INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS, VOL. 18, 1135–1151 (1994)

A PRECONDITIONED ALTERNATING INNER–OUTER ITERATIVE SOLUTION METHOD FOR THE MIXED FINITE ELEMENT FORMULATION OF THE NAVIER–STOKES EQUATIONS

S. Ø. WILLE

Department of Informatics, University of Oslo, PO Box 1080, Blindern, N-0316 Oslo 3, Norway

SUMMARY

In the present work a new iterative method for solving the Navier–Stokes equations is designed. In a previous paper a coupled node fill-in preconditioner for iterative solution of the Navier–Stokes equations proved to increase the convergence rate considerably compared with traditional preconditioners. The further development of the present iterative method is based on the same storage scheme for the equation matrix as for the coupled node fill-in preconditioner. This storage scheme separates the velocity, the pressure and the coupling of pressure and velocity coefficients in the equation matrix. The separation storage scheme allows for an ILU factorization of both the velocity and pressure unknowns. With the inner–outer solution scheme the velocity unknowns are eliminated before the resulting equation system for the pressures is solved iteratively. After the pressure unknown has been found, the pressures are substituted into the original equation system and the velocities are also found iteratively. The behaviour of the inner–outer iterative solution algorithm is investigated in order to find optimal convergence criteria for the inner iterations and compared with the solution algorithm for the original equation system. The results show that the coupled node fill-in preconditioner of the original equation system is more efficient than the coupled node fill-in preconditioner of the reduced equation system. However, the solution technique of the reduced equation system reveals properties which may be advantageous in future solution algorithms.

KEY WORDS Navier–Stokes Mixed formulation Bi-CGSTAB Incomplete LU Inner iterations

INTRODUCTION

The need for an efficient Navier-Stokes solver has been demonstrated in many research areas. Great efforts in research and development have been made in areas such as oceanography,[1–3] aerodynamics[4,5] and haemodynamics[6–8] as well as in general fluid dynamics. As progress has been made in designing efficient algorithms for generating meshes[2,9] around arbitrary bodies, attention has focused on designing large-scale algorithms for solving the Navier–Stokes equations.

For large problems the use of direct equation solvers is prohibitive owing to both the large storage required and the computationsl time necessary for solving the problem.[10] Iterative methods have advantages[10–15] compared with direct solvers, since only coefficients in the equation matrix different from zero are stored. However, the success of most iterative equation solvers seems to depend on using a good preconditioner.

Several iterative equation solvers for non-symmetric equation systems are available.[16] In this work the Bi-CGSTAB method of Van der Vorst[17,18] has been selected. Various preconditioning

CCC 0271–2091/94/121135–17

Received March 1993
Revised December 1993

methods for iterative equation solvers for flow problems have been subjected to extensive studies.[12–14,19] In the present work the linear equation solver has been tested with the ILU preconditioner with coupled fill-in for both Stokes and the Navier–Stokes equations. The mixed finite element formulation of the Navier–Stokes equations, first described by Taylor and Hood,[20] has been chosen. This mixed element method has certain advantages compared with the penalty method, since it does not introduce 'checker-board' pressure variations which could well occur with the penalty method.

In the present work the new incomplete LU preconditioner with coupled fill-in is designed specially for preconditioning the inner–outer formulation of the Navier–Stokes equations. This preconditioner permits fill-in at certain predefined locations in the pressure coefficient matrix rather than allowing fill-in when the size of the fill-in coefficients exceed a certain limit. Since the locations of the fill-in are predefined, the housekeeping during the factorization process is considerably reduced. These predefined locations are at the locations in the pressure matrix where the corner nodes are coupled. The corner nodes are coupled if they belong to the same finite element.

In the inner–outer iteration algorithm the inner iterations are preconditioned by incomplete LU factorization of the velocity matrix and the outer iterations are preconditioned by incomplete LU factorization of the coupled fill-in pressure matrix.

FINITE ELEMENT FORMULATION

The Stokes equations are linear and are given by

$$-\mu\nabla^2\mathbf{v} + \nabla p = 0 \quad \text{in } \Omega, \tag{1}$$

$$-\nabla\cdot\mathbf{v} = 0 \quad \text{in } \Omega, \tag{2}$$

where $\mathbf{v}$ is the velocity vector, p is the pressure and μ is the viscosity coefficient. The first equation is the equation of motion which contains a diffusion and a pressure gradient term. The second equation is the equation of continuity. A minus sign is introduced in the continuity equation in order to obtain the same sign for the pressure gradient as for the continuity equation in the finite element formulation. In the finite element formulation the velocities are approximated by quadratic polynomials and the pressure by linear polynomials to satisfy the Babuska–Brezzi condition. Denote the quadratic polynomials N_i and the linear polynomials L_i. Then by the Galerkin residual method and integration by parts the finite element formulation of the equation system becomes

$$\begin{aligned}\mathbf{F}_v &= \int_\Omega \mu\nabla N_i\cdot\nabla\mathbf{v}\,\mathrm{d}\Omega + \int_\Omega N_i\nabla p\,\mathrm{d}\Omega - \int_{\delta\Omega}\mu N_i\frac{\partial\mathbf{v}}{\partial n}\,\mathrm{d}\delta\Omega = 0,\\ \mathbf{F}_p &= -\int_\Omega L_i\nabla\cdot\mathbf{v}\,\mathrm{d}\Omega = 0.\end{aligned} \tag{3}$$

The non-linear Navier–Stokes equations are given by

$$-\mu\nabla^2\mathbf{v} + \mathbf{v}\cdot\nabla\mathbf{v} + \nabla p = 0 \quad \text{in } \Omega, \tag{4}$$

$$-\nabla\cdot\mathbf{v} = 0 \quad \text{in } \Omega, \tag{5}$$

where $\mathbf{v}$ is the velocity vector, p is the pressure and μ is the viscosity coefficient. In the Navier–Stokes equations an additional non-linear term is included in the equation of motion

compared with the Stokes equations. The finite element formulation of the Navier–Stokes equations is obtained similarly as for the Stokes equations:

$$\mathbf{F}_v = \int_\Omega \mu \nabla N_i \cdot \nabla \mathbf{v}\, d\Omega + \int_\Omega \rho N_i \mathbf{v} \cdot \nabla \mathbf{v}\, d\Omega + \int_\Omega N_i \nabla p\, d\Omega - \int_{\delta\Omega} \mu N_i \frac{\partial \mathbf{v}}{\partial n}\, d\delta\Omega = 0,$$
$$\mathbf{F}_p = -\int_\Omega L_i \nabla \cdot \mathbf{v}\, d\Omega = 0. \tag{6}$$

Instead of introducing a minus sign in the continuity equation, the pressure terms could be integrated by parts. A fully symmetric equation matrix could then be obtained for the Stokes equations. However, a different boundary condition

$$\int_{\delta\Omega} N_i \left(-p + \mu \frac{\partial \mathbf{v}}{\partial n} \right) d\delta\Omega \tag{7}$$

would then appear on the right-hand side. Since the pressure is not known on external boundaries, numerical difficulties are introduced in computing these pressure boundary conditions. The non-symmetry, however, in the present formulation is only occurring at those boundaries where the Dirichlet boundary conditions are not specified. For the Navier–Stokes equations the equation matrix is not symmetric anyway, so the slight non-symmetry in the pressure gradient continuity terms is probably of minor importance.

NEWTON FORMULATION

The Navier–Stokes equation has one non-linear term, the convective acceleration, which requires a non-linear iterative solution procedure. The non-linear formulation chosen is the Newton method, which is known to have a second-order convergence rate. The Navier–Stokes equations (6) then have to be differentiated and the linear equation system which has to be solved at each Newton step is

$$\begin{bmatrix} \partial \mathbf{F}_v^n / \partial \mathbf{v} & \partial \mathbf{F}_v^n / \partial p \\ \partial \mathbf{F}_p^n / \partial \mathbf{v} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x}_v \\ \mathbf{x}_p \end{bmatrix} = - \begin{bmatrix} \mathbf{F}_v^n \\ \mathbf{F}_p^n \end{bmatrix}. \tag{8}$$

If the initial solution $\mathbf{v}^0$, ρ^0 is chosen close enough to the final solution, convergence is guaranteed. The initial solution is chosen to be the Dirichlet boundary conditions where these are known and zero elsewhere. The solution is then updated at each Newton step with the correction found by solving (8):

$$\mathbf{v}^{n+1} = \mathbf{v}^n + \mathbf{x}_v, \tag{9}$$

$$\mathbf{p}^{n+1} = \mathbf{p}^n + \mathbf{x}_p. \tag{10}$$

The number of Newton steps to obtain a converged solution is usually of the order of 5–10 depending on the convergence criterion required.

The Newton method for non-linear equation systems can also be applied favourably to the linear Stokes equations. For linear equation systems only one Newton iteration is required. The advantage of a Newton formulation for linear equation systems appears when introducing the Dirichlet boundary conditions. The columns and rows in the equation matrix can then be zeroed with a one on the corresponding diagonal and a zero on the right-hand side. The Dirichlet value is included in the initial guess of the solution vector. The application of Dirichlet boundary conditions in the original equation matrix would be more complicated to maintain the

advantageous symmetric property, since the Dirichlet condition multiplied by the corresponding column vector has to be subtracted from the right-hand side.

STRUCTURE OF THE EQUATION MATRIX

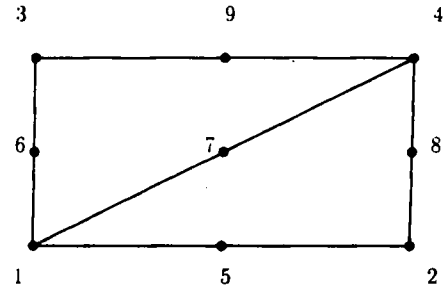
The finite element formulation of the system of equations (8) which is solved in the Newton process may be written in the form

$$\mathbf{Qx} = \mathbf{b}, \tag{11}$$

where

$$\mathbf{Q} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{0} \end{bmatrix} = \begin{bmatrix} \partial \mathbf{F}_v^n/\partial \mathbf{v} & \partial \mathbf{F}_v^n/\partial p \\ \partial \mathbf{F}_p^n/\partial \mathbf{v} & 0 \end{bmatrix}, \qquad \mathbf{x} = \begin{bmatrix} \mathbf{x}_v \\ \mathbf{x}_p \end{bmatrix}, \qquad \mathbf{b} = \begin{bmatrix} \mathbf{b}_v \\ \mathbf{b}_p \end{bmatrix} = - \begin{bmatrix} \mathbf{F}_v^n \\ \mathbf{F}_p^n \end{bmatrix}. \tag{12}$$

For the Stokes problem $\mathbf{x}_v$ and $\mathbf{x}_p$ contain the node values of the velocity components and pressure respectively. For the Navier–Stokes problem they contain the Newton corrections of the same quantities. the storage of the matrix **Q** is important when an iterative equation solver is used. For large problems only non-zero coefficients should be stored. Several storage schemes exist for storing sparse matrices. If the grid is regular, the different diagonals can be stored as one-dimensional vectors.[21] For irregular grids a more complex pointing structure is used to identify the rows and coefficients in the matrix used. When solving Navier–Stokes problems on irregular grids, a special storage scheme believed to have several advantages has been developed (Figures 1 and 2). Figure 1 shows the structure of the matrix **Q** for a simple two-dimensional grid consisting of two triangular elements. The upper part of the figure shows the grid and the enumeration of nodes. The corner nodes are numbered first. This way of numbering is obtained by the unstructured grid generation algorithm given by Wille[9] and is advantageous both in storing the matrix and during incomplete LU preconditioning. The indices above and to the left of the matrix shown in the lower part of Figure 1 refer to the node numbers. The equation matrix has non-zero coefficients at locations where two nodes are coupled in the grid. Two nodes are said to be coupled if they belong to the same finite element. For example, in the grid in Figure 1 nodes 1 and 5 are coupled while nodes 2 and 9 are not. The equation system is symmetric in shape. The matrix **A** contains the coefficients associated with the velocity degrees of freedom. The upper and lower parts of the submatrices of **A** are stored in separate one-dimensional vectors **U** and **L** as shown in Figure 2. This splitting is advantageous during the preconditioning, since the lower triangular part is accessed by columns and the upper triangular part is accessed by rows during factorization. The pointing structure has two pointing vectors; the first, PAC, points to where the node numbers for corresponding rows are stored in the other pointing vector, PAR. Let 'dim' be the spatial dimension for the sytem of differential equations; then the dimension of each of the submatrices $\mathbf{A}_{ij}$ is [dim × dim]. The position of each of these submatrices in both **U** and **L** is then easily calculated from the corresponding index in the vector PAR. Corresponding pointing structures are established for the matrices **B**, **C** and **P**. The matrix **P** is initially zero, but fill-in will occur during the ILU preconditioning. The dimension of $\mathbf{B}_{ij}$ is [dim × 1] and that of $\mathbf{C}_{ij}$ is [1 × dim]. The locations of the submatrices $\mathbf{B}_{ij}$ and $\mathbf{C}_{ij}$ are also easily calculated from the index in the pointing vector PBCR. The fill-in in the submatrices for the pressure $\mathbf{P}_{ij}$ consists of simple scalars. The pointing structure for the matrix **P** is equal to the first elements, which correspond to the pointing structure for the corner nodes of the matrices **B** and **C**. The pointing structure for the matrices **B** and **C** can then also be used for addressing the matrix **P**. The right-hand-side and solution vectors. $\mathbf{b}_v$ and $\mathbf{x}_v$ can



	1	2	3	4	5	6	7	8	9	1	2	3	4
1	A_{11}	A_{12}	A_{13}	A_{14}	A_{15}	A_{16}	A_{17}	A_{18}	A_{19}	B_{11}	B_{12}	B_{13}	B_{14}
2	A_{21}	A_{22}	0	A_{24}	A_{25}	0	A_{27}	A_{28}	0	B_{21}	B_{22}	0	B_{24}
3	A_{31}	0	A_{33}	A_{34}	0	A_{36}	A_{37}	0	A_{39}	B_{31}	0	B_{33}	B_{34}
4	A_{41}	A_{42}	A_{43}	A_{44}	A_{45}	A_{46}	A_{47}	A_{48}	A_{49}	B_{41}	B_{42}	B_{43}	B_{44}
5	A_{51}	A_{52}	0	A_{54}	A_{55}	0	A_{57}	A_{58}	0	B_{51}	B_{52}	0	B_{54}
6	A_{61}	0	A_{63}	A_{64}	0	A_{66}	A_{67}	0	A_{69}	B_{61}	0	B_{63}	B_{64}
7	A_{71}	A_{72}	A_{73}	A_{74}	A_{75}	A_{76}	A_{77}	A_{78}	A_{79}	B_{71}	B_{72}	B_{73}	B_{74}
8	A_{81}	A_{82}	0	A_{84}	A_{85}	0	A_{87}	A_{88}	0	B_{81}	B_{82}	0	B_{84}
9	A_{91}	0	A_{93}	A_{94}	0	A_{96}	A_{97}	0	A_{99}	B_{91}	0	B_{93}	B_{94}
1	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}	C_{16}	C_{17}	C_{18}	C_{19}	P_{11}	P_{12}	P_{13}	P_{14}
2	C_{21}	C_{22}	0	C_{24}	C_{25}	0	C_{27}	C_{28}	0	P_{21}	P_{22}	0	P_{24}
3	C_{31}	0	C_{33}	C_{34}	0	C_{36}	C_{37}	0	C_{39}	P_{31}	0	P_{33}	P_{34}
4	C_{41}	C_{42}	C_{43}	C_{44}	C_{45}	C_{46}	C_{47}	C_{48}	C_{49}	P_{41}	P_{42}	P_{43}	P_{44}

Figure 1. The upper part of the figure shows a simple two-dimensional grid consisting of two elements. In this grid the corner nodes are numbered first, then the mid-edge nodes. The structure of the corresponding equation matrix is shown below. The numbers at the top and to the left of the matrix are node numbers. The matrix P is initially zero and is used in the ILU preconditioning fill-in

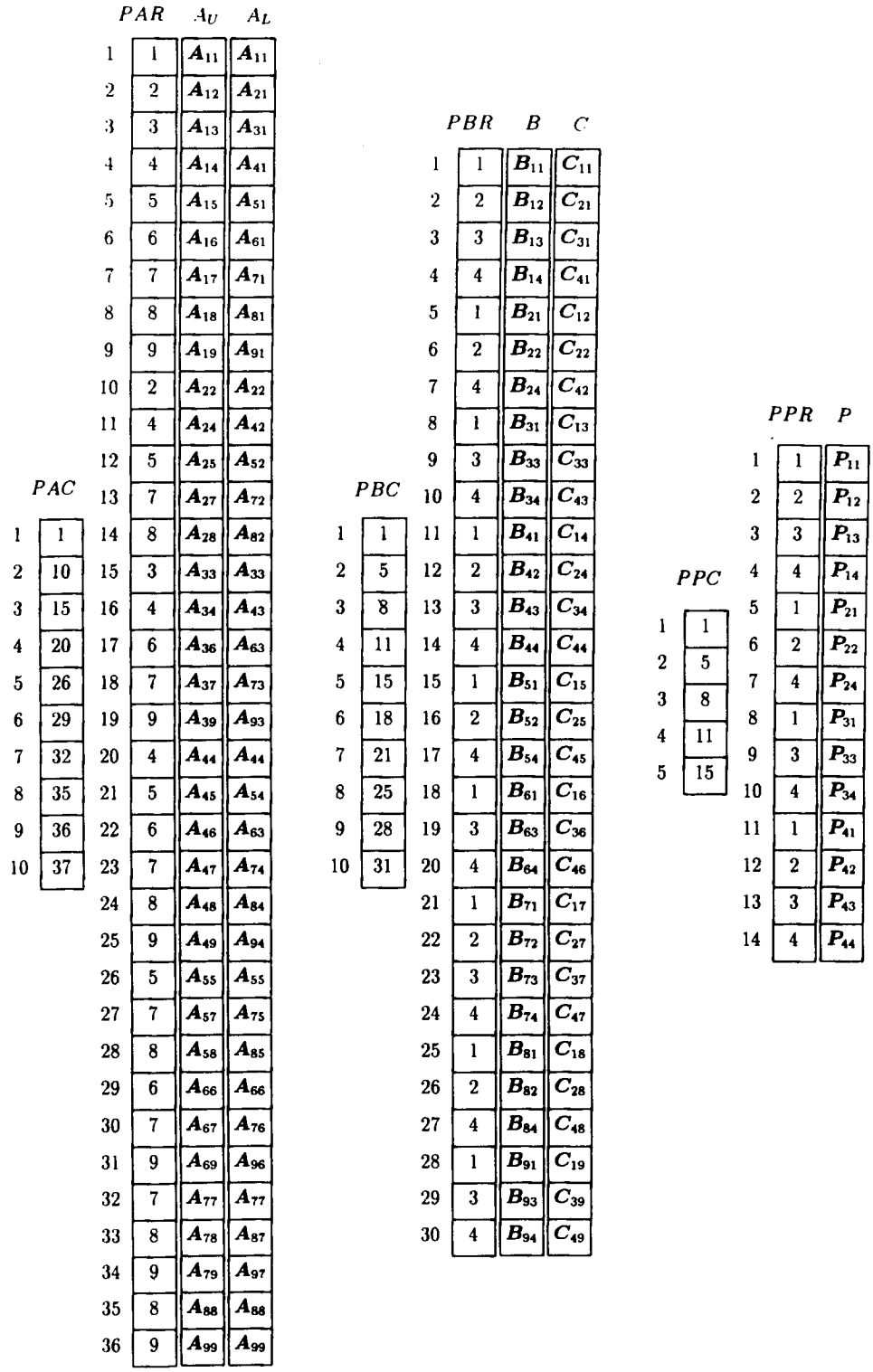


Figure 2. The equation matrix is stored as five one-dimensional vectors. The zero submatrices shown in Figure 1 are not stored. The storage structure requires two pointing vectors, one pointing to where each row begins in the vector which contains the nodes included in that row. The upper part **U** and the lower part **L** of **A** are stored in separate vectors. The same pointing structure can then be used for both **U** and **L**. Similarly, the same pointing structure is used for **B** and **C**. Note that the first part of the pointing structure for **B** and **C** can also be used for **P**

also be considered to consist of subvectors $\mathbf{b}_{vi}$ and $\mathbf{x}_{vi}$ with dimension [dim × 1]. These subvectors will represent the right-hand side and the solution at node i. The vectors $\mathbf{b}_p$ and $\mathbf{x}_p$ consist of scalars b_{pi} and x_{pi} which are the right-hand side of the continuity equation and the pressure at node i. All the submatrices $\mathbf{A}_{ij}$, $\mathbf{B}_{ij}$ and $\mathbf{C}_{ij}$ are stored row-by-row.

An important point to notice is that for the Navier–Stokes equations all the coefficients in the submatrices $\mathbf{A}_{ij}$ are non-zero. However, for the Stokes problem there are only non-zero coefficients on the diagonals of $\mathbf{A}_{ij}$. This implies that only the diagonals of $\mathbf{A}_{ij}$ need to be stored for the Stokes equations.

INNER–OUTER FORMULATION

For the linear Newton equation system (8) the equation system to be solved can be written as

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{x}_v \\ \mathbf{x}_p \end{bmatrix} = \begin{bmatrix} \mathbf{b}_v \\ \mathbf{b}_p \end{bmatrix}. \tag{13}$$

By multiplication the equivalent equation system is

$$\begin{aligned} \mathbf{A}\mathbf{x}_v + \mathbf{B}\mathbf{x}_p &= \mathbf{b}_v, \\ \mathbf{C}\mathbf{x}_v \qquad\quad &= \mathbf{b}_p. \end{aligned} \tag{14}$$

By elimination of $\mathbf{x}_v$ in the above equation ssytem the inner–outer formulation is obtained. First the pressure unknown is found from

$$\mathbf{C}\mathbf{A}^{-1}\mathbf{B}\mathbf{x}_p = \mathbf{C}\mathbf{A}^{-1}\mathbf{b}_v - \mathbf{b}_p. \tag{15}$$

Then solve for the velocity unknowns in the equation system

$$\mathbf{A}\mathbf{x}_v = \mathbf{b}_v - \mathbf{B}\mathbf{x}_p. \tag{16}$$

Both these equation systems can be solved by iterative equation solvers. For the first equation system (15) there will be both an outer and an inner iteration. In the inner iterations an equation system is also solved iteratively.

In all iterative equation solver algorithms a matrix–vector product has to be computed. The matrix–vector product

$$\mathbf{q} = \mathbf{C}\mathbf{A}^{-1}\mathbf{B}\mathbf{u} \tag{17}$$

for the pressure equation (15) is determined by first solving

$$\mathbf{A}\mathbf{w} = \mathbf{B}\mathbf{u} \tag{18}$$

and then calculating

$$\mathbf{q} = \mathbf{C}\mathbf{w}. \tag{19}$$

The inner equation system (18) is also solved iteratively. As seen from the Newton formulation (8), the original equation system is not positive definite. This will require high robustness of the linear iterative equation solver. However, positive definiteness is obtained in the reduced equation system (15).

LINEAR EQUATION SOLVER

In a previous paper[22] two linear equation solvers were tested. The truncated Orthomin method[23] with 10 search direction vectors and the Bi-CGSTAB method of Van der Vorst.[10,17,18] The latter method is a new varient of Bi-CG and has to some extent less irregular convergence behaviour than the CG-S method. When comparing the performance of Orthomin and CGSTAB, the CGSTAB method proved to require less work and was more stable than the Orthomin method for both the Stokes and the Navier–Stokes equations. In the present work only the CGSTAB method is therefore used.

Let $\|\cdot\|$ be the Euclidean norm. When solving the entire equation system $\mathbf{Qx} = \mathbf{b}$, the following convergence criterion for the linear equation solvers was used:

$$\frac{\|\mathbf{r}\|}{\|\mathbf{b}\|} < \varepsilon, \tag{20}$$

where $\mathbf{r} = \mathbf{b} - \mathbf{Q}\tilde{\mathbf{x}}$ is the residual of the computed approximate solution $\tilde{\mathbf{x}}$ of the system $\mathbf{Qx} = \mathbf{b}$.

For the non-linear Navier–Stokes equations the convergence criterion for the Newton iterations was

$$\frac{\|\Delta\mathbf{x}\|}{\|\mathbf{x}\|} < \varepsilon_n, \tag{21}$$

where $\Delta\mathbf{x}$ is the Newton update and $\mathbf{x}$ is the solution of the equation system.

In the inner–outer algorithm two convergence criteria for the iterative equation solver were used. For the inner iteration (18) $\mathbf{Aw} = \mathbf{Bu}$ the convergence criterion

$$\frac{\|\mathbf{r}_v\|}{\|\mathbf{Bu}\|} < \varepsilon_v, \tag{22}$$

where the residual $\mathbf{r}_v = \mathbf{Bu} - \mathbf{A}\tilde{\mathbf{w}}$ and $\tilde{\mathbf{w}}$ is the approximate solution to $\mathbf{Aw} = \mathbf{Bu}$, was used. For the outer iteration (15) the convergence criterion used was

$$\frac{\|\mathbf{r}_p\|}{\|\mathbf{CA}^{-1}\mathbf{b}_v - \mathbf{b}_p\|} < \varepsilon_p, \tag{23}$$

where the residual $\mathbf{r}_p = \mathbf{CA}^{-1}\mathbf{b}_v - \mathbf{b}_p - \mathbf{CA}^{-1}\mathbf{B}\tilde{\mathbf{x}}_p$ and $\tilde{\mathbf{x}}_p$ is the approximate solution to $\mathbf{CA}^{-1}\mathbf{Bx}_p = \mathbf{CA}^{-1}\mathbf{b}_v - \mathbf{b}_p$.

The non-linear Newton convergence criterion ε_n is the same as when iterating on the original equation system $\mathbf{Qx} = \mathbf{b}$.

PRECONDITIONING

Let $\mathbf{M}$ be a non-singular matrix. The original equation system $\mathbf{Qx} = \mathbf{b}$ can be replaced with $\mathbf{M}^{-1}\mathbf{Qx} = \mathbf{M}^{-1}\mathbf{b}$, where $\mathbf{M}^{-1}$ is the preconditioning matrix. The quality of the preconditioner depends very much on the choice of $\mathbf{M}$. The preconditioning matrix $\mathbf{M}$ should have the following properties: $\mathbf{M}$ is a good approximation to $\mathbf{Q}$, $\mathbf{M}$ is easily computed, $\mathbf{M}$ is reasonably sparse and equations of the form $\mathbf{Mx} = \mathbf{c}$ are easily solved.

The original equation system

Several preconditioners have been investigated[22] for preconditioning the equation system

$$\mathbf{Qx} = \mathbf{b}. \tag{24}$$

The preconditioners tested were modified diagonal preconditioning, SSOR preconditioning, ILU preconditioning of the velocity equations, ILU preconditioning with coupled fill-in for the entire equation system and ILU preconditioning with coupled fill-in for the symmetric part of the entire equation system. The results of these experiments showed that the ILU preconditioning with coupled fill-in for the symmetric part of the entire equation system was the most efficient. In the present work only this preconditioner was therefore used.

Incomplete LU factorization on the symmetric part of the complete matrix **Q**,

$$\mathbf{Q}^s = \frac{\mathbf{Q} + \mathbf{Q}^T}{2} \tag{25}$$

(see Figure 1), can be performed if certain fill-ins are accepted. These fill-ins are associated with the pressure and correspond to the matrix **P** given in Figure 1. Using the previous definition of coupled nodes, the submatrices $\mathbf{A}_{ij}$ are updated during the forward elimination of the matrix **A** if the two nodes ij are coupled in the element graph. The same philosophy is applied to the fill-ins in the pressure matrix **P**. Let nodes i and j be corner nodes; then the fill-in $\mathbf{P}_{ij}$ are accepted if the nodes ij are coupled. The general algorithm for forward and backward substitution is given in Dahl and Wille.[22] In this algorithm an inversion of the submatrices $\mathbf{A}_{kk}$ on the diagonal is included. As mentioned earlier, when the Stokes equations are considered, only the diagonals of these submatrices need to be stored. When using the Stokes equations as preconditioner, fill-in will never occur outside this diagonal. The inversion of the diagonal submatrices for the Stokes preconditioner will therefore only consist of inverting the diagonals in these submatrices. The preconditioning matrix of this preconditioner, ILU^s, then becomes

$$\mathbf{M} = \tilde{\mathbf{L}}^s\tilde{\mathbf{U}}^s. \tag{26}$$

In this work the matrix **Q** corresponding to the Stokes problem has been used when computing the preconditioning matrix **M**. Hence it has not been necessary to compute a new preconditioning matrix for each Newton step. Here the lower part of $\mathbf{Q}^s$, $\tilde{\mathbf{L}}^s$, has unit diagonal and it was observed that the diagonal elements of $\tilde{\mathbf{U}}^s$, corresponding to the zero block matrix in the lower right corner of $\mathbf{Q}^s$, were all negative and of the same order of magnitude. Let $\mathbf{M} = \mathbf{Q}^s$ and $\mathbf{M}^{-1}$ be the matrix obtained by incomplete factorization with coupled fill-in then the equation system $\mathbf{Qx} = \mathbf{b}$ can be preconditioned and the equation system to be solved is

$$\mathbf{M}^{-1}\mathbf{Qx} = \mathbf{M}^{-1}\mathbf{b}. \tag{27}$$

The reduced equation system

In the reduced equation system both the inner and the outer iteration can be preconditioned by a suitable matrix. Let **N** be the part of **A** corresponding to the Laplacian operator and $\mathbf{N}^{-1}$ be the matrix obtained by incomplete factorization of **N**. Then in the inner iteration the equation system

$$\mathbf{Aw} = \mathbf{Bu} \tag{28}$$

can be preconditioned with

$$\mathbf{N}^{-1}\mathbf{A}\mathbf{w} = \mathbf{N}^{-1}\mathbf{v}. \tag{29}$$

The outer equation system, the pressure equation (15), can be preconditioned similarly. Let $\mathbf{P}^{-1}$ be the matrix which is obtained by incomplete factorization with coupled fill-in in the pressure matrix. The preconditioned pressure equation system will then be

$$\mathbf{P}^{-1}\mathbf{C}\mathbf{A}^{-1}\mathbf{B}\mathbf{x}_p = \mathbf{P}^{-1}(\mathbf{C}\mathbf{A}^{-1}\mathbf{b}_v - \mathbf{b}_p). \tag{30}$$

By substituting the different preconditioners into equations (16) and (15), the equation system which is finally solved is

$$\mathbf{P}^{-1}\mathbf{C}\mathbf{A}^{-1}\mathbf{B}\mathbf{x}_p = \mathbf{P}^{-1}(\mathbf{C}\mathbf{A}^{-1}\mathbf{b}_v - \mathbf{b}_p), \tag{31}$$

$$\mathbf{N}^{-1}\mathbf{A}\mathbf{x}_v = \mathbf{N}^{-1}(\mathbf{b}_v - \mathbf{B}\mathbf{x}_p). \tag{32}$$

NUMERICAL RESULTS

The test problems for evaluating the different algorithms are shown in Figure 3. For the Stokes equations channel flow is considered and for the Navier–Stokes equations driven cavity flow is

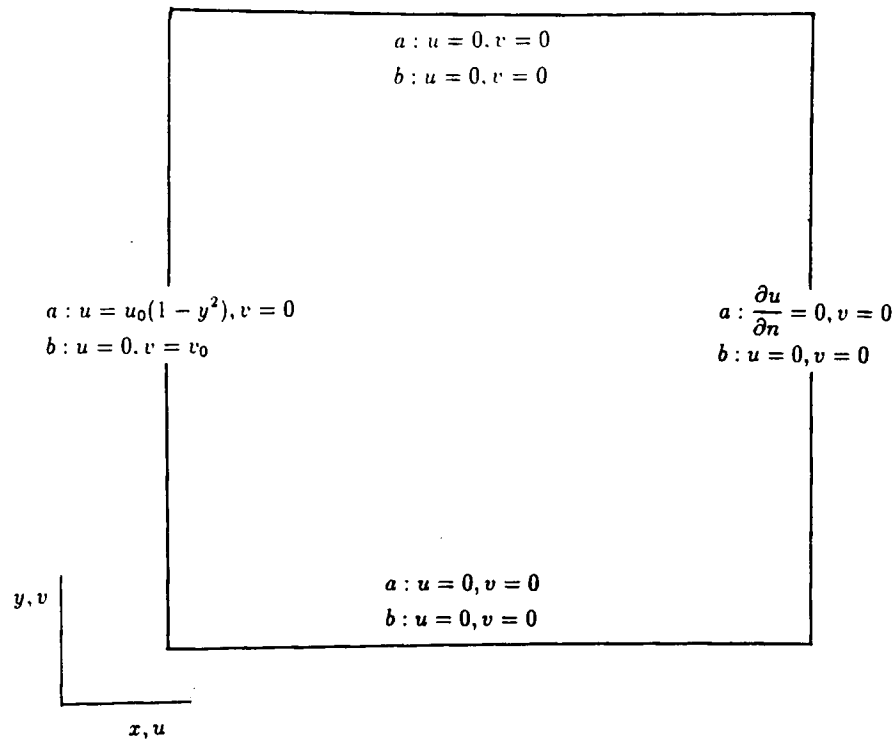


Figure 3. Boundary condtions for the two-dimensional test problems. The boundary contions 'a' are the boundary conditions for channel flow for the Stokes equations. The boundary conditions 'b' are the boundary conditions for the non-linear cavity flow problem described by the Navier–Stokes equations

Table I. Amounts of work, expressed as number of multiplications $\times 10^{-3}$, executed in the different parts of the algorithms for different size grids. The first column shows the grid size. The second column shows the work in the factorization of the preconditioning matrix $\mathbf{M}^{-1}$. This amount of work is the same as for factorizing both $\mathbf{N}^{-1}$ and $\mathbf{P}^{-1}$. The other columns show the work for the different matrix–vector multiplications and preconditioners

Grid	$\mathbf{M}^{-1}$ w_{PN}	$\mathbf{u} = \mathbf{Qv}$ w_Q	$\mathbf{u} = \mathbf{M}^{-1}\mathbf{v}$ w_M	$\mathbf{u} = \mathbf{Av}$ w_A	$\mathbf{u} = \mathbf{N}^{-1}\mathbf{v}$ w_N	$\mathbf{u} = \mathbf{CBv}$ w_{CB}	$\mathbf{u} = \mathbf{P}^{-1}\mathbf{v}$ w_P
4 × 4	58	5	5	3	3	0	0
8 × 8	225	16	19	12	13	4	0
16 × 16	892	68	74	48	52	20	1
32 × 32	3553	268	293	190	207	78	7
64 × 64	14176	1069	1167	757	824	312	29

computed. The boundary conditions for the different flow problems are also shown in Figure 3. The pressure has not been specified anywhere, or normalized in any other way in the grid, in order to get a unique solution. Iterative solvers do not need such a specification, since this is provided by the initial start vector. In all cases the start vector was $\mathbf{x}_0 = \mathbf{M}^{-1}\mathbf{b}$. In Table I the amounts of work for the different kinds of computational operations involved in the algorithms are given. The work is given in terms of the number of multiplications. The total work for solving the Stokes equations with the incomplete coupled fill-in is

$$w_t = w_{PN} + 2n(w_Q + w_M), \tag{33}$$

where n is the number of linear iterations. For the reduced equation system the total work is

$$w_t = w_{PN}\bar{n}_o + 2n_o[(w_P + w_{CB}) + 2n_i(w_N + w_A)] \tag{34}$$

where n_i is the number of inner iterations and n_o is the number of outer iterations; $\bar{n}_o = 1$ for the Stokes equations and $\bar{n}_o = n_o$ for the Navier–Stokes equations. When the reduced equation system is not preconditioned, $w_{PN} = w_P = w_N = 0$.

In Table II the numbers of iterations are given for solving the Stokes equations with different

Table II. Numbers of inner and outer iterations with and without preconditioning for solving the Stokes equations. In the second column the number of iterations with coupled node fill-in preconditioning of the original equation system is shown. The numbers of outer and inner iterations are given in the next columns. The number of inner iterations given is an average. However, the number of inner iterations for each outer iteration deviates only by a few iterations from the mean number of iterations. The convergence criterion is $\varepsilon_p = 10^{-4}$ for the outer iterations

Grid	Original preconditioned	Reduced, $\varepsilon_v = 10^{-5}$ preconditioned		Reduced, $\varepsilon_v = 10^{-6}$ not preconditioned	
		Outer	Inner	Outer	Inner
4 × 4	9	7	8	17	24
8 × 8	19	8	13	21	49
16 × 16	44	9	28	24	83
32 × 32	136	10	57	25	155
64 × 64	448	11	93	26	317

grid sizes. The second column shows the number of iterations by solving the original equation system. The third and fourth columns show the numbers of outer and inner iterations respectively when preconditioning the reduced equation system. In order to obtain convergence of the unpreconditioned reduced algorithm for the Stokes equations, the convergence criterion for the inner iterations had to be reduced by a factor of 10. In addition, a restart had to be performed during the inner iterations. The 16 × 16 grid was restarted every 20th iteration, the 32 × 32 grid was restarted every 40th iteration and the 64 × 64 grid was restarted every 80th iteration. For the preconditioned and unpreconditioned reduced algorithms the remarkable result is that on refining the grid by a factor of two, the number of outer iterations is only increased by one both for the preconditioned algorithm and on the three finest grids for the unpreconditioned algorithm. In Table III the amounts of work required by the three algorithms are given. It is seen that preconditioning of the original equation system is by far the most efficient algorithm in terms of work compared with the preconditioned reduced equation system algorithm. The preconditioned reduced algorithm is again more efficient than the unpreconditioned reduced algorithm.

In Table IV the numbers of iterations for obtaining convergence of the preconditioned reduced equation system are given for different convergence criteria of the inner iterations. These results

Table III. Total amounts of work, expressed as number of multiplications $\times 10^{-3}$ for the three methods, i.e. original system preconditioned, reduced system preconditioned and reduced system without preconditioning, for solving the Stokes equations in two dimensions

Grid	Original preconditioned	Reduced preconditioned	Reduced not preconditioned
4 × 4	148	1402	4096
8 × 8	1555	10689	50064
16 × 16	13388	102070	383424
32 × 32	158389	910413	2948900
64 × 64	2017632	6491130	24973000

Table IV. Numbers of inner and outer iterations for the Stokes equations with different convergence criteria ε_v for the inner iterations. The outer convergence criterion is $\varepsilon_p = 10^{-4}$ for all iterations. Both the outer and inner iterations are preconditioned. The number of inner iterations is an average. A 'dash' indicates that the solution did not converge

Grid	4 × 4		8 × 8		16 × 16		32 × 32		64 × 64	
ε_v	Outer	Inner	Outer	Inner	Outer	Inner	Outer	Inner	Outer	Inner
10^{-3}	8	5	—	9	—	20	—	38	—	69
10^{-4}	7	6	8	11	11	23	47	45	>100	87
10^{-5}	7	8	8	13	9	28	10	57	11	93
10^{-6}	7	9	8	15	9	32	10	62	11	116
10^{-7}	7	9	8	17	9	35	10	68	11	125
10^{-8}	7	10	8	19	9	37	10	74	11	138

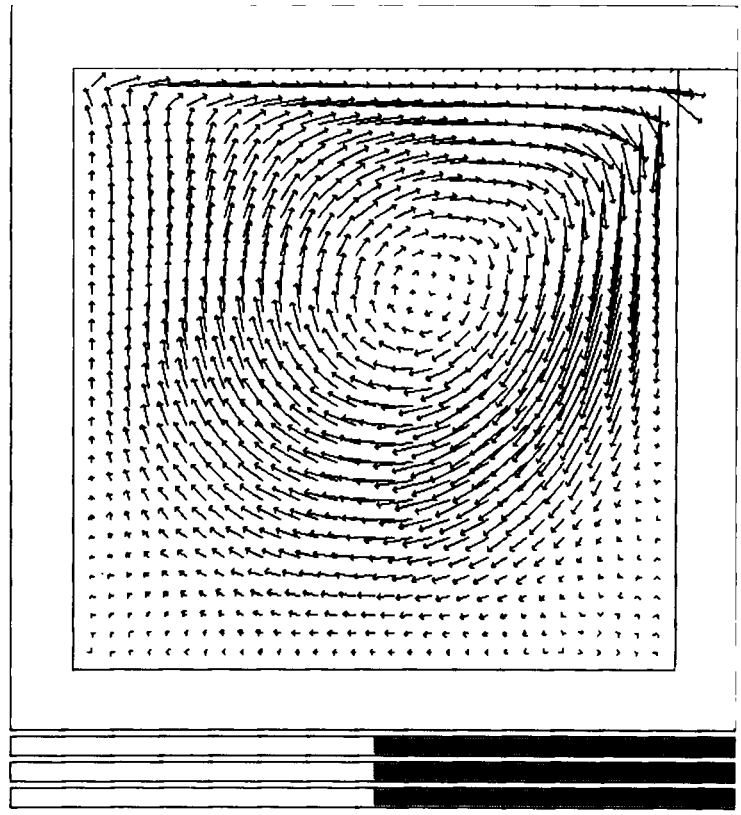


Figure 4. Solution of the Navier–Stokes problem for driven cavity flow in terms of velocity vectors for a 16 × 16 grid and Reynolds number 300

show that the outer iterations converge if the convergence criterion for the inner iterations is at least one order below the convergence criterion for the outer iterations. However, this relation between the two convergence criteria seems to be a sufficient condition for the Stokes equations. In Figure 4 the solution of the two-dimensional Navier–Stokes equations for the cavity flow is shown. In Table V the numbers of linear iterations are given for each Newton iteration for solving the Navier–Stokes equations. For comparison the numbers of outer and inner iterations for each Newton iteration for solving the reduced Navier–Stokes equations for the cavity problem are given in Table VI. When solving the reduced Navier–Stokes equations, the preconditioning using the Stokes equation matrix was not sufficient to obtain convergence. At each Newton step the preconditioning matrix used was equal to the equation matrix and a new incomplete factorization had to be performed at each step. The work used in recomputing the ILU factorization at all Newton steps is less than 10% of the total iterative work. In addition, to avoid stagnant solutions, the convergence criterion for the inner iterations had to be lowered to $\varepsilon_v = 10^{-7}$. For the 32 × 32 grid the inner iterations were restarted every 300th iteration and for the 64 × 64 grid restarts were performed every 1000th iteration. By inspection of the individual inner iterations it was observed for both the 32 × 32 grid and the 64 × 64 grid that for some inner iterations the restarts occurred too often and for some inner iterations too seldom. In some outer iterations the inner iterations were restarted as often as five times. The error

Table V. Numbers of iterations for solving the original Navier–Stokes equations with preconditioning in two dimensions for Reynolds number 300. For the original equation system the numbers of Newton and linear iterations are given. The convergence criterion for the Newton iterations is $\varepsilon_n = 10^{-4}$. The solution of the original equation system was restarted every 60th iteration

Grid	Newton	Linear					
8 × 8	6	16	45	42	53	64	49
16 × 16	6	60	71	95	117	100	96
32 × 32	6	107	334	222	278	240	318
64 × 64	6	375	824	1137	1390	516	978

Table VI. Numbers of iterations for solving the reduced Navier–Stokes equations with preconditioning in two dimensions for Reynolds number 300. For the reduced equation system the numbers of Newton, outer and inner iterations are given. The number of inner iterations is an average of the number of inner iterations for all outer iterations. For the inner iterations the convergence criterion was $\varepsilon_v = 10^{-7}$ and for the outer iterations $\varepsilon_p = 10^{-4}$

Grid	Newton	Outer					Inner
8 × 8	6	11	27	29	39	43	43	27
16 × 16	6	12	38	31	86	92	44	76
32 × 32	6	13	57	64	71	82	77	422
64 × 64	6	14	70	92	107	113	97	1311

history in the solution vector shows that this is too often. In other inner iterations the error history shows a stagnant solution before restarts occur. This way of performing restarts at constant restart intervals is obviously not an efficient restart algorithm. For the 32 × 32 grid this probably led to more inner iterations than were necessary to obtain convergence.

In Table VI the number of inner iterations is given as an average of the number of inner iterations for all outer iterations. The iterative equation solver was restarted every 300th iteration for the 32 × 32 grid and every 1000th iteration for the 64 × 64 grid. The high number of inner iterations for both the 32 × 32 and 64 × 64 grids is probably because these iterations were restarted too often. The amounts of work in solving the original and the reduced Navier–Stokes equations are given in Table VII. The method of solving the original equation system is again the most efficient.

Table VII. Total amounts of work, expressed as number of multiplications × 10^{-3}, for the original and reduced solution methods with preconditioning of the Navier–Stokes equations in two dimensions for Reynolds number 300

Grid	Original	Reduced
8 × 8	19	543
16 × 16	163	9211
32 × 32	1685	245223
64 × 64	23358	4094670

DISCUSSION

A different inner–outer algorithm where the pressure is solved by a preconditioned iterative method is presented by Vincent and Boyer.[24] Their preconditioner of the outer equation system is based on a stabilization matrix constructed from a 'penalty formulation' or 'local jump formulation'. The resulting preconditioning matrix is block diagonal and is inverted block-by-block. The inner equation system is solved by a Cholesky decomposition, although they suggest that it can be of major interest to solve the inner equation system iteratively. The construction and use of their preconditioner for the outer iterations require little work, but it is difficult to judge whether the total amount of work in solving the outer equation system is favourable compared the method presented here. However, one is tempted to believe that a preconditioner based on an aritificial 'penalty' or 'jump' matrix would exhibit poorer convergence properties.

In the present work a new preconditioned algorithm for solving the Navier–Stokes equations has been developed. This algorithm consists of inner–outer iterations. It has been shown that it is possible to solve both the Stokes and the Navier–Stokes equations with iterative equation solvers working inside each other. The solution method is shown to be speeded up by preconditioning. Both the inner and outer iterations are preconditioned by ILU factorization with coupled node fill-in. This way of solving and preconditioning the Navier–Stokes equations has been made possible by the special storage scheme, where the velocity coefficients, the pressure coefficient and the coupling between the pressure and velocity matrices are stored separately.

The results of these investigations show that the coupled node fill-in preconditioning algorithm of the original equation system[22] is superior to the preconditioned inner–outer algorithm. However, some interesting properties of the preconditioned inner–outer algorithm need to be pursued. When the grid is refined by a factor of two when solving the Stokes equations, it is rather remarkable that the number of outer iterations is only increased by one.

As the grid is refined when solving the Navier–Stokes equations, the numbers of outer iterations is reduced in two of the Newton iterations for the 32×32 grid compared with the 16×16 grid. For the 64×64 grid the increase in the number of outer iterations in the non-linear iterations is relatively small compared with the first Newton iteration. This is not unexpected, since the viscous term becomes more dominant compared with the convective term with decreasing size of the elements. The equation matrix $\mathbf{A}$ for the Newton formulation of the Navier–Stokes equations may be considered as the sum of a part $\mathbf{A}_L$ which consists of the Laplacian operator and a part $\mathbf{A}_P$ arising from the non-linear term of the differential system, i.e. $\mathbf{A} = \mathbf{A}_L + \mathbf{A}_P$. If h is a one-dimensional parameter describing the size of the elements, then we may consider $\mathbf{A}_P$ as a perturbation of order h of $\mathbf{A}_L$. Hence for small h we can expect the matrix $\mathbf{C}\mathbf{A}^{-1}\mathbf{B}$ to behave similarly to the corresponding matrix for the Stokes problem, $\mathbf{C}\mathbf{A}_L^{-1}\mathbf{B}$. The number of inner and outer iterations for the Navier–Stokes equations should then approach the number of outer iterations for the Stokes equations.

During the inner iterations it is important that the iterations are restarted only when necessary. If the iterations are restarted too often, all the information from the previous iterations is lost and the number of iterations is increased unnecessarily. If the iterations are restarted too seldom, the solution may become stagnant and superfluous iterations may be performed. The sufficient and necessary condition for performing a restart is that a restart should take place only when the solution becomes stagnant. Efforts are now being made to obtain a criterion for a stagnant solution and then to develop an automatic restart algorithm.

For the Stokes equations the inner iterations consist of solving an equation system with the Laplacian operator. An efficient means of solving an equation system consisting of the Laplacian operation has been shown to be multigrid methods.[25] In future work multigrid methods will

therefore be explored for solving the inner equation system for the Stokes equations and for either solving or preconditioning the inner equation system for the Navier–Stokes equations. By combining a multigrid method with preconditioned iterative equation solvers, it is believed that a very efficient implicit Navier–Stokes equation solver can be obtained.

ACKNOWLEDGEMENTS

The author is grateful to Olav Dahl for valuable discussions of the numerical methods and to Harald Svendsen for discussion of hydrodynamic problems. Knut Liestøl and Lars Gustav Snipen have been helpful with suggestions and discussions on the manuscript.

REFERENCES

1. G. Furnes, 'A three-dimensional numerical sea model with eddy viscosity varying piecewise linearly in the vertical', *Continental Shelf Res.*, **2**, 231–241 (1983).
2. R. R. Tharera, J. R. Stewart, O. Hassan, K. Morgan and J. Peraire, 'A point implicit unstructured grid solver for the Euler and Navier–Stokes equations', *Proc. AIAA 26th Aerospace Sciences Meet.*, Reno, NV, 1988, pp. 157–173.
3. T. Utnes, 'Finite element modeling of quasi-three-dimensional nearly horizontal flows', *Int. j. numer. methods fluids*, **12**, 559–576 (1991).
4. M. Jayaram and A. Jameson, 'An efficient vector and parallel multigrid method for viscous transonic flow', *Proc. Int. Symp. on High Performance Computing*, Montpellier, 1989, pp. 225–237.
5. T. J. R. Hughes, L. P. Franca and G. M. Hulbert, 'A new finite element formulation for computational fluid dynamics: VIII. The Galerkin/least-squares method for advective–diffusive equations', *Comput. Methods Appl. Mech. Eng.*, **73**, 173–189 (1989).
6. S. Ø. Wille, 'Numerical simulations of steady flow inside a three dimensional aortic bifurcation model', *J. Biomed. Eng.*, **6**, 49–55 (1984).
7. S. Ø. Wille and L. Walløe, 'Pulsatile pressure and flow in arterial stenosis simulated in a mathematical model', *J. Biomed. Eng.*, **3**, 17–24 (1981).
8. S. Ø. Wille, 'Pulsatile pressure and flow in arterial aneurysm simulated in a mathematical model', *J. Biomed. Eng.*, **3**, 153–158 (1981).
9. S. Ø. Wille, 'A structured tri-tree search method for generation of optimal unstructured finite element grids in two and three dimensions', *Int. j. numer. methods fluids*, **14**, 861–881 (1992).
10. H. A. Van der Vorst, 'Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems', *SIAM J. Sci. Stat. Comput.*, in press.
11. E. Barragy and G. F. Carey, 'A partitioning scheme and iterative solution for sparse bordered systems', *Comput. Methods Appl. Mech. Eng.* **70**, 321–327 (1988).
12. H. P. Langtangen, T. Rusten, A. Tveito and S. Ø. Wille. 'An element by element preconditioner for iterative equation solvers', *Proc. VI Int. Conf. on Finite Elements in Water Resources*, Vol 2, Lisbon, 1986, pp. 767–776.
13. S. Ø. Wille, 'An element by element preconditioner for refined finite element grids', *Proc. First Int. Conf. on Numerical Grid Generation in Computational Fluid Dynamics*, Vol 1, Landshut, 1986, pp. 449–460.
14. S. Ø. Wille, 'Local gaussian preconditioning of symmetric and non symmetric finite element equation solvers', *Proc. First Int. Conf. on Industrial and Applied Mathematics, ICIAM 87*, Paris, 1987, p. 23.
15. G. F. Carey, K. C. Wang and W. D. Joubert, 'Performance of iterative methods for Newtonian and generalized Newtonian flows', *Int. j. numer. methods fluids*, **9**, 127–150 (1989).
16. D. M. Young and K. C. Yea, 'Generalized conjugate-gradient acceleration of non-symmetrizable iterative methods', *Linear Algebra Appl.*, **34**, 159–194 (1980).
17. P. Sonneveld, 'CGS, a fast Lanczos-type solver for nonsymmetric linear systems', *SIAM J. Stat. Comput.*, **10**, 36–52 (1987).
18. H. A. Van der Vorst and P. Sonneveld, 'A more smoothly converging variant of CG-S', *Rep. 90–50*, Faculty of Technical Mathematics and Informatics, Delft University of Technology, 1990.
19. O. G. Johnson, C. A. Michelli and G. Paul, 'Polynomial preconditioners for conjugate gradient calculations', *SIAM J. Numer. Anal.*, **20**, 362–376 (1983).
20. C. Taylor and P. Hood, 'A numerical solution of the Navier–Stokes equations using the finite element technique', *Comput. Fluids*, **1**, 73–100 (1973).
21. A. George and J. W. Liu, *Computer Solutions of Large Sparse Positive Definite Systems*, Prentice-Hall, Englewood Cliffs, NJ, 1981.

22. O. Dahl, and S. Ø. 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. methods fluids*, **15**, 525–544 (1992).
23. P. K. W. Vinsome, 'Orthomin, an iterative method for solving sparse sets of simultaneous linear equations', *Proc. Fourth Symp. on Reservoir Simulation, Society of Petroleum Engineers of AIME*, Dallas, 1976, pp. 147–159.
24 C. Vincent and R. Boyer, 'A preconditioned conjugate gradient Uzawa-type method for the solutin of the Stokes problem by mixed Q1–P0 stabilized finite elements', *Int. j. numer. methods fluids*, **14**, 289–298 (1992).
25. W. Hackbush, *Multi-grid Methods and Applications*, Springer, Berlin, 1985.