Solution of the incompressible Navier-Stokes equations in general coordinates by Krylov subspace and multigrid methods

Report 93-64

S. Zeng
C. Vuik
P. Wesseling

TU Delft
Faculteit der Technische Wiskunde en Informatica
Faculty of Technical Mathematics and Informatics
Technische Universiteit Delft
Delft University of Technology

ISSN 0922-5641

Copies of these reports may be obtained from the bureau of the Faculty of Technical Mathematics and Informatics, Julianalaan 132, 2628 BL Delft, phone +3115784568.
A selection of these reports is available in PostScript form at the Faculty's anonymous ftp-site. They are located in the directory /pub/publications/tech-reports at ftp.twi.tudelft.nl

Solution of the Incompressible Navier-Stokes Equations in General Coordinates by Krylov Subspace and Multigrid Methods

S. Zeng C. Vuik P. Wesseling

June 21, 1993

Abstract

In this paper three iterative methods are studied: preconditioned GMRES with ILU preconditioning, GMRESR with multigrid as inner loop and multigrid for the solution of the incompressible Navier-Stokes equations in general coordinates. Robustness and efficiency of the three methods are investigated and compared. Numerical results show that the second method is very promising.

1 Introduction

In this paper we investigate three iterative methods for the solution of the incompressible Navier-Stokes equations discretized by a finite volume method on a staggered grid in general coordinates.

The resulting algebraic equations are solved by using a pressure correction scheme [3], which at each time step gives rise to two systems of equations: one for the momentum equations and one for the pressure equation. These systems are usually very large, and in general, the matrices are non-symmetric. For the solution of such non-symmetric large problems, GMRES type methods ([5],[7],[9]) are popular. They are robust and have relatively good rate of convergence. Multigrid methods, have developed rapidly during the past decade. For information on multigrid, see [2] and [11]. Multigrid methods are very suitable for solving large systems of equations resulting from discretization of partial differential equations. Multigrid methods are efficient and are able to solve problems to the accuracy of truncation error at $O(N)$ computational cost, where N is the number of unknowns.

It is desirable that a method is both robust and efficient. A method is robust if it can be applied to a large class of problems, and a method is efficient if it needs little CPU time (in comparison with other methods, of course). The original GMRES method, introduced in [5], is relatively expensive, since CPU time per iteration and memory grow as the number of iterations increases. An effective way to improve performance of iterative methods is preconditioning. Some investigations on the improvement of performance of GMRES by preconditioning can be found in, for example, [9] and [10]. Another variant, called GMRESR, is proposed in [7]. This method uses GMRES twice in an inner loop and an outer loop, with the inner loop providing a good search direction for the outer. With this method, one can easily use a different preconditioner at each iteration. The performance of GMRESR is investigated in [7] and [8] by numerical experiments. Results show that GMRES with ILU type preconditioners and GMRESR are satisfactorily robust and efficient. GMRES type methods can be rather easily implemented on vector computers, because most of the arithmetic operations concerned in these methods are matrix-vector multiplications, vector updates and inner products. So numerical experiments show that preconditioned GMRES type methods have satisfactory efficiency. For multigrid methods, the performance depends highly on the performance of smoothers. Often simple smoothers are efficient but not robust, whereas complicated smoothers are not easily efficiently implemented on vector computers, but are robust. For difficult problems, as for example Navier-Stokes in general coordinates, simple smoothers like those of point Jacobi type often fail. Therefore complicated smoothers like ILU should be used. However, vectorization and parallelization potential of such smoothers is not great. It is observed that with refinement of grids, GMRES type methods become less efficient, as the number of iterations required increases. But multigrid methods, as long as they work, preserve the property of computational cost proportional to $O(N)$. This fact suggests that a combination of GMRES type methods with multigrid methods could give good results.

The combination can be realized through GMRESR, in which the inner loop is replaced by a multigrid method.

In this paper, three methods are studied numerically and compared, which are a preconditioned GMRES method (Method 1), the GMRESR method with a multigrid method as its inner loop (Method 2) and a multigrid method (Method 3). The outline of this paper is as follows. In section 2, the discrete systems are discussed. The three iterative methods are described in section 3. Section 4 deals with test problems and presents results. Finally, in section 5 we draw conclusions.

2 The Discrete Systems

2.1 The Discrete Systems

We consider the discrete systems resulting from finite volume discretization of the incompressible Navier-Stokes equations in general boundary fitted coordinates on a standard staggered grid. For details about our discretization method, see [4], [6] and [12]. With the so-called θ-method for time discretization, we obtain the following discrete systems:

$$\begin{aligned} \frac{\mathbf{V}^{n+1}-\mathbf{V}^n}{\Delta t} + \theta\mathbf{Q}'(\mathbf{V}^{n+1}) + \theta\mathbf{G}\mathbf{p}^{n+1} + (1-\theta)\mathbf{Q}'(\mathbf{V}^n) + (1-\theta)\mathbf{G}\mathbf{p}^n & \\ = \theta\mathbf{B}^{n+1} + (1-\theta)\mathbf{B}^n, & \qquad (2.1) \\ \mathbf{D}\mathbf{V}^{n+1} = 0 & \qquad (2.2) \end{aligned}$$

for the momentum equations and the continuity equation, respectively. Here $\mathbf{V}$ and $\mathbf{p}$ are discrete grid functions, representing velocity components and pressure. The variable t is the time. The superscripts indicate the time level, and Δt is the time interval. The parameter θ is in [0,1]. The operator $\mathbf{Q}'$ is nonlinear, and is linearized, for instance for a typical nonlinear term $(VU)^{n+1}$ in $\mathbf{Q}'$ at time level $n+1$ by using Newton's method:

$$(VU)^{n+1} = V^{n+1}U^n + V^nU^{n+1} - (VU)^n. \qquad (2.3)$$

This gives

$$\mathbf{Q}'(\mathbf{V}^{n+1}) = \mathbf{Q}_1\mathbf{V}^{n+1} + \mathbf{Q}_2(\mathbf{V}^n) \qquad (2.4)$$

with $\mathbf{Q}_1$ linear and both $\mathbf{Q}_1$ and $\mathbf{Q}_2$ calculated at time level n. Central differencing is used in space discretization.

2.2 The Pressure Correction Scheme

The system of equations (2.1) and (2.2) is solved by using the pressure correction method [3], as follows. Let us denote a generic system to be solved by

$$\mathbf{A}\mathbf{x} = \mathbf{b}. \qquad (2.5)$$

First the momentum equations are solved. So (2.5) with

$$\begin{aligned} \mathbf{A} &= \frac{1}{\Delta t}\mathbf{I} + \theta\mathbf{Q}_1, \quad \mathbf{x} = \mathbf{V}^*, \\ \mathbf{b} &= \theta\mathbf{B}^{n+1} + (1-\theta)\mathbf{B}^n + \frac{1}{\Delta t}\mathbf{V}^n - \theta\mathbf{Q}_2(\mathbf{V}^n) - (1-\theta)\mathbf{Q}'(\mathbf{V}^n) - \mathbf{G}\mathbf{p}^n \qquad (2.6) \end{aligned}$$

is solved to give $\mathbf{V}^*$, which is an intermediate result for the velocity. Then the pressure equation, which is derived from the momentum equations and the continuity equation, is solved:

$$\mathbf{A} = \theta \mathbf{D}\mathbf{G}, \quad \mathbf{x} = \mathbf{p}^{n+1} - \mathbf{p}^n, \quad \mathbf{b} = -\frac{\mathbf{D}\mathbf{V}^*}{\Delta t}. \tag{2.7}$$

Now $\mathbf{p}^{n+1}$ is obtained. $\mathbf{V}^{n+1}$ is easily computed from $\mathbf{V}^*$ and $\mathbf{p}^{n+1}$ by means of

$$\frac{\mathbf{V}^{n+1} - \mathbf{V}^*}{\Delta t} = \theta \mathbf{G}(\mathbf{p}^{n+1} - \mathbf{p}^n). \tag{2.8}$$

In our numerical experiments, the parameter θ will be fixed at 1, which leads to the backward Euler method.

3 Algorithms

3.1 GMRES with ILU Preconditioning

If the linear equation system to be solved is represented as (2.5), then the original GMRES algorithm with restart after every m iterations is denoted as GMRES(m) and is given by:

Algorithm GMRES(m)
begin
    Choose: m, initial $\mathbf{x}$
    restart = **.false.**
10 $\mathbf{r} = \mathbf{b} - \mathbf{A}\mathbf{x}$
    $r = \|\mathbf{r}\|$
    if (**not**.*restart*) $r_0 = r$
    if ($r/r_0 > tol$) **then**
        $\mathbf{u}_1 = \mathbf{r}/r$
        for $1 \le j \le m$ **do**
            $\mathbf{c} = \mathbf{A}\mathbf{u}_j$
            $\mathbf{u}_{j+1} = \mathbf{c}$
            for $1 \le i \le j$ **do**
                $h_{i,j} = \mathbf{c}^T \cdot \mathbf{u}_i$
                $\mathbf{u}_{j+1} := \mathbf{u}_{j+1} - h_{i,j}\mathbf{u}_i$
            od
            $h_{j+1,j} = \|\mathbf{u}_{j+1}\|$
            $\mathbf{u}_{j+1} := \mathbf{u}_{j+1}/h_{j+1,j}$
        od
        $\mathbf{x} := \mathbf{x} + \mathbf{U}_m\mathbf{y}_m : \mathbf{y}_m$ minimizes $\|r\mathbf{e}_1 - \bar{\mathbf{H}}_m\mathbf{y}\|$, $\mathbf{y} \in \mathcal{R}^m$
        restart = **.true.**
        goto 10
    end if
end *Algorithm* GMRES(m)

Here, $\mathbf{U}_m$ is a matrix whose columns consist of the l_2-orthonormal basis $\{\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_m\}$, $\bar{\mathbf{H}}_m$ is an $(m+1) \times m$ matrix whose non-zero elements are $h_{i,j}$ for $i = 1, 2, \cdots, m+1$ and $j = 1, 2, \cdots, m$. $\mathbf{e}_1$ is the first column of the $(m+1) \times (m+1)$ identity matrix. *tol* is the accuracy tolerance factor. How to compute $\mathbf{y}_m$ such that $\mathbf{y}_m$ minimizes $\|r\mathbf{e}_1 - \bar{\mathbf{H}}_m \mathbf{y}\|$, $\mathbf{y} \in \mathcal{R}^m$ is described in [5], where also practical implementation of the algorithm is discussed. So we do not get into further details. When incorporating preconditioning, GMRES(m) solves the preconditioned system

$$\mathbf{A}'\mathbf{x} = \mathbf{b}' \tag{3.1}$$

instead of (2.5), where $\mathbf{A}' = \mathbf{C}^{-1}\mathbf{A}$ and $\mathbf{b}' = \mathbf{C}^{-1}\mathbf{b}$, with $\mathbf{C}$ being the preconditioner. The RILU preconditioning (cf. [9],[10]) is used, combining the ILUD preconditioning with the MILUD preconditioning for the momentum equations and the standard ILU preconditioning with the MILU preconditioning for the pressure equation, as follows:

for the momentum equations,

$$\text{RILUD} = \alpha\text{ILUD} + (1-\alpha)\text{MILUD}; \tag{3.2}$$

for the pressure equation,

$$\text{RILU} = \alpha\text{ILU} + (1-\alpha)\text{MILU}. \tag{3.3}$$

The ILUD preconditioner is constructed as follows:

1. $\mathbf{C} = \mathbf{L}\mathbf{D}^{-1}\mathbf{U}$;
2. $\text{diag}(\mathbf{L}) = \text{diag}(\mathbf{U}) = \mathbf{D}$;
3. the off-diagonal parts of $\mathbf{L}$ and $\mathbf{U}$ = the off-diagonal parts of $\mathbf{A}$;
4. $\text{diag}(\mathbf{L}\mathbf{D}^{-1}\mathbf{U}) = \text{diag}(\mathbf{A})$.

MILUD is obtained by using

4a. the sum of the row elements of $\mathbf{L}\mathbf{D}^{-1}\mathbf{U}$ = the sum of the row elements of $\mathbf{A}$.

instead of the last line for ILUD. The standard ILU preconditioner is obtained by requiring

1. $\mathbf{C} = \mathbf{L}\mathbf{U}$;
2. $\text{diag}(\mathbf{L}) = \mathbf{I}$;
3. the non-zero structure of $\mathbf{L} + \mathbf{U}$ = the non-zero structure of $\mathbf{A}$;
4. the non-zero part of $\mathbf{A}$ = the corresponding non-zero part of $\mathbf{L}\mathbf{U}$

We have MILU by replacing the 4-th line for standard ILU by

4a. the non-zero off-diagonal part of $\mathbf{A}$ = the corresponding non-zero off-diagonal part of $\mathbf{L}\mathbf{U}$;

4b. the diagonal elements of $\mathbf{U}$ are modified such that for a row, the sum of the row elements of $\mathbf{LU}$ = the sum of the row elements of $\mathbf{A}$.

Details about GMRES combined with preconditioning and applications to the solution of the incompressible Navier-Stokes equations can be found in [9] and [10]. In our experiments, $m = 20$ and $\alpha = 1$ for the momentum system and $m = 40$ and $\alpha = 0.975$ for the pressure system.

3.2 GMRESR with Multigrid

The GMRESR algorithm introduced in [7] allows us to use various and different preconditioners at each iteration and is given by:

Algorithm GMRESR
begin
    Choose: tol, initial $\mathbf{x}$
    $\mathbf{r} = \mathbf{b} - \mathbf{Ax}$
    $k = -1$
    comment Outer iteration
10 $r = \|\mathbf{r}\|$
    if $(k = -1)$ $r_0 = r$
    if $(r/r_0 > tol)$ **then**
        $k = k + 1$
        comment Inner iteration is in the procedure C
        $\mathbf{u}_k = C(\mathbf{A}, \mathbf{r})$
        $\mathbf{c}_k = \mathbf{Au}$
        for $0 \le i \le k - 1$ **do**
            $\alpha = \mathbf{c}_i^T \cdot \mathbf{c}_k$
            $\mathbf{c}_k := \mathbf{c}_k - \alpha \mathbf{c}_i$
            $\mathbf{u}_k := \mathbf{u}_k - \alpha \mathbf{u}_i$
        od
        $\mathbf{c}_k := \mathbf{c}_k / \|\mathbf{c}_k\|$
        $\mathbf{u}_k := \mathbf{u}_k / \|\mathbf{c}_k\|$
        $\beta = \mathbf{c}_k^T \cdot \mathbf{r}$
        $\mathbf{x} := \mathbf{x} + \beta \mathbf{u}_k$
        $\mathbf{r} := \mathbf{r} - \beta \mathbf{c}_k$
        goto 10
    end if
end *Algorithm* GMRESR

$C(\mathbf{A}, \mathbf{r})$ is the preconditioning procedure, which is to be replaced by any algorithm that gives an approximation for the solution, with $\mathbf{r}$ as the right-hand side. Here, it is a call to a linear multigrid algorithm, and gives $\mathbf{u}_k$ as return. Clearly, as k increases, the memory required increases. So in [7], the truncated GMRESR algorithm is suggested, or better still the so-called *min* α variant of truncated GMRESR ([8]). Here, we use the truncated GMRESR algorithm (trunclast version, see [8]), which is given here for completeness:

Algorithm Truncated GMRESR
begin
  choose nt, tol, initial $\mathbf{x}$
  $\mathbf{r} = \mathbf{b} - \mathbf{A}\mathbf{x}$
  $k = -1$
  comment Outer iteration
10 $r = \|\mathbf{r}\|$
  if $(k = -1)$ $r_0 = r$
  if $(r/r_0 > tol)$ **then**
    $k = k + 1$
    $k1 = \mathrm{mod}(k, nt) + 1$
    comment Inner iteration is in the procedure C
    $\mathbf{u}_{k1} = C(\mathbf{A}, \mathbf{r})$
    $\mathbf{c}_{k1} = \mathbf{A}\mathbf{u}$
    if $(k \geq nt)$ **then**
      $is = k - nt + 1$
    else
      $is = 0$
    end if
    for $is \leq i \leq k - 1$ **do**
      $k2 = \mathrm{mod}(i, nt) + 1$
      $\alpha = \mathbf{c}_{k1}^T \cdot \mathbf{c}_{k2}$
      $\mathbf{c}_{k1} := \mathbf{c}_{k1} - \alpha \mathbf{c}_{k2}$
      $\mathbf{u}_{k1} := \mathbf{u}_{k1} - \alpha \mathbf{u}_{k2}$
    od
    $\mathbf{c}_{k1} := \mathbf{c}_{k1} / \|\mathbf{c}_{k1}\|$
    $\mathbf{u}_{k1} := \mathbf{u}_{k1} / \|\mathbf{c}_{k1}\|$
    $\beta = \mathbf{c}_{k1}^T \cdot \mathbf{r}$
    $\mathbf{x} := \mathbf{x} + \beta \mathbf{u}_{k1}$
    $\mathbf{r} := \mathbf{r} - \beta \mathbf{c}_{k1}$
    goto 10
  end if
end *Algorithm* Truncated GMRESR

In this algorithm, the vectors from the last $nt - 1$ outer iterations are used. This truncated GMRESR algorithm is the algorithm used in our numerical experiments. The number $nt = 15$ (which, however, is not exceeded in our experiments, meaning that in this case the truncated GMRESR is equivalent to full GMRESR).

3.3 The Linear Multigrid Algorithm

The linear multigrid algorithm called in GMRESR is as follows. The F-cycle is used, with one pre- and one post-smoothing. The smoother performs an alternating Jacobi line smoothing, which consists of one horizontal line iteration followed by one vertical line iteration. The momentum equations are smoothed in a decoupled way, i.e., the alternating line smoothing is applied sequentially to the momentum equation in successive directions. Variables are updated after each line Jacobi iteration with damping:

$$\mathbf{x} := \mathbf{x} + \omega\delta\mathbf{x}, \tag{3.4}$$

where ω is an underrelaxation factor. Now we restrict ourselves for brevity temporarily to two dimensions. The coarsest grid in the numerical experiments is fixed at 2×2 and exact solution is obtained by using a direct solver. The underrelaxation factor ω is taken to be 0.7 for both the momentum equations and the pressure equation.

Coarse grid equation systems are formulated by using Galerkin coarse grid approximation (GCA):

$$\mathbf{A}^l = \mathbf{R}\mathbf{A}^{l+1}\mathbf{P}, \quad \mathbf{b}^l = \mathbf{R}\mathbf{b}^{l+1}, \tag{3.5}$$

where l is the grid level index, which is 1 for the coarsest grid, and $\mathbf{R}$ and $\mathbf{P}$ are the restriction and prolongation operators. The momentum equations (2.6) in two dimensions can be represented by

$$\begin{pmatrix} \mathbf{A}^{11} & \mathbf{A}^{12} \\ \mathbf{A}^{21} & \mathbf{A}^{22} \end{pmatrix} \begin{pmatrix} \mathbf{V}^1 \\ \mathbf{V}^2 \end{pmatrix} = \begin{pmatrix} \mathbf{b}^1 \\ \mathbf{b}^2 \end{pmatrix} \tag{3.6}$$

and the pressure equation by

$$\mathbf{A}^{33}\mathbf{p} = \mathbf{b}^3. \tag{3.7}$$

Therefore, Galerkin coarse grid approximation is carried out from grid level $l+1$ to grid level l as follows:

$$\begin{pmatrix} \mathbf{A}^{11(l)} & \mathbf{A}^{12(l)} \\ \mathbf{A}^{21(l)} & \mathbf{A}^{22(l)} \end{pmatrix} = \begin{pmatrix} \mathbf{R}^1\mathbf{A}^{11(l+1)}\mathbf{P}^1 & \mathbf{R}^1\mathbf{A}^{12(l+1)}\mathbf{P}^2 \\ \mathbf{R}^2\mathbf{A}^{21(l+1)}\mathbf{P}^1 & \mathbf{R}^2\mathbf{A}^{22(l+1)}\mathbf{P}^2 \end{pmatrix}, \tag{3.8}$$

$$\begin{pmatrix} \mathbf{b}^{1(l)} \\ \mathbf{b}^{2(l)} \end{pmatrix} = \begin{pmatrix} \mathbf{R}^1\mathbf{b}^{1(l+1)} \\ \mathbf{R}^2\mathbf{b}^{2(l+1)} \end{pmatrix} \tag{3.9}$$

for the momentum equations and

$$\mathbf{A}^{33(l)} = \mathbf{R}^3\mathbf{A}^{33(l+1)}\mathbf{P}^3, \quad \mathbf{b}^{3(l)} = \mathbf{R}^3\mathbf{b}^{3(l+1)} \tag{3.10}$$

for the pressure equation. An algorithm is presented in [16] for efficient implementation of GCA for systems of equations. The restriction operators $\mathbf{R}^1$ and $\mathbf{R}^2$ use the so-called hybrid interpolation, which, for example for $\mathbf{R}^1$, takes place by using the adjoint of bilinear interpolation for $\mathbf{V}^1$ in direction 1 but the adjoint of piecewise constant interpolation in direction 2. $\mathbf{R}^3$ uses the adjoint of piecewise constant interpolation. The prolongation operators $\mathbf{P}^1$, $\mathbf{P}^2$ and $\mathbf{P}^3$ use bilinear interpolations for $\mathbf{V}^1$, $\mathbf{V}^2$ and $\mathbf{p}$. Near boundaries, $\mathbf{R}$ and $\mathbf{P}$ need to be modified. For restriction operators, we use Dirichlet boundary conditions. But for prolongation operators, we employ Neumann boundary conditions. These prolongations and restriction are also applied to the prolongation of coarse grid corrections and the restriction of residuals. See [15] for more detailed descriptions of transfer operators.

When the multigrid algorithm is used as the inner loop in GMRESR (Method 2), only one multigrid iteration (one F-cycle) is performed. When it is used as a multigrid solver (Method 3), the maximum number of cycles is limited to 20.

4 Numerical Experiments

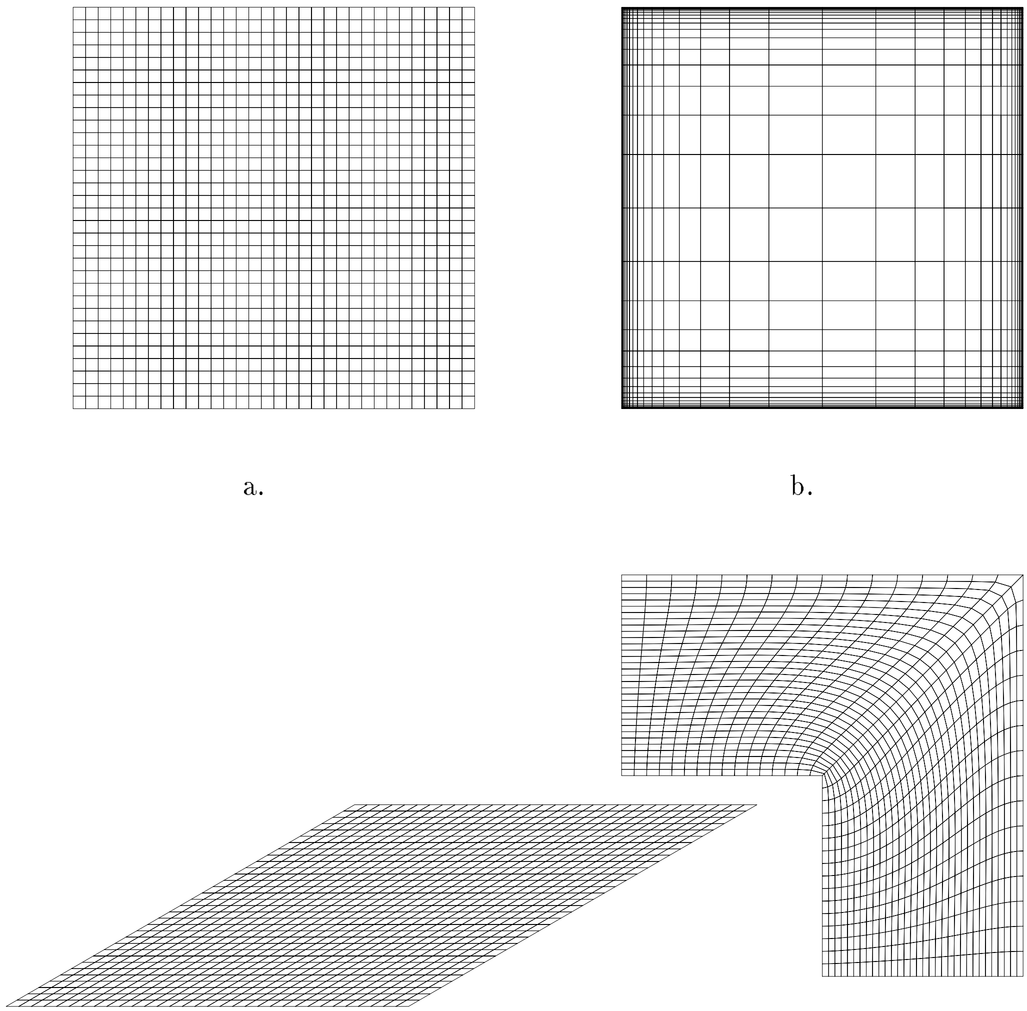
4.1 Test Problems

Four test problems are considered, which are the square driven cavity problem with uniform and non-uniform grids, the skewed driven cavity problem and the L-shaped driven cavity problem, as illustrated in figure 4.1. For convenience, we refer to these problems as Problem 1, Problem 2, Problem 3 and Problem 4, respectively. These problems give rise to different difficulties. We study these problems for two Reynolds numbers $Re = 1$, 1000, three time intervals $\Delta t = 0.0625, 0.125, 0.25$, and three grid sizes 32×32, 64×64, 128×128. The number of time steps is 40. Solution at each time step terminates if the ratio of the residual norm to the initial residual norm $\|\mathbf{r}\|/\|\mathbf{r}_0\| < tol$, where $tol = 10^{-4}$ for the momentum equations and $tol = 10^{-6}$ for the pressure equation. Computations are performed on an HP 730 workstation.

4.2 Results

Tables 4.1–4.4 give the total CPU time t_t, the CPU times t_v and t_p spent on the solution of the momentum equations and the pressure equation, respectively, and the numbers of iterations k_v and k_p at the final time step. For Method 1 (GMRES), the number of iterations is the number of GMRES iterations; for Method 2 (GMRESR with multigird), it is the number of GMRESR iterations; for Method 3 (multigrid), it is the number of Multigrid iterations. Also presented are the reduction factors ρ_v and ρ_p, which for Method 2 are the reduction factors of the multigrid algorithm in the last GMRESR iteration at the final time step, and for Method 3 are the reduction factors of the multigrid algorithm in the last multigrid iteration at the final time step, for the solution of the momentum equations and the pressure equation, respectively. CPU time is given in seconds. Note that $t_t \neq t_v + t_p$, because t_t includes generation of matrices and some other things. The CPU time spent on the computation of GCA is not counted in t_v and t_p, and is small and negligible. In the columns for t_t, 'd' means that the method does not converge. A number following a 'd' indicates the time step when the computation is broken down. These numbers with a star '$*$', indicate that the limit of number of iterations is reached before the accuracy requirement $\|\mathbf{r}\|/\|\mathbf{r}_0\| < tol$ is satisfied, but the corresponding methods still work.

We first discuss efficiency. On the 32×32 grids, Method 1 is the fastest one. Method 2 and Method 3 are approximately equivalent. On the 64×64 grids, Method 2 becomes competitive with Method 1. Method 3 now is the slowest. On the 128×128 grids, Method 2 turns to be the most efficient one in most of the cases. Method 3 surpasses Method 1 in many cases. As grids are refined, computational cost for Method 1 grows significantly, since the number of iterations needed to solve the pressure equation is largely increased. Method 3 can keep about a factor of 4 increasement of computational cost from a coarse grid to the next fine grid, which conforms the multigrid theory that computational cost is proportional to $O(N)$. Method 2 is somewhat superior to Methods 1 and 3, combining the advantages of the two methods. Method 2 also seems to have $O(N)$ computational complexity, and needs less CPU time than Method 1 in most cases and than Method 3 in almost all cases, on the 128×128 grids. The solution of the pressure equation consumes most of CPU time in Method 1, while



a. b. c. d.

Figure 4.1: The four test problems and the 32×32 grids: a. The square driven cavity problem; b. The non-uniform square driven cavity problem; c. The skewed driven cavity problem; d. The L-shaped driven cavity problem

in Methods 2 and 3, solution of the momentum equations is more expensive than the pressure equation. With larger time step Δt, the solution of the momentum equations needs more time. For Method 2, the number of (outer) iterations for the solution of the pressure equation is almost independent of Δt and grid size. Method 2 is faster than Methods 1 and 3 on fine grids. Method 2 is a method to accelerate Method 3, and is indeed faster than Method 3. For the pressure Method 2 is significantly faster than Method 1. It might be worthwhile to use Method 2 for the pressure and Method 1 for the momentum, when the Reynolds number is large.

Now we discuss robustness. Method 3 has more cases in which it fails than the other two methods. It is known that even if an operator on the finest grid has the K-matrix property, which is necessary for good smoothing, it gradually looses the property on coarser grids under GCA (cf. [13],[14],[17]). Furthermore, because of central differencing, diagonal dominance disappears when the time step and the Reynolds number are too large, which also deteriorates smoothing. With this Jacobi line smoother, Method 3 is not very robust. But when it is incorporated with GMRES, yielding Method 2, robustness is improved very much; Method 2 is of the same robustness as Method 1. Although Method 2 has 4 failure cases and Method 1 has 6, it is hard to say now which one is the most robust. It is surprising that when the inner loop of Method 2, which uses Method 3 with only one cycle, fails ($\rho > 1$), Method 2 still sometimes works rather well, within the 40 time steps used, and the number of outer iterations is smaller than for Method 1. It seems that the low Reynolds number cases are harder to solve for Method 1, but for Method 3, the high Reynolds number cases are harder. Both the high and low Reynolds number cases become easier for Method 2, combining the advantages of Method 1 and Method 3.

5 Conclusions

Three iterative solution methods, namely GMRES with ILU preconditioning, GMRESR combined with multigrid and a multigrid method, are applied to solve the incompressible Navier-Stokes equations in general curvilinear coordinates. Their efficiency and robustness are investigated numerically for four test problems. On coarser grids, Method 1 is the most efficient. With grid refinement, it is surpassed by Method 2, and also by Method 3 in many cases. Method 2 is most efficient on larger grids. Method 1 and Method 2 are equally robust. Method 3 is less robust one.

Computing time are reported for a scalar machine. On vector computers, the conclusions for efficiency may be different, because, as pointed out earlier, Method 1 has greater potential of vectorization than Method 3 and therefore than Method 2 as well. A subject of future research is whether for Method 1 the gain from increasing computation speed can compensate the loss due to the significant growth of number of iterations as the grid gets finer. For Method 3, we used a rather weak smoother. If we use more powerful smoothers such as ILU, its robustness will certainly be improved. Another benefit from using smoothers like ILU is the reduction factor can be reduced. However, for the reasons stated before, more time is needed to carry out one iteration, which deteriorates efficiency. So whether application of more powerful smoothers can be made efficient while enhancing robustness is another subject of future research. It might pay off to use different methods for the pressure and the momentum.

Method 2 is very promising. Our future research, therefore, will pay equal attention to Method 2.

Acknowledgement

The authors would like to thank their colleagues C.W. Oosterlee and E. Brakkee for help in generating the grids and for useful discussions.

Table 4.1: Problem 1: the total CPU time t_t, the CPU times t_v and t_p, the numbers of iterations k_v and k_p at the final time step, and the reduction factors ρ_v and ρ_p of the multigrid algorithm in the last iteration at the final time step

		$Re = 1$				$Re = 1000$			
Grid	Δt	t_t	t_v, t_p	k_v, k_p	ρ_v, ρ_p	t_t	t_v, t_p	k_v, k_p	ρ_v, ρ_p
Method 1									
32	.0625	18	7, 4	13, 17		13	2, 5	4,17	
×	.125	19	8, 4	15, 17		14	3, 5	6,18	
32	.25	20	9, 5	16, 17		15	4, 5	8,16	
64	.0625	141	68, 41	21, 25		85	15, 41	6,26	
×	.125	154	84, 42	24, 25		88	18, 42	7,26	
64	.25	171	99, 42	29, 26		94	25, 41	10,25	
128	.0625	1405	820,463	31, 45		669	97,455	7,45	
×	.125	1635	1044,467	40, 45		730	142,470	11,50	
128	.25	1811	1218,467	49, 47		811	232,460	18,50	
Method 2									
32	.0625	62	33, 12	3, 4	.0908,.0495	59	31, 12	4, 4	.145,.0486
×	.125	64	35, 12	4, 4	.137,.0512	73	44, 12	5, 4	.269,.0423
32	.25	65	35, 12	4, 4	.126,.0525	91	60, 12	7, 4	.348,.0476
64	.0625	228	131, 44	4, 4	.131,.0523	241	143, 44	4, 4	.257,.0456
×	.125	228	131, 44	4, 4	.139,.0539	274	170, 44	5, 4	.355,.0469
64	.25	226	131, 44	4, 4	.145,.0554	309	205, 44	6, 4	.360,.0511
128	.0625	948	573,197	4, 4	.140,.0553	938	562,197	3, 4	.158,.0444
×	.125	948	571,197	4, 4	.150,.0565	1023	676,197	4, 4	.261,.0464
128	.25	949	572,197	4, 4	.162,.0574	1050	703,197	5, 4	.250,.0500
Method 3									
32	.0625	63	37, 14	4, 5	.128,.0592	56	29, 14	3, 5	.135,.0690
×	.125	67	40, 14	4, 5	.152,.0607	111	84, 14	9, 5	.540,.0709
32	.25	70	43, 14	5, 5	.188,.0617	188	161, 14	20*, 5	.993,.0704
64	.0625	252	162, 52	5, 5	.184,.0617	278	189, 52	5, 5	.451,.0702
×	.125	258	168, 52	5, 5	.202,.0619	452	363, 52	12, 5	.666,.0675
64	.25	266	176, 52	5, 5	.221,.0620	689	600, 52	20*, 5	.806,.0643
128	.0625	1099	725,224	5, 5	.218,.0619	968	595,223	3, 3	.159,.0617
×	.125	1147	774,224	5, 5	.232,.0617	1191	821,220	6, 5	.408,.0637
128	.25	1162	788,224	6, 5	.252,.0616	1352	978,223	7, 5	.434,.0662

Table 4.2: Problem 2: the total CPU time t_t, the CPU times t_v and t_p, the numbers of iterations k_v and k_p at the final time step, and the reduction factors ρ_v and ρ_p of the multigrid algorithm in the last iteration at the final time step

		$Re = 1$				$Re = 1000$			
Grid	Δt	t_t	t_v, t_p	k_v, k_p	ρ_v, ρ_p	t_t	t_v, t_p	k_v, k_p	ρ_v, ρ_p
Method 1									
32	.0625	22	6, 9	12, 28		26	10, 9	17, 28	
×	.125	22	6, 9	12, 29		39	23, 9	39, 28	
32	.25	21	6, 9	12, 28		d(18)			
64	.0625	189	52, 100	19, 58		184	63, 91	22, 51	
×	.125	184	54, 100	20, 58		d(15)			
64	.25	186	58, 99	21, 58		d(4)			
128	.0625	2563	1022,1402	52,137		1617	340,1149	21,103	
×	.125	2774	1211,1422	67,137		1859	573,1160	31, 96	
128	.25	3328	1745,1442	78,143		2201	898,1172	49,108	
Method 2									
32	.0625	58	30, 16	3, 5	.0561,.0669	71	43, 16	4, 5	.364,.0737
×	.125	69	40, 16	4, 5	.175,.0693	90	62, 16	6, 5	.574,.0690
32	.25	68	39, 17	4, 5	.238,.0682	138	110, 16	12, 4	10^5,.0824
64	.0625	204	107, 60	3, 5	.0566,.0860	267	172, 58	4, 5	1.59,.0743
×	.125	203	106, 59	3, 5	.0572,.0819	516	423, 55	14, 4	1.65,.0802
64	.25	204	107, 60	3, 5	.0684,.0808	d(9)			
128	.0625	859	467, 243	3, 4	.0961,.0701	1767	1351, 266	8, 5	1.08,.0765
×	.125	882	467, 266	3, 5	.0963,.0781	d(9)			
128	.25	883	467, 266	3, 5	.0961,.0809	d(4)			
Method 3									
32	.0625	53	26, 16	3, 6	.0770,.0949	113	86, 15	10, 5	.660,.0954
×	.125	d(6)				d(9)			
32	.25	d(2)				d(4)			
64	.0625	213	117, 57	4, 6	.110, .110	d(6)			
×	.125	212	117, 57	4, 6	.110, .110	d(4)			
64	.25	212	117, 57	4, 6	.120, .110	d(3)			
128	.0625	931	527, 252	4, 6	.156, .128	d(6)			
×	.125	930	527, 252	4, 6	.156, .127	d(4)			
128	.25	931	526, 253	4, 6	.156, .125	d(3)			

Table 4.3: Problem 3: the total CPU time t_t, the CPU times t_v and t_p, the numbers of iterations k_v and k_p at the final time step, and the reduction factors ρ_v and ρ_p of the multigrid algorithm in the last iteration at the final time step

		$Re = 1$				$Re = 1000$			
Grid	Δt	t_t	t_v, t_p	k_v, k_p	ρ_v, ρ_p	t_t	t_v, t_p	k_v, k_p	ρ_v, ρ_p
Method 1									
32	.0625	30	12, 11	20, 33		21	2, 12	4, 33	
×	.125	31	13, 11	22, 33		21	3, 12	7, 32	
32	.25	32	14, 11	23, 32		23	5, 11	12, 32	
64	.0625	293	141,122	35, 69		161	17, 116	7, 65	
×	.125	306	156,119	41, 69		169	23, 117	11, 59	
64	.25	309	159,117	44, 67		185	39, 117	19, 58	
128	.0625	d(1)				1677	130,1421	10,133	
×	.125	d(1)				1758	195,1439	15,107	
128	.25	d(1)				1915	373,1415	28,120	
Method 2									
32	.0625	83	36, 26	4, 9	.265,.326	69	27, 26	3, 9	.0586,.331
×	.125	80	36, 27	4, 9	.259,.291	79	32, 26	4, 9	.131,.290
32	.25	80	36, 26	4, 9	.243,.294	89	43, 26	6, 9	.210,.301
64	.0625	286	131,100	4, 9	.240,.302	252	99, 99	3, 9	.0642,.308
×	.125	286	132,100	4, 9	.214,.310	285	130, 99	4, 9	.138,.314
64	.25	287	132,100	4, 9	.196,.350	291	131, 100	4, 9	.264,.343
128	.0625	1217	598,470	4, 9	.200,.328	1250	613, 483	4, 9	.191,.313
×	.125	1217	597,469	4, 9	.189,.326	1209	594, 465	4, 9	.213,.307
128	.25	1216	597,469	4, 9	.185,.310	1215	598, 468	4, 9	.209,.297
Method 3									
32	.0625	106	42, 51	6,20*	.329,.519	83	23, 48	3, 17	.0607,.453
×	.125	106	43, 51	5,20*	.283,.519	87	26, 49	4, 18	.149,.516
32	.25	107	45, 50	5,20*	.263,.519	97	36, 48	6, 17	.235,.511
64	.0625	394	171,179	6,20*	.297,.519	306	90, 177	3, 17	.0680,.497
×	.125	383	175,170	6, 19	.276,.519	312	105, 169	3, 16	.0873,.491
64	.25	380	176,166	6, 18	.267,.519	338	119, 181	4, 17	.271,.510
128	.0625	1595	741,702	6, 18	.270,.519	1451	508, 792	4, 17	.202,.511
×	.125	1561	741,668	6, 17	.272,.519	1511	536, 824	4, 19	.226,.513
128	.25	1536	741,643	6, 16	.277,.518	1599	619, 828	4,20*	.237,.516

Table 4.4: Problem 4: the total CPU time t_t, the CPU times t_v and t_p, the numbers of iterations k_v and k_p at the final time step, and the reduction factors ρ_v and ρ_p of the multigrid algorithm in the last iteration at the final time step

		$Re = 1$				$Re = 1000$			
Grid	Δt	t_t	t_v, t_p	k_v, k_p	ρ_v, ρ_p	t_t	t_v, t_p	k_v, k_p	ρ_v, ρ_p
Method 1									
32	.0625	21	7, 8	13, 25		16	2, 8	4, 25	
×	.125	22	8, 8	14, 25		18	3, 8	8, 25	
32	.25	23	9, 8	15, 25		20	6, 8	15, 25	
64	.0625	165	57, 79	19, 40		123	16, 78	6, 41	
×	.125	180	65, 80	20, 42		130	23, 79	12, 43	
64	.25	180	73, 79	23, 42		154	47, 79	24, 40	
128	.0625	2244	655,1414	29,102		1548	113,1258	9,105	
×	.125	2626	803,1688	32,145		1738	198,1371	19,150	
128	.25	2923	925,1864	40,164		2098	528,1440	43,117	
Method 2									
32	.0625	70	36, 17	4, 5	.129,.139	60	29, 15	3, 5	.0661,.135
×	.125	71	36, 17	4, 6	.127,.141	70	38, 15	5, 5	.173,.123
32	.25	72	36, 18	4, 6	.131,.122	92	57, 15	7, 6	.348,.116
64	.0625	245	134, 47	4, 5	.130,.115	238	130, 57	4, 6	.166,.155
×	.125	249	131, 58	4, 5	.147,.133	332	214, 61	7, 6	.489,.157
64	.25	253	132, 61	4, 5	.156,.168	486	357, 62	12, 6	.910,.160
128	.0625	1051	623, 271	4, 5	.176,.108	1232	820, 261	5, 6	.786,.160
×	.125	1007	598, 259	4, 5	.194,.106	2129	1690, 285	13, 6	.825,.163
128	.25	1006	597, 258	4, 5	.205,.116	d(6)			
Method 3									
32	.0625	69	37, 20	4, 7	.144,.159	57	26, 19	3, 7	.0648,.168
×	.125	70	38, 20	4, 7	.147,.157	71	38, 20	4, 8	.154,.195
32	.25	72	39, 20	4, 7	.161,.178	110	77, 20	12, 8	.472,.197
64	.0625	253	145, 71	4, 7	.148,.435	265	153, 74	4, 8	.264,.199
×	.125	266	158, 71	4, 7	.166,.408	627	512, 77	20*, 8	.736,.228
64	.25	270	160, 72	5, 7	.196,.440	d(11)			
128	.0625	1033	617, 264	5, 6	.214,.123	2445	2001, 292	20*, 8	1.01,.489
×	.125	1030	616, 263	5, 6	.236,.126	d(9)			
128	.25	1032	617, 264	5, 6	.243,.129	d(5)			

References

[1] Axelsson, O. and G. Lindskog, *On the eigenvalue distribution of a clase of preconditioning methods.* Numer. Math., **48**, 479–498, 1986.

[2] Hackbusch, W., *Multi-grid methods and applications.* Springer, Berlin, 1985.

[3] Kan, J.J.I.M. van, *A second-order accurate pressure-correction scheme for viscous incompressible flow.* SIAM J. Sci. Stat. Comput., **7**, 870–891, 1986.

[4] Mynett, A.E., P. Wesseling, A. Segal and C.G.M. Kassels, *The ISNaS incompressible Navier-Stokes solver: invariant discretization.* Appl. Sci. Research, **48**, 175–191, 1991.

[5] Saad, Y. and M.H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems.* SIAM J. Sci. Stat. Comput., **7**, 856–869, 1986.

[6] Segal, A., P. Wesseling, J. van Kan, C.W. Oosterlee and C.G.M. Kassels, *Invariant discretization of the incompressible Navier-Stokes equations in boundary fitted co-ordinates.* Int. J. Numer. Methods in Fluids, **15**, 411–426, 1992.

[7] Vorst, H.A. van der and C. Vuik, *GMRESR: A family of nested GMRES methods.* Report 91-80, Faculty of Technical Mathematics and Informatics, TU Delft, The Netherlands, 1991. To appear in J. Numer. L. A. A.

[8] Vuik, C., *Further experiences with GMRESR.* Report 92-12, Faculty of Technical Mathematics and Informatics, TU Delft, The Netherlands, 1992.

[9] Vuik, C., *Solution of the discretized incompressible Navier-Stokes equations with the GMRES method.* Int. J. Num. Methods in Fluids, **16**, 507–523, 1993.

[10] Vuik, C., *The solution of the discretized incompressible Navier-Stokes equations with iterative methods.* Report 93-54, Faculty of Technical Mathematics and Informatics, TU Delft, The Netherlands, 1993.

[11] Wesseling, P., *An introduction to multigrid methods.* John Wiley & Sons, Chichester, 1992.

[12] Wesseling, P., A. Segal, J. van Kan, C.W. Oosterlee and C.G.M. Kassels, *Finite volume discretization of the incompressible Navier-Stokes equations in general coordinates on staggered grids.* Comp. Fluid Dyn. J., **1**, 27–33, 1992.

[13] De Zeeuw, P.M., *Matrix-dependent prolongations and restrictions in a block multigrid method solver.* J. Comput. Appl. Math. **3**, 1–27, 1990.

[14] De Zeeuw, P.M. and E.J. van Asselt, *The convergence rate of multi-level algorithms applied to the convection-diffusion equation.* SIAM J. Sci. Stat. Comput., **6**, 492–508, 1985.

[15] Zeng, S. and P. Wesseling, *Galerkin coarse grid approximation for the incompressible Navier-Stokes equations in general coordinates.* Report 92-35, Faculty of Technical Mathematics and Informatics, TU Delft, The Netherlands, 1992.

[16] Zeng, S. and P. Wesseling, *An efficient algorithm for the computation of Galerkin coarse grid approximation for the incompressible Navier-Stokes equations in general coordinates.* Report 92-40, Faculty of Technical Mathematics and Informatics, TU Delft, The Netherlands, 1992.

[17] Zeng, S. and P. Wesseling, *Galerkin coarse grid approximation in multigrid for the incompressible Navier-Stokes equations.* Report 92-103, Faculty of Technical Mathematics and Informatics, TU Delft, The Netherlands, 1992. To appear in SIAM J. Num. Anal.