SIAM J. NUMER. ANAL.
Vol. 31, No. 6, pp. 1764-1784, December 1994
© 1994 Society for Industrial and Applied Mathematics
012

MULTIGRID SOLUTION OF THE INCOMPRESSIBLE NAVIER–STOKES EQUATIONS IN GENERAL COORDINATES*

S. ZENG† AND P. WESSELING†

Dedicated to Seymour Parter on the occasion of his 65th birthday.

Abstract. Galerkin coarse grid approximation (GCA) in multigrid methods is investigated for the incompressible Navier–Stokes equations in general coordinates. An efficient algorithm performing GCA is presented. The behavior of coarse grid matrices is studied under GCA with different transfer operators. For square and L-shaped driven cavity problems, the performance of the multigrid method using different combinations of transfer operators for the computation of coarse grid matrices and of coarse grid correction is investigated. Further computations are carried out in general coordinates for a channel flow problem with backward facing step in three dimensions.

Key words. multigrid methods, incompressible Navier–Stokes equations, staggered grid, general coordinates

AMS subject classifications. 65N55, 76D05

1. Introduction.

In multigrid methods for partial differential equations, a sequence of computational grids is used simultaneously. The finest grid discretization has to be approximated on the coarse grids. There are two ways of doing this: by discretization of the differential problem (discretization coarse grid approximation (DCA)) or by algebraic manipulation of the finest grid discretization (one way of doing this is called Galerkin coarse grid approximation (GCA)). Which of the two is best depends on circumstances. GCA is practically limited to linear problems, so that iterations on nonlinearities have to be taken outside multigrid, which is used as a linear systems solver; more (but perhaps cheaper, especially on vector computers) iterations may be required than with the nonlinear multigrid method. With GCA the discretization and solution phases are clearly separated, which leads to better structured, maintainable, and reusable software; on the other hand, the future may belong to intertwined discretization and solution in adaptive computing strategies. DCA may be inaccurate on the very coarse grids that normally occur in multigrid, leading to bad multigrid convergence; an example is given in [17]. For equations with discontinuous coefficients GCA is the method of choice [1], [3], [6], [7], [8], [19], [20], [23]. For systems of equations, efficient and correct implementation of GCA requires some care. This paper intends to contribute to this effort for the particular case of the incompressible Navier–Stokes equations in general nonorthogonal boundary-fitted coordinates discretized with the finite volume method on a staggered grid. In this case incorporation of the discretization method inside multigrid causes practical problems, because discretization in general coordinates with staggered arrangement of unknowns is a complicated affair; for more on this, see [10], [15], [21], and references quoted there. Improvements in discretization and boundary condition implementation lead to code changes in many places in a nonlinear multigrid code, and the vectorization potential of the code is not great. Nevertheless, this approach is also pursued (see [12], [11], and [13]), but here we concentrate on linear multigrid, using GCA. Although we have

* Received by the editors March 1, 1993; accepted for publication (in revised form) December 1, 1993.

† Faculty of Technical Mathematics and Informatics, Delft University of Technology, P. O. Box 5031, 2600 GA Delft, the Netherlands.

the incompressible Navier–Stokes equations in mind, the principles of the approach to be described carry over to systems of partial differential equations in general.

How the efficiencies of linear and nonlinear multigrid compare depends on circumstances, especially the rate of convergence of the nonlinear outer iteration that accompanies linear multigrid. For the incompressible Navier–Stokes equations, Newton converges fast, but Picard does so only for low Reynolds numbers. In any case, in our circumstances the convenience in software development that arises from separating the discretization and solution phases is worth a price.

We have thought it useful to include many practical details.

2. Linear multigrid applied to nonlinear problems. For our present purpose it suffices to consider two-grid methods. Let there be given a fine grid $\mathcal{G}$ and a coarse grid $\bar{\mathcal{G}}$. Let U be the space of fine grid functions, and $\bar{U}$ the space of coarse grid functions:

$$U = \{u : \mathcal{G} \longmapsto \mathcal{R}\}, \qquad \bar{U} = \{\bar{u} : \bar{\mathcal{G}} \longmapsto \mathcal{R}\}. \tag{1}$$

After discretization of the partial differential equation under consideration and linearization, a linear system of algebraic equations on the fine grid $\mathcal{G}$ is obtained, which is denoted by

$$Au = b. \tag{2}$$

This linear problem can be solved with a linear multigrid method as follows. Let $P : \bar{U} \longmapsto U$ and $R : U \longmapsto \bar{U}$ be a prolongation and a restriction operator, respectively, and let $S(u; A; b)$ stand for a call to a smoothing subroutine, i.e., a specified small number of iterations with a basic iterative method; u is both the initial guess (input) and the improved approximation (output). The linear two-grid method is given by the following algorithm.

ALGORITHM TWO-GRID METHOD
Choose u
begin
$S(u; A; b)$
$\bar{b} = R(b - Au)$
Solve $\bar{A}\bar{u} = \bar{b}$
$u := u + P\bar{u}$
$S(u; A; b)$
end Two-Grid Method

With GCA, $\bar{A}$ is given by

$$\bar{A} = RAP. \tag{3}$$

This is called Galerkin coarse grid approximation for the following reason. The problem to be solved is equivalent to the following variational formulation:

$$(Au, v) = (b, v), \quad u \in U \quad \forall v \in U \tag{4}$$

with $(\cdot, \cdot)$ the standard inner product. The Galerkin method consists of looking for an approximation of u in a lower-dimensional space, and restricting the test functions v to a lower-dimensional space. In multigrid we approximate u by $P\bar{u}, \bar{u} \in \bar{U}$, and restrict

v to $v = R^*\bar{v}, \bar{v} \in \bar{U}$, with R^* the adjoint of R, so that we have $(v, R^*\bar{w}) = (Rv, \bar{w})$ $\forall v \in U$ and $\forall \bar{w} \in \bar{U}$. This means that (4) is approximated by

$$(AP\bar{u}, R^*\bar{v}) = (b, R^*\bar{v}), \quad \bar{u} \in \bar{U} \quad \forall \bar{v} \in \bar{U} \tag{5}$$

or

$$(RAP\bar{u}, \bar{v}) = (Rb, \bar{v}), \quad \bar{u} \in \bar{U} \quad \forall \bar{v} \in \bar{U}, \tag{6}$$

which is equivalent to

$$\bar{A}\bar{u} = Rb \tag{7}$$

with $\bar{A} = RAP$. Of course, in the two-grid method b in (4)–(7) is to be replaced by $b - Au$.

3. Stencil notation. One of the goals of this paper is to discuss the efficient computation of $\bar{A} = RAP$ for systems of differential equations, particularly the incompressible Navier–Stokes equations in general coordinates on staggered grids. As a preliminary it is useful to specify a system of stencil notation; more on this in [20], to which we refer for proofs or derivations that are not outlined here. In the case of a single unknown grid function u the action of a linear operator A can be represented by

$$(Au)_i = \sum_j A(i,j)u_{i+j}, \tag{8}$$

where $i = (i_1, i_2, \ldots, i_d) \in \mathcal{G} \subset \mathcal{Z}^d$ are grid point indices, d is the number of space dimensions of $\mathcal{G}$, and $\mathcal{G}$ is conveniently identified with its set of grid point indices. The operator A is fully specified by the numbers $A(i,j)$. Often, these numbers are presented in a certain pattern called stencil, which is why (8) is called stencil notation, as opposed to the familiar matrix-vector notation of linear algebra. An example is the classic stencil for the approximation of the Laplace equation in two dimensions:

$$[A] = \begin{bmatrix} & -1 & \\ -1 & 4 & -1 \\ & -1 & \end{bmatrix}, \tag{9}$$

which does not depend on i when i is in the interior of $\mathcal{G}$. Equation (9) means that $A(i,0) = 4$, $A(i,(\pm 1,0)) = A(i,(0,\pm 1)) = -1$, while all other values of $A(i,j)$ are zero. The structure of a stencil is defined in the following way.

DEFINITION 1. *The structure S_A of a linear operator A is defined by*

$$S_A = \{j \in \mathcal{Z}^d : \exists i \in \mathcal{G} | A(i,j) \neq 0\}. \tag{10}$$

Hence, the structure corresponding to (9) can be represented as

$$[A] = \begin{bmatrix} & * & \\ * & * & * \\ & * & \end{bmatrix}. \tag{11}$$

The arrays in (9) and (11) and the numbers $A(i,j)$ are all referred to as the stencil $[A]$ of A; this is not found to be confusing.

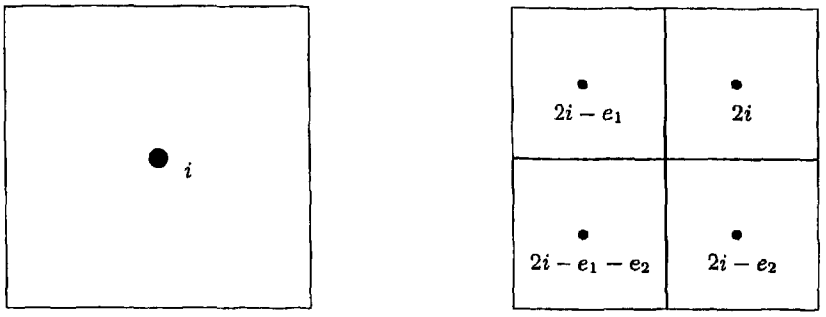

FIG. 1. *Coarse and fine grid cells.*

In order to specify a stencil notation for a restriction operator $\boldsymbol{R}$ a correspondence between the grid point numberings on $\mathcal{G}$ and $\tilde{\mathcal{G}}$ has to be given. A two-dimensional example is given in Fig. 1. Here we have cell centered coarsening: coarse grid cells are unions of fine grid cells, and the grid points are the cell centers. The coarse grid cell with center in i consists of the fine grid cells with centers in $2i$, $2i - e_\alpha$, $2i - e_1 - e_2$, with $e_1 = (1,0)$ and $e_2 = (0,1)$. Stencil notation of $\boldsymbol{R}$ can be given as

$$(\boldsymbol{R}u)_i = \sum_j \boldsymbol{R}(i,j)u_{2i+j}. \tag{12}$$

An example is

$$[\boldsymbol{R}] = \begin{bmatrix} 1 & \underline{1} \\ 1 & 1 \end{bmatrix}. \tag{13}$$

Indicating the element corresponding to $j = 0$ (here done by an underscore) is necessary to completely specify the correspondence between (13) and $\boldsymbol{R}(i,j)$. Here we have $\boldsymbol{R}(i,0) = \boldsymbol{R}(i,-e_1) = \boldsymbol{R}(i,-e_2) = \boldsymbol{R}(i,-e_1 - e_2) = 1$. We see that $(\boldsymbol{R}u)_i$ is obtained by summing the values in the four fine cells that lie in the coarse cell i.

It can be shown that the adjoint $\boldsymbol{R}^*$ of $\boldsymbol{R}$ satisfies

$$(\boldsymbol{R}^*\tilde{u})_i = \sum_j \boldsymbol{R}(j, i-2j)\tilde{u}_j. \tag{14}$$

Since $\boldsymbol{R}^* : \tilde{U} \longmapsto U$ is a prolongation, (14) immediately suggests the following stencil notation for prolongation operators:

$$(\boldsymbol{P}\tilde{u})_i = \sum_j \boldsymbol{P}^*(j, i-2j)\tilde{u}_j. \tag{15}$$

After specifying how $\boldsymbol{P}\tilde{u}$ is to be determined (for example, by interpolation), $\boldsymbol{P}^*(i,j)$ can be determined as follows. Let $\tilde{\delta}^i \in \tilde{U}$ be equal to 1 in $i \in \tilde{\mathcal{G}}$, and zero elsewhere. Then

$$\boldsymbol{P}^*(i,j) = (\boldsymbol{P}\tilde{\delta}^i)_{2i+j}. \tag{16}$$

For example, bilinear interpolation gives, in the cell-centered case,

$$[\boldsymbol{P}^*] = \frac{1}{16}\begin{bmatrix} 1 & 3 & 3 & 1 \\ 3 & 9 & \underline{9} & 3 \\ 3 & 9 & 9 & 3 \\ 1 & 3 & 3 & 1 \end{bmatrix}. \tag{17}$$

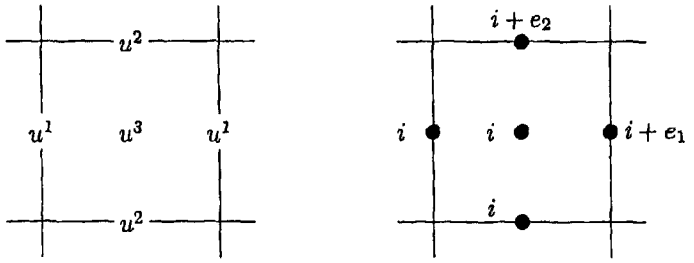

FIG. 2. *Staggered grid for the Navier–Stokes equations, and numbering system.*

At the same time, $\boldsymbol{R} = \boldsymbol{P}^*$ gives us another example of a restriction.

Extension to systems of differential equations is straightforward, but suitable notation must be developed. Let there now be M unknowns $u^1, u^2, \ldots, u^M$. In order to accomodate discretizations on staggered grids it is assumed that $u^m : \mathcal{G}^m \longmapsto \mathcal{R}$, $\mathcal{G}^m$ not necessarily identical to $\mathcal{G}^n$, $m \neq n$. An example is the staggered grid for the incompressible Navier–Stokes equations given in Fig. 2. $\mathcal{G}^1$ consists of the centers of vertical cell sides, $\mathcal{G}^2$ the centers of horizontal cell sides, and $\mathcal{G}^3$ cell centers. A convenient grid point numbering system is as follows: if the center of a cell has index $i = (i_1, i_2)$, then the centers of the lower and left sides also receive this index (cf. Fig. 2). Extension to three dimensions is obvious. The discretized system to be solved can be denoted as

$$\sum_n \boldsymbol{A}^{mn}\boldsymbol{u}^n = \boldsymbol{b}^m, \qquad m = 1, 2, \ldots, M, \tag{18}$$

with $\boldsymbol{A}^{mn} : \boldsymbol{U}^n \longmapsto \boldsymbol{U}^m$, $\boldsymbol{U}^m = \{u^m\}$, and $M = 3$ for Navier–Stokes (in two dimensions). In stencil notation, (18) becomes

$$\sum_n \sum_j \boldsymbol{A}^{mn}(i,j)u^n_{i+j} = b^m_i, \qquad m = 1, 2, \ldots, M, \tag{19}$$

where i in $\boldsymbol{A}^{mn}(i,j)$ refers to a point in $\mathcal{G}^m$, and i in u^n_{i+j} to the corresponding point in $\mathcal{G}^n$. For example, let the third equation be the continuity equation

$$u^1_{i+e_1} - u^1_i + u^2_{i+e_2} - u^2_i = 0. \tag{20}$$

Then

$$[\boldsymbol{A}^{31}]_i = \begin{bmatrix} \underline{-1} & 1 \end{bmatrix}, \quad [\boldsymbol{A}^{32}]_i = \begin{bmatrix} 1 \\ \underline{-1} \end{bmatrix}, \quad [\boldsymbol{A}^{33}]_i = 0. \tag{21}$$

4. The incompressible Navier–Stokes equations in general coordinates. Most of our considerations carry over to general systems of differential equations, but we will focus primarily on the incompressible Navier–Stokes equations, describing incompressible viscous flows. In order to be able to handle flows in complicated geometries, general boundary-fitted coordinates are used. In general coordinates, the incompressible Navier–Stokes equations are given in standard tensor notation by

$$\frac{\partial U^\alpha}{\partial t} + U^\beta U^\alpha_{,\beta} = -g^{\alpha\beta}p_{,\beta} + \mathrm{Re}^{-1}\left(g^{\beta\gamma}U^\alpha_{,\beta} + g^{\alpha\beta}U^\gamma_{,\beta}\right)_{,\gamma}, \tag{22}$$

$$U^\alpha_{,\alpha} = 0, \tag{23}$$

where U^α is the velocity vector field, p the pressure, Re the (dimensionless) Reynolds number, and $g^{\alpha\beta}$ the metric tensor. The range of Greek indices is $\{1,2,\ldots,d\}$. Summation takes place over twice repeated indices belonging to a variable or a product of variables. The reader not versed in tensor analysis may simply interpret "$,\alpha$" as $\partial/\partial x^\alpha$, which gives the correct form in Cartesian coordinates.

Fig. 2 shows part of the uniform computational grid, which is the image under a boundary-fitted coordinate mapping of a general curvilinear nonorthogonal grid in physical space, an example of which is given in Fig. 6. The grid in Fig. 2 is staggered; u^1, u^2, and u^3 correspond to $\sqrt{g}U^1$, $\sqrt{g}U^2$, and p, respectively, with $\sqrt{g}$ the Jacobian of the mapping. The discretization of (22) and (23) is discussed in [10], [21], and [11]; special measures, such as the use of $\sqrt{g}U^\alpha$ as unknown, have to be taken to ensure accuracy on strongly nonuniform grids. With Euler backward time discretization and Newton linearization of the nonlinear terms, a discrete system of the following form is obtained:

$$\text{(24)}\quad \begin{pmatrix} u^1 \\ u^2 \end{pmatrix}^{n+1} - \begin{pmatrix} u^1 \\ u^2 \end{pmatrix}^{n} = \begin{pmatrix} f^1 \\ f^2 \end{pmatrix}^{n} - \begin{pmatrix} A^{11} & A^{12} & A^{13} \\ A^{21} & A^{22} & A^{23} \end{pmatrix} \begin{pmatrix} u^1 \\ u^2 \\ u^3 \end{pmatrix}^{n+1},$$

$$\text{(25)}\quad \begin{pmatrix} A^{31} & A^{32} \end{pmatrix} \begin{pmatrix} u^1 \\ u^2 \end{pmatrix}^{n+1} = 0,$$

where u^1, u^2, and u^3 approximate $\sqrt{g}\,U^1$, $\sqrt{g}\,U^2$, and p, respectively. In the nonstationary case, the spatial derivatives are approximated with central differences; in the stationary case, obtained by replacing the left-hand side of (24) by zero, the convection terms are approximated by the hybrid scheme [16], switching between central and upwind depending on the mesh-Reynolds number. Furthermore, in the stationary case we use Picard linearization instead of Newton (lagging the coefficients). The structures of the discrete operators are given by

$$\text{(26)}\quad A = \begin{pmatrix} A^{11} & A^{12} & A^{13} \\ A^{21} & A^{22} & A^{23} \\ A^{31} & A^{32} & 0 \end{pmatrix},$$

with

$$\text{(27)}\quad [A^{11}] = \begin{bmatrix} * & * & * \\ * & \underline{*} & * \\ * & * & * \end{bmatrix}, \quad [A^{12}] = \begin{bmatrix} * & * & * & * \\ * & * & \underline{*} & * \end{bmatrix}, \quad [A^{13}] = \begin{bmatrix} * & * \\ * & \underline{*} \\ * & * \end{bmatrix},$$

$$[A^{21}] = \begin{bmatrix} * & * \\ \underline{*} & * \\ * & * \\ * & * \end{bmatrix}, \quad [A^{22}] = \begin{bmatrix} * & * & * \\ * & \underline{*} & * \\ * & * & * \end{bmatrix}, \quad [A^{23}] = \begin{bmatrix} * & \underline{*} & * \\ * & * & * \end{bmatrix},$$

whereas $[A^{31}]$ and $[A^{32}]$ have been given in (21). Extension to three dimensions is straightforward.

5. Prolongation and restriction. We construct coarse grids by taking unions of fine grid cells, as illustrated in Fig. 1. In Fig. 3 we present a cell of the grid $\bar{\mathcal{G}}^1$ and the corresponding four cells for the variables $\bar{u}^1$ and u^1. Let $P^1 : \bar{U}^1 \longmapsto U^1$ be

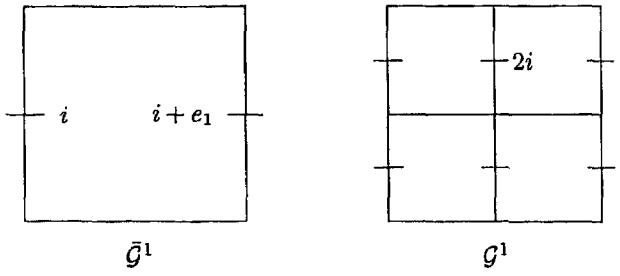

FIG. 3. *A cell of* $\bar{\mathcal{G}}^1$ *and the corresponding four cells of* $\mathcal{G}^1$; *the grid points are indicated by* —.

defined by bilinear interpolation. This means, for example, that u^1_{2i} is obtained from $\bar{u}^1_i$, $\bar{u}^1_{i+e_1}$, $\bar{u}^1_{i+e_2}$, $\bar{u}^1_{i+e_1+e_2}$. Using equation (16) one easily obtains

$$[P^{1*}] = \frac{1}{8}\begin{bmatrix} 1 & 2 & 1 \\ 3 & 6 & \underline{3} \\ 3 & 6 & 3 \\ 1 & 2 & 1 \end{bmatrix}. \tag{28}$$

If one of the elements of P^{1*} refers to a grid point outside $\mathcal{G}^1$ this element can be replaced by zero, but we have found that somewhat better results are obtained with

$$[P^{1*}] = \frac{1}{8}\begin{bmatrix} nw & 2n & ne \\ (4-n)w & 2(4-n) & \underline{(4-n)e} \\ (4-s)w & 2(4-s) & (4-s)e \\ sw & 2s & se \end{bmatrix}. \tag{29}$$

Here $n = 0$ on the "north" boundary and $n = 1$ elsewhere, and similarly for s, e, and w on the south boundary, etc. Equation (29) is obtained by constant extrapolation of the coarse grid quantity to be interpolated outside the domain. The corresponding version for P^2 is found to be

$$[P^{2*}] = \frac{1}{8}\begin{bmatrix} nw & (4-w)n & \underline{(4-e)n} & ne \\ 2w & 2(4-w) & 2(4-e) & 2e \\ sw & (4-w)s & (4-e)s & se \end{bmatrix}. \tag{30}$$

Another possibility, to be called hybrid interpolation, is to use linear interpolation in ξ^1 but first-order interpolation in ξ^2, resulting in

$$P^{1*} = \frac{1}{2}\begin{bmatrix} w & 2 & \underline{e} \\ w & 2 & e \end{bmatrix}, \qquad P^{2*} = \frac{1}{2}\begin{bmatrix} n & \underline{n} \\ 2 & 2 \\ s & s \end{bmatrix}. \tag{31}$$

For prolongation of $\bar{u}^3$ we use zeroth-order interpolation:

$$[P^{3*}] = \begin{bmatrix} 1 & \underline{1} \\ 1 & 1 \end{bmatrix}. \tag{32}$$

Restrictions are chosen as follows. Staggered grid discretization is performed by using staggered finite volumes, as illustrated in Fig. 4. Restriction is applied to residuals,

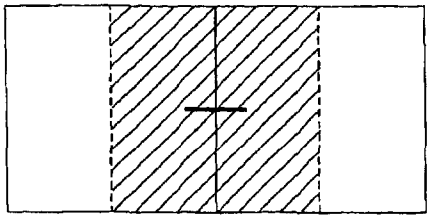

FIG. 4. *Staggered finite volume (hatched) for the u^1 equation.*

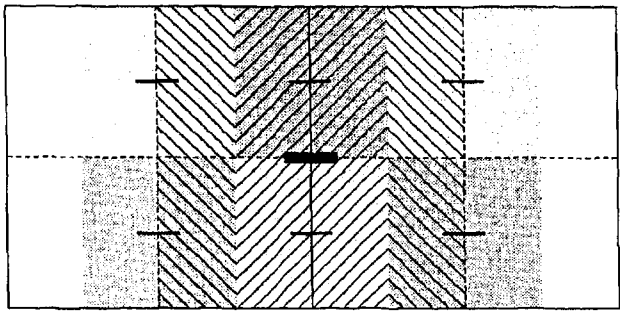

FIG. 5. *Fine grid staggered finite volumes contained wholly (////) or partly (\\\\) in coarse grid staggered finite volume.*

and is performed by averaging the residuals in the fine grid staggered volumes that are (partly) contained in the coarse grid staggered finite volume under consideration, weighted with the part of the fine grid finite volume that is included; cf. Fig. 5. This leads to $\boldsymbol{R}^m = \boldsymbol{P}^{m*}$, $m = 1, 2$, with $\boldsymbol{P}^{m*}$ given by (31).

Prolongation and restriction should be sufficiently accurate. A well-known rule for this is as follows [2], [4]. Let the order m_P of a prolongation be defined as the highest degree of polynomials that are interpolated exactly, plus one. The order m_R of a restriction is one plus the highest degree of polynomials that are interpolated exactly by $\boldsymbol{R}^*$ after suitable scaling. Then the rule for a single differential equation of order M is

$$m_P + m_R > M. \tag{33}$$

This is a sufficient condition. In [18] a numerical example is given of slow multigrid convergence in a case where

$$m_P + m_R = M. \tag{34}$$

What is really needed is that the coarse grid correction does not amplify short wavelength error components. Hemker [5] shows that (34) is necessary and sufficient with a different definition of m_P and m_R, referring to short wavelength behavior. Here we refrain from further analysis. The extension to systems would seem to be (cf. [4, §11.1.1]) that if $\boldsymbol{A}^{mn}$ is a discretization of a differential operator of order M^{mn}, then one should have at least

$$m_{P^n} + m_{R^m} \geq M^{mn} \tag{35}$$

or, better still,

$$m_{P^n} + m_{R^m} > M^{mn}. \tag{36}$$

In the Navier–Stokes case we have $M^{11} = M^{22} = 2$, $M^{12} \leq 2$, $M^{21} \leq 2$ (depending on circumstances: Cartesian or general coordinates), $M^{3n} = M^{n3} = 1$. For (29) and (30) we have $m_P = 2$; for (31) and (32) we have $m_P = 1$. By numerical experiments we have determined suitable choices for $\boldsymbol{P}$ and $\boldsymbol{R}$, which will be discussed later.

6. Efficient programming of RAP. The GCA of $\boldsymbol{A}^{mn}$ is

$$\bar{\boldsymbol{A}}^{mn} = \boldsymbol{R}^m \boldsymbol{A}^{mn} \boldsymbol{P}^n. \tag{37}$$

A generalization is

$$\bar{\boldsymbol{A}}^{mn} = \boldsymbol{R}^{mn} \boldsymbol{A}^{mn} \boldsymbol{P}^{mn}, \tag{38}$$

with $\boldsymbol{R}^{mn} : \boldsymbol{U}^m \longmapsto \bar{\boldsymbol{U}}^m$, $\boldsymbol{P}^{mn} : \bar{\boldsymbol{U}}^n \longmapsto \boldsymbol{U}^n$. With (38) the variational interpretation of §2 does not apply, but this is of minor consequence if the method works well in practice. We will now discuss algorithmic details of (38). For this purpose, the superscripts can be deleted.

In stencil notation we have, using (8), (12), and (15),

$$\bar{\boldsymbol{A}}(i,n) = \sum_{m \in \mathcal{S}_R} \sum_{k \in \mathcal{S}_A} \boldsymbol{R}(i,m) \boldsymbol{A}(2i+m,k) \boldsymbol{P}^*(i+n, m+k-2n). \tag{39}$$

In order to determine $\mathcal{S}_{\bar{A}}$ the following algorithm may be used.

```
ALGORITHM STRURAP
comment Calculation of S_Ā
begin S_Ā = ∅
    for q ∈ S_P· do
        for m ∈ S_R do
            for k ∈ S_A do
                n = (m + k − q)/2
                if n ∈ Z^d then
                    S_Ā = S_Ā ∪ n
                end if
    od od od
end STRURAP
```

Here $\mathcal{Z} = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$ and d is the number of space dimensions.

We now turn to the algorithmic implementation of (39). It is slightly more convenient to consider the following equivalent form:

$$\bar{\boldsymbol{A}}(i,n) = \sum_{m \in \mathcal{S}_R} \sum_{q \in \mathcal{S}_{P^*}} \boldsymbol{R}(i,m) \boldsymbol{A}(2i+m, q+2n-m) \boldsymbol{P}^*(i+n, q). \tag{40}$$

Equation (40) leads to nested do-loops over the d-tuples i, n, m, and q. For high computing efficiency the longest loop, i.e., the loop over $\bar{\mathcal{G}}$, should be the innermost loop. Tests on whether $2i + m \in \mathcal{G}$ and $i + n \in \bar{\mathcal{G}}$ should not occur in the inner loop for vectorization. Therefore, the correct set (called $\bar{\mathcal{G}}_2$) over which i varies is determined beforehand. Furthermore, $\boldsymbol{R}(i,m)$ and $\boldsymbol{P}^*(i,m)$ are not stored because

they are constant in the interior and change only near boundaries. We therefore divide $\bar{\mathcal{G}}_2$ into parts $\bar{\mathcal{G}}_2(b)$, $b \in \mathcal{B}$, with $\mathcal{B}$ some index set, in each of which $\boldsymbol{R}(i,m)$ and $\boldsymbol{P}^*(i+n,q)$ are constant; their values will be denoted by $\boldsymbol{R}_b(m)$ and $\boldsymbol{P}^*_b(n,q)$. This leads to the following algorithm.

```
ALGORITHM RAP
comment Calculation of Ā = RAP
begin Ā = 0
    for m ∈ S_R do
        Ḡ_1 = {i ∈ Ḡ : 2i + m ∈ G}
        for n ∈ S_Ā do
            Ḡ_2 = {i ∈ Ḡ_1 : i + n ∈ Ḡ}
            for q ∈ S_P* do
                k = q + 2n − m
                if k ∈ S_A then
                    for b ∈ B do
                        μ = R_b(m) P*_b(n, q)
                        for i ∈ Ḡ_2(b) do
                            Ā(i, n) = Ā(i, n) + μ A(2i + m, k)
                    od od
                end if
    od od od
end RAP
```

It is necessary to be more specific about the partitioning $\bar{\mathcal{G}}_2(b)$ of $\bar{\mathcal{G}}_2$ and about the determination of $\boldsymbol{R}_b(m)$ and $\boldsymbol{P}^*_b(n,q)$. Let

$$\bar{\mathcal{G}}_2 = \prod_{c=1}^{d} \{ib(c), ib(c)+1, \ldots, ie(c)\}. \tag{41}$$

Define

$$\bar{\mathcal{G}}_2(b) = \prod_{c=1}^{d} \bar{\mathcal{G}}_2(b_c, c), \quad b = (b_1, b_2, \ldots, b_d), \quad b_c = -1, 0, 1, \tag{42}$$

with $\bar{\mathcal{G}}_2(-1,c) = ib(c)$, $\bar{\mathcal{G}}_2(1,c) = ie(c)$, $\bar{\mathcal{G}}_2(0,c) = \{ib(c)+1, ib(c)+2, \ldots, ie(c)-1\}$. It is easily seen that $\bar{\mathcal{G}}_2 = \bigcup_{b\in\mathcal{B}} \bar{\mathcal{G}}_2(b)$ with $\mathcal{B} = \prod^d \{-1,0,1\}$. Next we assume that $\boldsymbol{R}$ and $\boldsymbol{P}^*$ are constant (but perhaps different) on the sides, corners, and interior of $\bar{\mathcal{G}}$. All prolongations and restrictions discussed in §5 satisfy this assumption. Since $\bar{\mathcal{G}}_2 \subset \bar{\mathcal{G}}$ the same is true for $\bar{\mathcal{G}}_2(b)$. For $\boldsymbol{R}_b(m)$ one simply checks whether $\bar{\mathcal{G}}_2(b)$ is in the interior or on a boundary of $\bar{\mathcal{G}}$, and selects the corresponding value of $\boldsymbol{R}(i,m)$ accordingly. With $i \in \bar{\mathcal{G}}_2(b)$, $i+n$ is in a part of $\bar{\mathcal{G}}$ where $\boldsymbol{P}^*(i+n,q)$ is constant, as is seen after some reflection; this is the value selected for $\boldsymbol{P}^*_b(n,q)$.

Extension to systems of differential equations is done block-by-block, according to (38); this needs no further discussion.

7. Properties of the coarse grid operator. For efficient multigrid convergence the coarse grid should have the approximation property, as defined and studied in [4] for the case of a single differential equation; for an elementary introduction, see [20]. For systems of equations the approximation property is analyzed in [22]. For the Stokes equations (obtained from the Navier–Stokes equations by deletion of

the nonlinear terms) discretized on a staggered grid (as described before), it is shown in [22] that with GCA we have the approximation property if $\boldsymbol{P}^m$ corresponds to biquadratic interpolation and if $\boldsymbol{R}^m = \boldsymbol{P}^{m*}$. However, our numerical experience indicates that this condition (much stronger than (36)) is not necessary. This is fortunate, because with these transfer operators the structure of the stencil of the operator grows under GCA, leading to a superlinear dependence of computing work on the number of unknowns.

The structure of $\bar{\boldsymbol{A}}^{mn}$ as given by (38), with $\boldsymbol{A}^{mn}$ given by (27) and (21), is as follows. We choose $\boldsymbol{R}^{mn} = \boldsymbol{R}^m = \boldsymbol{P}^{m*}$, $m, n = 1, 2, 3$ with $\boldsymbol{P}^{m*}$ given by (31) and (32). This means that $m_R = 1$. Next, $\boldsymbol{P}^{m3}$, $m = 1, 2$ is given by (32), and $\boldsymbol{P}^{3m}$, $m = 1, 2$ is given by (31). This gives $m_P = 1$. For $\boldsymbol{P}^{mn}$, $m, n = 1, 2$ we choose $\boldsymbol{P}^{mn} = \boldsymbol{P}^n$, with two possibilities: bilinear or hybrid interpolation. In the bilinear case $\boldsymbol{P}^n$ is given by (29), (30), giving $m_P = 2$; in the hybrid case $\boldsymbol{P}^n$ is given by (31), giving $m_P = 1$. We find that with all of these choices the structure of $\bar{\boldsymbol{A}}^{mn}$ equals that of $\boldsymbol{A}^{mn}$.

In order to realize efficient smoothing it is desirable that the operators $\boldsymbol{A}$ and $\bar{\boldsymbol{A}}$ of §2 correspond to M-matrices. In the Navier–Stokes case the structure of the system to be solved is given by (27) and (21). It is desirable that $\boldsymbol{A}^{11}$ and $\boldsymbol{A}^{22}$ correspond to M-matrices. This can be realized by upwind or hybrid differencing. It is also desirable that the coarse grid approximations $\bar{\boldsymbol{A}}^{11}$ and $\bar{\boldsymbol{A}}^{22}$ correspond to M-matrices, or retain at least as much diagonal dominance as possible. With GCA according to (38) this depends on $\boldsymbol{R}^{mm}$ and $\boldsymbol{P}^{mm}$. We will now investigate this. Neglecting the viscous term (which brings the operator closer to the M-matrix property), typical upwind stencils for $\boldsymbol{A}^{11}$ are

$$[\boldsymbol{A}^{11}] = \begin{bmatrix} -1 & \underline{1} \end{bmatrix}, \qquad [\boldsymbol{A}^{11}] = \begin{bmatrix} \underline{1} \\ -1 \end{bmatrix}, \tag{43}$$

corresponding to a horizontal and a vertical flow direction, respectively. By numerical experimentation we find that m-fold GCA according to

$$\boldsymbol{A}^{11(m)} = \boldsymbol{R}^{11}\boldsymbol{A}^{11(m-1)}\boldsymbol{P}^{11}, \qquad \boldsymbol{A}^{11(0)} = \boldsymbol{A}^{11}, \tag{44}$$

with $\boldsymbol{P}^{11}$ given by the first equation of (31) and $\boldsymbol{R}^{11} = \boldsymbol{P}^{11*}$ results in (in the interior)

$$[\boldsymbol{A}^{11(m)}] = \frac{1}{2}\begin{bmatrix} -1 & \underline{2} & -1 \end{bmatrix} + \frac{2^m}{2}\begin{bmatrix} -1 & \underline{0} & 1 \end{bmatrix} \tag{45}$$

in the first case of (43), and in

$$[\boldsymbol{A}^{11(m)}] = \frac{2^m}{6}\begin{bmatrix} 0 & 0 & 0 \\ 1 & \underline{4} & 1 \\ -1 & -4 & -1 \end{bmatrix} + \frac{2^{-m}}{6}\begin{bmatrix} 0 & 0 & 0 \\ -1 & \underline{2} & -1 \\ 1 & -2 & 1 \end{bmatrix} \tag{46}$$

in the second case. With $\boldsymbol{P}^{11}$ given by (29) and $\boldsymbol{R}^{11}$ as before we obtain

$$\begin{aligned}[\boldsymbol{A}^{11(m)}] = {} & \frac{2^m}{12}\begin{bmatrix} -1 & 0 & 1 \\ -4 & \underline{0} & 4 \\ -1 & 0 & 1 \end{bmatrix} + \frac{1}{12}\begin{bmatrix} -1 & 2 & -1 \\ -4 & \underline{8} & -4 \\ -1 & 2 & -1 \end{bmatrix} \\ & + \frac{2^{-m}}{12}\begin{bmatrix} 1 & 0 & -1 \\ -2 & \underline{0} & 2 \\ 1 & 0 & -1 \end{bmatrix} + \frac{4^{-m}}{12}\begin{bmatrix} 1 & -2 & 1 \\ -2 & \underline{4} & -2 \\ 1 & -2 & 1 \end{bmatrix}\end{aligned} \tag{47}$$

in the first case of (43), whereas the second case results in a stencil that is obtained from (47) by a 90° rotation.

A finite difference operator A will be called a K-matrix if

$$A(i,0) > 0, \quad A(i,j) \leq 0 \quad \text{for } j \neq 0, \quad \sum_j A(i,j) \geq 0. \tag{48}$$

In [20] it is shown that a K-operator corresponds to an M-matrix. We see that all of the coarse grid operators (45)–(47) deviate more from the K-property as m increases. Accordingly, smoothing performance might deteriorate on coarser grids. Furthermore, poor coarse grid approximation of certain smooth components may occur. These phenomena are well known for applications of GCA to convection-diffusion-type equations [23], [24]. A redeeming feature may be that certain smoothers also take care of certain smooth components. Whether these difficulties are serious for our applications will be investigated by numerical experiments to be described later.

For completeness we also show how a typical viscous term, given by

$$[A^{11}] = \begin{bmatrix} 0 & -1 & 0 \\ -1 & \underline{4} & -1 \\ 0 & -1 & 0 \end{bmatrix}, \tag{49}$$

transforms under GCA. We find, with P^{11} given by the first equation of (31) and $R^{11} = P^{11*}$ (violating rule (33) if the viscous term dominates):

$$[A^{11(m)}] = \frac{2^m}{6}\begin{bmatrix} -1 & -4 & -1 \\ 2 & \underline{8} & 2 \\ -1 & -4 & -1 \end{bmatrix} + \begin{bmatrix} -1 & \underline{2} & -1 \end{bmatrix} + \frac{2^{-m}}{6}\begin{bmatrix} 1 & -2 & 1 \\ -2 & \underline{4} & -2 \\ 1 & -2 & 1 \end{bmatrix}. \tag{50}$$

Note that the anisotropic nature of P^{11} and R^{11} causes the horizontal derivative to gradually disappear. With P^{11} given by (29) and R^{11} as before we find

$$[A^{11(m)}] = \frac{1}{3}\begin{bmatrix} -1 & -1 & -1 \\ -1 & \underline{8} & -1 \\ -1 & -1 & -1 \end{bmatrix} + \frac{4^{-m}}{3}\begin{bmatrix} 1 & -2 & 1 \\ -2 & \underline{4} & -2 \\ 1 & -2 & 1 \end{bmatrix}. \tag{51}$$

Note that in (51) the K-property is conserved, but in (50) it is lost.

Using central differences, we found that satisfactory smoothing could be obtained in the nonstationary case by choosing Δt small enough.

8. Numerical experiments. In §5 we have described two types of prolongation: bilinear and hybrid, denoted as type B and type H for short. Prolongation is used in the computation of the coarse grid matrix according to (37) and for the prolongation of corrections. These may be chosen differently. The prolongation for the coarse grid matrix will be denoted by P_A and that for the correction by P_C. We always use (32) as the prolongation operator for the pressure. Keeping this in mind, the notation $P_A(B)$ means that the bilinear prolongation was used for the coarse grid matrix, etc. Our concern here is that $P_A(H)$ violates (36); on the other hand, $P_A(B)$ bring us farther from the K-matrix property on the coarse grids in some situations. We will

describe some numerical experiments to see whether the prolongations described can be used to form suitable combinations of P_A and P_C.

Our first test problem is the two-dimensional square driven cavity flow problem, with a uniform computational grid. We use the stationary equations. The multigrid algorithm is designed as follows. The initial solution is chosen as zero. The W-cycle is used with one pre- and one postsmoothing. Since our main concern here is coarse grid approximation, it is not necessary to give details about the smoothing method, which was of distributive ILU type, not unlike the methods described in [22]. For details about the smoother, we refer to [25]. On the coarsest grid of size 2×2, exact solution takes place. The residual norm is measured by

$$r = \|b - Au\| = \left(\frac{1}{M} \sum_{m=1}^{M} \left(\sum_{j \in \mathcal{G}^m} (b^m - A^m u)_j^2 / N_g^m \right) \right)^{\frac{1}{2}}, \tag{52}$$

with M the number of unknowns and N_g^m the number of grid points in $\mathcal{G}^m$. Let r_0 be the initial residual norm on the finest grid before the multigrid iterations, and let r_i be the residual norm on the finest grid after i iterations. The average reduction factor $\bar{\rho}$ is defined by

$$\bar{\rho} = \left(\frac{r_n}{r_0} \right)^{\frac{1}{n}}. \tag{53}$$

The reduction factor ρ_i at the ith iteration is defined by

$$\rho_i = \frac{r_i}{r_{i-1}}. \tag{54}$$

First, two outer (Picard) iterations are carried out, each with one multigrid cycle as the inner iteration. In the third outer iteration 20 multigrid cycles are performed. Tables 1–6 present results. The numbers accompanied by a "*" are not accurate because machine accuracy was reached before the 20th cycle. Picard iteration was not continued beyond the third iteration because Picard is quite inefficient for high Reynolds numbers, but Newton iteration would have worked fine. Our purpose here is not to recompute this frequently computed cavity flow in Cartesian coordinates (as we have done in [26]), but merely to test our linear multigrid method on some representative systems. Complete calculations will be made after this for two other flows in general coordinates.

With $P_C(H)$ and $P_A(H)$, the algorithm works for both the low and the high Reynolds number cases. The reduction factor grows with refinement of grids, but the rate of convergence remains satisfactory. Apparently, the violation of (36) does not spoil convergence enough to make the method useless. With $P_C(B)$ and $P_A(B)$, the algorithm has very good convergence, except on the 128×128 grid for $\mathrm{Re} = 10,000$, where it diverges. The failure is observed to be caused by bad smoothing on coarse grids. This is explained by (47); a large departure from the K-matrix property occurs. Other combinations of P_C and P_A give results lying between those from the above two extreme choices. Generally, for a fixed P_A, $P_C(B)$ gives better results, and for a fixed P_C, $P_A(B)$ gives better results, if the algorithm works. $P_A(H)$ is more suitable for high Reynolds number cases, while $P_A(B)$ is more suitable for low Reynolds number cases.

Now we go to general coordinates and compute the flow in a two-dimensional L-shaped driven cavity, as illustrated in Fig. 6. A benchmark solution is available in

TABLE 1
Reduction factors for the square driven cavity problem (Re = 1, *finest grid* = 32 × 32, $r_0 = 2.433$).

P_C \ P_A		H	B
H	r_{20}	0.4228×10^{-07}	0.1495×10^{-10}
	$\bar{\rho}$	0.4093	0.2751
	ρ_{16}	0.4250	0.2760
	ρ_{17}	0.4253	0.2763
	ρ_{18}	0.4254	0.2763
	ρ_{19}	0.4255	0.2765
	ρ_{20}	0.4256	0.2765
B	r_{20}	0.7000×10^{-08}	0.1616×10^{-11}
	$\bar{\rho}$	0.3741	0.2461
	ρ_{16}	0.3961	0.2710
	ρ_{17}	0.3964	0.2713
	ρ_{18}	0.3966	0.2715
	ρ_{19}	0.3963	0.2714
	ρ_{20}	0.3970	0.2714

TABLE 2
Reduction factors for the square driven cavity problem (Re = 1, *finest grid* = 64 × 64, $r_0 = 9.449$).

P_C \ P_A		H	B
H	r_{20}	0.1163×10^{-04}	0.4889×10^{-09}
	$\bar{\rho}$	0.5064	0.3060
	ρ_{16}	0.5292	0.3051
	ρ_{17}	0.5299	0.3058
	ρ_{18}	0.5304	0.3058
	ρ_{19}	0.5308	0.3063
	ρ_{20}	0.5312	0.3064
B	r_{20}	0.1936×10^{-05}	0.4146×10^{-11}
	$\bar{\rho}$	0.4630	0.2411
	ρ_{16}	0.4971	0.2695
	ρ_{17}	0.4981	0.2706
	ρ_{18}	0.4990	0.2712
	ρ_{19}	0.4998	0.2717
	ρ_{20}	0.5005	0.2732

[14] for Re = 100 and Re = 1000. We now use central differencing for the convection terms and time-stepping. Due to time-stepping, we have a sequence of systems of equations at the corresponding time levels, which are to be solved by the linear multigrid method. With different combinations of P_C and P_A, our purpose now is to see whether the multigrid method is also applicable in general coordinates. So we solve the problem until steady state. The time required to reach steady state increases with Re. We simply took the final time $t = 20$, which was sufficient in all cases, based on numerical evidence. Because of central differencing, diagonal dominance is lost when the Reynolds number is sufficiently large, which could damage smoothing. Time-stepping increases the main diagonal, and so one may expect that a proper choice of the time step Δt could still maintain smoothing even if the Reynolds number is large. We found that time-stepping allows us to use central differences, which give better accuracy than upwind discretization. Based on some numerical experiments, $\Delta t = 0.2$ is used for Re = 1000. For Re = 100, Δt can be somewhat larger; here we choose $\Delta t = 1$ (these Δt are absolute values). At each time step, one multigrid

TABLE 3
Reduction factors for the square driven cavity problem (Re = 1, *finest grid* = 128 × 128, r_0 = 23.20).

P_C \ P_A		H	B
H	r_{20}	0.4281×10^{-03}	0.5581×10^{-08}
	$\bar{\rho}$	0.5798	0.3304
	ρ_{16}	0.6008	0.3293
	ρ_{17}	0.6019	0.3211
	ρ_{18}	0.6028	0.3317
	ρ_{19}	0.6036	0.3329
	ρ_{20}	0.6043	0.3335
B	r_{20}	0.7447×10^{-04}	0.8202×10^{-11}
	$\bar{\rho}$	0.5313	0.2385
	ρ_{16}	0.5679	0.2522
	ρ_{17}	0.5695	0.2571
	ρ_{18}	0.5709	0.2612
	ρ_{19}	0.5722	0.2646
	ρ_{20}	0.5733	0.2694

TABLE 4
Reduction factors for the square driven cavity problem (Re = 10,000, *finest grid* = 32×32, r_0 = 0.6799×10^{-1}).

P_C \ P_A		H	B
H	r_{20}	0.5454×10^{-12}	0.8591×10^{-13}
	$\bar{\rho}$	0.2788	0.2541
	ρ_{16}	0.3229	0.2699
	ρ_{17}	0.3239	0.2724
	ρ_{18}	0.3245	0.2750
	ρ_{19}	0.3247	0.2824*
	ρ_{20}	0.3241	0.3599*
B	r_{20}	0.7356×10^{-13}	0.8673×10^{-13}
	$\bar{\rho}$	0.2522	0.2543
	ρ_{16}	0.2731	0.2711
	ρ_{17}	0.2744	0.2735
	ρ_{18}	0.2764	0.2758
	ρ_{19}	0.2773	0.2837*
	ρ_{20}	0.2943*	0.3757*

iteration is carried out. Denote the residual norm defined by (52) for the stationary Navier–Stokes equations at $t = 0$ as r^0, and that at $t = 20$ as r^{20}. Figures 7 and 8 present the streamlines for Re = 100 and Re = 1000, respectively. The streamlines are in good agreement with those given in [14]. In Table 7, we give the reduction $\bar{\rho}$ factor of the one multigrid iteration at $t = 20$ on the 32 × 32, 64 × 64, and 128 × 128 grids, and the ratio r^{20}/r^0 as well, for all the combinations of $\boldsymbol{P}_C$ and $\boldsymbol{P}_A$; the numbers indicated by a "*" are obtained with a heavier underrelaxation in the smoothing method (not discussed here). The reduction factor gradually approaches a stable value after several time levels.

The conclusions are as follows. When the Reynolds number is large, for a fixed $\boldsymbol{P}_A$, $\boldsymbol{P}_C(B)$ gives better results, but for a fixed $\boldsymbol{P}_C$, $\boldsymbol{P}_A(B)$ sometimes leads to slightly worse results. Conversely, we have the same conclusion as for the square driven cavity problem when the Reynolds number is low. Again, $\boldsymbol{P}_A(H)$ is better for

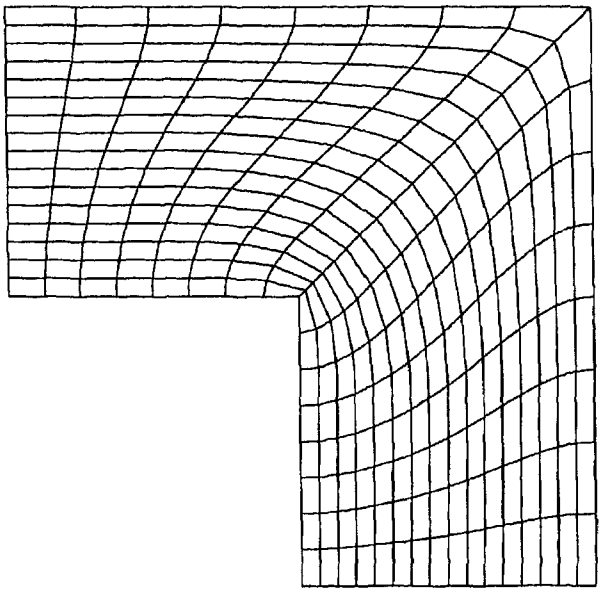

FIG. 6. *The L-shaped driven cavity with a biharmonic grid.*

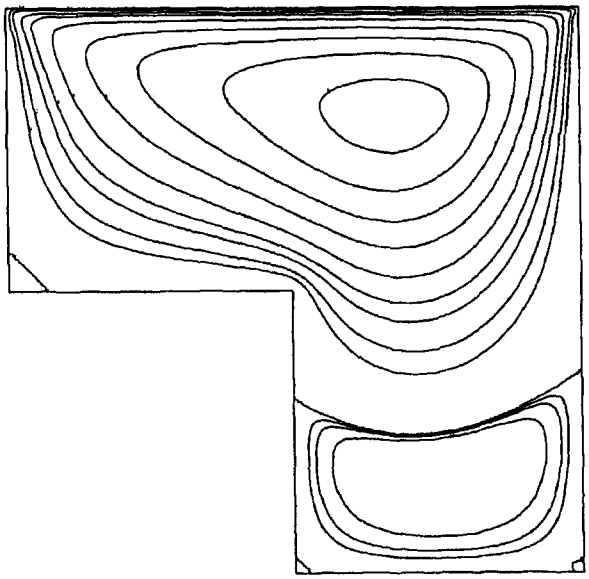

FIG. 7. *Streamlines for the L-shaped driven cavity problem* (Re = 100, *on a* 128 × 128 *grid*).

TABLE 5

Reduction factors for the square driven cavity problem (Re = 10,000, *finest grid* = 64×64, $r_0 = 0.1264$).

P_C \ P_A		H	B
H	r_{20}	0.2009×10^{-10}	0.1877×10^{-12}
	$\bar{\rho}$	0.3236	0.2562
	ρ_{16}	0.3988	0.2635
	ρ_{17}	0.4031	0.2652
	ρ_{18}	0.4045	0.2698
	ρ_{19}	0.4041	0.2974*
	ρ_{20}	0.4031	0.4996*
B	r_{20}	0.3652×10^{-11}	0.3543×10^{-12}
	$\bar{\rho}$	0.2972	0.2645
	ρ_{16}	0.3534	0.2648
	ρ_{17}	0.3530	0.2677
	ρ_{18}	0.3520	0.2815*
	ρ_{19}	0.3507	0.4062*
	ρ_{20}	0.3493	0.7557*

TABLE 6

Reduction factors for the square driven cavity problem (Re = 10,000, *finest grid* = 128×128, $r_0 = 0.1953$).

P_C \ P_A		H	B
H	r_{20}	0.3895×10^{-09}	
	$\bar{\rho}$	0.3637	
	ρ_{16}	0.4155	
	ρ_{17}	0.4211	div
	ρ_{18}	0.4245	
	ρ_{19}	0.4261	
	ρ_{20}	0.4263	
B	r_{20}	0.5460×10^{-10}	
	$\bar{\rho}$	0.3329	
	ρ_{16}	0.3811	
	ρ_{17}	0.3855	div
	ρ_{18}	0.3866	
	ρ_{19}	0.3856	
	ρ_{20}	0.3835	

high Reynolds number cases and $P_A(B)$ is better for low Reynolds number cases.

A computation is carried out for a three-dimensional flow in a channel with a backward facing step, as shown in Fig. 9. The grid is uniform in direction 3, and is generated in plane 1-2 by using a biharmonic grid generator. Only Re = 100 is considered here. The combination of $P_A(B)$ and $P_C(B)$ is employed. Transfer operators are replaced by their corresponding three-dimensional versions, which are obtained from the two-dimensional versions in a straightforward way. The final time is again 20, and Δt is chosen to be 0.25 based on numerical experiments. Because this is found to be more efficient, the W-cycle used in the previous computations is replaced by the F-cycle with one pre- and one postsmoothing. At each time step, one multigrid iteration is carried out. A smoothing sweep consists of one plane sweep marching in direction 3, and in a plane (parallel to plane 1-2) the ILU smoother used before is applied. The coarsest grid solver uses the smoother, performing 10 sweeps. Table 8

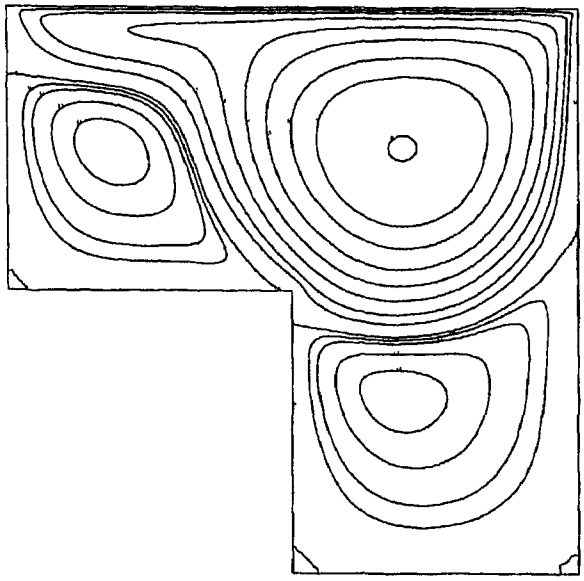

FIG. 8. *Streamlines for the L-shaped driven cavity problem* (Re = 1000, *on a* 128 × 128 *grid*).

TABLE 7

Reduction factors and ratios $\bar{\rho}$, r^{20}/r^0 *for the L-shaped driven cavity problem on three grids with* 2 × 2 *coarsest grid.*

Grid	P_C \ P_A	H	B
32 × 32	Re = 100, $\Delta t = 1$		
	H	0.4120, 8.650×10^{-8}	0.3401, 9.143×10^{-9}
	B	0.3189, 5.542×10^{-8}	0.3164, 6.758×10^{-9}
	Re = 1000, $\Delta t = 0.2$		
	H	0.1437, 1.380×10^{-3}	0.1366, 1.348×10^{-3}
	B	0.1323, 1.361×10^{-3}	0.1294, 1.352×10^{-3}
64 × 64	Re = 100, $\Delta t = 1$		
	H	0.5043, 4.817×10^{-7}	0.4035, 7.023×10^{-9}
	B	0.4793, 2.978×10^{-7}	0.3781, 2.974×10^{-9}
	Re = 1000, $\Delta t = 0.2$		
	H	0.1979, 3.757×10^{-4}	0.1802, 3.699×10^{-4}
	B	0.1696, 3.682×10^{-4}	0.1721, 3.698×10^{-4}
128 × 128	Re = 100, $\Delta t = 1$		
	H	0.7992, $6.252 \times 10^{-4*}$	0.4564, 6.437×10^{-8}
	B	0.7252, $3.870 \times 10^{-5*}$	0.4009, 1.904×10^{-9}
	Re = 1000, $\Delta t = 0.2$		
	H	0.2071, 1.167×10^{-4}	0.2270, 1.189×10^{-4}
	B	0.1432, 1.153×10^{-4}	0.1523, 1.172×10^{-4}

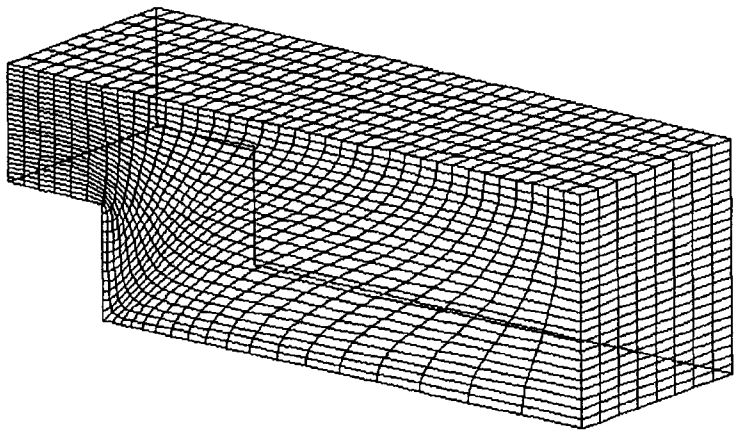

FIG. 9. *The channel with a backward facing step and a* $24 \times 36 \times 8$ *grid.*

TABLE 8
Reduction factors and ratios $\bar{\rho}$, r^{20}/r^0 *for the channel with backward facing step on two grids,* $\mathrm{Re} = 100$, $\Delta t = 0.25$.

Grid	$\bar{\rho}$, r^{20}/r^0
$12 \times 18 \times 4$	$0.0489, 0.7457 \times 10^{-4}$
$24 \times 36 \times 8$	$0.1118, 0.9494 \times 10^{-5}$

gives the reduction factor of the one multigrid iteration at the final time and the ratio r^{20}/r^0, on the finest grid. The reduction factor gradually approaches a constant after some time steps. Figure 10 presents two particle traces in the channel flow on the $24 \times 36 \times 8$ grid. Clearly, the flow pattern is very different from that obtained in two-dimensional cases (see, for instance, [9]), where the flow in the backward facing step forms a closed recirculation region.

9. Conclusions. Galerkin coarse grid approximation for multigrid methods has been investigated for systems of equations, using the incompressible Navier–Stokes equations in general coordinates as an example. Restriction uses the adjoint of the hybrid interpolation for the velocities and a piecewise constant interpolation for the pressure, and is fixed. Two prolongations are used for the velocities: the hybrid interpolation and the bilinear interpolation. For the pressure, piecewise constant interpolation is used. Prolongation (denoted as $\boldsymbol{P}_A$) in the computation of coarse grid matrices can differ from that (denoted as $\boldsymbol{P}_C$) in the computation of coarse grid corrections. An efficient algorithm performing GCA is presented, which can be easily vectorized.

The properties of coarse grid matrices computed from upwind differencing and central differencing in Cartesian coordinates are analyzed. It is found that diagonal dominance of coarse grid matrices is lost or the derivative in a direction disappears for upwind differencing. For central differencing, the derivative in a direction vanishes with the hybrid prolongation. Furthermore, with the hybrid prolongation, the well-known accuracy condition (36) is violated. However, none of this has caused serious

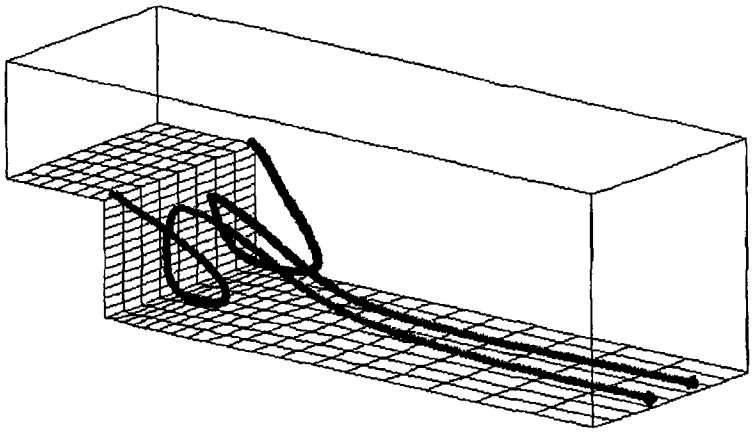

FIG. 10. *Two particle traces in the channel flow on the* $24 \times 36 \times 8$ *grid.*

problems. Numerical experiments show that for a fixed P_A, bilinear prolongation for P_C is more efficient, unless the method diverges, which may occur. Bilinear prolongation for P_A is preferable for low Reynolds numbers, and hybrid prolongation for P_A is preferable for high Reynolds numbers. With bilinear prolongation for P_A and P_C, we apply the method to a three-dimensional flow in a channel with a backward facing step. Satisfactory results are obtained.

Acknowledgment. The authors would like to thank A. Segal and C.G.M. Kassels for their efforts in making the ISNaS code available for this computation. The authors are also grateful to C.W. Oosterlee and E. Brakkee for providing the two-dimensional L-shaped problem and grids, as well as some useful discussions. Thanks go to C. Kasbergen as well, for his help in visualizing the results in three dimensions.

REFERENCES

[1] R. E. ALCOUFFE, A. BRANDT, JR., J. E. DENDY, AND J. W. PAINTER, *The multigrid method for diffusion equations with strongly discontinuous coefficients*, SIAM J. Sci. Stat. Comput., 2 (1981), pp. 430–454.

[2] A. BRANDT, *Multi-level adaptive techniques* (MLAT) *for partial differential equations: ideas and software*, in Proc. Symposium on Mathematical Software, J. Rice, ed., Academic Press, New York, 1977, pp. 277–318.

[3] J. E. DENDY, *Black box multigrid*, J. Comp. Phys., 48 (1982), pp. 366–386.

[4] W. HACKBUSCH, *Multi-grid methods and applications*, Springer-Verlag, Berlin, 1985.

[5] P. W. HEMKER, *On the order of prolongations and restrictions in multigrid procedures*, J. Comp. Appl. Math., 32 (1990), pp. 423–429.

[6] R. KETTLER, *Analysis and comparison of relaxation schemes in robust multigrid and conjugate gradient methods*, in Multigrid Methods, Lecture Notes in Mathematics 960, W. Hackbusch and U. Trottenberg, eds., Springer-Verlag, Berlin, 1982, pp. 502–534.

[7] R. KETTLER AND J. A. MEIJERINK, *A multigrid method and a combined multigrid-conjugate gradient method for elliptic problems with strongly discontinuous coefficients in general domains*, Shell Publ. 604, KSEPL, Rijswijk, the Netherlands, 1981.

[8] M. KHALIL AND P. WESSELING, *Vertex-centered and cell-centered multigrid for interface problems*, J. Comp. Phys., 98 (1992), pp. 1–10.

[9] K. J. MORGAN, J. PÉRIAUX, AND F. THOMASSET, EDS., *Analysis of laminar flow over a backward facing step*, a GAMM-Workshop, Bievres, France, 1984, Vieweg, Braunschweig, 1984.

[10] A. E. MYNETT, P. WESSELING, A. SEGAL, AND C. G. M. KASSELS, *The ISNaS incompressible Navier-Stokes solver: Invariant discretization*, Appl. Sci. Res., 48 (1991), pp. 175–191.

[11] C. W. OOSTERLEE AND P. WESSELING, *A multigrid method for a discretization of the incompressible Navier-Stokes equations in general coordinates*, in Proc. 9th GAMM Conf. on Numer. Methods in Fluid Mech. Notes on Num. Fluid Mech., Vol. 35, J. B. Vos, A. Rizzi, and I. L. Ryhming, eds., Vieweg, Braunschweig, 1992, pp. 99–106.

[12] ———, *A multigrid method for an invariant formulation of the incompressible Navier-Stokes equations in general coordinates*, Comm. Appl. Numer. Methods, 8 (1992), pp. 721–734.

[13] ———, *A robust multigrid method for a discretization of the incompressible Navier-Stokes equations in general coordinates*, in Computational Fluid Dynamics '92, Proc. First European Computational Fluid Dynamics Conference, Ch. Hirsch, J. Périaux, and W. Kordulla, eds., Elsevier, Amsterdam, 1992, pp. 101–107.

[14] C. W. OOSTERLEE, P. WESSELING, A. SEGAL, AND E. BRAKKEE, *Benchmark solutions of the incompressible Navier-Stokes equations in general coordinates on staggered grids*, Internat. J. Numer. Methods in Fluids, 17 (1993), pp. 301–321.

[15] A. SEGAL, P. WESSELING, J. VAN KAN, C. W. OOSTERLEE, AND C. G. M. KASSELS, *Invariant discretization of the incompressible Navier-Stokes equations in boundary fitted coordinates*, Internat. J. Numer. Methods in Fluids, 15 (1992), pp. 411–426.

[16] D. B. SPALDING, *A novel finite difference formulation for differential expressions involving both first and second derivatives*, Internat. J. Numer. Methods in Fluids, 4 (1972), pp. 551–559.

[17] P. WESSELING, *Theoretical and practical aspects of a multigrid method*, SIAM J. Sci. Stat. Comput., 3 (1982), pp. 387–407.

[18] ———, *Linear multigrid methods*, in Multigrid Methods, Frontiers in Applied Mathematics, Vol. 3, S. F. McCormick, ed., Society for Industrial and Applied Mathematics, Philadelphia, PA, 1987, pp. 31–56.

[19] ———, *Cell-centered multigrid for interface problems*, J. Comput. Phys., 79 (1988), pp. 85–91.

[20] ———, *An Introduction to Multigrid Methods*, John Wiley & Sons, Chichester, 1992.

[21] P. WESSELING, 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 (1992), pp. 27–33.

[22] G. WITTUM, *On the convergence of multi-grid methods with transforming smoothers—theory with applications to the Navier-Stokes equations*, Numer. Math., 57 (1990), pp. 15–38.

[23] P. M. DE ZEEUW, *Matrix-dependent prolongations and restrictions in a block multigrid method solver*, J. Comp. Appl. Math., 3 (1990), pp. 1–27.

[24] P. M. DE ZEEUW AND E. J. VAN ASSELT, *The convergence rate of multi-level algorithms applied to the convection-diffusion equation*, SIAM J. Sci. Stat. Comput., 6 (1985), pp. 492–508.

[25] S. ZENG AND P. WESSELING, *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.

[26] ———, *Numerical study of a multigrid method with four smoothing methods for the incompressible Navier-Stokes equations in general coordinates*, Report 93-04, Faculty of Technical Mathematics and Informatics, TU Delft, the Netherlands, 1993.