\hat{Q}_v^{-1}$. We refer to this method as MSIMPLER (Modified SIMPLER). MSIMPLER is described by Algorithm 4.3.

It is clear that the cost of the MSIMPLER preconditioner is equal to the cost of the SIMPLER preconditioner. However, in solving the Navier-Stokes problem, at each non-linear iteration, the Schur complement approximation in MSIMPLER does not need to be build again because the operators used in the Schur complement approximation are independent of any change taken place at each non-linear iteration. This in contrast to the SIMPLER preconditioner where we have to rebuild the Schur complement approximation in each non-linear step.

Algorithm 4.3: MSIMPLER preconditioner

```
1.  Solve  $\hat{S}p^* = r_p - B\hat{Q}_v^{-1} r_u$.
2.  Solve  $Fu^* = r_u - B^T p^*$.
3.  Solve  $\hat{S}\delta p = r_p - Bu^*$.
4.  update  $u = u^* - \hat{Q}_v^{-1} B^T \delta p$.
5.  update  $p = p^* + \delta p$.
```

5. A saddle point ILU type preconditioner

A disadvantage of the block preconditioners from the prior sections is that straightforward application of standard finite element codes is impossible. Both the standard matrix builder and solver must be adapted, since splitting of velocity and pressure unknowns is required. It would be attractive, if standard classical iterative solution schemes, like preconditioned Krylov solvers, could be applied, without any changes. However, in the case of non-stabilized elements, the zero pressure block in the continuity equation, prevents straightforward application of LU and ILU factorizations. If the common ordering of unknowns is used, i.e. placing first all unknowns of node 1, then those of node 2 and so on, one might get a zero pivot, especially if velocities at some boundaries are prescribed and therefore both factorizations may fail. Pivoting, [31], on the other hand, will result in a large increase of memory usage and, as a consequence, computation time. Also estimating the amount of memory is hard, which from an implementation point of view is, not very practical. To avoid this problem, it is better to use a suitable a priori reordering of unknowns. As pointed out by Wille and others [7–9], pivoting is not necessary, when the unknowns are ordered in the sequence, so that all velocity unknowns come first and then all the pressure unknowns; like in the block preconditioners. This is due to the fact that during (incomplete) factorization the zeros at the main diagonal will become non-zero.

A better approach is to combine a node renumbering scheme with a special ordering of the unknowns. In Subsection 5.1, we treat a method that results in an optimal band width or profile (envelope) of the coefficient matrix, in case of a direct method. The combination of this reordering technique, with ILU preconditioned Krylov solvers, usually results in better solver performance. The extra advantage is of course the simplicity of the implementation.

5.1. Ordering for direct solvers

If we consider the complete matrix, using the same ordering as in case of the block-preconditioners, i.e. placing first all velocity unknowns and then all pressure unknowns, we end up with a very large profile of the matrix. This is even true if we use an optimal node renumbering. The main advantage of this ordering is that no pivoting is necessary in case of a direct method. In the remaining part of this paper we shall refer to this reordering as *p-last*. Fig. 2 shows an example of the non-zero structure of the matrix for the p-last ordering, applied to a 4×4 rectangular grid of $Q2-Q1$ elements, where a lexicographic numbering of nodes is used.

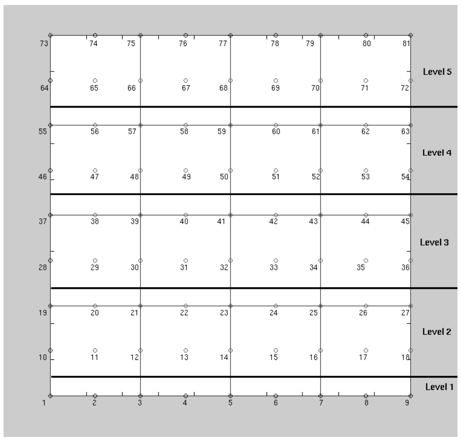

Figure 1: Levels defined for 4×4 $Q2-Q1$ grid.

We get a much smaller profile, if we order the unknowns in the sequence of the (renumbered) nodal points. However, this may lead to zero pivots, especially in the case of Dirichlet boundary conditions for the velocity. So this ordering is not applicable if we try to avoid pivoting during elimination. Our goal is to develop a reordering, that avoids zero pivots, but has a profile that is comparable to the one corresponding to node-wise ordering. To that end we need to define the concept of levels, which originates from the classical Cuthill McKee renumbering scheme.

Let us first define the notion of levels for Cuthill McKee. Suppose we have created levels 1 to i-1. Then level i is defined as the set of nodes that are connected directly to level $i-1$, and are not in one of the prior levels. Nodes are connected if they belong to the same element.

The first level may be defined as a point, or even a line in R^2 or a surface in R^3. This definition also applies in case of structured grids. For example in the 4×4 structured grid of Fig. 1, with $Q2-Q1$ elements, the first level might consist of the nodes 1 to 9. Level 2 consists of the nodes 10 to 29 and so on. It is clear that nodes in level i are only connected to nodes in level $i-1$ and level i+1.

In case of a different renumbering scheme, like Sloan [32] or the one proposed by Wille et al. [8], we define levels in the following way:

Suppose level 1 to $i-1$ have been constructed. Let node, k, be the node with highest node number, that is directly connected to nodes in level $i-1$. Then level, i, consist of node k, and all nodes with node number less than k but not belonging to one of the levels 1 to $i-1$. The first level is defined as node, 1.

Once the levels have been defined, we reorder the unknowns in the following way. First we take all the velocities of level 1, then all pressures of level 1. Next we do the same for level 2, and repeat this process for all nodes. So instead of a global block ordering we apply a block ordering per level. Such an approach has two advantages.

First, the profile is hardly enlarged, compared to the optimal ordering, since the local bandwidth is defined by the largest distance in node numbers.

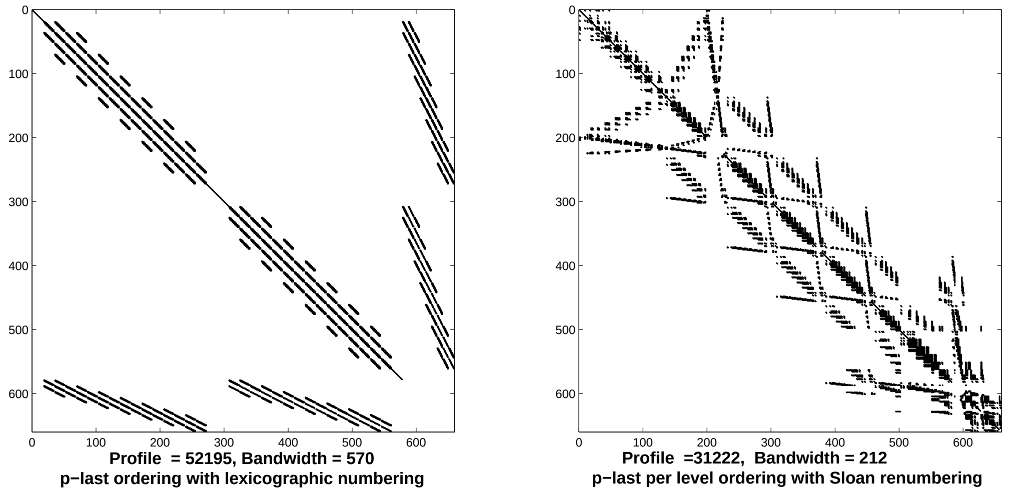

Figure 2: Effect of Sloan renumbering of grid points and p-last per level reordering of unknowns on the profile and bandwidth of the matrix.

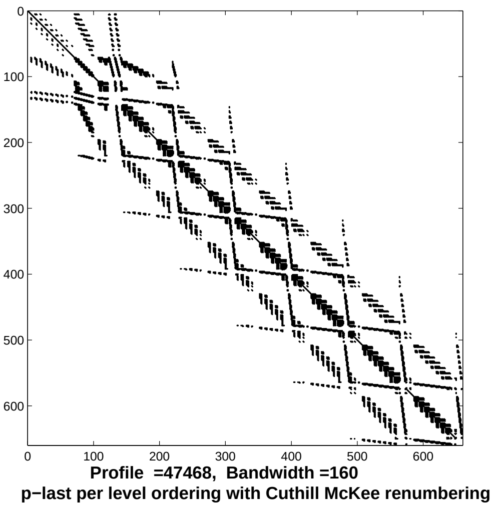

Figure 3: Effect of Cuthill McKee renumbering of grid points and p-last per level reordering of unknowns on the profile and bandwidth of the matrix.

Second, due to the local block reordering, zero pivots become non-zero, during factorization, and no a postiori pivoting is required. In the remainder we shall refer to this ordering technique as *p-last per level*.

One has to be careful at the start of this process. If, for example, the velocities in node 1, are prescribed, we start with a pressure unknown that gives rise to a zero pivot. Therefore, we always combine levels 1 and 2, into a new level. If the number of free velocity unknowns in this new level, is less than the number of pressure unknowns, we also add the next level to level 1, and if necessary this process is repeated. In practice combinations of 2 or 3 levels is sufficient. Note that the starting level has always a small contribution to the global profile. Figs. 2 and 3, show the effect of the p-last per level renumbering, combined with Sloan and Cuthill McKee renumbering, respectively. The grid used, consists of 8×8 $Q2-Q1$ elements. The gain in memory is clear, even for this small example.

So our reordering technique p-last per level, in combination with a suitable node renumbering strategy, produces a nearly optimal profile and avoids the need for pivoting in case of direct solvers. It has been applied to many practical problems, without ever producing small pivots. Since optimal reordering of unknowns for direct methods, are usually also suitable for ILU preconditioners, we consider this method in the next subsection.

5.2. Renumbering for ILU preconditioning

The block preconditioners of the previous sections, all require adaptation of standard finite element software packages. ILU-preconditioned Krylov subspace solvers, on the other hand, can be applied without any change at all. Since an optimal ordering of unknowns for a direct solver, usually improves the behavior of an ILU preconditioner, in Section 8 we investigate p-last per level ordering, as well as p-last ordering, in combination with ILU.

We define the set, S, of fill-in positions as the set of unknowns, that are directly connected. This implies that, zeros in the pressure block, may also be part of the set S, provided that there is a connectivity with velocity unknowns. The ILU decomposition $\mathscr{A} \approx LD^{-1}U$ is defined by the following rules:

$$\begin{aligned} l_{i,j} &= 0 \quad \text{for } (i,j) \notin \mathscr{S}, \\ u_{i,j} &= 0 \quad \text{for } (i,j) \notin \mathscr{S}, \\ (LD^{-1}U)_{i,j} &= a_{i,j} \quad \text{for } (i,j) \in \mathscr{S}, \end{aligned}$$

Experiments in Section 8, show that in a large number of practical cases, this method performs very well and is competitive, with the block preconditioners. However, in some cases the Krylov method converges slowly, or even diverges, for example in case of stretched grids, using elements with a large aspect ratio. In that case, we apply *extra fill-in* referred to as ILUF. Extra fill-in is defined by adding all neighbor points of the standard ILU node structure to the connectivity set, provided these nodes do not affect the envelope of the original matrix. In many cases, extra fill-in solves the convergence problem, but at the cost of extra memory and computing time per iteration.

6. Special methods for the Stokes equations

If one is only interested in solving the Stokes equations, one can use specially developed methods that perform only well in case of absence of the convective terms. Such problems can for example be found in the field of geodynamics, but also in case of extrusion problems, like extrusion of aluminum. Usually the viscosity in these problems is far from being constant. Since the Stokes matrix is symmetric indefinite an option is to use

MINRES as iteration method. However, MINRES requires an SPD preconditioner and this limits our choice. The combination of non-symmetric preconditioners and GCR leads to a faster convergence. In this section we shall treat two methods that are very efficient for these kind of problems. We limit ourselves to the stable case, since at this moment no good preconditioner for stabilized elements is available.

6.1. PMM preconditioner

If we use exact arithmetic, GMRES using a version of the $\mathscr{D}_b\mathscr{U}_b$ decomposition (3.2) as preconditioner, converges in two iterations. In Section 3 we have seen some approximations of this preconditioner that can be applied for the Navier-Stokes equations. In case of the Stokes equations a very simple approximation of the Schur complement matrix S can be constructed. In case of stable elements and constant viscosity, S is defined by $-BF^{-1}B^T$, which can be considered as an approximation of $-\nabla \cdot \frac{1}{\nu}\Delta^{-1}\nabla$. Here Δ must be considered component-wise. Suppose one could interchange the operators this could be written as $-\frac{1}{\nu}\nabla \cdot \nabla\Delta^{-1}$. Since $-\nabla \cdot \nabla$ is equal to Δ and since we use the finite element method, this suggests that minus the pressure mass matrix, M_p, with elements:

$$m_{ij} = \int\limits_{\Omega} \frac{1}{\nu}\psi_i\psi_j \, d\Omega, \tag{6.1}$$

with ψ_i basis functions corresponding to the pressure, can be used as preconditioner for the Schur matrix. Note that putting $\frac{1}{\nu}$ into the integral makes this definition applicable to non-constant viscosities.

The complete PMM preconditioner [33,34] is defined by

$$P_t = \begin{bmatrix} F & B^T \\ 0 & -M_p \end{bmatrix}. \tag{6.2}$$

Solving $P_t z = r$ requires the steps in Algorithm 6.1.

Algorithm 6.1: PMM algorithm

```
1.  **Solve** $M_p z_2 = r_2$.
2.  **update** $r_1 = r_1 - B^T z_2$.
3.  **Solve** $F z_1 = r_1$.
```

Each of the steps can be performed with a lower accuracy than the accuracy required for the outer GCR iterations.

We applied the PMM preconditioner successfully to various Stokes problems with constant viscosity, but also with variable viscosity with large contrasts in the domain. In case of the Stokes problem it appears to be more efficient than the general methods of Section 3 to 5. One of the reasons is that solving the pressure step is very cheap compared to the methods treated before. However, one should be careful with the termination criterion as will be demonstrated in Section 7.

An alternative method to precondition the Stokes equations is the so-called Schur method, which is considered in the next section.

6.2. The Schur method

The Schur method is based on the $L_{bt}U_b$ factorization ((3.1) and (4.1)). The Schur-complement matrix, S, present in the factorization is treated implicitly. In order to apply pressure-correction type methods, we split the coefficient matrix as follows:

$$\begin{bmatrix} F & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} u \\ p \end{bmatrix} = \begin{bmatrix} F & 0 \\ B & S \end{bmatrix} \begin{bmatrix} u^* \\ \delta p \end{bmatrix}, \tag{6.3}$$

where

$$\begin{bmatrix} u^* \\ \delta p \end{bmatrix} = \begin{bmatrix} I & F^{-1}B^T \\ 0 & I \end{bmatrix} \begin{bmatrix} u \\ p \end{bmatrix}. \tag{6.4}$$

Then the systems of equations can be solved in the steps in Algorithm 6.2.

Algorithm 6.2: The Schur method

```
Initialize $u^{(0)}$, $p^{(0)}$ and $maxiter$ (maximum iterations)
Compute: $r_u = f - Fu^{(0)} - B^T p^{(0)}$
 $r_p = g - Bu^{(0)}$
For $k = 0$ to $maxiter$
    1. Solve $Fu_f = r_u$
    2. Solve $Sp_\delta = r_p - Bu_f$
    3. Update $u_\delta = u_f - u_l$, where $u_l$ is obtained by solving $Fu_l = B^T p_\delta$
    4. Update $u^{(k+1)} = u^{(k)} + u_\delta$
    5. Update $p^{(k+1)} = p^{(k)} + p_\delta$
    6. Update $r_u = f - Fu^{(k+1)} - B^T p^{(k+1)}$
    7. Update $r_p = g - Bu^{(k+1)}$
    8. If converged Exit
End For
```

As already mentioned, the Stokes problem is symmetric and indefinite. However, the subsystems corresponding to the velocity (vector Poisson) and pressure used in the solves in Steps 1, 2 and 3 are symmetric and definite. F^{-1} in Steps 1 to 3 is computed approximately by solving the velocity subsystem with an inexact solver. The best option is to use MG preconditioned CG or some multigrid technique since both of these methods are known to give optimal convergence for Poisson-type problems.

The pressure subsystem in Step 2 can in principle, be solved efficiently by CG. However, we do not construct $-BF^{-1}B^T$ explicitly, but approximately solve $(-BF^{-1}B^T)p_\delta = r_p - Bu_f$

within each step of CG. Since F^{-1} is computed inexactly, CG can only be applied if the number of iterations used to do this is kept constant in each step. This is due to the fact that CG requires a constant matrix and preconditioner. This problem can be overcome by either using a stand-alone solver, such as multigrid, or a flexible Krylov method (GCR in our case). The efficiency of the Schur method requires efficient treatment of Step 2. Because the Schur-complement matrix is not constructed explicitly, we need a special type of preconditioner. Like in PMM, we use the pressure mass matrix as preconditioner for the Schur subsystem. If we use the same accuracy to solve the system with PMM and to solve Step 2 of the Schur method, the number of pressure mass matrix preconditioned GCR iterations in both methods are almost the same. These iterations govern the efficiency of both techniques. This observation also motivates the use of GCR instead of flexible CG [35].

Based on the Schur method, Algorithm 6.2, we propose two schemes:

1. **Schur method as direct method:** By requiring the accuracy of the subsystem solves to be the same or higher than the outer accuracy, the Schur method can be used as a direct solver. To solve the Schur pressure system $Sp_\delta = r_p - Bu_f$, we use CG with the pressure mass matrix, M_p, as a preconditioner. The system, $Sp_\delta = r_p - Bu_f$, is solved with the help of GCR in which preconditioned matrix-vector products within $(BF^{-1}B^T)p_\delta$ are obtained by computing preconditioned residual $S^{(k+1)} = Bu_m$, where u_m is obtained by solving a subsystem $Fu_m = B^T M_p{}^{-1} r^{(k)}$ and $r^{(k)}$ is the residual computed in the previous GCR iteration.

2. **Schur as an iterative method:** When the inner systems are solved with a lower accuracy than desired of the final solution, outer iterations are required. Again, the pressure mass matrix is used as preconditioner for the pressure subsystem.

From the above discussion, it is clear that three subsystems are solved for the velocity unknowns u_f, u_m, and u_l and one subsystem is solved for the pressure unknown p_δ in each iteration of the Schur method. The most expensive part of the algorithm is the computation of u_m since the number of times $Fu_m = r$ must be solved is equal to the total number of GCR iterations that are required to solve the pressure subsystem.

One of the advantages that can be seen from the above algorithm is that most of the computations are done at subsystem level. The system level computations can be reduced by a proper choice of the inner accuracy. For example, if subsystems are solved as accurately as the outer tolerance requires, only one outer iteration is required. However, more outer iterations are typically required.

7. Discussion of the convergence criteria

It has been observed that for the same preconditioner, different implementations (Split, left or right preconditioner) leads to the same eigenvalue spectrum of the preconditioned system [36, Chapter 13, page 175]. Therefore, the same number of iterations are expected for a desired accuracy.

In our software we use GCR in combination with a right preconditioner. This suggests that the stopping criterion is independent of the preconditioner. However, if we solve the Stokes problem in combination with the PMM preconditioner, we can see that for increasing grid size the actual error becomes much larger than the required accuracy (see Table 1). Therefore we study the stopping criterion in more detail.

The convergence criterion we use is

$$\left|\left|b - \mathscr{A}x_k\right|\right|_2 < \varepsilon \, ||b||_2, \tag{7.1}$$

From this relation it can be proved that

$$\frac{\left|\left|x - x_k\right|\right|_2}{||x||_2} \leq K_2(\mathscr{A}) \frac{\left|\left|r_k\right|\right|_2}{||b||_2} \leq K_2(\mathscr{A})\varepsilon.$$

This implies that the relative error depends on the condition number of the matrix. The right preconditioner has the advantage that it only effects the operator and not the right-hand side. This implies that stopping criteria for various preconditioners are the same.

Remark 7.1. Since in right preconditioning we solve $\mathscr{A}P^{-1}y = b$ with $x = P^{-1}y$ one must be careful with stopping criteria that are based on the error $\left|\left|y - y_k\right|\right|_2$ because this error may be much smaller than the error norm $\left|\left|x - x_k\right|\right|_2$ (equal to $\left|\left|P^{-1}(y - y_k)\right|\right|_2$) [36, Chapter 13, page 175].

In case of left preconditioning, the convergence criterion depends also on the preconditioner

$$\left|\left|P^{-1}(b - \mathscr{A}x_k)\right|\right|_2 < \varepsilon \left|\left|P^{-1}b\right|\right|_2.$$

Now the condition number is changed to $K_2(P^{-1}\mathscr{A})$ which is expected to be much smaller than $K_2(\mathscr{A})$ and the convergence criterion may be quite different then the criterion based on the unpreconditioned residual.

In case of a variable preconditioner one should use a right preconditioner because its independence of the stopping criterion. In order to avoid the dependence of the termination criterion on the condition number of $\mathscr{A}$ we apply a scaling of the complete system.

The idea is to minimize the condition number of the matrix to $K_2(S_m^{-1}\mathscr{A})$, where S_m is a scaling matrix. Diagonal scaling is considered to be optimal if the original system is SPD [37] but also for the symmetric indefinite system we use this type of scaling.

Tables 1 to 3 compare the error in the pressure part in the Stokes problem preconditioned with PMM and LSC with different termination criteria. Note that the pressure part is the most sensitive part in (Navier-)Stokes, the velocity part usually behaves very well. It is clear that, if we base the termination on the norm of the residual, the real error increases for increasing grid size. On the other hand if we use the norm of the residual multiplied by the inverse of the preconditioner, we see that the real error is almost constant. Of course the number of iterations increases in this case. Unfortunately the computation of $P^{-1}r_k$ is very costly and therefore not practical. The alternative, scaling the complete system results in a termination based on $\left|\left|S_m^{-1}r_k\right|\right|$. This gives an increase in iterations, but clearly a much

Table 1: Backward facing step Stokes problem (PMM preconditioner).

Grid	$\|r_k\|$		$\|P^{-1}r_k\|$		$\|S_m^{-1}r_k\|$	
	Iter.	Error-p	Iter.	Error-p	Iter.	Error-p
8×24	18	2e-5	22	2e-7	20	2e-6
16×48	17	3e-4	24	8e-7	20	4e-6
32×96	16	1e-3	23	5e-7	20	1e-5

Table 2: Driven cavity Stokes problem (PMM preconditioner).

Grid	$\|r_k\|$		$\|P^{-1}r_k\|$		$\|S_m^{-1}r_k\|$	
	Iter.	Error-p	Iter.	Error-p	Iter.	Error-p
16×16	9	9e-5	15	9e-8	13	9e-7
32×32	9	3e-4	16	7e-8	13	1e-6
64×64	9	2e-3	16	9e-8	13	2e-6
128×128	8	4e-3	16	1e-7	12	8e-6

Table 3: Driven cavity Stokes problem (LSC preconditioner).

Grid	$\|r_k\|$		$\|P^{-1}r_k\|$		$\|S_{m1}^{-1}r_k\|$	
	Iter.	Error-p	Iter.	Error-p	Iter.	Error-p
64×64	10	1e-4	19	1e-8	15	3e-7
128×128	11	4e-4	24	1e-8	19	8e-7

more reliable result. The real error in all these cases is computed by solving the original system with a direct solver.

Because scaling of the system does not effect the eigenvalue spectrum of the preconditioned matrix, but only changes the convergence criterion, it is much cheaper to scale only the residual by S_m^{-1}, before checking the accuracy.

Various choices are possible for the scaling matrices. In this section, we consider two variants. Vuik et al. [26] and May and Moresi [38] use the following scaling for the complete system with different preconditioning strategies

$$S_m = \begin{bmatrix} \sqrt{diag(BD^{-1}B^T)} & 0 \\ 0 & \sqrt{diag(BD^{-1}B^T)} \end{bmatrix}. \tag{7.2}$$

D is the diagonal of the velocity matrix. S_m is only well-defined when $F_{ii} > 0$ and $BD^{-1}B_{ii}^T > 0$. If we use this scaling, our system, $\mathscr{A}x = b$ becomes $S_m^{-1}\mathscr{A}S_m^{-1}S_m x = S_m^{-1}b$, the most important change is the termination criterion for the iterative method. Convergence criteria are based on the residual in a scaled L_2 norm.

Instead of scaling the complete system, we scale only the stopping criteria with the scaling matrix:

$$S_{m1} = \begin{bmatrix} diag(F) & 0 \\ 0 & diag(BD^{-1}B^T) \end{bmatrix}. \tag{7.3}$$

Another choice of scaling the stopping criteria can be that $diag(BD^{-1}B^T)$ is replaced by $diag(Q_p)$:

$$S_{m2} = \begin{bmatrix} diag(F) & 0 \\ 0 & diag(Q_p) \end{bmatrix}. \tag{7.4}$$

For the pressure part, the pressure mass matrix is a good scaling operator because in case of a lumped pressure mass matrix, this is a diagonal matrix.

Table 4 shows the results of this approach for two different choices of the scaling matrices, S_{m1} andS_{m2}. Both scaling matrices are comparable, but S_{m1} gives a slightly better error estimate than S_{m2}. It is clear that this cheap approach is a very good alternative for complete scaling. For more information concerning stopping criteria we refer to [39] and [40].

Table 4: Driven cavity Stokes problem solved using scaled stopping criteria.

Grid	PMM				LSC			
	$\|\|S_{m1}^{-1}r_k\|\|$		$\|\|S_{m2}^{-1}r_k\|\|$		$\|\|S_{m1}^{-1}r_k\|\|$		$\|\|S_{m2}^{-1}r_k\|\|$	
	Iter.	Error-p	Iter.	Error-p	Iter.	Error-p	Iter.	Error-p
16×16	16	1e-8	15	9e-8	11	2e-8	11	2e-8
32×32	17	2e-8	16	7e-8	14	2e-8	13	2e-7
64×64	17	4e-8	16	9e-8	19	1e-8	17	7e-8
128×128	18	2e-8	16	1e-7	24	1e-8	22	7e-8

8. Comparison of the various preconditioners

In this section we give an overview of some of the numerical experiments we have performed to test the various preconditioners. In case one only solves the Stokes equations the special methods of Section 6 perform better than the more general methods of Sections 3 to 5.

To test the performance of the special methods for the Stokes equations we use the SINKER problem [38]. It represents a benchmark problem for geodynamic models. Inside a square region we have another square with a different (but constant) viscosity, resulting in a sharp viscosity contrast. Besides that there is also a jump in the density between both regions. The configuration is shown in [38] and in Fig. 4. Boundary conditions are no normal flow and no shear stress (at all boundaries). For this boundary condition, the pressure can be determined only up to an arbitrary additive constant.

For the more general Navier-Stokes problem we have performed experiments with the 2d lid driven cavity (Fig. 5) and with the 2d (Fig. 6) and 3d backward facing step.

In this paper we only compare the best performing of the methods we treated, i.e. MSIMPLER, LSC and SILU. Experiments have been done in Matlab using the IFISS package[¶] and in Fortran with the SEPRAN package[‖].

In our experiments SILU is combined with Bi-CGSTAB. Outer iterations in the block preconditioners are always done with GCR.

[¶]http://www.maths.manchester.ac.uk
[‖]http://ta.twi.tudelft.nl/sepran/sepran.html

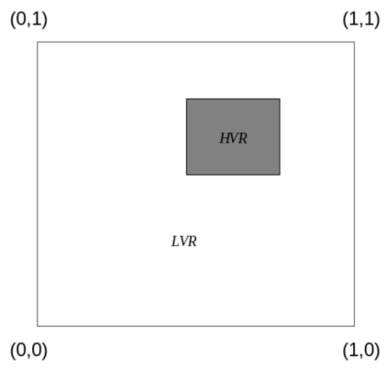

Figure 4: Domain for the SINKER model, LVR corresponds to ν_1 and HVR to ν_2.

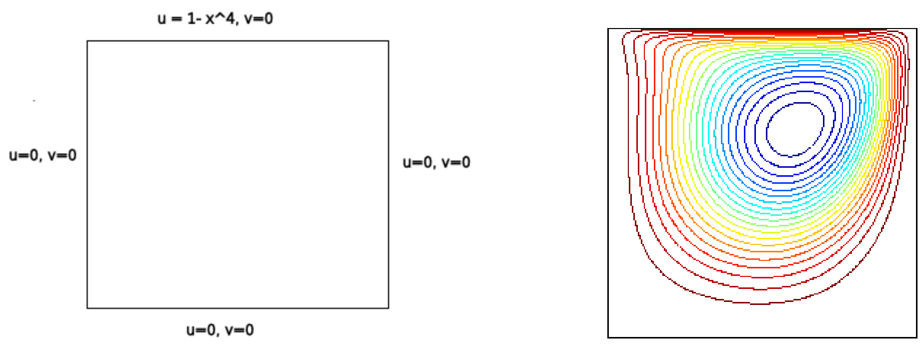

Figure 5: Definition of square cavity problem and corresponding streamlines.

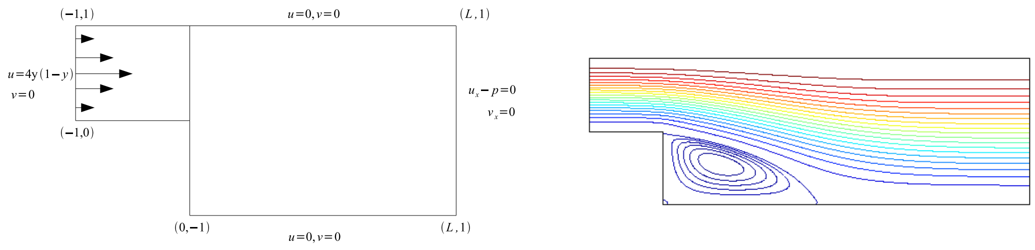

Figure 6: Definition of backward facing step and corresponding streamlines.

8.1. Comparison of the Stokes solvers

The SINKER problem is defined in Fig. 4. Here, we consider a forcing term $f = (0, -\rho g)$, where $g = 9.8m/s^2$. Since LSC either diverges or converges very slowly for the high viscosity contrasts considered here, we do not report any results for this method. However, a variant of LSC, LSC_D [34], where the velocity mass matrix is replaced by the diagonal of the velocity matrix, is able to solve problems with large contrasts in viscosity.

Before applying the iterative solvers, the matrices have been scaled with matrix S_m (7.2) in order to get a better condition. The velocity solution for all preconditioners is accurate. Therefore, we only remark on the accuracy of the pressure solution. This is done by comparing with the "exact solution", computed by a direct method.

Table 5 shows the number of iterations and the norm of the error in pressure. In the first experiments, ν_1 is kept fixed at one and ν_2 is increased; thereafter, ν_2 is kept equal to one and ν_1 is increased. We see that the number of iterations for PMM and Schur is almost the same for both the 30×30 and 60×60 grids, which suggests h-independent convergence. LSC_D shows a clear increase with the increase of grid points. For constant ν_2, the difference in accuracy between all methods is small.

Table 5: Iterative solution of the Stokes problem, accuracy $= 10^{-6}$.

ν	PMM		LSC_D		Schur	
	iter.	$\|p - p_{PMM}\|_2$	iter.	$\|p - p_{LSC_D}\|_2$	iter. (inner)	$\|p - p_{Schur}\|_2$
30×30						
$\nu_1 = 1,\ \nu_2 = 10^6$	12	9×10^{-4}	26	7×10^{-6}	2(18)	2×10^{-8}
$\nu_1 = 1,\ \nu_2 = 10^3$	12	2×10^{-5}	26	3×10^{-6}	2(20)	2×10^{-10}
$\nu_1 = 1,\ \nu_2 = 10^1$	11	5×10^{-6}	24	1×10^{-6}	2(16)	2×10^{-10}
$\nu_1 = 1,\ \nu_2 = 1$	11	4×10^{-7}	25	2×10^{-6}	2(5)	5×10^{-11}
$\nu_1 = 10^1,\ \nu_2 = 1$	15	1×10^{-6}	27	2×10^{-6}	1(14)	2×10^{-6}
$\nu_1 = 10^3,\ \nu_2 = 1$	18	4×10^{-6}	26	3×10^{-6}	1(18)	2×10^{-6}
$\nu_1 = 10^6,\ \nu_2 = 1$	15	4×10^{-4}	23	1×10^{-4}	1(16)	2×10^{-5}
60×60						
$\nu_1 = 1,\ \nu_2 = 10^6$	13	8×10^{-3}	40	6×10^{-5}	2(19)	5×10^{-8}
$\nu_1 = 1,\ \nu_2 = 10^3$	13	3×10^{-5}	40	5×10^{-6}	2(20)	3×10^{-9}
$\nu_1 = 1,\ \nu_2 = 10^1$	13	1×10^{-6}	41	3×10^{-6}	2(18)	4×10^{-10}
$\nu_1 = 1,\ \nu_2 = 1$	3	6×10^{-6}	36	3×10^{-6}	2(5)	9×10^{-10}
$\nu_1 = 10^1,\ \nu_2 = 1$	16	2×10^{-6}	41	4×10^{-6}	1(14)	4×10^{-6}
$\nu_1 = 10^3,\ \nu_2 = 1$	20	1×10^{-5}	38	7×10^{-6}	1(20)	5×10^{-6}
$\nu_1 = 10^6,\ \nu_2 = 1$	17	4×10^{-4}	35	1×10^{-4}	1(18)	3×10^{-5}

However, in the problem where ν_1 is small, the accuracy obtained with PMM is less than the other two iterative methods, even though all subsystems are solved with high accuracy (10^{-6} or 10^{-7}). The Schur method gives much more accurate results than the other two preconditioners, because the first iteration of the Schur method gives an accurate inner solve, while the second iteration makes the solution more accurate than the desired tolerance. From the table, we see that, with respect to accuracy and efficiency, the Schur method seems a better option than the other two. The reason is that PMM requires more iterations than Schur to get the same accuracy, while the costs per iteration are comparable. Similar results have been observed for a 90×90 grid.

8.2. Comparison in 2D

First of all we investigate the difference between p-last and p-last per level ordering. Table 6 gives the time and number of iterations needed for solving the Stokes problem on

Table 6: Solution of the Stokes problem with the $Q_2 - Q_1$ discretization in the square cavity with an *accuracy* of 10^{-6}.

Solver.	Renumber	16 × 16		32 × 32		64 × 64	
		Iter.	Time(s)	Iter.	Time(s)	Iter.	Time(s)
Direct	p-last	-	0.61	-	20.34	-	1378
	p-last per level	-	0.13	-	2.28	-	37
GMRES(20)	p-last	95	0.16	354	1.72	1800	44.0
	p-last per level	50	0.12	207	1.14	792	20.0
Bi-CGSTAB	p-last	36	0.11	90	0.92	255	11.98
	p-last per level	25	0.09	59	0.66	135	6.74

Table 7: Solution of the Stokes problem with the stabilized $P_1 - P_1$ discretization in the square cavity with an *accuracy* of 10^{-6}.

number of nodes	SILU		ILU	
	Iter.	Time(s)	Iter.	Time(s)
17 × 17	48	0.008	22	0.004
33 × 33	84	0.052	41	0.016
65 × 65	200	0.620	91	0.160
129 × 129	653	9.060	245	1.860
257 × 257	1891	118.359	1468	48.463

a square cavity with Q2-Q1 elements. Results are compared for a direct solver, and SILU preconditioning in combination with GMRES and Bi-CGSTAB. This shows that renumbering per level is in all cases more efficient than the p-last ordering. Even for the relatively small 16 × 16 elements grid, SILU in combination with Bi-CGSTAB performs better than the direct solver. In case of the $P_1 - P_1$ mini element, which may be considered as a stabilized element, no zeros appear on the main diagonal of the pressure block. As a consequence there is no need to reorder the unknowns. Table 7 shows that in this case ILU (i.e. without reordering) is a better option than SILU. In both cases the number of iterations increases considerably with the increase of grid size.

In Fig. 7, preconditioners for $Q_2 - Q_1$ elements are compared based on the number of outer iterations, CPU time and inner iterations. In terms of all these parameters, we can see that the MSIMPLER performance is better than the rest of the preconditioners. In terms of CPU time, SILU performance is comparable with MSIMPLER and better than that of the other preconditioners. We can see that the number of iterations of SILU and SIMPLE increase more for increasing grid size than for the other preconditioners. SIMPLE appears to be robust but expensive, and therefore we do not report further experiments in 2D.

Table 8 compares LSC and MSIMPLER in the case that the pressure and velocity subequations are solved by one MG cycle. Although we see that for coarse grids the convergence depends on the Reynolds number, this is no longer the case for the finest grid. Furthermore it is clear that the number of iterations decreases for fixed Reynolds number for finer grids. Presumably this is due to the decrease in cell Reynolds number ($element\ size/\nu$). In all cases MSIMPLER requires less iterations than LSC but the difference becomes small for increasing mesh size. The relative good result of the last number in the MSIMPLER

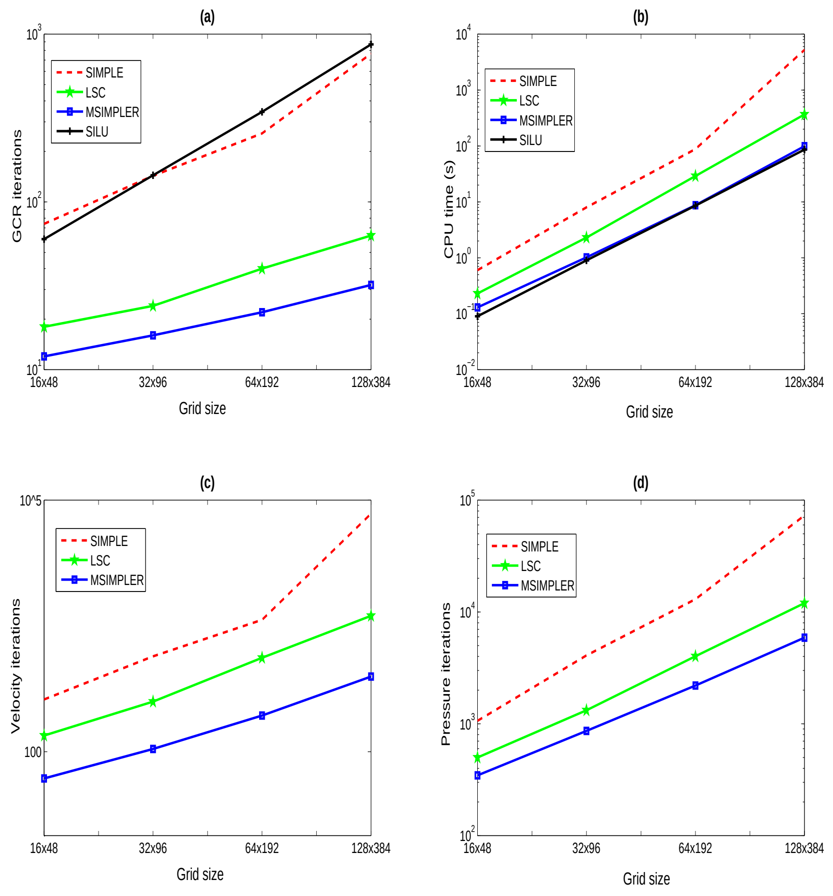

Figure 7: 2D Backward facing step (Q2-Q1): The Stokes problem is solved with accuracy 10^{-6}. PCG is used as inner solver in the block preconditioners (SEPRAN).

Table 8: Backward facing step Navier-Stokes problem with preconditioned Bi-CGSTAB with accuracy 10^{-6}. MG solver is used to solve subsystems (IFISS).

Grid	Re=100		Re=200		Re=400	
	LSC	MSIMPLER	LSC	MSIMPLER	LSC	MSIMPLER
	iter. (t_s)					
16×48	17(8)	9(4.5)	27(13)	15(7)	73(39)	29(16)
32×96	16(17)	11(13.7)	15(22)	10(17)	24(28.5)	15(21)
64×192	24(119)	20(99)	23(118)	15(84)	22(112)	18(102)

column must be because of the better cell Reynolds number.

Table 9 shows the convergence of MSIMPLER, LSC and SILU in case the subsystems in the block preconditioners are solved by ILU preconditioned Bi-CGSTAB. The accuracy for the inner solves is 10^{-2}, which is sufficient to reach the final accuracy of the Navier-Stokes problem without increasing the number of Picard iterations. In this table we report the sum

Table 9: Backward facing step: Preconditioned GCR is used to solve the Navier-Stokes problem with accuracy 10^{-2}, using Bi-CGSTAB as inner solver, the number of iterations are the accumulated iterations consumed by the outer and inner solvers (SEPRAN).

Grid	LSC	MSIMPLER	SILU(Bi-CGSTAB)
	iter. (t_s)		iter. (t_s)
Re=100 (11 Picard iterations)			
16×48	114(1.7)	73(1)	246(0.8)
32×96	193(22)	106(10.5)	731(8.7)
64×192	328(545)	182(162)	2071(95)
128×384	695(8863)	296(2806)	6352(1155)
Re=200 (17 Picard iterations)			
16×48	179(2.3)	137(1.7)	436(1.3)
32×96	302(31)	161(14)	1100(13)
64×192	598(983)	232(191)	3114(141)
128×384	946(10405)	541(6301)	2668(9038)
Re=400 (31 Picard iterations)			
16×48	441(4.93)	356(3.9)	716(2.13)
32×96	528(51)	328(25)	1706(20.7)
64×192	NC	405(408)	5366(246)
128×384	NC	663(7025)	NC

of the iterations in all Picard steps, which gives a complete picture of the whole problem. In this case the difference between MSIMPLER and LSC is much more pronounced. The reason must be the change of inner solver. Furthermore we see that SILU is faster than MSIMPLER except for the finest grid in combination with the larger Reynolds numbers.

Table 10: Driven cavity flow problem: The Navier-Stokes problem is solved with preconditioned Bi-CGSTAB with accuracy 10^{-6}. MG and direct solver is used to solve subsystems (IFISS).

Grid	Re=100		Re=500		Re=1000	
	LSC	MSIMPLER	LSC	MSIMPLER	LSC	MSIMPLER
	MG/Exact	MG/Exact	MG/Exact	MG/Exact	MG/Exact	MG/Exact
No. of iterations per Picard step						
16×16	14/10	10/9	53/29	28/25	102/55	50/53
32×32	19/15	13/12	35/26	26/20	82/55	45/43
64×64	22/22	19/19	34/29	25/25	63/55	34/32
128×128	27/27	25/28	47/44	42/44	62/59	43/43

In order to see if the block preconditioners are sensitive to the inner accuracies we compare one MG cycle for the inner solver with an exact inner solver in Table 10. From this table it is clear that MSIMPLER is hardly effected by the inner accuracy, whereas LSC is more sensitive in case of coarse grids in combination with a high Reynolds number.

8.3. Comparisons in 3D

Iterative solvers for the Navier-Stokes are especially important for 3D problems. In our experiments, we used both hexahedra and tetrahedra. Only Taylor-Hood elements have

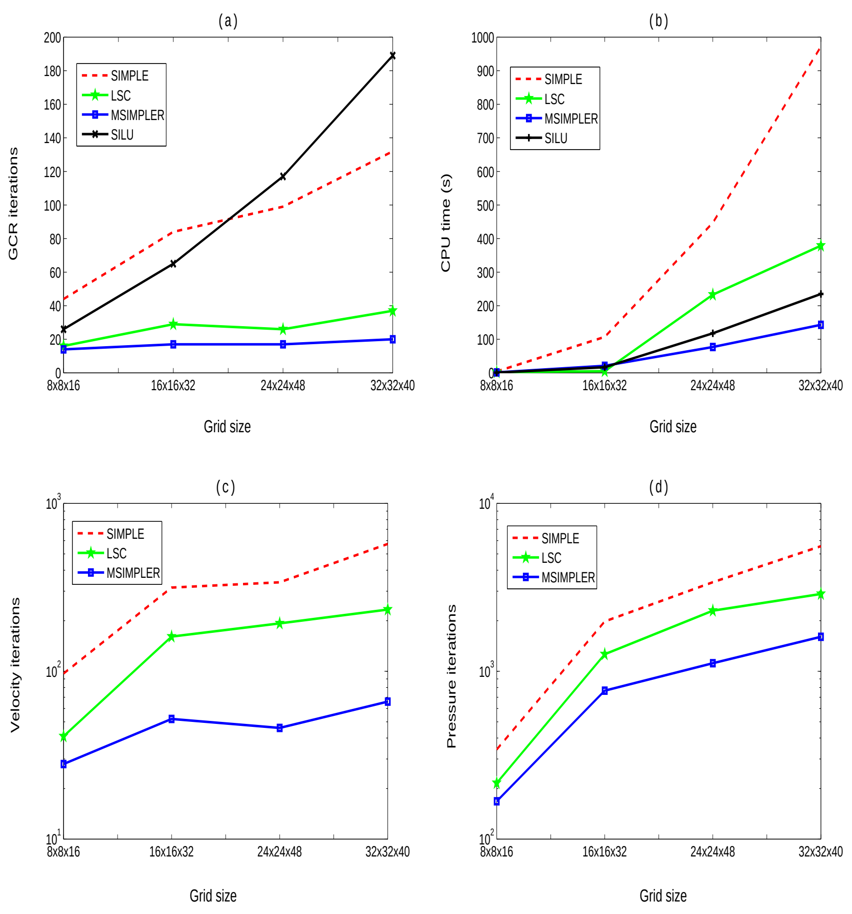

Figure 8: 3D Backward facing step (hexahedra): The Stokes problem is solved with accuracy 10^{-6}. PCG is used as inner solver in the block preconditioners (SEPRAN).

been applied.

Fig. 8 shows results of the various preconditioners for the Stokes problem solved on a 3D backward facing step with hexahedral elements. An IC preconditioned CG solver is used as inner solver for the block preconditioners. MSIMPLER requires the least number of iterations (inner/outer) and shows almost grid independent convergence behavior. The computation time of SILU is also good, but for finer grids it becomes more expensive than MSIMPLER.

In the Navier-Stokes problem it is sufficient to use an accuracy of 10^{-2} per Picard step. In this case SILU performs slightly better than MSIMPLER, see Table 11.

To investigate the behavior of the preconditioners for tetrahedral elements we solved the 3D lid driven cavity problem (Table 12). For the Stokes problem the result is comparable to the hexahedral case. MSIMPLER requires less CPU time than LSC and SILU. The number of GCR iterations is almost mesh-independent.

Table 11: 3D Backward facing step (hexahedra):The Navier-Stokes problem is solved with accuracy 10^{-4}, a linear system at each Picard step is solved with accuracy 10^{-2} using preconditioned Krylov subspace methods. Bi-CGSTAB is used as inner solver in block preconditioners (SEPRAN).

Re	SIMPLE	LSC	MSIMPLER	SILU
	GCR iter. (t_s)			Bi-CGSTAB iter. (t_s)
$8 \times 8 \times 16$				
100	200(23)	117(17.6)	74(9.6)	140(8.9)
200	314(31)	176(25)	112(14.8)	255(13.8)
400	509(47)	280(36)	168(21)	1688(49)
$16 \times 16 \times 32$				
100	447(591)	173(462)	96(162)	321(114)
200	718(839)	256(565)	145(223)	461(173)
400	1277(1223)	399(745)	235(312)	768(267)
$32 \times 32 \times 40$				
100	909(12000)	240(5490)	130(1637)	1039(1516)
200	> 1000	421(7784)	193(2251)	1378(2000)
400	> 2000	675(11000)	295(2800)	1680(2450)

Table 12: 3D Lid driven cavity problem (tetrahedra): The Stokes problem is solved with accuracy 10^{-6}. PCG is used as inner solver in block preconditioners (SEPRAN).

Grid	LSC	MSIMPLER	SILU (Bi-CGSTAB)
	iter. $(t_s)\frac{\text{in-it-}u}{\text{in-it-}p}$		iter. (t_s)
$8 \times 8 \times 8$	9(0.24) $\frac{17}{52}$	8(0.23) $\frac{16}{53}$	32(0.25)
$16 \times 16 \times 16$	12(4.8) $\frac{49}{152}$	11(3.4) $\frac{31}{150}$	73(5.6)
$32 \times 32 \times 32$	17(89) $\frac{129}{426}$	14(54) $\frac{68}{380}$	237(162)

Table 13: 3D Lid driven cavity problem (tetrahedra): The Navier-Stokes problem is solved with accuracy 10^{-4}, a linear system at each Picard step is solved with accuracy 10^{-2} using preconditioned Krylov subspace methods. Bi-CGSTAB is used as inner solver in block preconditioners (SEPRAN).

Re	LSC	MSIMPLER	SILU
	GCR iter. (t_s)	GCR iter. (t_s)	Bi-CGSTAB iter. (t_s)
$16 \times 16 \times 16$			
20	30(20)	20(16)	144(22)
50	57(37)	37(24)	234(35)
100	120(81)	68(44)	427(62)
$32 \times 32 \times 32$			
20	38(234)	29(144)	463(353)
50	87(544)	53(300)	764(585)
100	210(1440)	104(654)	1449(1116)

The situation for tetrahedral elements to discretize Navier-Stokes is different from that of Stokes. Table 13 gives the CPU time and number of iterations. Now MSIMPLER proves to be the best choice. The increase of iterations for increasing Reynolds number is caused by an increase of the number of Picard iterations. The Reynolds dependency of all methods per Picard iteration is only mild.

9. Conclusions

In this paper we studied the convergence behavior of block preconditioners for Stokes and Navier-Stokes problems both in 2D and 3D. Results for various grid sizes and Reynolds numbers have been investigated. We also compared the convergence with an algebraic preconditioner (SILU). Some common properties of MSIMPLER and LSC are discussed. In all our experiments MSIMPLER proved to be cheaper than LSC. This concerns both the number of outer iterations, inner iterations and CPU time. The number of outer iterations in MSIMPLER hardly increases if a direct solver for the subsystems is replaced by an iterative solver. This is in contrast with LSC where large differences are observed. It appears that the combination of LSC with MG is almost optimal but the combination of LSC with a PCG inner solver can take many iterations and much CPU time. When problems are solved with low accuracy, for example in case of Navier-Stokes, SILU sometimes shows better performance than other preconditioners. In case of stabilized elements there is no need to use SILU, since standard ILU is a better alternative. MSIMPLER proved to be cheaper than SILU, especially when the problem is solved with high accuracy.

If only the Stokes equations have to be solved, PMM and the Schur method are faster than the other preconditioners. For fine grids the best results can be expected in case the subsystems are solved by Algebraic Multigrid Preconditioned CG (AMG/CG).

It must be remarked that in case of stretched grids the performance of all preconditioners decreases as function of the stretching. In some cases even divergence occurs. Solving this problem is a source for future research.

Acknowledgments The second author thanks the Higher Education Commission (HEC) Pakistan for their support.

References

[1] F. Brezzi and M. Fortin. *Mixed and Hybrid Finite Element Methods*. Springer-Verlag New York, Inc., 1991.

[2] M. Fortin. Old and new finite elements for incompressible flows. *Int. J. Numer. Meth. Fluids*, 1(4):347–364, 1981.

[3] C. Cuvelier, A. Segal, and A. A. van Steenhoven. *Finite Element Methods and Navier-Stokes Equations*. Reidel Publishing Company, Dordrecht, Holland, 1986.

[4] C. R. Dohrmann and P. B. Bochev. A stabilized finite element method for the Stokes problem based on polynomial pressure projections. *Int. J. Numer. Meth. Fluids*, 46(2):183–201, 2004.

[5] M. ur Rehman, C. Vuik, and G. Segal. A comparison of preconditioners for incompressible navier-stokes solvers. *Int. J. Numer. Meth. Fluids*, 57:1731–1751, 2008.

[6] M. Fortin and R. Glowinski. *Augmented Lagrangian Methods: Applications to the Numerical Solution of Boundary Value Problems*. Elsevier Science Ltd. Noth-Holland, Amsterdam, 1983.

[7] O. Dahl and S. . Wille. An ILU preconditioner with coupled node fill-in for iterative solution of the mixed finite element formulation of the 2D and 3D Navier-Stokes equations. *Int. J. Numer. Meth. Fluids*, 15(5):525–544, 1992.

[8] S. . Wille and A. F. D. Loula. A priori pivoting in solving the Navier-Stokes equations. *Commun. Numer. Meth. Engng.*, 18(10):691–698, 2002.

[9] S. . Wille, O. Staff, and A. F. D. Loula. Efficient a priori pivoting schemes for a sparse direct Gaussian equation solver for the mixed finite element formulation of the Navier-Stokes equations. *Appl. Math. Modelling*, 28(7):607–616, July 2004.

[10] S. V. Patankar. *Numerical heat transfer and fluid flow*. McGraw-Hill, New York, 1980.

[11] D. Kay, D. Loghin, and A. Wathen. A Preconditioner for the Steady-State Navier-Stokes Equations. *SIAM J. Sci. Comput.*, 24(1):237–256, 2002.

[12] D. Silvester, H. Elman, D. Kay, and A. Wathen. Efficient preconditioning of the linearized Navier-Stokes equations for incompressible flow. *J. Comput. Appl. Math.*, 128(1-2):261–279, 2001.

[13] H. Elman, V. E. Howle, J. Shadid, R . Shuttleworth, and R. Tuminaro. Block Preconditioners Based on Approximate Commutators. *SIAM J. Sci. Comput.*, 27(5):1651–1668, 2006.

[14] M. Benzi and M. A. Olshanskii. An Augmented Lagrangian-Based Approach to the Oseen Problem. *SIAM J. Sci. Comput.*, 28(6):2095–2113, 2006.

[15] A. C. de Niet and F. W. Wubs. Two preconditioners for saddle point problems in fluid flows. *Int. J. Numer. Meth. Fluids*, 54(4):355–377, 2007.

[16] M. Benzi. Preconditioning Techniques for Large Linear Systems: A Survey. *J. Comput. Phys.*, 182(2):418–477, 2002.

[17] M. Benzi, G. H. Golub, and J. Liesen. Numerical solution of saddle point problems. *Acta Numerica*, 14:1–137, 2005.

[18] H. C. Elman, D. Silvester, and A. J. Wathen. *Finite Elements and Fast Iterative Solvers with applications in incompressible fluids dynamics*. Oxford University Press, Oxford, 2005.

[19] C. Taylor and P. Hood. A numerical solution of the Navier-Stokes equations using the finite element techniques. *Computers and Fluids*, 1:73–100, 1973.

[20] M. Crouzeix and P. A. Raviart. Conforming and nonconforming finite element methods for solving the stationary Stokes equations. *Rairo Analyse Numerique*, 7:33–76, 1973.

[21] D.N. Arnold, F. Brezzi, and M. Fortin. A stable finite element method for the Stokes equation. *Calcolo*, 21:337–344, 1984.

[22] M. F. Murphy, G. H. Golub, and A. J. Wathen. A Note on Preconditioning for Indefinite Linear Systems. *SIAM J. Sci. Comput.*, 21(6):1969–1972, 2000.

[23] H. Elman, V. Howle, J. Shadid, D Silvester, and R. Tuminaro. Least squares preconditioners for stabilized discretizations of the Navier-Stokes equations. *SIAM J. Sci. Comput.*, 30(1):290–311, 2007.

[24] M. Benzi, M. A. Olshanskii, and Z. Wang. Augmented Lagrangian preconditioners for the incompressible Navier-Stokes equations. *Submitted to Int. J. Numer. Meth. Fluids*, 2009.

[25] P. Wesseling. *Principles of Computational Fluid Dynamics*, volume 29. Springer Series in Computational Mathematics, Springer, Heidelberg, 2001.

[26] C. Vuik, A. Saghir, and G. P. Boerstoel. The Krylov accelerated SIMPLE(R) method for flow problems in industrial furnaces. *Int. J. Numer. Meth. Fluids*, 33(7):1027–1040, 2000.

[27] C. Li and C. Vuik. Eigenvalue analysis of the SIMPLE preconditioning for incompressible flow. *Numer. Lin. Alg. Appl.*, 11(5-6):511–523, 2004.

[28] J.B. Perot. An analysis of the fractional step method. *J. Comp. Phys.*, 108:51–58, 1993.

[29] J. Blasco, R. Codina, and A. Huerta. A fractional-step method for the incompressible Navier-Stokes equations related to a predictor-multicorrector algorithm. *International Journal for Numerical Methods in Fluids*, 28(10):1391–1419, 1998.

[30] A.J. Chorin. Numerical solution of the Navier-Stokes equations. *Math. Comp.*, 22:745–762, 1968.

[31] M. Benzi, H. Choi, and D. Szyld. Threshold Ordering for Preconditioning Nonsymmetric Problems. In *G. Golub et al., editors, Sci. Comput., Proc. Workshop. Hong Kong*, pages 159–

165. Springer Verlag, 10–12 March 1997.

[32] S. W. Sloan. An algorithm for profile and wavefront reduction of sparse matrices. *Int. J. Numer. Meth. Engng.*, 23(2):239–251, 1986.

[33] T. Geenen, M. ur Rehman, S.P. MacLachlan, G. Segal, C. Vuik, A. P. van den Berg, and W. Spakman. Scalable robust solvers for unstructured FE geodynamic modeling applications: Solving the Stokes equation for models with large localized viscosity contrasts. *Geochemistry Geophysics Geosystems*, 10:1–12, 2009. doi:10.1029/2009GC002526.

[34] M. ur Rehman, T. Geenen, C. Vuik, G. Segal, and S.P. MacLachlan. On iterative methods for the incompressible Stokes problem. *Int. J. Numer. Meth. Fluids to appear*. doi: 10.1002/fld.2235.

[35] Y. Notay. Flexible Conjugate Gradients. *SIAM J. Sci. Comput.*, 22(4):1444–1460, 2000.

[36] H. A. van der Vorst. *Iterative Krylov Methods for Large Linear Systems*. Cambridge University Press, Cambridge, 2003.

[37] A. van der Sluis. Condition numbers and Equilibration of Matrices. *Numer. Math.*, 14:14–23, 1969.

[38] D. A. May and L. Moresi. Preconditioned iterative methods for Stokes flow problems arising in computational geodynamics. *Physics of the Earth and Planetary Interiors*, 171:33–47, 2008.

[39] A Wathen. Preconditioning and convergence in the right norm. *Int. J. Comput. Math.*, 84(8):1199–1209, 2007.

[40] M. Arioli and D. Loghin. Stopping criteria for mixed finite element problems. *ETNA*, 29:178–192, 2008.