NUMERICAL STUDY OF A MULTIGRID METHOD WITH FOUR SMOOTHING METHODS FOR THE INCOMPRESSIBLE NAVIER-STOKES EQUATIONS IN GENERAL COORDINATES

S. Zeng and P. Wesseling
Faculty of Technical Mathematics and Informatics
Delft University of Technology
P.O. Box 5031, 2600 GA Delft, The Netherlands

SUMMARY

The performance of a linear multigrid method using four smoothing methods, called SCGS, CLGS, SILU and CILU, is investigated for the incompressible Navier-Stokes equations in general coordinates, in association with Galerkin coarse grid approximation. Robustness and efficiency are measured and compared by application to test problems. The numerical results show that CILU is the most robust, SILU the least, with CLGS and SCGS in between. CLGS is the best in efficiency, SCGS and CILU follow, and SILU is the worst.

INTRODUCTION

Robustness and efficiency of a multigrid method are strongly influenced by the smoother used. Because there are so many factors influencing robustness and efficiency, it is hard to say in general which method is the most appropriate choice for certain applications. In this paper, we study four smoothing methods for the incompressible Navier-Stokes equations in general coordinates, namely the SCGS (Symmetrical Coupled Gauß-Seidel [18]), CLGS (Collective Line Gauß-Seidel, adapted from SCAL [16]), SILU (Scalar ILU, or TILU in [23]) and CILU (Collective ILU [30]), respectively, which are used in a linear multigrid method. Galerkin coarse grid approximation (GCA) is used. An elementary introduction to GCA can be found in [21]. Application to the Navier-Stokes equations is discussed in [29] and [31].

The multigrid method using the above four smoothers solves the velocity and the pressure simultaneously (collectively). Decoupled solution is also used in practice, solving the velocity and the pressure separately. A comparison is given in [1] of multigrid methods using coupled solution with SCGS and CLSOR (Coupled Line Successive Over Relaxation) smoothing and multigrid methods using decoupled solution. Comparisons are presented in [13] and [14] for multigrid methods using the SCGS method and methods using the uncoupled MGPC method (Multigrid Pressure Correction) and the SPC (SIMPLE Pressure Correction) smoothing methods by means of local Fourier analysis as well as numerical experiments. It is stated in [17] that it is advantageous to use the coupled approach. However, both coupled and decoupled solution methods are widely used in practice.

The comparisons mentioned above are made for nonlinear multigrid methods in which the coarse grid operators are computed by using discretization coarse grid approximation (DCA). The relative merits of DCA and GCA are discussed in [21]. A nonlinear multigrid using DCA for the applications discussed in the present paper is presented in [8], [9], [10]. Here we apply GCA. Our main reason is that discretization of the Navier-Stokes equations in general coordinates on a staggered grid is a complicated affair, and GCA enables us to completely separate discretization and multigrid solution. In this paper, attention is focussed on smoothing.

EQUATIONS AND DISCRETIZATION

The incompressible Navier-Stokes equations in tensor formulation in general curvilinear coordinates are given as follows:

$$\frac{\partial U^\alpha}{\partial t} + U^\beta U^\alpha_{,\beta} + g^{\alpha\beta} p_{,\beta} - Re^{-1}\left(g^{\beta\gamma} U^\alpha_{,\beta} + g^{\alpha\beta} U^\gamma_{,\beta}\right)_{,\gamma} = B^\alpha, \tag{1}$$

$$U^\alpha_{,\alpha} = 0. \tag{2}$$

Here U^α, $\alpha = 1, 2, \cdots, d$, are the contravariant velocity components with d the number of space dimensions, p is the pressure, t is the time, B^α is the contravariant component of the body force, and $g^{\alpha\beta}$ is the metric tensor. About tensor notation, see [2] for more details. $U^\alpha_{,\beta}$ is the contravariant derivative. Readers not familar with tensor analysis can understand what is going on by assuming that Cartesian coordinates are used, and interpreting $U^\alpha_{,\beta}$ as $\partial U^\alpha / \partial x^\beta$. U^α and B^α are defined by $U^\alpha = \mathbf{a}^\alpha \cdot \mathbf{u}$, $B^\alpha = \mathbf{a}^\alpha \cdot \mathbf{b}$, where $\mathbf{u}$ and $\mathbf{b}$ are the physical velocity vector and the body force, respectively, and $\mathbf{a}^\alpha$ is the contravariant base vector of the general coordinate system. Let $\mathbf{x} = (x^1, x^2, \cdots, x^d)$ be a Cartesian coordinate system and $\xi = (\xi^1, \xi^2, \cdots, \xi^d)$ be a general coordinate system. Then the contravariant base vector $\mathbf{a}^\alpha$ is defined as $\mathbf{a}^\alpha = \mathrm{grad}(\xi^\alpha)$, and the metric tensor $g^{\alpha\beta}$ is defined by $g^{\alpha\beta} = \mathbf{a}^\alpha \cdot \mathbf{a}^\beta$. It is found that to achieve better accuracy the variable $V^\alpha = \sqrt{g} U^\alpha$ should be used instead of U^α ([7], [12], [22]), with $\sqrt{g}$ the Jacobian of the transformation $\mathbf{x} \longmapsto \xi : \sqrt{g} = |\partial x^\alpha / \partial \xi^\beta|$.

A finite volume discretization of equations (1) and (2) is presented in [7], [12], [22] on staggered grids in general coordinates. From now on we concentrate on two dimensions. Cells may be indexed by a two-tuple of integers $i = (i_1, i_2) \in \mathcal{G}$, $\mathcal{G} = \{1, 2, \cdots, I\} \times \{1, 2, \cdots, J\}$, with I and J the number of cells in the ξ^1- and the ξ^2-direction. The index system for discrete variables is defined as follows. The V^1 variable at the center of the left face, the V^2 variable at the center of the lower face and the p variable at the center of a cell have the same index as the cell. Cells can be numbered in many ways. But unless indicated otherwise, we use the lexicographic order. Variables can also be numbered in different ways, for example, blockwise ordering. We use blockwise ordering for representation of equations; orderings used in the smoothers may be different and are specified together with the smoothers. In blockwise ordering, V^1, V^2 and p are ordered separately: $(\cdots, V^1_k, V^1_{k+1}, \cdots, V^2_k, V^2_{k+1}, \cdots, p_k, p_{k+1}, \cdots)$. Let $\mathbf{V} = (V^1, V^2)$, $\mathbf{B} = (B^1, B^2)$ and $\mathbf{p}$ represent the discrete velocity, the body force and the pressure grid functions, respectively. The discretization results in the following discrete system:

$$\begin{array}{rcl} \dfrac{1}{\Delta t}\mathbf{V}^{n+1} + \theta \mathbf{Q}'(\mathbf{V}^{n+1}) + \theta \mathbf{G}\mathbf{p}^{n+1} & = & \mathbf{f}^{v,(n+1)}, \\ \mathbf{D}\mathbf{V}^{n+1} & = & \mathbf{f}^{c(n+1)} \end{array} \tag{3}$$

with

$$\begin{aligned} \mathbf{f}^{v\prime(n+1)} &= \theta\mathbf{B}^{n+1} + (1-\theta)\mathbf{B}^n + \frac{1}{\Delta t}\mathbf{V}^n - (1-\theta)\mathbf{Q}'(V^n) - (1-\theta)\mathbf{G}\mathbf{p}^n, \\ \mathbf{f}^{c(n+1)} &= 0. \end{aligned} \tag{4}$$

The superscript n denotes the time level. The parameter θ is in [0,1]. The backward Euler method is obtained by setting $\theta = 1$, which is the method used in our numerical experiments. Note that (3) is nonlinear, and is to be solved by a linear multigrid method. Therefore it should be linearized outside multigrid iterations at each time level. Linearization with Newton's method gives

$$(U^\alpha U^\beta)^{n+1} = (U^\alpha)^{n+1}(U^\beta)^n + (U^\alpha)^n(U^\beta)^{n+1} - (U^\alpha U^\beta)^n. \tag{5}$$

So we have

$$\mathbf{Q}'(\mathbf{V}^{n+1}) = \mathbf{Q}_1\mathbf{V}^{n+1} + \mathbf{Q}_2(\mathbf{V}^n) \tag{6}$$

with both $\mathbf{Q}_1$ and $\mathbf{Q}_2$ evaluated at time level n. Note that $\mathbf{Q}_1$ is linear. As a consequence, using blockwise ordering, the linear system to be solved at each time level can be written as

$$\mathbf{K}\mathbf{x} = \mathbf{f}, \tag{7}$$

where

$$\mathbf{K} = \begin{pmatrix} \mathbf{Q} & \theta\mathbf{G} \\ \mathbf{D} & 0 \end{pmatrix}, \quad \mathbf{x} = \begin{pmatrix} \mathbf{V}^{n+1} \\ \mathbf{p}^{n+1} \end{pmatrix}, \quad \mathbf{f} = \begin{pmatrix} \mathbf{f}^{v(n+1)} \\ \mathbf{f}^{c(n+1)} \end{pmatrix} \tag{8}$$

with

$$\mathbf{Q} = \frac{1}{\Delta t} + \theta\mathbf{Q}_1, \quad \mathbf{f}^{v(n+1)} = \mathbf{f}^{v\prime(n+1)} - \theta\mathbf{Q}_2(\mathbf{V}^n). \tag{9}$$

A stationary solution is reached if

$$\mathbf{K}_s\mathbf{x} = \mathbf{f}_s \tag{10}$$

is satisfied, where

$$\mathbf{K}_s = \begin{pmatrix} \mathbf{Q}' & \mathbf{G} \\ \mathbf{D} & 0 \end{pmatrix}, \quad \mathbf{f}_s = \begin{pmatrix} \mathbf{B} \\ \mathbf{f}^c \end{pmatrix}. \tag{11}$$

THE SMOOTHING METHODS

In this section, the four smoothing methods to be used, SCGS, CLGS, SILU and CILU, are described briefly. SCGS is of collective point Gauß-Seidel type. It is a well-known fact that Gauß-Seidel smoothing is not robust when cells in physical space are stretched, which occurs often in general boundary fitted coordinates. Line smoothers are better than point smoothers in handling such problems. Based on the idea of SCGS, a line version called SCAL is presented in [16]. Successful applications of the SCGS and the SCAL methods to problems in Cartesian coordinates can be found in, for instance, [4], [15], [16], [18]. Satisfactory results are also reported for problems in general coordinates ([8], [9], [10], [11]). The results show that SCAL seems to be more attractive than SCGS. Good smoothers may also be derived by employing ILU factorization. For a survey of ILU smoothers, see [20]. Two versions of ILU smoothers, called SILU and CILU, for the incompressible Navier-Stokes equations are presented in [23] and [30], respectively.

The SCGS Method

The SCGS method updates variables cell by cell in a smoothing sweep, first in lexicographical order and then in backward lexicographical order. The five variables, say for cell $i \in \mathcal{G}$, V_i^1, $V_{i+e_1}^1$, V_i^2, $V_{i+e_2}^2$, p_i, which are located at the centers of the four cell faces and the center of cell i, are updated simultaneously, with $e_1 = (1,0)$, $e_2 = (0,1)$. For convenience, we introduce $e_0 = (0,0)$. Let the array $\mathbf{y}$ contain the above five variables, and let the local system for the correction $\delta\mathbf{y}$ of $\mathbf{y}$ be given by

$$\mathbf{A}\delta\mathbf{y} = \mathbf{c}, \tag{12}$$

with $\mathbf{c}^T = (c_i^{v1}, c_{i+e_1}^{v1}, c_i^{v2}, c_{i+e_2}^{v2}, c_i^p)$ and $\mathbf{A}$ a 5×5 matrix. The local system is formulated as follows. Equation (8) can be written, in more detail, as

$$\mathbf{K} = \begin{pmatrix} \mathbf{Q}^{11} & \mathbf{Q}^{12} & \theta\mathbf{G}^1 \\ \mathbf{Q}^{21} & \mathbf{Q}^{22} & \theta\mathbf{G}^2 \\ \mathbf{D}^1 & \mathbf{D}^2 & 0 \end{pmatrix}, \quad \mathbf{x} = \begin{pmatrix} \mathbf{V}^1 \\ \mathbf{V}^2 \\ \mathbf{p} \end{pmatrix}, \quad \mathbf{f} = \begin{pmatrix} \mathbf{f}^{v1} \\ \mathbf{f}^{v2} \\ \mathbf{f}^c \end{pmatrix}. \tag{13}$$

$\mathbf{c}$ contains the residuals of the five equations corresponding to the five variables and is computed by

$$\begin{aligned} c_i^{v1} &= (\mathbf{f}^{v1} - \mathbf{Q}^{11}\mathbf{V}^1 - \mathbf{Q}^{12}\mathbf{V}^2 - \mathbf{G}^1\mathbf{p})_i, & c_{i+e_1}^{v1} &= (\mathbf{f}^{v1} - \mathbf{Q}^{11}\mathbf{V}^1 - \mathbf{Q}^{12}\mathbf{V}^2 - \mathbf{G}^1\mathbf{p})_{i+e_1}, \\ c_i^{v2} &= (\mathbf{f}^{v2} - \mathbf{Q}^{21}\mathbf{V}^1 - \mathbf{Q}^{22}\mathbf{V}^2 - \mathbf{G}^2\mathbf{p})_i, & c_{i+e_2}^{v2} &= (\mathbf{f}^{v2} - \mathbf{Q}^{21}\mathbf{V}^1 - \mathbf{Q}^{22}\mathbf{V}^2 - \mathbf{G}^2\mathbf{p})_{i+e_2}, \\ c_i^p &= (\mathbf{f}^c - \mathbf{D}^1\mathbf{V}^1 - \mathbf{D}^2\mathbf{V}^2)_i. \end{aligned} \tag{14}$$

Using stencil notation ([21]), $\mathbf{A}$ can be written as

$$\mathbf{A} = \begin{pmatrix} \mathbf{Q}^{11}(i,e_0) & & & & \mathbf{G}^1(i,e_0) \\ & \mathbf{Q}^{11}(i+e_1,e_0) & & & \mathbf{G}^1(i+e_1,-e_1) \\ & & \mathbf{Q}^{22}(i,e_0) & & \mathbf{G}^2(i,e_0) \\ & & & \mathbf{Q}^{22}(i+e_2,e_0) & \mathbf{G}^2(i+e_2,-e_2) \\ \mathbf{D}^1(i,e_0) & \mathbf{D}^1(i,e_1) & \mathbf{D}^2(i,e_0) & \mathbf{D}^2(i,e_2) & 0 \end{pmatrix}. \tag{15}$$

Equation (15) is solved analytically. The correction $\delta\mathbf{y}$ is added immediately to $\mathbf{y}$:

$$\mathbf{y} := \mathbf{y} + \omega\delta\mathbf{y}, \tag{16}$$

where ω is an underrelaxation factor.

The CLGS Method

The CLGS method is in fact the same as the SCAL method proposed in [16], except that a smoothing sweep is composed of line Gauß-Seidel in CLGS instead of alternating zebra in SCAL. So CLGS updates variables line by line successively. Let the vector $\mathbf{y}$ accommodate the variables for a whole horizontal i_2-line of cells:

$$\begin{aligned} \mathbf{y}^T = (&\cdots, V_i^1, V_i^2, V_{i+e_2}^2, p_i, V_{i+e_1}^1, V_{i+e_1}^2, V_{i+e_1+e_2}^2, p_{i+e_1}, \\ &V_{i+2e_1}^1, V_{i+2e_1}^2, V_{i+2e_1+e_2}^2, p_{i+2e_1}, \cdots), \quad i = (i_1, i_2) \in \mathcal{G}. \end{aligned} \tag{17}$$

Updating $\mathbf{y}$ gives horizontal line Gauß-Seidel smoothing. Similarly, if the line is taken vertical for a fixed i_1, we have the following arrangement of variables:

$$\begin{aligned}\mathbf{y}^T &= (\cdots, V_i^2, V_i^1, V_{i+e_1}^1, p_i, V_{i+e_2}^2, V_{i+e_2}^1, V_{i+e_2+e_1}^1, p_{i+e_2},\\ &\quad V_{i+2e_2}^2, V_{i+2e_2}^1, V_{i+2e_2+e_1}^1, p_{i+2e_2}, \cdots), \quad i = (i_1, i_2) \in \mathcal{G},\end{aligned} \tag{18}$$

which gives vertical line Gauß-Seidel smoothing. We will use forward horizontal line smoothing, unless indicated otherwise. Other types of line smoothings can be constructed easily by changing the sequence of visiting lines. Let the equation system for the correction $\delta\mathbf{y}$ of $\mathbf{y}$ be denoted again by equation (12), which is readily derived from equation (7), with $\mathbf{c}$ the residuals of the equations corresponding to the variables in $\mathbf{y}$. For a line, for example with i_2 fixed, the variables having the same i_1 are grouped together to form a 4-vector $(V_i^1, V_i^2, V_{i+e_2}^2, p_i)$. This collective arrangement of variables results in a 4×4 block matrix representation of the matrix $\mathbf{A}$, which has non-zero elements (4×4 matrices) at positions $(i_1, i_1 \pm 1, i_1 - 2)$ in the i_1-th row of $\mathbf{A}$. Solution of equation (12) can be carried out easily by using block LU factorization, which needs no further discussion. Updating is performed by (16). Because variables are collectively updated and line Gauß-Seidel relaxation is employed, this method is called CLGS.

The SILU Method

The SILU method is constructed as follows. Because $\mathbf{K}$ in (7) is indefinite, it is hard to find a regular splitting ([19])

$$\mathbf{K} = \mathbf{M} - \mathbf{N} \tag{19}$$

such that the classical iteration

$$\mathbf{x}^{i+1} = \mathbf{x}^i - \mathbf{M}^{-1}(\mathbf{K}\mathbf{x}^i - \mathbf{b}) \tag{20}$$

converges. Therefore, an r-transformation $\bar{\mathbf{K}}$ is used ([23], [24], [25], [26]), and a regular splitting

$$\mathbf{K}\bar{\mathbf{K}} = \mathbf{M} - \mathbf{N} \tag{21}$$

is easier to find. Equation (21) corresponds to the following splitting of $\mathbf{K}$:

$$\mathbf{K} = \mathbf{M}\bar{\mathbf{K}}^{-1} - \mathbf{N}\bar{\mathbf{K}}^{-1}. \tag{22}$$

So with underrelaxation, the iteration (20) is revised as

$$\mathbf{x}^{i+1} = \mathbf{x}^i - \omega\bar{\mathbf{K}}\mathbf{M}^{-1}(\mathbf{K}\mathbf{x}^i - \mathbf{b}). \tag{23}$$

The matrix $\bar{\mathbf{K}}$ chosen and the product $\mathbf{K}\bar{\mathbf{K}}$ are given by

$$\bar{\mathbf{K}} = \begin{pmatrix} \mathbf{I} & -\mathbf{Q}^{-1}\mathbf{G}\mathbf{E}^{-1}\mathbf{F} \\ 0 & \mathbf{E}^{-1}\mathbf{F} \end{pmatrix}, \quad \mathbf{K}\bar{\mathbf{K}} = \begin{pmatrix} \mathbf{Q} & 0 \\ \mathbf{D} & -\mathbf{F} \end{pmatrix} \tag{24}$$

with $\mathbf{E} = \mathbf{D}\mathbf{Q}^{-1}\mathbf{G}$ and $\mathbf{F} = \mathbf{D}\mathbf{G}$. Since $\bar{\mathbf{K}}$ involves the computation of $\mathbf{Q}^{-1}$ and $\mathbf{E}^{-1}$, which is not practical, the following approximation $\tilde{\mathbf{K}}$ of $\bar{\mathbf{K}}$ is applied:

$$\tilde{\mathbf{K}} = \begin{pmatrix} \mathbf{I} & -\mathbf{G} \\ 0 & \tilde{\mathbf{F}}^{-1}\mathbf{D}\tilde{\mathbf{Q}}^{-1}\mathbf{G} \end{pmatrix}, \tag{25}$$

where $\tilde{\mathbf{F}} = \text{diag}(\mathbf{DG})$ and $\tilde{\mathbf{Q}} = \text{diag}(\mathbf{Q})$. Hence we use:

$$\mathbf{x}^{i+1} = \mathbf{x}^i - \omega \bar{\mathbf{K}} \mathbf{M}^{-1}(\mathbf{K}\mathbf{x}^i - \mathbf{b}). \tag{26}$$

Note that the approximation $\tilde{\mathbf{K}}$ of $\bar{\mathbf{K}}$ is different from that used in [23], which is

$$\tilde{\mathbf{K}} = \begin{pmatrix} \mathbf{I} & -\mathbf{G} \\ 0 & \mathbf{DG} \end{pmatrix}, \tag{27}$$

because we found with (27) the multigrid method does not work. The ILU factorization of $\mathbf{K}\bar{\mathbf{K}}$ uses a nine-point ILU factorization, in which the ordering of variables is nested, that is, $(\cdots, V_k^1, V_k^2, p_k, V_{k+1}^1, V_{k+1}^2, p_{k+1}, \cdots)$. So equation (21) can be rewritten as

$$\mathbf{K}\bar{\mathbf{K}} = (\mathbf{L} + \mathbf{D})\mathbf{D}^{-1}(\mathbf{D} + \mathbf{U}) - \mathbf{N} \tag{28}$$

with $\mathbf{L}$ and $\mathbf{U}$ strictly lower and upper triangular matrices and $\mathbf{D}$ a diagonal matrix. For the k-th row of the matrix $\mathbf{M}$, the non-zero pattern G for incomplete factorization is chosen to be a nine-point pattern $G = (k, k \pm 3,\ k \pm 3I,\ k \pm 3I \pm 3,\ k \pm I \mp 3)$ with I the number of cells in the ξ^1-direction, and the elements of $\mathbf{M}$ in G are chosen to be equal to the corresponding elements of $\mathbf{K}\bar{\mathbf{K}}$. In this paper, this method is referred to as SILU because it works with scalar elements of matrices and to distinguish it from CILU, which works with block elements (here 3×3 matrices) and is explained now.

The CILU Method

CILU differs from SILU in two aspects: the choice of r-transformation and a collective treatment of unknowns. The r-transformation $\bar{\mathbf{K}}$ and the corresponding $\mathbf{K}\bar{\mathbf{K}}$ are given by

$$\bar{\mathbf{K}} = \begin{pmatrix} \mathbf{I} & -\mathbf{Q}^{-1}\mathbf{C} \\ 0 & \zeta\mathbf{I} \end{pmatrix}, \quad \mathbf{K}\bar{\mathbf{K}} = \begin{pmatrix} \mathbf{Q} & (\zeta - 1)\mathbf{G} \\ \mathbf{D} & -\mathbf{D}\mathbf{Q}^{-1}\mathbf{G} \end{pmatrix}. \tag{29}$$

Note that a parameter ζ is introduced. It is observed that ζ sometimes has significant effect on convergence (cf. [30]), but here for simplicity it is fixed at 2, which is found to be a good compromise for different problems. Obviously, $\bar{\mathbf{K}}$ and $\mathbf{K}\bar{\mathbf{K}}$ both should be approximated since the computation of $\mathbf{Q}^{-1}$ is impracticable. They are approximated by:

$$\tilde{\mathbf{K}} = \begin{pmatrix} \mathbf{I} & -\tilde{\mathbf{Q}}^{-1}\mathbf{G} \\ 0 & \zeta\mathbf{I} \end{pmatrix}, \quad \widetilde{\mathbf{K}\bar{\mathbf{K}}} = \begin{pmatrix} \mathbf{Q} & (\zeta - 1)\mathbf{G} \\ \mathbf{D} & -\mathbf{D}\tilde{\mathbf{Q}}^{-1}\mathbf{G} \end{pmatrix}, \tag{30}$$

respectively. $\widetilde{\mathbf{K}\bar{\mathbf{K}}}$ is approximately factorized as follows:

$$\widetilde{\mathbf{K}\bar{\mathbf{K}}} = \mathbf{M} - \mathbf{N} = (\mathbf{L} + \mathbf{D})\mathbf{D}^{-1}(\mathbf{D} + \mathbf{U}) - \mathbf{N}. \tag{31}$$

Similar to CLGS, variables are grouped together. For cell i, three variables having the same cell index are grouped in a 3-vector (V_i^1, V_i^2, p_i). Of course, this corresponds to nested ordering. This collective treatment of variables leads to a 3×3 block matrix representation of $\widetilde{\mathbf{K}\bar{\mathbf{K}}}$. The ILU factorization works with the 3×3 blocks as elements. Because of the collective treatment, we call the resulting ILU method CILU. In a typical row, for example row k, $\widetilde{\mathbf{K}\bar{\mathbf{K}}}$ has non-zero elements (3×3 matrices) at positions $(k,\ k \pm 1,\ k \pm I,\ k \pm I \pm 1,\ k \pm I \mp 1,\ k - 2,\ k + I - 2,\ k - 2I,\ k - 2I + 1)$. We choose the following non-zero pattern $G = (k, k \pm 1, k \pm I, k \pm I \pm 1, k \pm I \mp 1)$ for the approximate factorization.

THE LINEAR MULTIGRID ALGORITHM

The linear multigrid algorithm solves the linearized equation system (7) at each time step. The F-cycle will be used. The number of pre-smoothings and the number of post-smoothings are both 1. The coarsest grid will be as coarse as possible; the coarsest grid is 2×2 in our cases. A direct solver is applied on the coarsest grid. The transfer operators for prolongation may be different in the computation of coarse grid matrices by means of Galerkin coarse grid approximation and in the computation of coarse grid correction. For the computation of coarse grid matrices, the prolongation operators for the velocities in the momentum equations are bilinear interpolation, but the hybrid interpolation [28] is used in the continuity equation in order to preserve the structure of the matrix on every grid. The prolongation operator for the pressure is a piecewise constant interpolation. For completeness, we describe the hybrid interpolation here. Cell-centered coarsening is used, taking unions of four fine grid cells to form a coarse grid cell, as illustrated in figure 1. The correspondence between the numbering of

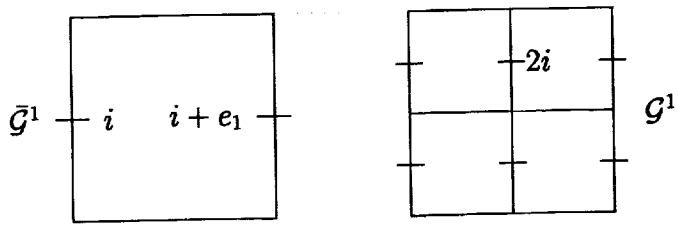

Figure 1. A cell of $\bar{\mathcal{G}}^1$ and the corresponding four cells of $\mathcal{G}^1$; the grid points are indicated by —.

the variables $\bar{V}^1 \subset \bar{\mathbf{U}} : \bar{\mathcal{G}}^1 \longmapsto \mathcal{R}$ on the coarse grid $\bar{\mathcal{G}}^1$ and of $V^1 \subset \mathbf{U} : \mathcal{G}^1 \longmapsto \mathcal{R}$ on the fine grid $\mathcal{G}^1$ is also presented in this figure; coarse grid quantities are indicated by an overbar. The hybrid interpolation $\mathbf{P}^1 : \bar{\mathbf{U}}^1 \longmapsto \mathbf{U}^1$ is constructed by using linear interpolation in the ξ^1-direction but zeroth order interpolation in the ξ^2-direction:

$$[\mathbf{P}^{1*}] = \frac{1}{2} \begin{bmatrix} we & 2 & \underline{we} \\ we & 2 & we \end{bmatrix}. \tag{32}$$

where $\mathbf{P}^{1*}$ is the adjoint of $\mathbf{P}^1$ (cf. [21] for this way of specifying a prolongation). Here $w = 0$ when the "west" point refers to a point outside domain and $w = 1$ elsewhere, and similarly for e relative to "east" points. The underlined element indicates that the corresponding point has index $2i$ on the fine grid, if the operator is applied to point i on the coarse grid. The hybrid interpolation $\mathbf{P}^2$ for V^2 is constructed similarly. Coarse grid correction is computed by using bilinear interpolation for the velocities and piecewise constant interpolation for the pressure. The restriction operators use the adjoint of the hybrid interpolation for the momentum equations and that of the piecewise constant interpolation for the continuity equation. More details about the choice of transfer operators are given in [28] and [31], and an efficient computation of Galerkin coarse grid approximation is presented in [29] and [31] for systems of equations.

Reduction factors are used as measure of the performance of multigrid. The average and the asymptotic reduction factor will be presented. Let $r = \|\mathbf{r}\|$, with $\mathbf{r} = \mathbf{f} - \mathbf{K}\mathbf{x}$ the residual of equation (7) and the norm $\|\mathbf{r}\|$ defined by

$$\|\mathbf{r}\| = \left(\frac{1}{M} \sum_{m=1}^{M} \left(\sum_{j \in \mathcal{G}^m} (\mathbf{f}^m - \mathbf{K}^m \mathbf{x})_j^2 / N_g^m \right) \right)^{\frac{1}{2}}, \tag{33}$$

with M the number of partial differential equations and N_g^m the number of grid points in $\mathcal{G}^m$. At each time step we have a linearized equation system which is solved by a number of linear multigrid iterations. Let r_0 and r_n denote respectively the residual norm before and after n cycles of multigrid iterations on the finest grid. The average reduction factor $\bar{\rho}$ is defined by

$$\bar{\rho} = (r_n / r_0)^{\frac{1}{n}}. \tag{34}$$

The reduction factor ρ_i at the i-th iteration is defined by

$$\rho_i = r_i / r_{i-1}. \tag{35}$$

If a limit of ρ_i exists, then it is the asymptotic reduction factor. Define $r_s = \|\mathbf{r}_s\|$, with $\mathbf{r}_s = \mathbf{f}_s - \mathbf{K}_s \mathbf{x}$ the residual of equation (10). A steady state is approximately obtained if

$$r_s^t / r_s^0 \leq \epsilon \ll 1 \tag{36}$$

is satisfied, where r_s^0 is r_s at time 0 and r_s^t is r_s at time t. The values of ϵ are reported in figures 3–8.

From the results of the following experiments, we choose the most robust method and undertake a further test, which aims at finding a proper choice of prolongation operators for the formulation of coarse grid operators. So the prolongation operators for the velocity in the momentum equations now use the hybrid interpolation for the velocities in the continuity equation. This specification of prolongation operators violates the well-known accuracy condition ([6]) for transfer operators. In [31], it is found that with such specification the multigrid method still works fine. The conclusion is that bilinear prolongation is better for low Reynolds number cases, whereas hybrid interpolation is better for high Reynolds number cases. With application to various test problems, which are described later, we perform some further experiments and try to select the best choice.

NUMERICAL EXPERIMENTS AND RESULTS

Three test problems are chosen, which are the square driven cavity problem, the skewed driven cavity problem and the L-shaped driven cavity problem, as illustrated in figure 2. These impose various difficulties. For brevity, we refer to the square driven cavity problem as problem 1, the skewed driven cavity problem as problem 2, and the L-shaped driven cavity as problem 3. In problem 1, the grid is uniform Cartesian. This gives the simplest discretization, because stretched mesh cells and mixed derivatives do not occur. In problem 2, the grid is still uniform but the grid lines are not orthogonal, so mixed derivatives occur. Giving rise to more difficulties, problem 3 has a stretched non-uniform non-orthogonal grid. For each test problem, two Reynolds numbers are considered, $Re = 1$ and

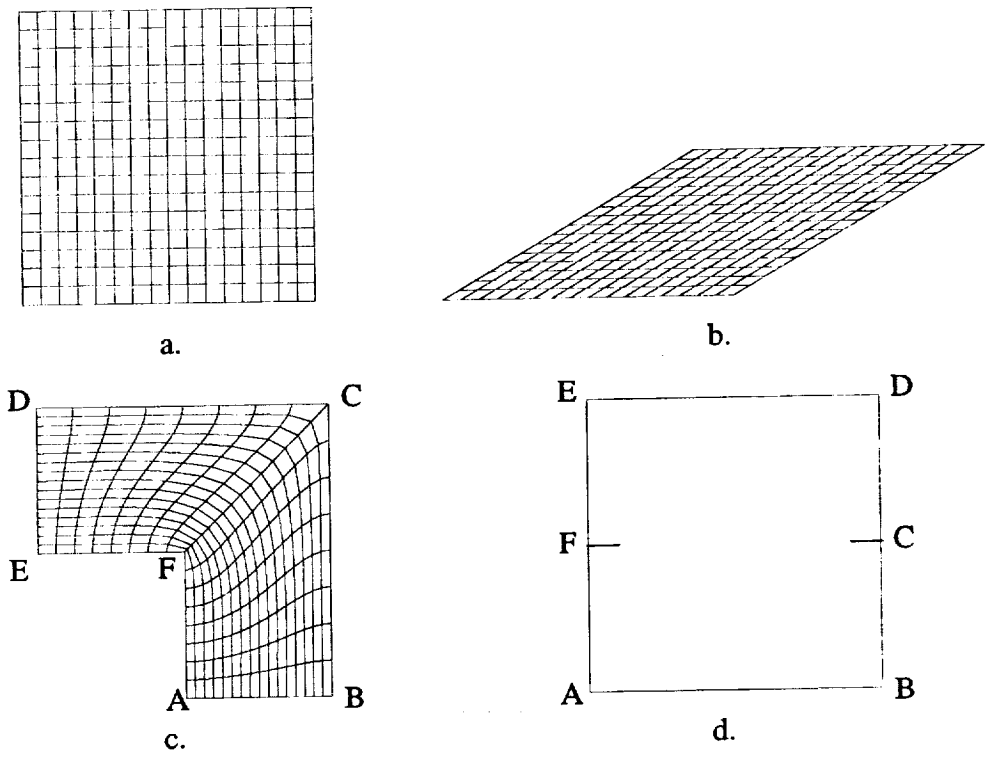

Figure 2: The three test problems and corresponding grids: a. The square driven cavity problem; b. the skewed driven cavity problem; c. the L-shaped driven cavity problem; d. the computational domain of the L-shaped driven cavity problem

$Re = 1000$. The two cases represent viscosity-dominated flows and (mostly) convection-dominated flows. Benchmark solutions for $Re = 1000$ are provided in [3], [5], [11], respectively, for problems 1–3. All computations are carried out on an HP-730 computer.

Prior to the measurement of reduction factors a linear system should be specified. It is natural to use equation (7) at steady state (more precisely, almost steady state). For $Re = 1$, 20 time steps with $\Delta t = 1$ are carried out to give the matrix and the right-hand side at the 'steady state', with each time step accompanied by one multigrid iteration. Only one multigrid iteration is used because we do not want to compute the real time history and so it is not necessary to solve the linear system at each time step very accurately. For $Re = 1000$, the number of time steps is changed to 100 with $\Delta t = 0.2$. The smoother used in the computations for the 'steady states' is CILU, with the underrelaxation parameter ω fixed at 0.7. A smaller time step is needed for larger Reynolds numbers to increase the main diagonal because the discretization uses central differencing, which results in bad smoothing for Re and Δt being too large. Figures 3–8 present the streamlines of the test problems. They match well with the corresponding results in [3], [5], [11].

In order to determine the best performance of each smoother, the underrelaxation parameter is sampled at an interval 0.1 to find a good value. Tables 1–3 give the reduction factors for the multigrid methods using different smoothers on the 128×128 grids corresponding to the best values of the underrelaxation factor ω. If machineaccuracy is not reached, the reduction factors for the last 5

iterations are given, otherwise the reduction factors for the last 5 successive iterations before machine accuracy is reached. The maximum number of grid levels l_f is also given; exceeding this causes the algorithm to fail.

From these tables, we deduce the following observations.

- The SCGS smoother works for all test cases. However, it is clearly very problem-sensitive. The underrelaxation parameter changes significantly as the problems and the Reynolds number change. For problem 3 and $Re = 1000$ it is very slow. For problem 2, overrelaxation has to be employed instead of underrelaxation. The number of grid levels must be reduced in problem 3, i.e., the coarse grids cannot be very coarse in order to obtain smoothing.

- The CLGS seems to work better than the SCGS smoother, because the underrelaxation parameter does not vary so much and usually the reduction factors are smaller. The one exceptional case is problem 1, where the number of grids has to be reduced by 1, even for $Re = 1$. What is more surprising in this case is that for $Re = 1000$ convergence cannot be achieved with forward horizontal line smoothing. But using forward vertical line smoothing and strong damping, we recover convergence, which is, however, worse than that of the SCGS smoother. To improve the performance for this case, perhaps symmetrical line smoothing should be used. So further tested are symmetrical horizontal line smoothing (SHCLGS), symmetrical vertical line smoothing (SVCLGS) and symmetrical alternating line smoothing (SACLGS). It is found that SHCLGS and SACLGS both do not show any improvement, because the horizontal sweeps destroy smoothing seriously; SVCLGS gives some improvement, giving the best average reduction $\bar{\rho} = 0.5610$ for $\omega = 0.2$.

- With the SILU smoother, the underrelaxation parameter is less sensitive to change of problem and Reynolds number than with SCGS and CLGS, but the reduction factors are usually larger than those of SCGS and CLGS. The number of grid levels cannot exceed 4 or 5, otherwise the method does not work due to loss of smoothing on coarser grids. The well-known dependence of ILU smoothers on grid point ordering plays a role in problem 3. SILU is here found to be a bad smoother with lexicographic grid point ordering. The results presented have been obtained with a backward ordering, starting from corner D (cf. figure 2d) and moving first down and then to the left.

- The CILU smoother is not problem-sensitive. Very good convergence is obtained for all test problems. It is possible to fix the underrelaxation parameter at one value, which here is found to be 0.8. The dependence on the grid point ordering is pronounced for problem 3, for which the backward ordering described for SILU was used.

According to the above observations, we can arrange the four smoothers in the following order from the best to worst: CILU, CLGS, SCGS, SILU. Of course, this conclusion is not general, because discretization and transfer operators both certainly affect the overall performance of an algorithm.

Apart from robustness, efficiency should also be taken into account. Table 4 gives the CPU time in seconds per cycle (t_c) for the smoothers. The most robust smoother CILU takes twice as much time per cycle as the other three smoothers. The efficiency of two multigrid methods using two smoothers

(referred to as method 1 and method 2) may be compared as follows. Let the average reduction factor of method 1 be $\bar{\rho}_1$ and that of method 2 be $\bar{\rho}_2$, and let the CPU time per multigrid cycle be t_{c1} and t_{c2}, respectively. For a required accuracy, for example a reduction ϵ of the initial residual norm, method 1 takes $t_{c1} \ln \epsilon / \ln \bar{\rho}_1$ CPU time and method 2 takes $t_{c2} \ln \epsilon / \ln \bar{\rho}_2$ CPU time. Define the efficiency factor E_f of method 1 with respect to method 2 by

$$E_f = \frac{t_{c2} \ln \bar{\rho}_1}{t_{c1} \ln \bar{\rho}_2}. \tag{37}$$

So if $E_f > 1$, then method 1 is more efficient; if $E_f < 1$ then method 2 is more efficient. For comparisons among more than 2 methods, one of them is used as a standard, in place of method 2. Using $\bar{\rho}$ and t_c given in tables 1–4 and taking CILU as the standard for the comparison, table 5 presents E_f in all the test cases. Bigger numbers mean higher efficiency. Apparently, The SCGS smoother and the CLGS smoother are mostly more, but not very much, efficient than the CILU smoother; the SILU smoother is mostly less efficient. Because SCGS and CLGS can be easily altered to parallellizable versions by using black-white or zebra ordering, one may argue that SCGS and CLGS have more parallellization potential than CILU, and higher efficiency can be obtained. But this may be true only in two dimensions.

Now with CILU, we investigate convergence of the multigrid method using the hybrid interpolation instead of bilinear interpolation for the velocities in the momentum equations in the formulation of coarse grid operators. The results are given in table 6 in terms of the reduction factors for the best values of ω. Clearly, the method works much better for $Re = 1000$ than for $Re = 1$. Using the hybrid prolongation for $Re = 1000$ the method performs equally as well as the method using the bilinear prolongation. It is easy to see that for low Reynolds number cases bilinear prolongation is better, but this is not so clear for high Reynolds number cases. We found that for high Reynolds numbers there are some cases in which bilinear prolongation does not work but the hybrid prolongation still works well. Therefore it is safer to use the hybrid prolongation for high Reynolds numbers. One may conclude again that the hybrid prolongation is more suitable for high Reynolds numbers and bilinear prolongation is more suitable for low Reynolds numbers.

CONCLUSIONS

The performance of the multigrid method using SCGS, CLGS, SILU and CILU smoothers are studied for the incompressible Navier-Stokes equations in general coordinates. Galerkin coarse grid approximation is used in the computation of coarse grid matrices. Both robustness and efficiency of the methods are investigated and measured in terms of reduction factors and efficiency factors. The results show that the most robust smoother is CILU; CLGS and SCGS follow. SILU is the worst. For efficiency, the order from the best to the worst is CLGS, SCGS, CILU and SILU. Although CILU is somewhat less efficient than CLGS and SCGS and it has less parallellization potential in two dimensions, it may be more promising in three dimensions because it is much more robust than all the others and parallellization can also be established among planes.

For prolongation operators in the computation of coarse grid operators, the hybrid interpolation is a more appropriate choice for high Reynolds numbers, whereas bilinear interpolation is a more appropriate choice for low Reynolds numbers.

REFERENCES

1. Arakawa, Ch.; Demuren A.O.; Rodi, W.; and Schönung, B.: *Application of Multigrid Methods for the Coupled and Decoupled Solution of the Incompressible Navier-Stokes Equations.* In M. Deville (ed.): Proc. 7th GAMM Conf. on Numer. Methods in Fluid Mech., Louvain-la-Neuve, Sept. 9–11, 1987. Notes on Numer. Fluid Mech., vol. 20, pp.1–8, Vieweg, Braunschweig, 1988.

2. Aris, R.: *Vectors, Tensor and the Basic Equations of Fluid Mechanics.* Prentice-Hall, Englewood Cliffs, NJ, 1962.

3. Demirdžić, I.; Lilek, Ŭ.; and Perić, M.: *Fluid Flow and Heat Transfer Test Problems for Non-Orthogonal Grids: Bench-Mark Solutions.* Int. J. Numer. Methods in Fluids, vol. 15, pp.329–354, 1992.

4. Gaskell, P.H.; and Wright, N.G.: *A Multigrid Algorithm for the Investigation of Thermal Recirculating Fluid Flow Problems.* In R.W. Lewis, K. Morgan and W.G. Habashi (eds.): Proc. 5th Int. Conf. on Numer. Methods for Thermal Problems, Montreal, Canada, June 29–July 3, 1987. Numerical Methods in Thermal Problems, vol. 5, pp.1194–1215, Pineridge, Swansea, 1987.

5. Ghia, U.; Ghia, K.N.; and Shin C.T.: *High-Re Solutions for Incompressible Flow Using the Navier-Stokes Equations and a Multigrid Method.* J. Comp. Phys., vol. 48, pp.387–411, 1982.

6. Hackbusch, W.: *Multi-Grid Methods and Applications.* Springer, Berlin, 1985.

7. Mynett, A.E.; Wesseling, P.; Segal, A.; and Kassels, C.G.M.: *The ISNaS Incompressible Navier-Stokes Solver: Invariant Discretization.* Applied Scientific Research, vol. 48, pp.175–191, 1991.

8. Oosterlee, C.W.; and Wesseling, P.: *A Multigrid Method for an Invariant Formulation of the Incompressible Navier-Stokes Equations in General Coordinates.* Comm. Appl. Numer. Methods., vol. 8, pp.721–734, 1992.

9. Oosterlee, C.W.; and Wesseling, P.: *A Multigrid Method for a Discretization of the Incompressible Navier-Stokes Equations in General Coordinates.* In J.B. Vos, A. Rizzi, and I.L. Ryhming (eds.): Proc. 9th GAMM Conf. on Numer. Methods in Fluid Mech. Notes on Num. Fluid Mech., vol. 35, pp.99–106, Vieweg, Braunschweig, 1992.

10. Oosterlee, C.W.; and Wesseling, P.: *A Robust Multigrid Method for a Discretization of the Incompressible Navier-Stokes Equations in General Coordinates.* In Ch. Hirsch, J. Périaux and W. Kordulla (eds.): Computational Fluid Dynamics'92. Proc. First European Computational Fluid Dynamics Conference, pp.101–107, Elsevier, Amsterdam, 1992.

11. Oosterlee, C.W.; Wesseling, P.; Segal, A.; and Brakkee, E.: *Benchmark Solutions of the Incompressible Navier-Stokes Equations in General Coordinates on Staggered Grids.* Report 92-67, Faculty of Technical Mathematics and Informatics, TU Delft, The Netherlands, 1992. To appear in Int. J. Numer. Methods in Fluids.

12. Segal, A.; Wesseling, P.; van Kan, J.; Oosterlee, C.W.; and Kassels, C.G.M.: *Invariant Discretization of the Incompressible Navier-Stokes Equations in Boundary Fitted Co-ordinates.* Int. J. Numer. Methods in Fluids, vol. 15, pp.411–426, 1992.

13. Sivaloganathan, S.: *The Use of Local Mode Analysis in the Design and Comparison of Multigrid Methods.* Comp. Phys. Comm., vol. 65, pp.246–252, 1991.

14. Sivaloganathan, S.; Shaw, G.J.; Shah, T.M.; and Mayers, D.F.: *A Comparison of Multigrid Methods for the Incompressible Navier-Stokes Equations.* In K.W. Morton and M.J. Baines (eds.): Numerical Methods for Fluid Dynamics III, pp.410–417. Oxford University Press, 1988.

15. Thompson, C.P.; Leaf, G.K.; and Vanka, S.P.: *Application of a Multigrid Method to a Buoyancy-Induced Flow Problem.* In S.F. McCormick (ed.): Multigrid Methods. Lecture Notes in Pure and Applied Math., vol. 110, pp.605–629, Marcel Dekker, New York, 1988.

16. Thompson, M.C.; and Ferziger, J.H.: *An Adaptive Multigrid Technique for the Incompressible Navier-Stokes Equations.* J. Comp. Phys., vol. 82, pp.94–121, 1989.

17. Vanka, S.P.: *Block-Implicit Calculation of Steady Turbulent Recirculating Flows.* Int. J. Heat Mass Transfer, vol. 28, pp.2093–2193, 1985.

18. Vanka, S.P.: *Block-Implicit Solution of Navier-Stokes Equations in Primitive Variables.* J. Comp. Phys., vol. 65, pp.138–158, 1986.

19. Varga, R.S., *Matrix Iterative Analysis.* Prentice-Hall, Englewood Cliffs, NJ, 1962.

20. Wesseling, P.: *A Survey of Fourier Smoothing Analysis Results.* In W. Hackbusch and U. Tottenberg (eds.): Multigrid Methods III. Proc. 3rd European Conf. on Multigrid Methods, Bonn, Oct.1–4, 1990. International Series of Numerical Mathematics, vol. 98, pp.105–127, Birkhäuser, 1991.

21. Wesseling, P.: *An Introduction to Multigrid Methods.* John Wiley & Sons, Chichester, 1992.

22. Wesseling, P.; Segal, A.; van Kan, J.; Oosterlee, C.W.; and Kassels, C.G.M.: *Finite Volume Discretization of the Incompressible Navier-Stokes Equations in General Coordinates on Staggered Grids.* Comp. Fluid Dyn. J., vol. 1, pp.27–33, 1992.

23. Wittum, G.: *Multi-Grid Methods for Stokes and Navier-Stokes Equations—Transforming Smoothers: Algorithms and Numerical Results.* Numer. Math., vol. 54, pp.27–33, 1989.

24. Wittum, G.: *On the Convergence of Multi-Grid Methods with Transforming Smoothers—Theory with Applications to the Navier-Stokes Equations.* Numer. Math., vol. 57, pp.15–38, 1990.

25. Wittum, G.: *The Use of Fast Solvers in Computational Fluid Dynamics.* In P. Wesseling (ed.): Proc. 8th GAMM-Conf. on Numer. Methods in Fluid Mech., Sept. 27–29, 1989, Delft, The Netherlands. Notes on Fluid Numer. Mech., vol. 29, pp.547–581, Vieweg, Braunschweig, 1990.

26. Wittum, G.: *R-Transforming Smoothers for the Incompressible Navier-Stokes Equations.* In W. Hackbusch (ed.): Proc. 5-th GAMM-Seminar, Jan. 20-22, 1989, Kiel, Germany. Notes on Numer. Fluid Mech., vol. 30, pp.153–162, Vieweg, Braunschweig, 1990.

27. De Zeeuw, P.M.: *Matrix-Dependent Prolongations and Restrictions in a Block Multigrid Method Solver.* J. Comput. Appl. Math. vol. 3, pp.1–27, 1990.

28. Zeng, S.; and Wesseling, P.: *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.

29. Zeng, S.; and Wesseling, P.: *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.

30. Zeng, S.; and Wesseling, P.: *An ILU Smoother for the Incompressible Navier-Stokes Equations in General Coordinates.* Report 92-91, Faculty of Technical Mathematics and Informatics, TU Delft, The Netherlands, 1992.

31. Zeng, S.; and Wesseling, P.: *Multigrid Solution of the Incompressible Navier-Stokes Equations in General Coordinates.* To appear in SIAM J. of Num. Anal.

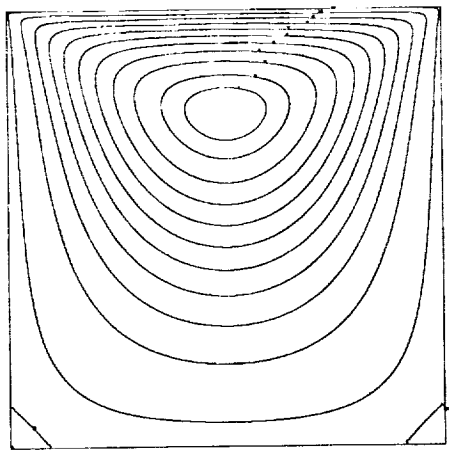

Figure 3: Streamlines for problem 1, $Re = 1$, $r_s^t/r_s^0 < 4.581 \times 10^{-11}$

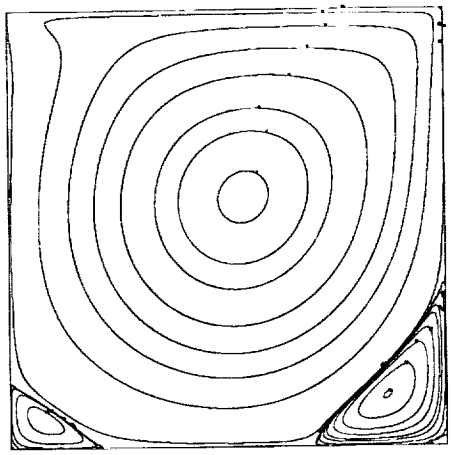

Figure 4: Streamlines for problem 1, $Re = 1000$, $r_s^t/r_s^0 < 1.804 \times 10^{-3}$

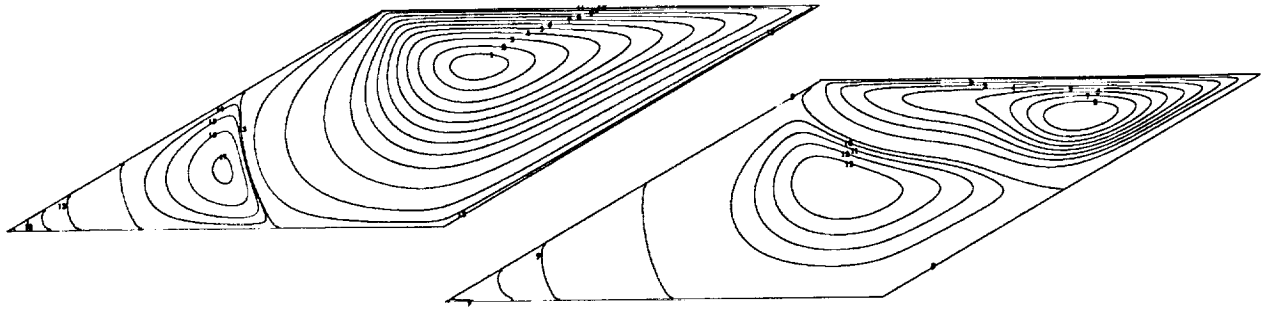

Figure 5: Streamlines for problem 2, $Re = 1$, $r_s^t/r_s^0 < 4.358 \times 10^{-10}$

Figure 6: Streamlines for problem 2, $Re = 1000$, $r_s^t/r_s^0 < 4.484 \times 10^{-6}$

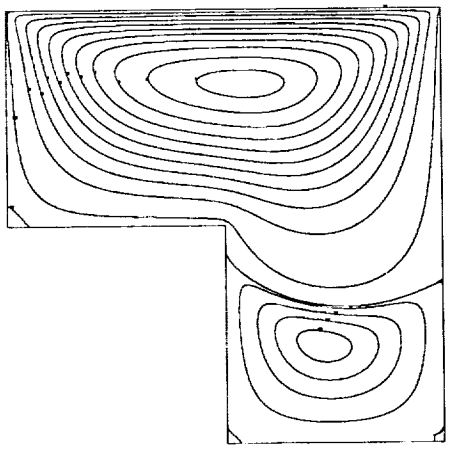

Figure 7: Streamlines for problem 3, $Re = 1$, $r_s^t/r_s^0 < 9.723 \times 10^{-9}$

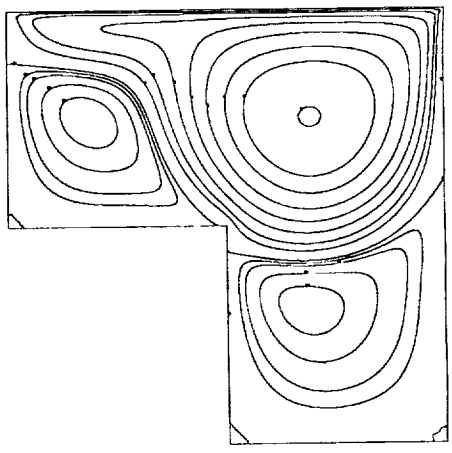

Figure 8: Streamlines for problem 3, $Re = 1000$, $r_s^t/r_s^0 < 1.172 \times 10^{-4}$

Table 1: Reduction Factors Corresponding to the Best Values of ω for Problem 1 on the 128×128 Grid

Smoother	SCGS	CLGS	SILU	CILU	SCGS	CLGS	SILU	CILU
	$Re = 1,\ r_0 = 12.96$				$Re = 1000,\ r_0 = 1.605 \times 10^{-02}$			
ω	0.8	1.0	0.9	1.0	0.3	0.1*	0.7	1.0
l_f	7	6	4	7	7	6	4	7
i	16	16	16	15	16	16	16	16
ρ_i	0.2787	0.2183	0.3708	0.2026	0.6464	0.8344	0.8168	0.4122
ρ_{i+1}	0.2811	0.2184	0.3761	0.2079	0.6420	0.8950	0.8009	0.4116
ρ_{i+2}	0.2789	0.2237	0.3807	0.2142	0.5994	0.8735	0.8244	0.4136
ρ_{i+3}	0.2816	0.2235	0.3849	0.2224	0.6088	0.9048	0.8846	0.4155
ρ_{i+4}	0.2791	0.2300	0.3880	0.2393	0.5869	0.8899	0.9346	0.4131
$\bar{\rho}$	0.2561	0.1973	0.2863	0.1732	0.4918	0.7773	0.7005	0.2996

* Forward vertical smoothing

Table 2: Reduction Factors Corresponding to the Best Values of ω for Problem 2 on the 128×128 Grid

Smoother	SCGS	CLGS	SILU	CILU	SCGS	CLGS	SILU	CILU
	$Re = 1,\ r_0 = 25.92$				$Re = 1000,\ r_0 = 2.697 \times 10^{-02}$			
ω	1.2	0.9	0.9	0.8	1.2	1.0	0.8	0.9
l_f	7	7	4	7	7	7	4	7
i	16	16	16	16	16	16	16	16
ρ_i	0.3377	0.3476	0.7401	0.3857	0.3693	0.4166	0.7784	0.3052
ρ_{i+1}	0.3406	0.3492	0.7418	0.3885	0.3650	0.4167	0.7798	0.3190
ρ_{i+2}	0.3452	0.3512	0.7437	0.3911	0.3613	0.4173	0.7811	0.3312
ρ_{i+3}	0.3432	0.3536	0.7456	0.3931	0.3582	0.4180	0.7825	0.3399
ρ_{i+4}	0.3463	0.3563	0.7476	0.3942	0.3558	0.4184	0.7839	0.3447
$\bar{\rho}$	0.3032	0.3205	0.6315	0.2968	0.3472	0.3941	0.6716	0.2802

Table 3: Reduction Factors Corresponding to the Best Values of ω for Problem 3 on the 128 x 128 Grid

Smoother	SCGS	CLGS	SILU	CILU	SCGS	CLGS	SILU	CILU
	$Re = 1,\ \tau_0 = 18.20$				$Re = 1000,\ \tau_0 = 1.969 \times 10^{-02}$			
ω	1.0	0.9	0.8*	0.8*	0.1	0.4	0.2*	0.8*
l_f	6	7	5	7	5	7	5	7
i	16	15	16	16	16	16	16	16
ρ_i	0.7302	0.2320	0.5960	0.6997	0.9381	0.6527	0.9293	0.3496
ρ_{i+1}	0.7319	0.1699	0.5878	0.4104	0.9399	0.6354	0.9337	0.3355
ρ_{i+2}	0.7334	0.2131	0.5914	0.2317	0.9400	0.6425	0.9376	0.3344
ρ_{i+3}	0.7347	0.1941	0.5927	0.6643	0.9383	0.6536	0.9411	0.3448
ρ_{i+4}	0.7359	0.2614	0.5909	0.4450	0.9352	0.6386	0.9442	0.3292
$\bar{\rho}$	0.5914	0.1645	0.4992	0.3673	0.7815	0.5422	0.8183	0.2795

* Backward lexicographical ordering of grid points

Table 4: CPU Time Needed by One Multigrid Cycle on 128 x 128 Grid

Smoother	SCGS	CLGS	SILU	CILU
t_c	25.0	23.4	28.9	56.3

Table 5. The Efficiency Factor E_f for All Test Cases

Smoother	SCGS	CLGS	SILU	CILU	SCGS	CLGS	SILU	CILU
	$Re = 1$				$Re = 1000$			
Problem 1	1.7	2.2	1.4	1.0	1.3	0.5	0.6	1.0
Problem 2	2.2	2.3	0.7	1.0	1.9	1.8	0.6	1.0
Problem 3	1.2	4.3	1.4	1.0	0.4	1.2	0.3	1.0

Table 6: Reduction Factors of the Multigrid Method Using CILU and the Hybrid Prolongation for Various Problems on the 128×128 Grids

Problem	Problem 1	Problem 2	Problem 3*
$Re = 1$			
ω	1.0	1.0	0.2
l_f	7	7	7
i	16	16	16
ρ_i	0.5878	0.7986	0.7560
ρ_{i+1}	0.5900	0.8028	0.7538
ρ_{i+2}	0.5919	0.8064	0.7520
ρ_{i+3}	0.5935	0.8095	0.7504
ρ_{i+4}	0.5949	0.8121	0.7492
r_0	$0.1296 \times 10^{+02}$	$0.2592 \times 10^{+02}$	$0.1802 \times 10^{+02}$
r_{i+4}	0.2396×10^{-05}	0.9662×10^{-03}	0.3300×10^{-02}
$\bar{\rho}$	0.4606	0.6006	0.6503
$Re = 1000$			
ω	1.1	1.0	0.7
l_f	7	7	7
i	16	16	16
ρ_i	0.3732	0.3282	0.3286
ρ_{i+1}	0.3778	0.3159	0.3282
ρ_{i+2}	0.3616	0.3222	0.3267
ρ_{i+3}	0.3746	0.3629	0.3253
ρ_{i+4}	0.3861	0.3837	0.3278
r_0	0.1605×10^{-01}	0.2697×10^{-01}	0.1969×10^{-01}
r_{i+4}	0.1491×10^{-12}	0.8231×10^{-12}	0.3485×10^{-12}
$\bar{\rho}$	0.2808	0.2980	0.2900

* Backward lexicographical ordering of grid points

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	November 1993	Conference Publication

4. TITLE AND SUBTITLE
Sixth Copper Mountain Conference on Multigrid Methods

5. FUNDING NUMBERS
WU 505-59-53-01

6. AUTHOR(S)
N. Duane Melson, T. A. Manteuffel, and S. F. McCormick, editors

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
NASA Langley Research Center
Hampton, VA 23681-0001

8. PERFORMING ORGANIZATION REPORT NUMBER
L-17275

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
National Aeronautics and Space Administration, Washington, DC 20546-0001; Air Force Office of Scientific Research, Bolling AFB, Washington, DC 20338; the Department of Energy, Washington, DC 20585; and the National Science Foundation, Washington, DC 20550.

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
NASA CP-3224
Part 2

11. SUPPLEMENTARY NOTES
Organizing Institutions: University of Colorado at Denver, Front Range Scientific Computations, Inc., and the Society for Industrial and Applied Mathematics.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Unclassified–Unlimited

Subject Category 64

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The Sixth Copper Mountain Conference on Multigrid Methods was held on April 4–9, 1993, at Copper Mountain, Colorado. This book is a collection of many of the papers presented at the conference and so represents the conference proceedings. NASA Langley graciously provided printing of this document so that all of the papers could be presented in a single forum. Each paper was reviewed by a member of the conference organizing committee under the coordination of the editors.

The multigrid discipline continues to expand and mature, as is evident from these proceedings. The vibrancy in this field is amply expressed in these important papers, and the collection clearly shows its rapid trend to further diversity and depth.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Multigrid; Algorithms; CFD	346
	16. PRICE CODE A15

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	