INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS, VOL. 16, 507–523 (1993)

SOLUTION OF THE DISCRETIZED INCOMPRESSIBLE NAVIER–STOKES EQUATIONS WITH THE GMRES METHOD

C. VUIK

Faculty of Technical Mathematics and Informatics, Delft University of Technology, P.O. Box 5031, Mekelweg 4, 2600 GA Delft, The Netherlands

SUMMARY

We describe some experiences using iterative solution methods of GMRES type to solve the discretized Navier–Stokes equations. The discretization combined with a pressure correction scheme leads to two different systems of equations: the momentum equations and the pressure equation. It appears that a fast solution method for the pressure equation is obtained by applying the recently proposed GMRESR method, or GMRES combined with a MILU preconditioner. The diagonally scaled momentum equations are solved by GMRES(m), a restarted version of GMRES.

KEY WORDS Navier–Stokes equations Incompressible boundary-fitted co-ordinates Non-symmetric linear systems Iterative solver

1. INTRODUCTION

In Reference 17 a numerical discretization of the incompressible Navier–Stokes equations in general curvilinear co-ordinates is treated. The space discretization consists of a finite volume technique on a structured grid. The motivation for these choices is that we want to solve large two- and three-dimensional problems. In these problems it is important to obtain fast iterative methods to solve the discretized equations. This is easier using a finite volume technique instead of a finite element technique. Finally, the structured grid enables us to develop a good implementation of the methods on vector computers.

In this paper we present some experiences with iterative solution methods applied to the equations given in Reference 17. We opt for iterative methods because we want to solve large two-dimensional problems and in the near future large three-dimensional problems. Fast iterative methods to solve a system of linear equations are: multigrid methods and Krylov subspace methods. In References 11 and 14 the discretized incompressible Navier–Stokes equations are solved with a multigrid method. We consider Krylov subspace methods. Since the matrices are non-symmetric, we are not able to use the conjugate gradient or conjugate residual method. So we use the GMRES method, which is a robust method to solve non-symmetric problems and has an optimal rate of convergence. A comparison of GMRES-type methods with CGS[3] and Bi-CGSTAB is a point of current research.

Other iterative methods to solve the incompressible Navier–Stokes equations are:

1. the SIMPLE method (semi-implicit method for pressure-linked equations),[12,13]
2. the distributive Gauss–Seidel smoothing method,[2,6]

0271–2091/93/060507–17$13.50

Received 11 October 1991
Revised 5 November 1992

3. the symmetric coupled Gauss–Seidel method,[19]
4. the distributive ILU smoothing method.[27]

For more details on these methods used in combination with the multigrid method, we also refer to Wesseling.[26]

The solution of the pressure equation is discussed in Section 3. It appears that GMRES should be combined with a preconditioner to obtain a fast iterative method. In Section 4 we use the GMRES method to solve the momentum equations. We show that a correct scaling of the equations is important. Furthermore, we describe a termination criterion, which is combined with the GMRES method. Finally, we end this paper with conclusions in Section 5.

2. STATEMENT OF THE PROBLEM

In this section we specify the incompressible Navier–Stokes equations and outline the discretization of these equations in time and space. Subsequently, we describe the geometry and initial and boundary conditions for a test problem, which is used in the remainder of this paper.

We consider the flow of an incompressible fluid in a two-dimensional configuration. In Reference 17 the Navier–Stokes equations, which can be used to describe this flow, are formulated in general co-ordinates. For the sake of simplicity we describe these equations in Cartesian co-ordinates

$$\frac{\partial u_i}{\partial t}-\left(\frac{\partial \tau_{i1}}{\partial x_1}+\frac{\partial \tau_{i2}}{\partial x_2}\right)+\frac{\partial u_i u_1}{\partial x_1}+\frac{\partial u_i u_2}{\partial x_2}+\frac{\partial p}{\partial x_i}=0, \quad i\in\{1,2\}, \tag{1}$$

where

$$\tau_{ii}=Re^{-1}\left(\frac{4}{3}\frac{\partial u_i}{\partial x_i}-\frac{2}{3}\frac{\partial u_j}{\partial x_j}\right), \qquad \tau_{ij}=Re^{-1}\left(\frac{\partial u_i}{\partial x_j}+\frac{\partial u_j}{\partial x_i}\right), \qquad i,j\in\{1.2\}, \quad i\neq j,$$

together with the incompressibility condition

$$\frac{\partial u_1}{\partial x_1}+\frac{\partial u_2}{\partial x_2}=0, \tag{2}$$

and appropriate initial and boundary conditions. In these equations u_i is the component of the velocity of the fluid in x_i-direction, p is the pressure and Re is a parameter called the Reynolds number.

In the time discretization, finite differences are used. In this test problem we use an equidistant time discretization. We note that the discretization described in Reference 17 is not necessarily equidistant. For a given positive integer N we define $k=T/N$. In the following v^n denotes the numerical approximation of $v(nk)$. Using the pressure correction method[8] we obtain the following equations:

$$\frac{\hat{u}_i^{n+1}-u_i^n}{k}-\left(\frac{\partial \hat{\tau}_{i1}^{n+1}}{\partial x_1}+\frac{\partial \hat{\tau}_{i2}^{n+1}}{\partial x_2}\right)+\frac{\partial \hat{u}_i^{n+1}\hat{u}_1^{n+1}}{\partial x_1}+\frac{\partial \hat{u}_i^{n+1}\hat{u}_2^{n+1}}{\partial x_2}+\frac{\partial p^n}{\partial x_i}=0, \quad i\in\{1,2\}, \tag{3}$$

$$\frac{\partial^2 p^{n+1}}{\partial x_1^2}+\frac{\partial^2 p^{n+1}}{\partial x_2^2}=\frac{\partial^2 p^n}{\partial x_1^2}+\frac{\partial^2 p^n}{\partial x_2^2}+\frac{1}{k}\left(\frac{\partial \hat{u}_1^{n+1}}{\partial x_1}+\frac{\partial \hat{u}_2^{n+1}}{\partial x_2}\right) \tag{4}$$

and, finally,

$$u_i^{n+1}=\hat{u}_i^{n+1}+k\frac{\partial(p^n-p^{n+1})}{\partial x_i}, \quad i=1,2. \tag{5}$$

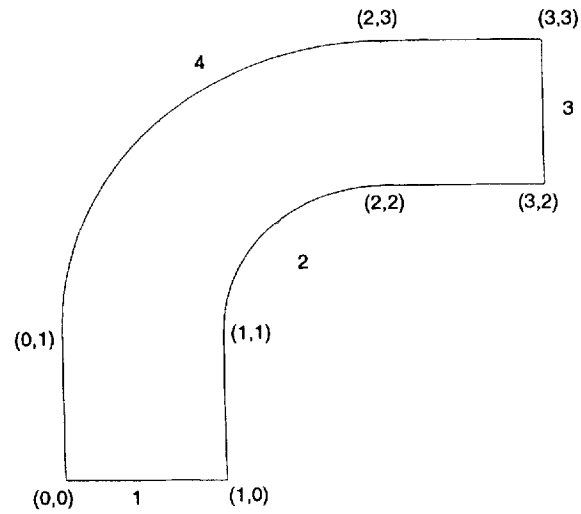

Figure 1. The physical domain of the test problem

The non-linear terms $\hat{u}_i^{n+1}\hat{u}_j^{n+1}$ appearing in (3) are linearized (Newton linearization) as follows:

$$\hat{u}_i^{n+1}\hat{u}_j^{n+1} \cong \hat{u}_i^{n+1}u_j^n + u_i^n\hat{u}_j^{n+1} - u_i^n u_j^n \quad \text{with } i,j \in \{1,2\}.$$

For the discretization in space the physical domain is mapped onto a rectangle (computational domain). Combining this co-ordinate transformation with finite volumes on a staggered grid in the computational domain, we obtain a space discretization of the equations (3), (4) and (5). The operator div·grad in the pressure equation (4) is discretized by $\text{div}_h \cdot \text{grad}_h$, where the same operators div_h and grad_h are used in the discretization of the momentum equations (3). For more details we refer to Reference 17.

The resulting equations can be divided into two linear systems. The first system, the momentum equations, is a discretized version of (3), whereas the second system, the pressure equation, is a discretized version of (4). The space discretization is such that it is possible to implement algorithms, for building the matrices and solving the systems of equations, which are suitable for vector and parallel computers.

In the remainder of this paper we consider one test problem, which describes the flow through a curved channel. Our experiments with this problem give valuable insight into the behaviour of GMRES solving the pressure and the momentum equations. The physical domain of the problem is displayed in Figure 1. Initially, the velocities are equal to zero. The boundary conditions are: a parabolic velocity profile, with the maximal velocity equal to one, on the inflow boundary (Boundary 1), a no-slip condition on Boundary 2 and 4 and the normal stress and tangential velocity given on the outflow boundary (Boundary 3). We chose $Re = 500$, $T = 0{\cdot}3$ and $N = 2$. For the space discretization the two squares (see Figure 1) are divided into 16×16 finite volumes, whereas the curve is divided into 16×32 finite volumes. The total number of finite volumes is $16 \times 64 = 1024$.

3. THE PRESSURE EQUATION

In this section we start with a specification of the discretized pressure equation. After that we describe the GMRES method, which is used to solve this equation. We obtain valuable insights

from the convergence properties of full GMRES. However, full GMRES is expensive with respect to computing time and memory requirements. A fast iterative method is obtained by a combination of GMRES with a polynomial or an ILU preconditioner. We end the section with a remark on the memory requirements of this iterative solution method.

The physical domain of our test problem is the curved channel of Figure 1. The computational domain, which is used in the discretization of the pressure equation (4) is a rectangle, of which the edges are parallel to the co-ordinate axes. The orientation of the computational domain is such that $n_1 \leqslant n_2$, where n_i is the number of grid points in the x_i-direction.

We use a lexicographical ordering of the grid points, so the difference between the indices of neighbouring grid points equals 1 in the x_1-direction and n_1 in the x_2-direction. The structure of the matrix $P \in \mathbb{R}^{(n_1 n_2)\times(n_1 n_2)}$ used in the discretized pressure equations is given in Figure 2. The curvilinear grid combined with a co-ordinate transformation results in a non-symmetric pressure matrix P. With considerable effort, it is possible to make the matrix P symmetric in the interior region; however, not in the vicinity of the boundary. This is a disadvantage of the finite volume method, since using a finite element technique one obtains a symmetric pressure matrix P.

In this paper we solve the pressure equation $Px=b$ using the GMRES(m) method (see Reference 15). In the GMRES method the vector z_k is chosen such that

$$z_k = \arg \min_{z \in K^k(P;r_0)} \| b - P(x_0+z) \|_2, \tag{6}$$

where $r_0 = b - Px_0$ and the Krylov subspace $K^k(P;r_0)$ is defined by

$$K^k(P;r_0) = \text{span}\ \{r_0, Pr_0, \ldots, P^{k-1}r_0\}.$$

In the following experiments we always take $x_0 = (0, \ldots, 0)^T$. After m iteration steps the method restarts using $x_m = x_0 + z_m$ as start vector. For an implementation of this method we refer to References 15 and 22.

GMRES(50)

Using the insights given in Reference 7, we choose the integer $m \in [5, 50]$ as large as possible with respect to memory requirements. In this example there are $16 \times 64 = 1024$ grid points, and it

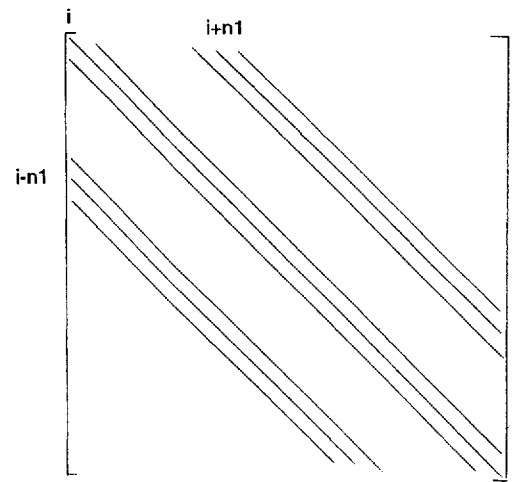

Figure 2. The pressure matrix P

turns out to be possible to take m equal to 50. After 1220 iterations and 17 s CPU time measured on one processor of a Convex C240, we obtain $\| r_{1220} \|_2 / \| r_0 \|_2 \leqslant 10^{-6}$. This is a bad result, which motivates us to analyse the convergence behaviour of the GMRES method in this application. To facilitate comparison we specify the amount of work and memory in Table I.

Full GMRES

In the following experiment we solve the pressure equation with full GMRES, so, the GMRES method is not restarted. It appears that $\| r_{184} \|_2 / \| r_0 \|_2 \leqslant 10^{-6}$. This is a better result with respect to computing time but now the amount of required memory is (too) large (see Table I).

Although full GMRES uses too much memory in this application, the results of this experiment give valuable information. From the local convergence behaviour (see Figures 3 and 4) it appears that in the first 100 iterations, the residual remains nearly the same. Taking into account the analysis of the convergence behaviour of GMRES given in References 7 and 23, we conclude that it takes more than 100 iteration before the Ritz values approximate the eigenvalues, which results

Table I. The amount of work, memory and CPU time

Method	Matvec	Vector update	Inner product	Memory vectors	CPU time (s) (Convex)	Precondition matvec
GMRES(50)	1220	35000	35000	50	17	0
Full GMRES	184	17000	17000	184	7·2	0
GMRESR	198	1386	1224	46	1·2	0
ILU	53	1405	1405	53	1·2	53
MILU ($\alpha = 0{\cdot}95$)	31	481	481	31	0·6	31

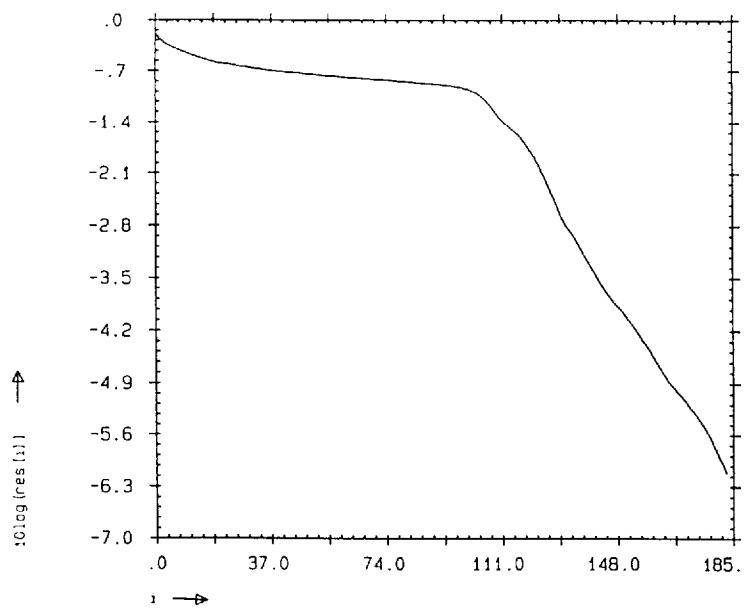

Figure 3. Full GMRES

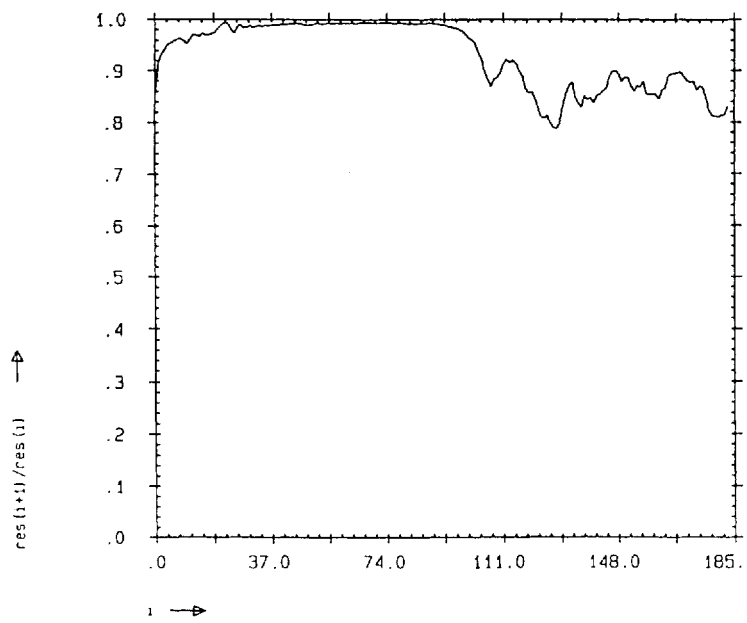

Figure 4. Full GMRES

in a super linear convergence behaviour of GMRES in a later phase. This explains that restarting GMRES is a bad idea in this application, because if we choose m less than 100, GMRES is only linearly convergent, and the reduction factor is nearly 1. This agrees with the results obtained with GMRES(50).

Polynomial preconditioning

From these experiments it follows that a preconditioner is necessary to get an acceptable computing time and a reasonable amount of memory required. Since a polynomial preconditioner has good vectorization properties, we start with such a preconditioner. A combination of GMRES with a polynomial preconditioner is presented in References 10 and 16.

Using GMRES combined with a polynomial preconditioner, we cannot expect to need fewer matrix vector products than using full GMRES, because GMRES has the minimal residual property (6). So, the main purpose of a polynomial preconditioner is to lower the number of vector updates and inner products, and the amount of required memory.

In this paragraph we describe a polynomial preconditioner, where the polynomial is adapted in every iteration. We get the idea for this preconditioner from the EN method as given in Reference 4 and analysed in Reference 25. In the EN method one approximates P^{-1} in every iterate by the polynomial $(I-P)$. We propose to approximate P^{-1} by a polynomial of higher degree, which is obtained by a call of full GMRES. The resulting method, which we denote by GMRESR, is given as follows (cf. Reference 24):

1. $u_0 = H_0 r_0 / \| PH_0 r_0 \|_2$, $c_0 = Pu_0$, $k=0$,
   $x_1 = x_0 + u_0 c_0^{\mathrm{T}} r_0$ and $r_1 = r_0 - c_0 c_0^{\mathrm{T}} r_0$;

2. while $\| r_{k+1} \|_2 > \text{eps}$ do $k := k+1$,

$$c_k^{(0)} = PH_k r_k,\ u_k^{(0)} = H_k r_k$$

$$\beta_i = c_i^T c_k^{(i)},\ c_k^{(i+1)} = c_k^{(i)} - \beta_i c_i,\ u_k^{(i+1)} = u_k^{(i)} - \beta_i u_i, \quad i = 0, \ldots, k-1,$$

$$c_k = c_k^{(k)} / \| c_k^{(k)} \|_2,\ u_k = u_k^{(k)} / \| c_k^{(k)} \|_2,$$

$$x_{k+1} = x_k + u_k c_k^T r_k \text{ and } r_{k+1} = r_k - c_k c_k^T r_k,$$

end.

In this algorithm $H_k r_k$ denotes the solution of the system $Py_k = r_k$ with mpost itertions of full GMRES.

Application of GMRESR with mpost = 10 gives $\| r_{18} \|_2 / \| r_0 \|_2 \leqslant 10^{-6}$. Since in every iteration of GMRESR we use 11 matrix vector products, we obtain the solution with 198 matrix vector products. Comparing this with GMRES(50) and full GMRES, we conclude that GMRESR has a favourable convergence behaviour in this application.

It follows from Table I that the number of vector updates and inner products shows a considerable reduction, with respect to full GMRES. These observations explain the large reduction of the computing time.

With respect to memory requirements, we note that, using m iterations of GMRESR, we need $2m + \text{mpost}$ vectors in memory. In this experiment $m = 18$ and mpost = 10; so, we need 46 vectors in memory, which is comparable with GMRES(50).

Note that both aims of polynomial preconditioning are achieved: reduction of the vector updates and inner products and a reduction of the memory required. The extra cost of this approach, 14 extra matrix vector multiplications, is negligible.

Application of GMRESR with mpost = 15 or 20 gives comparable results. So, the efficiency of the algorithm is not very sensitive to variation of mpost.

ILU preconditioning

Another successful preconditioning technique is to construct an incomplete $LD^{-1}U$ decomposition of P and to solve the system $U^{-1}DL^{-1}Px = U^{-1}DL^{-1}b$ instead of $Px = b$ (see References 9 and 20). To implement this preconditioning we use the following rules to obtain L, D and U:[20]

(a) $\text{diag}(L) = \text{diag}(U) = D$;
(b) the off-diagonal parts of L and U are equal to the corresponding parts of P;
(c) $\text{diag}(LD^{-1}U) = \text{diag}(P)$.

Note that this preconditioning needs only one extra vector in memory to store the diagonal matrix D.

Combination of the ILU preconditioner with full GMRES gives $\| r_{53} \|_2 / \| r_0 \|_2 \leqslant 10^{-6}$. The resulting computing time and memory requirements are comparable with GMRESR (see Table I). Note that in contrast to GMRESR we have not yet implemented a vectorizable version of the ILU preconditioning. Using such a version the computing time should be less than the computing time using GMRESR.

MILU preconditioning

Finally, we use the MILU preconditioning specified in Reference 5. In this preconditioning the construction of D is such that the rowsum of $LD^{-1}U$ equals the rowsum of P. In the following

Table II. Number of iterations for different α

α	0	0·9	0·925	0·95	0·98	1
Iterations	53	33	32	31	40	48

experiments, we use an average of the ILU and the MILU preconditioner.[1,22] The ILU preconditioner corresponds with $\alpha = 0$, whereas the MILU preconditioner corresponds with $\alpha = 1$.

The number of iterations of MILU combined with full GMRES such that $\| r_i \|_2 / \| r_0 \|_2 \leqslant 10^{-6}$ is given in Table II for different choices of α. From Table I it appears that this method, with $\alpha = 0{\cdot}95$, is optimal with respect to the amount of work, required memory and computing time. It appears from our experiments that $\alpha = 0{\cdot}95$ is a good choice for many different problems. Note that the matrix P only depends on the discretization of the physical domain. In this application combination of MILU and polynomial preconditioning gives only a small reduction of the computing time and memory requirements.

Memory requirements

From these experiments, it appears that GMRES combined with MILU preconditioning needs 31 vectors in memory. This is a large amount of memory with respect to the memory requirements of for instance CGS.[18] However, the momentum equations are built every timestep so the memory required to store and solve the momentum equations can be used to solve the pressure equations. In Section 4 we show that the momentum matrix consists of 13 vectors, which are twice as long as the vectors used in the pressure equations. So, we do not need much extra memory to solve the pressure equations using full GMRES combined with a MILU preconditioner.

4. THE MOMENTUM EQUATIONS

In this section we specify the momentum equations. An application of full GMRES shows that the momentum matrix is 'nearly' singular. It appears that this is a consequence of a wrong scaling of the equations. We show that a diagonally scaled version of the momentum matrix has much better properties. Thereafter we discuss some termination criteria. Finally, the section is concluded by some experiments with the GMRES(m) method.

The discretized momentum equations (see equation (3)) are denoted by $Mx = b$, where $M \in \mathbb{R}^{(2n_1n_2)\times(2n_1n_2)}$ and $x, b \in \mathbb{R}^{2n_1n_2}$. The structure of the matrix M is given in Figure 5. From equation (3) it follows that the matrix M depends on the space and the time discretization, the Reynolds number and the velocity of the fluid in the preceding timestep. This implies that, in general, M is different in every timestep, therefore, in the following the momentum matrix in the nth timestep is denoted by M_n. From our experiments it appears that $M_1 x = b$ can be solved with a small number of GMRES iterations. In order to give an explanation, we note that the initial condition and the choice of Re imply that M_1 is approximately equal to a scalar times, the identity matrix. It is known that GMRES converges fast for such a linear system. Since it appears that the convergence behaviour of GMRES applied to $M_n x = b$ is more or less the same for every $n \geqslant 2$, we consider the system $M_2 x = b$ in the remainder of this section.

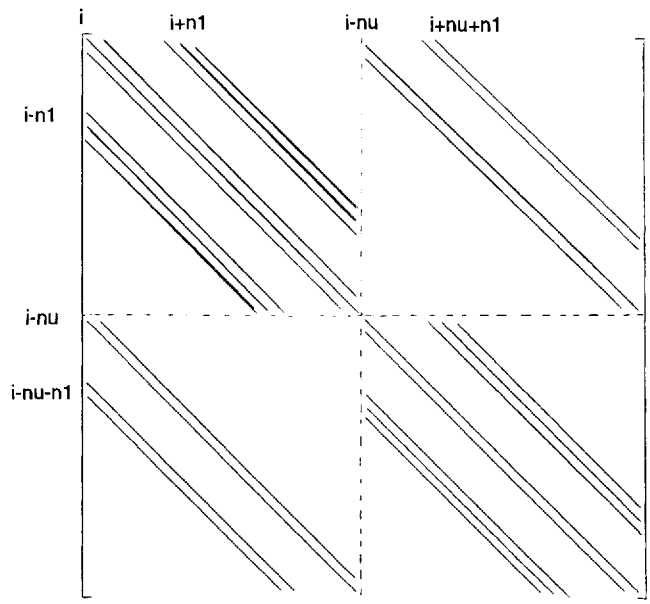

Figure 5. The momentum matrix M, where $nu = n_1 \cdot n_2$

Full GMRES

As in Section 3 we apply full GMRES to the momentum equations to obtain insights into its convergence properties. To solve $M_2 x = b$, we start GMRES with $x_0 = 0$ and stop when $\| r_i \|_2 \leqslant 10^{-8}$. In this experiment GMRES satisfies the stopping criterion after 135 iterations. The converge history is given in Figure 6. Note that the convergence stagnates from iteration 50 until iteration 70. To obtain more insight into this stagnation phase we calculate the Ritz values, which are plotted in the Figures 7–9. From these figures, and the analysis given in Reference 23 we conclude the following: It appears from Figure 9 that there is a 'small' eigenvalue, which means that its modulus is small with respect to the moduli of the other eigenvalues. Initially, the component in the corresponding eigenvector is small, so, there is no 'small' Ritz value in Figure 7 and the convergence is fast (see Figure 6). However, in the stagnation phase the process discovers that ther is a 'small' eigenvalue (Figure 8). After iteration 70, the 'small' Ritz value is converged (Figure 9) and GMRES converges as if the 'small' eigenvalue is absent, which corresponds with the results given in Reference 23.

Where does this 'small' eigenvalue come from? In the discretization the Dirichlet boundary conditions are included as equations in the system $M_2 x = b$. However, these extra equations are not scaled with respect to the other ones. As a result of this, the main diagonal elements in these equations, which are equal to one, differ by a factor 10^3–10^4 with the other non-zero main diagonal elements. To get rid of these 'small' eigenvalues we propose the following approaches:

1. do not include the Dirichlet boundary conditions as extra equations,
2. take the start vector x_0 such that it already satisfies the extra equations,
3. scale the momentum equations with the inverse of the non-zero main diagonal elements.

In the following paragraphs we analyse these approaches in more detail.

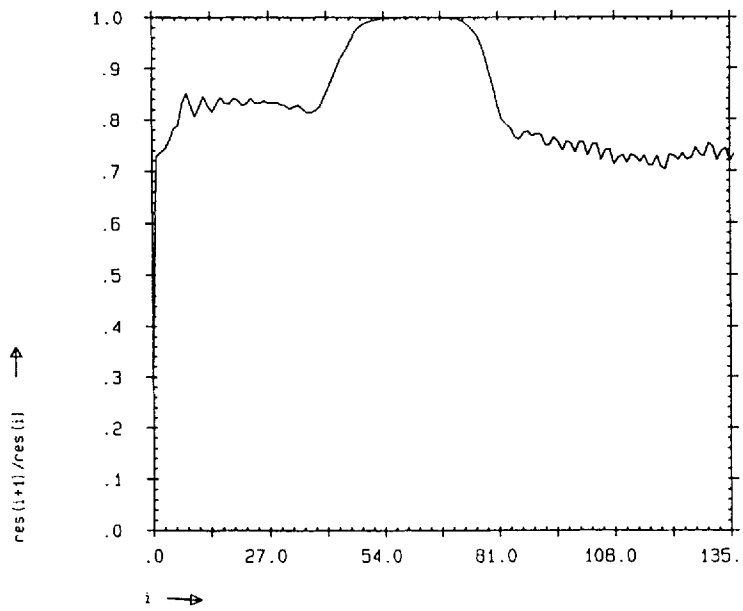

Figure 6. Full GMRES

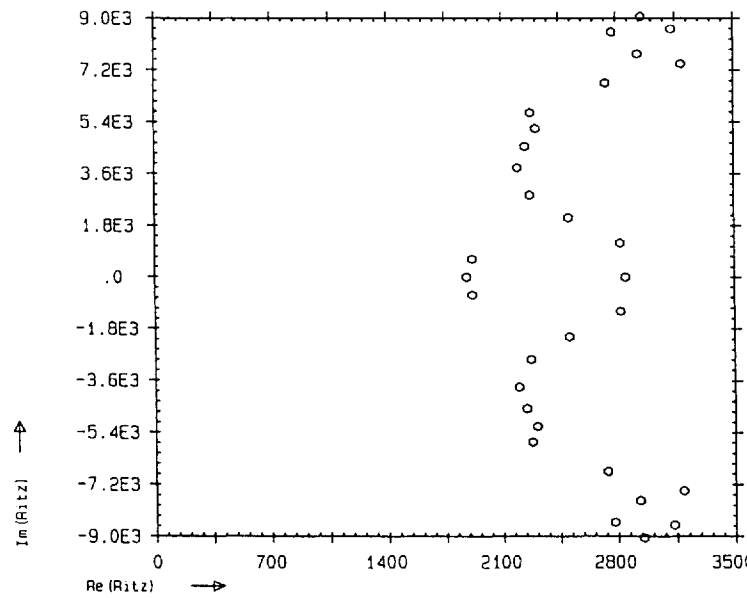

Figure 7. Ritz values after 30 iterations

Approach 1

The convergence should be better without the extra equations. However, since it is not easy to implement this in our discretization, we have not experimented with this approach.

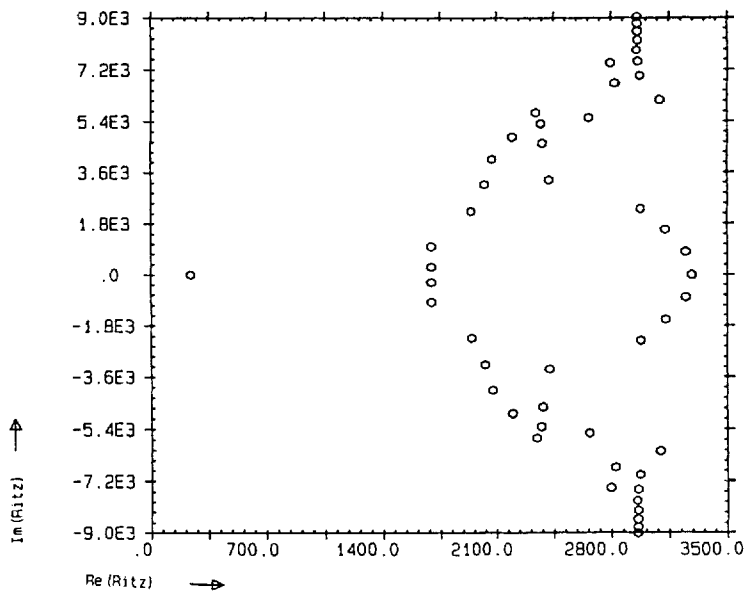

Figure 8. Ritz values after 50 iterations

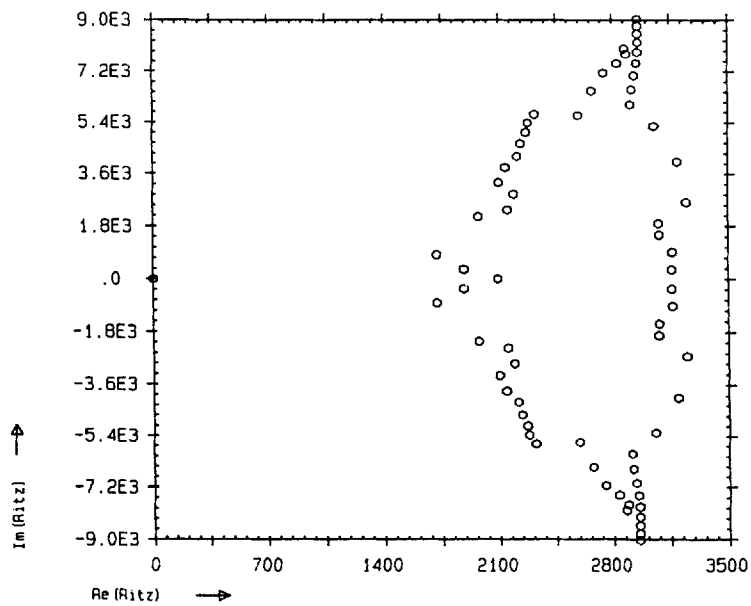

Figure 9. Ritz values after 70 iterations

Approach 2

This approach is the same as approach 1 in the sense that the extra equations, and thus the 'small' eigenvalues, no longer influence the convergence behaviour of GMRES. From an experiment we obtain $\|r_{112}\|_2 \leqslant 10^{-8}$. So this approach saves 25 iterations. The local convergence

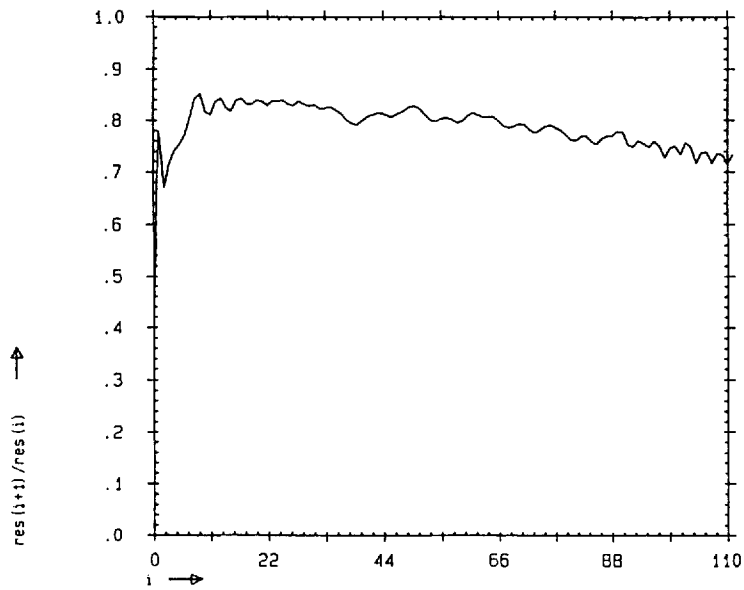

Figure 10. Full GMRES without the influence of the Dirichlet boundary conditions

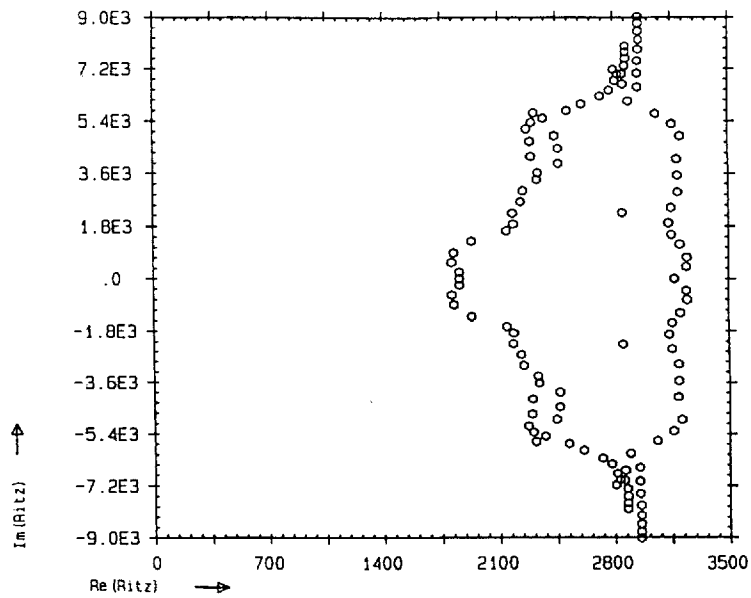

Figure 11. Ritz values after 112 iterations without the influence of the Dirichlet boundary conditions

behaviour is given in Figure 10. Note that there is no stagnation phase (compare Figure 6). Calculation of the Ritz values, which are plotted in Figure 11, shows that 'small' Ritz values are absent.

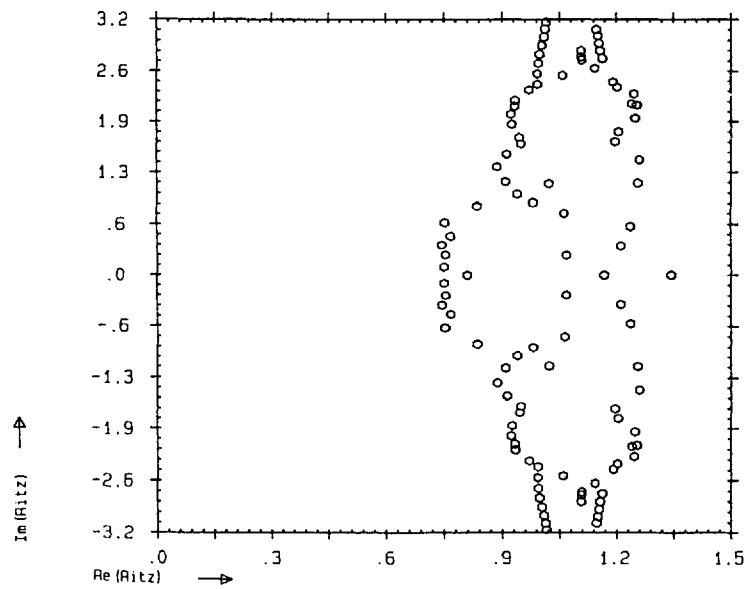

Figure 12. Ritz values after 107 iterations of GMRES combined with a diagonal scaling

Approach 3

It is known[7, 15] that the convergence of GMRES depends on the convex hull of the eigenvalues. After the scaling of the equations, we expect that all eigenvalues are clustered around one, which implies that the extra equations do not influence the convergence of GMRES. Since $\| r_0 \|_2$ depends on the scaling of the equations, GMRES is stopped after the ratio of the norm of the ith residual and the initial residual is the same as for the original process. Using this stopping criterion and starting with $x_0 = 0$, GMRES stops after 107 iterations. The local convergence behaviour is more or less the same as that given in Figure 10. In Figure 12 we plot the Ritz values of GMRES combined with diagonal scaling. Note that the eigenvalue equal to one, which comes from the Dirichlet boundary condition, is inside the convex hull of the eigenvalues, as expected; so, it does not influence the convergence of GMRES.

A termination criterion

A good termination criterion is important for iterative solution methods. In this paragraph we discuss the following criteria:

1. $\| r_k \|_2 \leqslant \text{eps}$
   The main disadvantage of this criterion is that it is not scaling invariant.
2. $\| r_k \|_2 / \| r_0 \|_2 \leqslant \text{eps}$
   This criterion is scaling-invariant, however, the number of iterates is independent of the initial estimate x_0. This is a drawback because we expect that after some time the solution of the foregoing timestep is a good starting solution.
3. $\| r_k \|_2 / \| b \|_2 \leqslant \text{eps}$
   This is a good stopping criterion.

4. $K_2(A)\|r_k\|_2/\|b\|_2 \leqslant \text{eps}$, where $K_2(A)$ is the condition number of A.
This is the best termination criterion. From the inequality

$$\frac{\|x-x_k\|_2}{\|x\|_2} \leqslant K_2(A)\frac{\|r_k\|_2}{\|b\|_2}$$

it follows that $\|x-x_k\|_2/\|x\|_2 \leqslant \text{eps}$. However, in general $K_2(A)$ is not known. To obtain an estimate for $K_2(A)$, we propose to calculate the singular values of $R_k \in \mathbb{R}^{(k+1)\times k}$. R_k is defined in Reference 15, p. 861. An underestimate of $K_2(A)$ is given by the ratio $\sigma_1^{(k)}/\sigma_k^{(k)}$ where $\sigma_1^{(k)}$ is the largest and $\sigma_k^{(k)}$ is the smallest singular value of R_k.

In Figure 13 we plot the estimate from full GMRES applied to the original momentum equations. Note that initially the estimate is small but after 50 iterations the estimate increases to 10^4. This corresponds with the results given in Figure 6.

In Figure 14 we give the estimate from full GMRES combined with a diagonal scaling of the momentum equations. This estimate remains small. Furthermore, it appears that after a small number of iterations the difference between the estimate and its final value is rather small. This motivates us to use the following termination strategy in the diagonal scaled momentum equations:

```
if    k=10   calculate the estimate K̂2(A)=σ1^(10)/σ10^(10),
if    k<10   then
             stop   when   ||rk||2/||b||2 ≤ eps
else
             stop   when   K̂2(A)||rk||2/||b||2 ≤ eps
endif
```

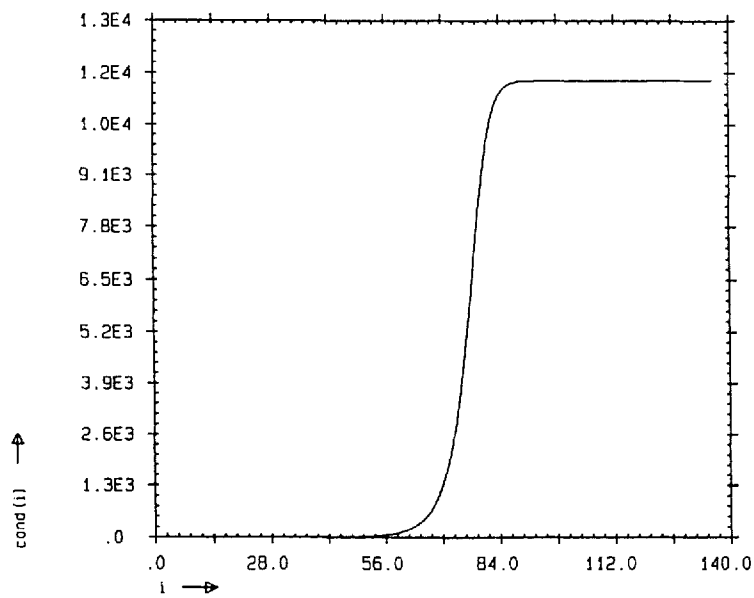

Figure 13. The estimate of the condition of M obtained from full GMRES

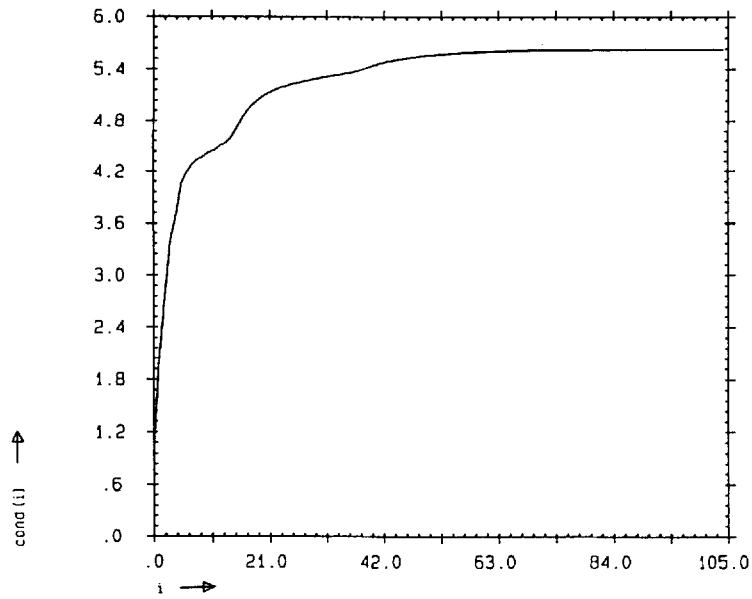

Figure 14. The estimate of the condition of M obtained from full GMRES combined with a diagonal scaling

The time to estimate $\hat{K}_2(A)$ is less than 0·006 s CPU time on the CONVEX. Comparing this with Table III we conclude that this overhead is negligible.

GMRES(m) with diagonal scaling

From the local convergence behaviour of full GMRES (Figure 10), we expect that GMRES(m) is a good iterative method. In Table III we give some experiments with eps $=10^{-5}$ and different choices of m. In these experiments $\| r_0 \|_2 = 0{\cdot}92$ and $\| b \|_2 = 2$.

If $k = \min(10, m)$, we calculate the estimate $\hat{K}_2(A)$. This implies that $\hat{K}_2(A)$ is less for $m=5$ ($\hat{K}_2(A) = 3{\cdot}4$) and $m=3$ ($\hat{K}_2(A) = 2{\cdot}9$) than for $m \geqslant 10$ ($\hat{K}_2(A) = 4{\cdot}2$). Since this estimate is used in the stopping criterion, we expect larger norms of the final residual if m is less than 10 (see Table III). It follows from Table III that if m decreases the amount of vector updates and inner products decreases, whereas the amount of matvec's increases. We observe from $m=10$ and $m=3$ that the decrease in computing time with respect to the vector updates and inner products is equal to the increase in computing time with respect to the matrix vector multiplications. Finally, we

Table III. The amount of work, memory and CPU time

m	Iterations	Matvec	Vector update	Inner product	Memory vectors	$\| r \|_2$	CPU time (s) CONVEX
53	53	53	1431	1431	53	$0{\cdot}44 \times 10^{-5}$	1·24
20	55	57	520	520	20	$0{\cdot}37 \times 10^{-5}$	0·72
10	55	60	263	263	10	$0{\cdot}44 \times 10^{-5}$	0·67
5	57	68	144	144	5	$0{\cdot}51 \times 10^{-5}$	0·62
3	62	82	100	100	3	$0{\cdot}63 \times 10^{-5}$	0·69

note that for the optimal choice ($m = 5$) the required number of vectors in memory is rather small. For m less than 6, the required memory is comparable with the required memory of CGS.[18]

5. CONCLUSIONS

In this paper we have described properties of GMRES-type iterative methods to solve a discretization of the Navier–Stokes equations. Our experiments consist of two parts: in the first part we solve the pressure equation, and in the second part we solve the momentum equations.

Solving the pressure equation, we note that only full GMRES gives a good iterative solution method. Restarting destroys the superlinear convergence behaviour of GMRES.[23] From these results it follows that a preconditioning is necessary to obtain reasonable computing time and memory requirements. It appears that polynomial preconditioning and (M)ILU preconditioning give good results. The required memory used in full GMRES combined with a MILU preconditioner is available because the memory required to store the momentum matrix can be used. Note that the momentum matrix is built anew every timestep.

Full GMRES applied to the momentum equations reveals that the scaling of the equations as originally given is unfavourable. Diagonal scaling of the matrix gives much better results. A termination criterion is proposed such that the norm of the relative error in the solution vector is less than a prescribed accuracy. Using diagonal scaling and this termination criterion, we present results of some experiments with GMRES(m). It appears that a small m is sufficient, which implies that only a small amount of extra memory is required. This amount is comparable with the amount of memory used in the CGS method.

REFERENCES

1. O. Axelsson and G. Lindskog, 'On the eigenvalue distribution of a class of preconditioning methods', *Numer. Math.*, **48,** 479–498 (1986).
2. A. Brandt and N. Dinar, 'Multigrid solutions to flow problems', in S. Parter (ed.), *Numerical Methods for Partial Differential Equations*, Academic Press, New York, 1979, pp. 53–147.
3. K. Dekker, 'Preconditined conjugate gradient techniques for the solution of time-dependent differential equations', *Report 92–19*, Faculty of Technical Mathematics and Informatics, Delft University of Technology, 1992.
4. T. Eirola and O. Nevanlinna, 'Accelerating with rank-one updates', *L.A.A.*, **121,** 511–520 (1989).
5. I. A. Gustafsson, 'A class of first order factorization methods', *BIT*, **18,** 142–156 (1978).
6. W. Hackbusch, *Multi-Grid Methods and Applications*, Springer, Berlin, 1985.
7. Y. Huang and H. A. van der Vorst, 'Some observations on the convergence behavior of GMRES', *Report 89 09*, Faculty of Technical Mathematics and Informatics, Delft University of Technology, 1989.
8. J. J. I. M. van Kan, 'A second-order accurate pressure-correction scheme for viscous incompressible flow', *SIAM J. Sci. Stat. Comput.*, **7,** 870–891 (1986).
9. J. A. Meijerink and H. A. van der Vorst, 'An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix', *Math. Comput.*, **31,** 148–162 (1977).
10. N. M. Nachtigal, L. Reichel and L. N. Trefethen, 'A hybrid GMRES algorithm for non symmetric linear systems', *SIAM J. Math. Anal.*, **13,** 796–825 (1992).
11. C. W. Oosterlee and P. Wesseling, 'A multigrid method for an invariant formulation of the incompressible Navier–Stokes equations in general coordinates', *Comm. Appl. Num. Meth.*, **8,** 721–734 (1992).
12. S. V. Patankar, *Numerical Heat Transfer and Fluid Flow*, McGraw-Hill, New York, 1980.
13. S. V. Patankar and D. B. Spalding, 'A calculation procedure for heat and mass transfer in three-dimensional parabolic flows', *Int. J. Heat Mass Transfer*, **15,** 1787–1806 (1972).
14. D. Rayner, 'Multigrid flow solutions in complex two-dimensional geometries', *Int. j. numer. methods fluids*, **13,** 507–518 (1991).
15. Y. Saad and M. H. Schultz, 'GMRES: a generalized minimal residual algorithm for solving non symmetric linear systems', *SIAM J. Sci. Stat. Comput.*, **7,** 856–869 (1986).
16. P. E. Saylor and D. C. Smolarski, 'Implementation of an adapted algorithm for Richardson's method', *L. A. A.*, **154–156,** 615–646 (1991).
17. A. Segal, P. Wesseling, J. van Kan, C. W. Oosterlee and K. Kassels, 'Invariant discretization of the incompressible Navier–Stokes equations in boundary fitted coordinates', *Int. j. numer. methods fluids*, **15,** 411–426 (1992).

18. P. Sonneveld, 'CGS: a fast Lanczos-type solver for nonsymmetric linear systems', *SIAM J. Sci. Stat. Comput.*, **10,** 36–52 (1989).
19. S. P. Vanka, 'Block-implicit multigrid solution of Navier–Stokes equations in primitive variables', *J. Comput. Phys.*, **65,** 138–158 (1986).
20. H. A. van der Vorst, 'Iterative solution method for certain sparse linear systems with a non-symmetric matrix arising from PDE-problems', *J. Comput. Phys.*, **44,** 1–19 (1981).
21. H. A. van der Vorst, 'High performance preconditioning', *SIAM J. Sci. Stat. Comput.*, **10,** 1174–1185 (1989).
22. H. A. van der Vorst, 'The convergence behavior of some iterative solution methods', in P. Gruber, J. Periaux and R. P. Shaw, (eds), *Proc. 5th Int. Symp. on Numerical Methods in Engineering, Vol. 1*, Springer, Berlin, 1989, pp. 61–72.
23. H. A. van der Vorst and C. Vuik, 'The superlinear convergence behaviour of GMRES', *J. Comput. Appl. Math.*, to appear.
24. H. A. van der Vorst and C. Vuik, 'GMRESR: a family of nested GMRES methods', *J. Numer. L. A. A.*, to appear.
25. C. Vuik and H. A. van der Vorst, 'A comparison of some GMRES-like methods', *L.A.A.*, **160,** 131–162 (1992).
26. P. Wesseling, *An Introduction to Multigrid Methods*, Wiley, Chichester, 1992.
27. G. Wittum, 'Multi-grid methods for Stokes and Navier–Stokes equations with transforming smoothers: algorithms and numerical results', *Numer. Math.*, **54,** 543–563 (1989).