JOURNAL OF COMPUTATIONAL PHYSICS 84, 207-241 (1989)

Time-Dependent Viscous Incompressible Navier–Stokes Equations: The Finite Difference Galerkin Formulation and Streamfunction Algorithms

JOHN W. GOODRICH

Computational Fluid Dynamics Branch,
NASA Lewis Research Center, Cleveland, Ohio 44135

AND

W. Y. SOH

Sverdrup Technology Inc., and
NASA Lewis Research Center, Cleveland, Ohio 44135

Received March 14, 1988; revised November 7, 1988

The finite difference Galerkin (FDG) method is extended to time dependent incompressible Navier–Stokes equations. Two algorithm development examples are given that use a staggered grid and centered differencing scheme for the primitive variables. Mass balance is used to solve the essential problems associated with applying the FDG method. The use of the FDG method with this underlying discretization is shows to be the discrete analog of the continuum manipulations that lead to the fourth-order streamfunction equation. Asymptotic and time evolution results obtained with a Crank–Nicolson Adams–Basforth algorithm are compared with published computations for Re 400, 1000, and 3200. © 1989 Academic Press, Inc.

1. INTRODUCTION

This paper is concerned with the development of finite difference algorithms for unsteady incompressible Navier–Stokes equations. The momentum and continuity equations for incompressible flows cannot be directly integrated in time because the continuity equation is not given in a time evolution form. In primitive variable formulations one approach has been to use a fractional time step method. The intermediate step introduces a velocity solution to the momentum equations without the pressure gradient, and then the pressure and velocity fields are successively corrected until the continuity equation is satisfied. The successive correction of pressure and velocity is equivalent to projecting the intermediate velocity solution onto the subspace of discretely divergence free velocity fields. This general approach was developed by Chorin [3], and a specific example is given in Kim and Moin [11]. Another general method for solving incompressible flow problems in

0021-9991/89 $3.00

Copyright © 1989 by Academic Press, Inc.
All rights of reproduction in any form reserved.

primitive variables is to directly couple the momentum and continuity equations. Moin and Kim [13] use direct coupling by simultaneously solving the momentum and continuity equations for velocity and pressure. Indirect coupling is used in the pressure Poisson method introduced for two dimensions by Harlow and Welch [9] and extended to three dimensions by Williams [22]. This approach combines the divergence of the momentum equation with the continuity equation to obtain a Poisson equation for the pressure. A third general approach that has been widely used in two dimensions is the streamfunction and vorticity formulation, as in Fromm [4] and Roache [16]. The vorticity transport equation is obtained by taking the curl of the momentum equation, and this eliminates the pressure gradient. The velocity solution is obtained from the stream function so that it automatically satisfies the continuity equation. Particular difficulties with this method have been caused by the vorticity boundary conditions, and by the coupling between the vorticity and the streamfunction along the boundaries.

Weak forms of the incompressible Navier–Stokes equations can be formulated without the pressure gradient and the continuity equation. Incompressibility is obtained by restricting the function space of velocity solutions, and the continuity equation is viewed as a constraint for defining the solution space. This approach is useful both for the analysis of the Navier–Stokes equations (Ladyzhenskaya [12]), and for the development of numerical algorithms (Temam [20]). A numerical algorithm is generally developed after manipulating the partial differential equations and their solution spaces. In the context of Galerkin ideas, Stephens, Bell, Solomon, and Hackerman [19] use their finite difference Galerkin (FDG) method for the numerical solution of steady incompressible equations. They begin with a discretization of the primitive variable equations, and then they manipulate the finite difference equations and their solution spaces to obtain a convenient algorithm. The essential features of the FDG method are the expansion of the discrete velocity solution using a basis for the discretely divergence free vector fields on the grid and derivation of equations for the expansion coefficients by taking the inner product of the expansion vectors and the discrete momentum equations. The only exceptional constraint is that the primitive variable discretization for the divergence operator in the continuity equation must be the adjoint (matrix transpose) of the discretization for the gradient operator applied to the pressure in the momentum equations. If the discrete divergence and gradient operators are the adjoints of each other, then the discrete pressure gradient will drop out of the derived equations. The FDG method applies in two or three dimensions and may be used with any primitive variable discretization that is chosen by the user. The essential problems in applying the FDG method are to find a basis for the nullspace of the discrete divergence operator and to find a particular solution of the discrete continuity equation that accounts for the velocity boundary values that are prescribed by the problem that is being solved. The FDG method is equivalent to the dual variable (DV) method of Amit, Hall, and Porsching [1]. The DV method has been used for practical problems with steady and unsteady flows and has been extended to both compressible flows and the finite element method (see Hall [8]).

The DV method also starts with a primitive variable discretization and is essentially a variable reduction method using network theory algorithms as a systematic method for obtaining a basis of nullvectors for the discrete divergence operator and for obtaining a particular solution to the continuity equation that assumes any boundary values prescribed for the flow. A similar variable reduction method has been used in the solution of nonlinear electrical network problems (see Rheinbolt [15]). Both Amit *et al.* [1] and Stephens *et al.* [19] have suggested that in two dimensions it is possible to relate the dual variables or expansion coefficients to a discrete streamfunction.

We begin this paper in Section 2 by presenting a general extension of the FDG method used by Stephens *et al.* [19] to unsteady incompressible flows. A continuous time treatment is used for this presentation, since specific algorithms may be developed with any time discretization. This general formulation is valid in two or three space dimensions and provides a discretely divergence free velocity solution without requiring a pressure solution. The presentation of the general method is more theoretical than the rest of the paper, and readers are urged to begin with Section 3 if they are more interested in the details of implementing this method. An example of the general method for developing algorithms is given a detailed treatment in Section 3. The primitive variable discretization used by the method in Section 3 is on a staggered MAC grid with central differencing, and we show how discrete mass balance can be used to solve the essential problems associated with applying the FDG method. The driven cavity problem is used as a simple example. In Section 4 we show in detail that the resulting FDG expansion variables may be directly interpreted as a discrete streamfunction and that our choices of primitive variable discretization and expansion vectors in the FDG method are equivalent to formulating an algorithm for the time dependent fourth-order streamfunction equation. This method of algorithm development can use a staggered grid and mass balance for a primitive variable formulation, with the FDG method leading to a reduction of variables in the derived streamfunction formulation, while the discrete streamfunction interpretation of the derived variables yields a primitive variable solution that is discretely divergence free with all velocity components defined at the same point. Section 5 contains a discussion of how mass balance may be used to adapt the general method to problems in two dimensions with throughflow and with obstacles in the flow field. Section 6 presents numerical results using an algorithm developed in Section 3, with comparisons to published computations. Results are reported for the asymptotic steady state flow in a driven cavity at $\mathrm{Re}=400$, 1000, and 3200. Results are also shown for the unsteady vortex dynamics at $\mathrm{Re}=1000$ in driven cavities with aspect ratios 1 and 2, including the dramatic evolution of secondary vortices from bubble recirculations starting on the downstream wall.

2. The Finite Difference Galerkin Method for Unsteady Viscous Incompressible Flows

Let Ω be a bounded open region in $\mathbf{R}^2$ or $\mathbf{R}^3$, with boundary $\partial\Omega$. The dimensionless incompressible Navier–Stokes equations in Ω are

$$\frac{\partial \mathbf{u}}{\partial t}+\nabla\cdot(\mathbf{uu})-\frac{1}{\mathrm{Re}}\Delta\mathbf{u}=-\nabla p+\mathbf{F}, \qquad \text{for} \quad \mathbf{x} \text{ in } \Omega \text{ and } t>0, \tag{1a}$$

$$\nabla\cdot\mathbf{u}=0, \qquad \text{for} \quad \mathbf{x} \text{ in } \Omega \text{ and } t>0, \tag{1b}$$

$$\mathbf{u}(\mathbf{x},0)=\mathbf{a}(\mathbf{x}), \qquad \text{for} \quad \mathbf{x} \text{ in } \Omega \text{ at } t=0, \tag{1c}$$

$$B[\mathbf{u}(\mathbf{u},t)]=\mathbf{b}(\mathbf{x},t), \qquad \text{for} \quad \mathbf{x} \text{ in } \partial\Omega \text{ and } t>0, \tag{1d}$$

where $\mathbf{u}$ is the velocity, p is the scaled pressure, Re is the Reynolds number, $\mathbf{F}$ is the volume force per unit mass, and B is the operator that defines the boundary conditions. If $\mathbf{u}=B[\mathbf{u}]=\mathbf{b}$ in $\partial\Omega$, then the boundary data must satisfy the constraint

$$\int_{\partial\Omega}\mathbf{b}(\mathbf{x},t)\cdot\boldsymbol{\eta}(\mathbf{x})\,ds=0, \qquad \text{for} \quad t>0, \tag{1e}$$

with $\boldsymbol{\eta}$ as the outer normal to the boundary $\partial\Omega$. The convenience of known velocity values on the boundary is usually absent from practical problems. In common applications at least part of the boundary is artifically imposed in the flow in order to restrict the domain of the computation, and the artifical boundary algorithms may not correspond to the usual boundary conditions that are treated theoretically. In order to accomodate problems like this, let $\partial\Omega_1$ and $\partial\Omega_2$ be disjoint with $\partial\Omega=\partial\Omega_1\cup\partial\Omega_2$, where $\mathbf{u}$ is specified on $\partial\Omega_1$ and unspecified on $\partial\Omega_2$. Weaker forms of the equations for incompressible flow can be derived by starting with the L^2 inner product equation

$$\frac{d}{dt}\int_{\Omega}\mathbf{u}\cdot\mathbf{v}\,dv+\int_{\Omega}\left[\nabla\cdot(\mathbf{uu})-\frac{1}{\mathrm{Re}}\Delta\mathbf{u}\right]\cdot\mathbf{v}\,dv=\int_{\Omega}[-\nabla p+\mathbf{F}]\cdot\mathbf{v}\,dv, \tag{2}$$

for t in $(0,T]$, where $\mathbf{v}$ is an arbitrary member of a suitably chosen space of divergence free functions defined in Ω, and where $T>0$ is a fixed time. Seeking a solution by using equations and algorithms derived from (2) can separate the problem of finding the velocity $\mathbf{u}$ from that of finding the pressure p. If the test function $\mathbf{v}$ in Eq. (2) satisfies $\nabla\cdot\mathbf{v}=0$ in Ω and $\mathbf{v}(\mathbf{x})=0$ in $\partial\Omega$, then the pressure term is eliminated from (2) by Gauss' Theorem. The usual Galerkin method makes a choice of function spaces for the solution $\mathbf{u}$ and test functions $\mathbf{v}$ which will ensure both that the pressure does not appear in the weak equations derived from (2), and that the solution is incompressible.

The FDG method is very similar to standard applications of the Galerkin method with a few important differences. The most important difference is that standard Galerkin methods begin with weak continuous equations like (2), while the FDG method begins with a primitive variable finite difference approximation of (1a)–(1d). This difference amounts to a choice of where to stop manipulating the continuum problem, and of where to begin manipulating its discrete approximation. The point where discretization begins naturally implies that there will be differences in the function spaces for admissible solutions and trial functions. For a typical application of the Galerkin method in developing a finite element algorithm, the function spaces consist of piecewise smooth functions that give a solution which is defined for every point in the domain Ω. The function spaces for the FDG method are defined on the discrete points of the computational grid. The spaces of solution and trial functions for applications of the Galerkin method are typically constrained by boundary conditions from the partial differential equations. The FDG method incorporates the boundary conditions in the primitive variable discretization of the momentum equations, and obtains the constraint for the solution and trial function spaces from the discrete continuity equation. The nullvectors for the discrete divergence operator are used as a basis for the trial function space, and any prescribed boundary values are assumed by a specific solution of the continuity equations. As in standard applications of the Galerkin method, the solution of the discrete momentum equations is expanded in terms of these basis vectors, and equations for the expansion coefficients are obtained from the inner product between these nullvectors and the discrete momentum equations. In order to apply the FDG method, the discretization of the gradient operator for the pressure in the momentum equations must be the adjoint (i.e., transpose) of the discretization of the divergence operator in the continuity equation. Stephens *et al.* [19] have already applied the Galerkin method in this way to steady viscous incompressible flow.

In order to apply the Galerkin ideas we must introduce a discrete grid and spaces of scalar and vector valued functions defined on the grid, and we must introduce a spatial discretization of equations (1a)–(1d). A finite difference grid denoted by G_h will be introduced in $\Omega \cup \partial\Omega_2$, and the grid in $\partial\Omega_1$ will be denoted by BG_h, where h is an indicator of the mesh size. The set of all possible vector and scalar valued functions on $G_h \cup BG_h$ will be denoted by V_h and F_h, respectively. Denote an element of V_h by $\tilde{\mathbf{u}}$, and let $\mathbf{II}_{Vi}$ and $\mathbf{II}_{Vb}$ be index sets for the scalar components of V_h defined on G_h and BG_h, respectively. If $\tilde{\mathbf{u}} \in V_h$ then $\tilde{\mathbf{u}} = \{u_i : i \in \mathbf{I}_{Vi} \cup \mathbf{I}_{Vb}\}$, where u_i are the discrete scalar components of $\tilde{\mathbf{u}}$, and not the discrete vector field elements of V_h. The discrete velocity components indexed by $\mathbf{I}_{Vi}$ and defined on G_h are not determined by prescribed boundary data, while the components on BG_h indexed by $\mathbf{I}_{Vb}$ are determined by boundary data. For a staggered grid some of the velocity components in cells along the boundary may be defined as boundary data on BG_h, while other components in the same cells must be found as part of the solution on G_h. Denote an element of F_h by $\tilde{p}$, and let $\mathbf{I}_{Fi}$ and $\mathbf{I}_{Fb}$ be index sets for the components of F_h defined on the grids G_h and BG_h, respectively. If $\tilde{p} \in F_h$ then $\tilde{p} = \{p_i : i \in \mathbf{I}_{Fi} \cup \mathbf{I}_{Fb}\}$. On a staggered grid it may be appropriate to define F_h on just G_h and not on BG_h, and the physical location in the grid cells of F_h values may be different from the locations of any of the velocity components. Define an inner product on V_h by

$$(\tilde{\mathbf{u}}, \tilde{\mathbf{v}}) = \sum_{i \in \mathbf{I}_{Vi} \cup \mathbf{I}_{Vb}} u_i v_i, \tag{3}$$

for $\tilde{\mathbf{u}}$ and $\tilde{\mathbf{v}}$ in V_h, and define an inner product on F_h by

$$\langle \tilde{p}, \tilde{q} \rangle = \sum_{i \in \mathbf{I}_{Fi} \cup \mathbf{I}_{Fb}} p_i q_i,$$

for $\tilde{p}$ and $\tilde{q}$ in F_h. The discrete inner product $(\tilde{\mathbf{u}}, \tilde{\mathbf{v}})$ is just the component by component product across the entire discrete vector field. Let di and gr represent discrete divergence and gradient operators, where di is an operator from V_h to F_h and gr is an operator from F_h to V_h. Let the discretely divergence free subspace of V_h be $D_h = \{\tilde{\mathbf{u}} \in V_h : \mathrm{di}(\tilde{\mathbf{u}}) = 0\}$. Let D_h^0 be the subspace of D_h with zero values in BG_h. Let $\mathrm{na}(\tilde{\mathbf{u}})$ be a discrete nonlinear operator defined on V_h that approximates the continuous terms $\nabla \cdot (\mathbf{uu}) - (1/\mathrm{Re}) \Delta \mathbf{u}$ from (1a), and that includes an approximation of the boundary operator B on $\partial \Omega_2$, where $\mathbf{u}$ is not specified. Let $\tilde{\mathbf{f}}$ be a discrete approximation of the volume forces $\mathbf{F}$ from (1a), where $\tilde{\mathbf{f}}$ may just be a constant vector in V_h. With this notation a continuous time discretization of Eqs. (1a)–(1d) may be written as

$$\frac{\partial \tilde{\mathbf{u}}}{\partial t} + \mathrm{na}(\tilde{\mathbf{u}}) = -\mathrm{gr}(\tilde{p}) + \tilde{\mathbf{f}}, \qquad \text{for} \quad t > 0, \tag{4a}$$

for the unknown discretely divergent free velocity solution as a function of time

$$\tilde{\mathbf{u}}(t) : [0, T] \mapsto D_h, \tag{4b}$$

and for the discrete pressure solution $\tilde{p}(t)$: $[0, T] \mapsto F_h$ as a function of time, with

$$\tilde{\mathbf{u}}(0) = \tilde{\mathbf{a}}, \qquad \text{in } G_h \text{ at } t = 0, \tag{4c}$$

$$\tilde{\mathbf{u}}(t) = \tilde{\mathbf{b}}(t), \qquad \text{in } BG_h \text{ for } t > 0, \tag{4d}$$

where $\tilde{\mathbf{a}}$ and $\tilde{\mathbf{b}}(t)$ are mesh functions on G_h and $BG_h \otimes (0, T]$ that discretize $\mathbf{a}$ in Ω from (1c) and $\mathbf{b}$ in $\partial \Omega_1$ from (1d), respectively. Equations (4a)–(4d) are a coupled first-order system of nonlinear ODE for the unknown velocity components $\{u_i(t) : i \in \mathbf{I}_{Vi}\}$ of the discrete solution $\tilde{\mathbf{u}}(t)$ for $t \in (0, T]$. Note that the divergence free constraint no longer appears since it has been incorporated in the definition (4b) of the function space for the solution, and that the boundary operator B on $\partial \Omega_2$ no longer appears since it has been incorporated in the space discretization. Note also that at this point we could discretize in time before proceeding with the discrete Galerkin formulation. We will continue with a continuous time and discrete space formulation because of the similarity with a general pattern of use for the Galerkin method, and because this approach will also easily lend itself to the formulation of a variety of time discretizations after the FDG method has been applied to the space discretization.

We will now proceed with the discrete Galerkin formulation for the time dependent case, based upon the primitive variable continuous time discrete space formulation (4a)–(4d). The solution vector $\tilde{\mathbf{u}}(t):[0, T] \mapsto D_h$ of (4a)–(4d) may be written as $\tilde{\mathbf{u}}(t) = \tilde{\mathbf{w}}_p(t) + \tilde{\mathbf{w}}_0(t)$, where $\tilde{\mathbf{w}}_p(t):[0, T] \mapsto D_h$ is discretely divergence free with $\tilde{\mathbf{w}}_p(t) = \tilde{\mathbf{b}}(t)$ in BG_h for $t > 0$, and where $\tilde{\mathbf{w}}_0(t)$ is an undetermined element of D_h^0 for $0 < t \leqslant T$. The role of $\tilde{\mathbf{w}}_p(t)$ is to reduce the computational problem to finding the discretely divergence free solution $\tilde{\mathbf{w}}_0(t)$ with homogeneous boundary data in D_h^0. The particular solution $\tilde{\mathbf{w}}_p(t)$ is introduced in order to handle the prescribed boundary values, and in order to simplify finding a basis for the discretely divergent free subspace D_h^0 which contains $\tilde{\mathbf{w}}_0(t)$. In the next section a detailed example is given of D_h^0 for a specific problem, and of how to find a basis for D_h^0 using mass balance. A detailed discussion of D_h^0 and its basis for general problems is given in Section 5. For the general situation under discussion, let the dimension of D_h^0 be $d = \dim(D_h^0)$, and let $\{\tilde{\mathbf{c}}_l\}_{l=1}^d$ be a basis for D_h^0. Since $\{\tilde{\mathbf{c}}_l\}_{l=1}^d$ is a basis for D_h^0, there exists unknown real valued functions $\{z_l(t):[0, T] \mapsto \mathbf{R}\}_{l=1}^d$, such that $\tilde{\mathbf{w}}_0(t) = \sum_{l=1}^d z_l(t)\,\tilde{\mathbf{c}}_l$. There is an equation in the system (4a) for each undetermined discrete velocity component defined on the grid, or for each separate index $i \in \mathbf{I}_{Vi}$, and the expansion vectors $\{\tilde{\mathbf{c}}_l\}_{l=1}^d$ of D_h^0 have nonzero components only on the grid G_h, or for each index $i \in \mathbf{I}_{Vi}$. If the separate scalar equations in (4a) are ordered consistently with the order of the unknown interior velocity components $\{u_i : i \in \mathbf{I}_{Vi}\}$, then the inner product between the vector equation (4a) and the expansion vectors may be computed as in (3). If di and gr are adjoints of each other in the sense that for all $\tilde{\mathbf{u}} \in V_h$ and for all $\tilde{p} \in F_h$ with $\tilde{p}\tilde{\mathbf{u}} = 0$ in BG_h,

$$\langle \mathrm{di}(\tilde{\mathbf{u}}), \tilde{p} \rangle = (\tilde{\mathbf{u}}, \mathrm{gr}(\tilde{p})), \tag{5}$$

then for $L = 1, \ldots, d$ we have

$$(\tilde{\mathbf{c}}_L, \mathrm{gr}(\tilde{p})) = \langle \mathrm{di}(\tilde{\mathbf{c}}_L), \tilde{p} \rangle = \langle \tilde{0}, \tilde{p} \rangle = 0,$$

since $\tilde{\mathbf{c}}_L \in D_h^0$ so that $\mathrm{di}(\tilde{\mathbf{c}}_L) = \tilde{0}$. If (5) is satisfied, then a discrete solution to the Navier–Stokes equations on the given mesh may be obtained by solving the system of nonlinear ODE,

$$\sum_{l=1}^{d} (\tilde{\mathbf{c}}_L, \tilde{\mathbf{c}}_l) \frac{dz_l}{dt} + \left(\tilde{\mathbf{c}}_L, \frac{d}{dt}\tilde{\mathbf{w}}_p(t)\right) = \left(\tilde{\mathbf{c}}_L, \tilde{\mathbf{f}} - \mathrm{na}\left(\tilde{\mathbf{w}}_p(t) + \sum_{l=1}^{d} z_l \tilde{\mathbf{c}}_l\right)\right), \tag{6a}$$

for $L = 1, \ldots, d$, where the discretely divergence free $\tilde{\mathbf{w}}_p(t) \in D_h$ is assumed to be known and to satisfy $\tilde{\mathbf{w}}_p(t) = \tilde{\mathbf{b}}(t)$ on BG_h. The initial data for $\{z_l(t)\}_{l=1}^d$ is obtained from

$$\tilde{\mathbf{a}} = \tilde{\mathbf{w}}_p(0) + \sum_{l=1}^{d} z_l(0)\,\tilde{\mathbf{c}}_l, \tag{6b}$$

and the final velocity solution is obtained as

$$\tilde{\mathbf{u}}(t) = \tilde{\mathbf{w}}_p(t) + \sum_{l=1}^{d} z_l(t)\,\tilde{\mathbf{c}}_l. \tag{6c}$$

Equation (6a) is a discrete analog of (2). Note that if (5) is satisfied, then the pressure terms have dropped out of (6a). Note also that there is no reference to the space dimension in the derivation that leads up to (6a)–(6c), so that the FDG method may be applied in two or three space dimensions. The discrete divergence equation no longer explicitly appears, but it can still play a role in finding the particular solution $\tilde{\mathbf{w}}_p(t)$ that satisfies $\tilde{\mathbf{w}}_p(t) = \tilde{\mathbf{b}}(t)$ on BG_h. Let us write $\tilde{\mathbf{w}}_p(t) = \tilde{\mathbf{w}}_i(t) + \tilde{\mathbf{w}}_b(t)$, where $\tilde{\mathbf{w}}_i(t)$ and $\tilde{\mathbf{w}}_b(t)$ are zero valued on BG_h and G_h, respectively. Let A be a matrix representation of di restricted to elements in V_h that are zero on BG_h. Since $\tilde{\mathbf{w}}_p(t) \in D_h$, $0 = \mathrm{di}(\tilde{\mathbf{w}}_p) = \mathrm{di}(\tilde{\mathbf{w}}_i + \tilde{\mathbf{w}}_b) = \mathbf{A}\tilde{\mathbf{w}}_i + \mathrm{di}(\tilde{\mathbf{w}}_b)$. We know $\tilde{\mathbf{w}}_b$ from the boundary data $\tilde{\mathbf{b}}$, so that we can find $\tilde{\mathbf{w}}_i$ from the underdetermined linear system $\mathbf{A}\tilde{\mathbf{w}}_i = -\mathrm{di}(\tilde{\mathbf{w}}_b)$.

The continuous time algorithm in Eqs. (6a)–(6c) may be solved or approximated in any manner that is convenient. We shall introduce two examples of discretization in time. For a two time level discretization we may use the general form

$$\begin{aligned}\sum_{l=1}^{d} (\tilde{\mathbf{c}}_L, \tilde{\mathbf{c}}_l)\,\frac{z_l^{n+1} - z_l^n}{\Delta t} + \frac{(\tilde{\mathbf{c}}_L, \tilde{\mathbf{w}}_p^{n+1} - \tilde{\mathbf{w}}_p^n)}{\Delta t} &= \theta\left(\tilde{\mathbf{c}}_L, \tilde{\mathbf{f}} - \mathrm{na}\left(\tilde{\mathbf{w}}_p^{n+1} + \sum_{l=1}^{d} z_l^{n+1}\tilde{\mathbf{c}}_l\right)\right)\\ &\quad + (1-\theta)\left(\tilde{\mathbf{c}}_L, \tilde{\mathbf{f}} - \mathrm{na}\left(\tilde{\mathbf{w}}_p^{n} + \sum_{l=1}^{d} z_l^{n}\tilde{\mathbf{c}}_l\right)\right),\end{aligned} \tag{7}$$

for $L = 1, ..., d$, where $0 \leqslant \theta \leqslant 1$. For the second time discretization we must write $\mathrm{na}(\tilde{\mathbf{u}}) = \mathrm{cv}(\tilde{\mathbf{u}}) - \mathrm{df}(\tilde{\mathbf{u}})$, where $\mathrm{cv}(\tilde{\mathbf{u}})$ and $\mathrm{df}(\tilde{\mathbf{u}})$ are discrete operators defined on V_h that approximate the continuous terms $\nabla \cdot (\mathbf{uu})$ and $(1/\mathrm{Re})\,\Delta\mathbf{u}$ in (1a). We may write a Crank–Nicolson Adams–Bashforth time discretized scheme as

$$\begin{aligned}\sum_{l=1}^{d} (\tilde{\mathbf{c}}_L, \tilde{\mathbf{c}}_l)\,\frac{z_l^{n+1} - z_l^n}{\Delta t} + \frac{(\tilde{\mathbf{c}}_L, \tilde{\mathbf{w}}_p^{n+1} - \tilde{\mathbf{w}}_p^n)}{\Delta t} &= \left(\tilde{\mathbf{c}}_L, \frac{1}{2}\left[\mathrm{df}\left(\sum_{l=1}^{d} z_l^{n+1}\tilde{\mathbf{c}}_l + \tilde{\mathbf{w}}_p^{n+1}\right) + \mathrm{df}\left(\sum_{l=1}^{d} z_l^{n}\tilde{\mathbf{c}}_l + \tilde{\mathbf{w}}_p^{n}\right)\right]\right)\\ &\quad - \left(\tilde{\mathbf{c}}_L, \frac{1}{2}\left[3\mathrm{cv}\left(\sum_{l=1}^{d} z_l^{n}\tilde{\mathbf{c}}_l + \tilde{\mathbf{w}}_p^{n}\right) - \mathrm{cv}\left(\sum_{l=1}^{d} z_l^{n-1}\tilde{\mathbf{c}}_l + \tilde{\mathbf{w}}_p^{n-1}\right)\right]\right) + (\tilde{\mathbf{c}}_L, \tilde{\mathbf{f}}),\end{aligned} \tag{8}$$

for $L = 1, ..., d$. Note that the specific formulations resulting in algorithms (7) and (8) could also have been obtained by first discretizing in time as well as in space, and then using the discrete Galerkin expansion and derivation of equations for the expansion coefficients. The general formulation in (6a)–(6c), and the two time discretizations in (7) and (8) are very abstractly stated and are the end result of applying the FDG method to the continuous time formulation (4a)–(4d). The FDG method manipulates the spatial discretization in order to reduce the computational effort required to obtain a solution, and is essentially independent from the time discretization. The point of the FDG method is to reduce the computational problem at the nth time step from finding $\tilde{\mathbf{u}}^n$ and $\tilde{p}^n$ to that of finding $\{z_l^n\}_{l=1}^{d}$. An example of the specific details required to actually implement this abstract and general algorithm is given in Section 3. Section 4 discusses the interpretation of the coefficients $\{z_l^n\}_{l=1}^{d}$, and Section 5 discusses the details required for general problems.

3. Algorithms on a Two-Dimensional Staggered Grid

In this section we will use the general FDG method to develop an algorithm specifically for the time dependent driven cavity problem in two space variables. The flow is normalized on a 1×1 square with zero initial flow and with an impulsively started lid moving to the right with unit velocity. This problem is illustrated in Fig. 1a. We will not include the volume force $\mathbf{F}$ in Eq. (1a). The velocity will be denoted by $\mathbf{u}(\mathbf{x}, t) = (u(x, y, t), v(x, y, t))$. The initial data in equation (1c) is $\mathbf{a}(\mathbf{x}) = 0$, and the boundary data in Eq. (1d) is $\mathbf{b}(\mathbf{x}, t) = 0$ everywhere on the boundary except for $\mathbf{b}(x, 1, t) = (1, 0)$. The flow field variables will be defined on a staggered uniform grid with $\Delta x = \Delta y = h$. The interior mesh cells will be indexed in the x direction by i for $1 \leqslant i \leqslant I$, and in the y direction by j for $1 \leqslant j \leqslant J$. A typical cell with positions for the variables is given in Fig. 1b. A velocity component defined on a cell face will be interpreted as a velocity average over the face.

The application of the FDG method to a continuous time discretization (4a) is useful theoretically, but for the practical development of an algorithm it is easier to apply the FDG method to a primitive variable discretization in both time and

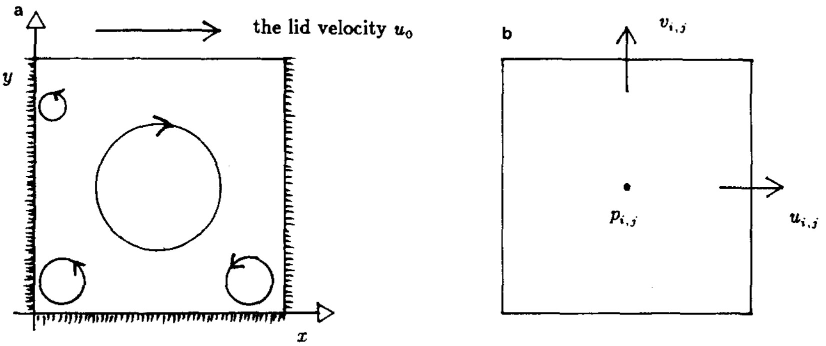


FIG. 1. (a) The driven cavity problem. (b) The (i, j) cell with variable locations.

space, and this is how we will proceed. For second-order time and space accuracy we will use a Crank–Nicolson time discretization and a centered space discretization for all primitive variable terms in Eq. (1a). We will follow Harlow and Welch [9] and Welch *et al.* [21] in the treatment of central space differencing on the staggered grid. The discretized momentum equation (1a) without the body force can be written as

$$\frac{\tilde{\mathbf{u}}^{n+1}-\tilde{\mathbf{u}}^{n}}{\Delta t}+\frac{1}{2}\left(\mathrm{na}(\tilde{\mathbf{u}}^{n+1})+\mathbf{A}^{\mathrm{T}}\tilde{p}^{n+1}+\mathrm{na}(\tilde{\mathbf{u}}^{n})+\mathbf{A}^{\mathrm{T}}\tilde{p}^{n}\right)=0, \tag{9}$$

where $\tilde{\mathbf{u}}^{m}$ and $\tilde{p}^{m}$ are the discrete velocity and pressure solutions at time t_m, and where $\mathrm{na}(\tilde{\mathbf{u}}^{m})$ and $\mathbf{A}^{\mathrm{T}}\tilde{p}^{m}$ are the space discretizations for the continuous terms $\nabla\cdot(\mathbf{uu})-(1/\mathrm{Re})\,\Delta\mathbf{u}$ and ∇p at time t_m. Note that $\mathbf{A}^{\mathrm{T}}$ is a matrix and that na is a nonlinear operator. Note also that centered differences are used for discretizing all of the space derivatives. This discretization applies directly to the equations in the interior cells but must be modified next to the boundaries to accomodate the boundary conditions. The finite difference approximations for the nonlinear derivatives are not effected next to the boundaries since the discrete forms compute the product of a tangential velocity component in an exterior cell with zero boundary values of the normal velocity component. The only term that is actually effected is for diffusion normal to the boundary, and we will use the standard procedure for staggered grids by defining an exterior cell velocity with the aid of the boundary conditions. Note that the only inhomogeneous boundary data is the lid velocity which is incorporated in the expression for $(1/\mathrm{Re})\,\partial^2 u/\partial y^2$ in cells next to the lid. Since the lid velocity is tangential to the cavity, it does not appear in cell centered divergence calculations. Consequently, this known boundary data may be incorparated as a source term in the discrete primitive variable momentum equations, the tangential velocity values on the boundary may be excluded from the discrete velocity field for the computational problem, and the continuity equations will be homogeneous. The discrete incompressible velocity solutions that we consider for this example will therefore be in the subspace D_h^0 of discrete divergence operator nullvectors that have zero boundary values, and the vector of discrete velocity components will only need to include those components defined at positions inside the cavity. The components will be ordered by cell across the mesh and ordered within cells as u then v. If we let $\tilde{\mathbf{u}}$ be a discrete velocity vector defined on the interior mesh faces, then the transpose of $\tilde{\mathbf{u}}$ is

$$\tilde{\mathbf{u}}^{\mathrm{T}}=(u_{1,1}, v_{1,1}, ..., v_{I-1,1}, v_{I,1}, u_{1,2}, ..., v_{I,J-1}, u_{1,J}, u_{2,J}, ..., u_{I-1,J}),$$

since the u values for cells with $i=0$ or $i=I$ and $1\leqslant j\leqslant J$ represent zero boundary velocities across the upstream and downstream walls, while the v values for cells with $j=0$ or $j=J$ and $1\leqslant i\leqslant I$ represent zero boundary velocities across the bottom wall and the upper lid. The pressure is defined at the center of each indexed cell on the interior mesh, so the transpose of the vector of discrete pressure values may be represented as

$$\tilde{p}^{\mathrm{T}}=(p_{1,1}, p_{2,1}, ..., p_{I,J}).$$

For this staggered grid there are $M_i = I(J-1) + (I-1)J = 2IJ - I - J$ components in $\tilde{\mathbf{u}}$, and $m_i = IJ$ components in $\tilde{p}$.

We may interpret the continuity or divergence free constraint as the requirement that there is no net flow across the boundary of each cell at every discrete time. The appropriate discretization in the (i, j) cell at time t_m is

$$\frac{u^m_{i,j} - u^m_{i-1,j}}{\Delta x} + \frac{v^m_{i,j} - v^m_{i,j-1}}{\Delta y} = 0. \tag{10}$$

Along the two side walls and the bottom there is an obvious modification of this constraint. For cells next to the upper lid, the v velocity on the upper cell face is zero, and this is the velocity on that face which must be used in the divergence constraint (10) next to the upper lid. The lid velocity $\mathbf{u} = (1, 0)$ is tangential to the lid and does not contribute to the mass balance for a boundary cell next to the lid. The linear system representing (10) on the entire mesh may therefore be written at each discrete time t_m as the matrix equation

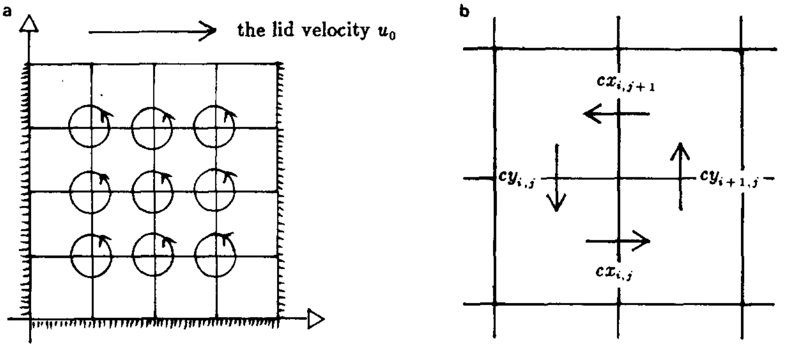
$$\mathbf{A}\tilde{\mathbf{u}}^m = 0, \tag{11}$$

where for this problem the matrix $\mathbf{A}$ is a discretization of di on the interior mesh. Since (10) applies to each cell, (11) represents IJ equations in $2IJ - I - J$ velocity components, so that $\mathbf{A}$ is an $IJ \times (2IJ - I - J)$ matrix. There is a discrete evolution equation in (9) for each unknown velocity component, so that (9) represents $2IJ - I - J$ equations. Since $\tilde{p}^m$ is a vector with IJ components, the discrete gradient operator gr used on $\tilde{p}^m$ must have a $(2IJ - I - J) \times IJ$ matrix representation. The matrix dimensions for the discrete representations of the divergence and gradient operators are consistent with each being the transpose of the other, and the discretizations have been chosen to ensure that this is true with the matrix $\mathbf{A}^{\mathrm{T}}$ representing the discrete gradient operator gr. Because there is no flow normal to the boundary, the discrete analog of Gauss' theorem implies that the rows of $\mathbf{A}$ add to zero, and system (11) is underdetermined. There is at least one redundant equation, so that the dimension of the nullspace D^0_h of $\mathbf{A}$ is

$$d = \dim(\mathrm{Null}[\mathbf{A}]) \leqslant (2IJ - I - J) - IJ + 1 = (I-1)(J-1).$$

To find typical nullvectors for $\mathbf{A}$ consider the flows in Figs. (2a)–(2b). Each of these flows go through four cells around their common corner at the intersection of the two grid lines between the cells, with an equal flow across each of the faces between adjacent cells. If $l = i + (j-1)(I-1)$ for $1 \leqslant i \leqslant I-1$ and $1 \leqslant j \leqslant J-1$, then there is a nullvector $\tilde{\mathbf{c}}_l$ like this associated with the grid intersection at the upper right-hand corner of the (i, j) cell. If $cx_{i,j}$ and $cy_{i,j}$ are the components of $\tilde{\mathbf{c}}_l$ that correspond to $u_{i,j}$ and $v_{i,j}$, then the only nonzero components of $\tilde{\mathbf{c}}_l$ are

$$cx_{i,j} = \frac{1}{h}, \qquad cy_{i+1,j} = \frac{1}{h}, \qquad cx_{i,j+1} = -\frac{1}{h}, \qquad cy_{i,j} = -\frac{1}{h}.$$



c

i	1	1	2	2	3	3	4	1	1	2	2	3	3	4	1	1	2	2	3	3	4	1	2	3
j	1	1	1	1	1	1	1	2	2	2	2	2	2	2	3	3	3	3	3	3	3	4	4	4
velocity component	*u*	*v*	*u*	*v*	*u*	*v*	*v*	*u*	*v*	*u*	*v*	*u*	*v*	*v*	*u*	*v*	*u*	*v*	*u*	*v*	*v*	*u*	*u*	*u*
	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	−1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	−1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	−1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	−1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	−1	0	0	0	−1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	−1	0	0	0	−1	0	1	1	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	−1	0	0	0	0	−1	0	1	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	−1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	−1	0	0	0	−1	0	1	1	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	−1	0	0	0	−1	0	1	1	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	−1	0	0	0	0	−1	0	1	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	−1	0	0	0	0	0	1	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	−1	0	0	0	−1	1	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	−1	0	0	−1	1
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	−1	0	0	−1

d

i	1	1	2	2	3	3	4	1	1	2	2	3	3	4	1	1	2	2	3	3	4	1	2	3
j	1	1	1	1	1	1	1	2	2	2	2	2	2	2	3	3	3	3	3	3	3	4	4	4
velocity component	*u*	*v*	*u*	*v*	*u*	*v*	*v*	*u*	*v*	*u*	*v*	*u*	*v*	*v*	*u*	*v*	*u*	*v*	*u*	*v*	*v*	*u*	*u*	*u*
	1	−1	0	1	0	0	0	−1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	1	−1	0	1	0	0	0	−1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	1	−1	1	0	0	0	0	−1	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	1	−1	0	1	0	0	0	−1	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	1	−1	0	1	0	0	0	−1	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	1	−1	1	0	0	0	0	−1	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	−1	0	1	0	0	0	−1	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	−1	0	1	0	0	−1	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	−1	1	0	0	−1

FIG. 2. (a) Nullvector flows on a 4 × 4 grid. (b) Nonzero components of a typical nullvector flow. (c) Components of **A** for a 4 × 4 grid, scaled by h. (d) Components of nullvectors for **A** on a 4 × 4 grid, scaled by h.

A velocity vector of this form is discretely divergence free with zero net flow through every cell in the mesh, and with $\mathbf{A}\tilde{\mathbf{c}}_l = \tilde{0}$. These nullvectors are clearly all linearly independent, and there are $d = (I-1)(J-1)$ of them. Consequently,

$$d = \dim(\mathrm{Null}[\mathbf{A}]) = (I-1)(J-1),$$

and we now have a basis for the nullspace D_h^0 of **A** to work with. Note that for $1 \leqslant l \leqslant d$, the inner product of $\tilde{\mathbf{c}}_l$ with any $\tilde{\mathbf{u}} \in V_h$ is

$$(\tilde{\mathbf{c}}_l, \tilde{\mathbf{u}}) = \frac{u_{i,j} - u_{i,j+1}}{\Delta y} + \frac{v_{i+1,j} - v_{i,j}}{\Delta x} \approx \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}, \tag{12}$$

ot the discrete curl, where the derivatives are evaluated at the grid intersection about which the nullvector $\tilde{\mathbf{c}}_l$ is defined. The nullvectors $\{\tilde{\mathbf{c}}_l\}_{l=1}^{d}$ for **A** will be arranged in order as the columns of a $(2IJ-I-J)\times d$ matrix **C**.

For the sake of completeness, the matrix **A** and its nullvectors $\{\tilde{\mathbf{c}}_l\}_{l=1}^{d}$ for a 4 × 4 uniform mesh are given in Figs. (2c)–(2d) as two arrays. The entries in each array have been labeled by cell and by velocity component within cell. On this mesh there are 3 unknown u velocities for each of the 4 mesh rows, and 4 unknown v velocities for each of the first 3 mesh rows, so that an element in D_h^0 will have 24 components. The pressure on this mesh will have 16 components. The 16×24 matrix **A** multiplied by the scale factor $h=\Delta x=\Delta y$ is presented in Fig. 2c. The $d=9$ scaled nullvectors for **A** are given in Fig. 2d. This basis is identical to the one given by Amit *et al.* [1], but can be obtained using local mass balance as above instead of network theory.

The divergence constraint is defined for discrete velocity vectors with homogeneous boundary data, so there is no need to follow the general method of Section 2 and find a particular solution of the continuity equations $\tilde{\mathbf{w}}_p(t)\in V_h$ to account for the boundary conditions of this problem. Since $\tilde{\mathbf{u}}^m\in D_h^0$ at time t_m, and since $\tilde{\mathbf{w}}_p(t)\in V_h$ is not needed, there exists scalars $\{z_l^m\}_{l=1}^{d}$ such that

$$\tilde{\mathbf{u}}^m=\sum_{l=1}^{d} z_l^m\tilde{\mathbf{c}}_l=\mathbf{C}\mathbf{z}^m, \tag{13}$$

where $\mathbf{z}^m=(z_1^m, z_2^m, ..., z_d^m)^{\mathrm{T}}$. Equations for $\{z_l^m\}_{l=1}^{d}$ are obtained by substituting the expansion (13) into (9), and then taking the inner product between each of the expansion vectors $\{\tilde{\mathbf{c}}_l\}_{l=1}^{d}$ and Eq. (9). The inner products between the expansion vectors and Eq. (9) can be simultaneously computed by premultiplying (9) with the transpose of **C** to obtain

$$\frac{1}{\Delta t}\mathbf{C}^{\mathrm{T}}\mathbf{C}\mathbf{z}^{n+1}-\frac{1}{\Delta t}\mathbf{C}^{\mathrm{T}}\mathbf{C}\mathbf{z}^{n}+\frac{1}{2}\mathbf{C}^{\mathrm{T}}\mathrm{na}(\mathbf{C}\mathbf{z}^{n+1})+\frac{1}{2}\mathbf{C}^{\mathrm{T}}\mathrm{na}(\mathbf{C}\mathbf{z}^{n})$$
$$+\frac{1}{2}\mathbf{C}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\tilde{p}^{n+1}+\frac{1}{2}\mathbf{C}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\tilde{p}^{n}=0. \tag{14}$$

But each column of **C** is a nullvector for **A**, so that

$$\mathbf{C}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}=(\mathbf{A}\mathbf{C})^{\mathrm{T}}=0,$$

and we may therefore write Eq. (14) as

$$\mathbf{C}^{\mathrm{T}}\mathbf{C}\mathbf{z}^{n+1}+\frac{\Delta t}{2}\mathbf{C}^{\mathrm{T}}\mathrm{na}(\mathbf{C}\mathbf{z}^{n+1})=\mathbf{C}^{\mathrm{T}}\mathbf{C}\mathbf{z}^{n}-\frac{\Delta t}{2}\mathbf{C}^{\mathrm{T}}\mathrm{na}(\mathbf{C}\mathbf{z}^{n}), \tag{15}$$

where na is a nonlinear operator. Equation (15) for this algorithm corresponds to the general equation (7) with $\theta=\frac{1}{2}$. The unknown velocity $\tilde{\mathbf{u}}^{n+1}$ is found from

Eq. (13) after the expansion coefficients $\mathbf{z}^{n+1}$ are obtained as the solution to the nonlinear system (15), using the known solution at time t_n. The essential issue in developing a code from (15) is the solution of the nonlinear system. The usual methods (see Ortega and Rheinboldt [14]) require the Jacobian $\mathbf{C}^{\mathrm{T}}(\partial \mathrm{na}/\partial \tilde{\mathbf{u}})\mathbf{C}$, which has only 13 nonzero diagonals in this case. Any component of this matrix product can be written as the sum of 16 terms from $\partial \mathrm{na}/\partial \tilde{\mathbf{u}}$, so the computational overhead for the calculation of this matrix product can be avoided.

For many applications the nonlinear implicit equations (15) present too great a computational overhead to be practical. An alternative algorithm that is more efficient may be derived from the primitive variable Crank–Nicolson Adams–Bashforth scheme (8) with lagged nonlinear terms. The diffusion terms are treated implicitly to avoid the numerical stability restrictions from the viscous terms, and the convection terms are lagged to avoid the computational effort of solving a nonlinear system at each time step. If we use the notation of (8), then this discretization of the primitive variable momentum equation (1a) without the body force may be written as

$$\frac{\tilde{\mathbf{u}}^{n+1}-\tilde{\mathbf{u}}^{n}}{\Delta t}+\frac{1}{2}\left(3\mathrm{cv}(\tilde{\mathbf{u}}^{n})-\mathrm{cv}(\tilde{\mathbf{u}}^{n-1})-\mathrm{df}(\tilde{\mathbf{u}}^{n+1})-\mathrm{df}(\tilde{\mathbf{u}}^{n})\right),$$

$$+\frac{1}{2}\left(\mathbf{A}^{\mathrm{T}}\tilde{p}^{n+1}+\mathbf{A}^{\mathrm{T}}\tilde{p}^{n}\right)=0, \tag{16}$$

where cv and df are discretizations of the convection and diffusion terms, respectively, and where $\mathbf{A}^{\mathrm{T}}$ is a matrix representation of the discrete gradient operator as above in this section. If $\mathbf{C}$ is the matrix with nullvectors of $\mathbf{A}$ for its columns as above in this section, then the FDG method applied to (16) as the underlying primitive variable discretization will lead to

$$\mathbf{C}^{\mathrm{T}}\mathbf{C}\mathbf{z}^{n+1}-\frac{\Delta t}{2}\mathbf{C}^{\mathrm{T}}\,\mathrm{df}(\mathbf{C}\mathbf{z}^{n+1})$$

$$=\mathbf{C}^{\mathrm{T}}\mathbf{C}\mathbf{z}^{n}+\frac{\Delta t}{2}\mathbf{C}^{\mathrm{T}}\,\mathrm{df}(\mathbf{C}\mathbf{z}^{n})-\frac{3\,\Delta t}{2}\mathbf{C}^{\mathrm{T}}\,\mathrm{cv}(\mathbf{C}\mathbf{z}^{n})+\frac{\Delta t}{2}\mathbf{C}^{\mathrm{T}}\,\mathrm{cv}(\mathbf{C}\mathbf{z}^{n-1}), \tag{17}$$

where cv is a nonlinear operator. There are various ways to efficiently deal with the linear problem posed by the resulting block pentadiagonal coefficient matrix for the implicit terms in (17), as in Roache and Ellis [17]. Amit *et al.* [1] use a frontal technique for solving analogous equations from the dual variable method. General techniques for solving banded or sparse linear systems are also available.

As an estimate of comparitive efficiency for either of these schemes, the primitive variable formulation in space dimension s has $O(M_i+m_i)=O((s+1)m_i)$ unknown discrete velocity components and pressure values at each time step, while the FDG method has only $O((s-1)m_i)$ unknown coefficients at each time step. The reduction in the number of variables is $O(2m_i)$, or a factor of $\frac{1}{3}$ in $\mathbf{R}^2$ and a factor of $\frac{1}{2}$

in $\mathbf{R}^3$. When storage requirements and operation counts are considered the reduction in effort becomes even greater. Note that the FDG method produces algorithms in which the order of the number of unknowns is equal to that for streamfunction or velocity potential algorithms with fourth-order governing equations in $\mathbf{R}^2$ or $\mathbf{R}^3$. It should also be noted that the algorithms in this form do not actually require the computation of the various matrix products with $\mathbf{C}$ and $\mathbf{C}^{\mathrm{T}}$, since each of the terms in Eqs. (15) and (17) can be obtained as the weighted sum of at most 16 scalars. The next section will show that the dual variable or finite difference Galerkin algorithms can actually be interpreted as streamfunction algorithms. This discovery resulted from trying to understand and simplify the product terms in the FDG algorithms as expressed in Eqs. (15) or (17).

4. The Discrete Streamfunction Interpretation

Even though the algorithm development using the FDG method is formally complete, it is useful and informative to consider (15) and (17) in greater detail. As above, let us write $\mathrm{na}(\tilde{\mathbf{u}}) = \mathrm{cv}(\tilde{\mathbf{u}}) - \mathrm{df}(\tilde{\mathbf{u}})$, where cv and df are the discrete representations of the convection and diffusion terms. It will be convenient to introduce an indexing for the expansion coefficients $\{z_l^m\}_{l=1}^d$ as a 2-dimensional array

$$\{z_{i,j}^m : 1 \leqslant i \leqslant I-1,\ 1 \leqslant j \leqslant J-1\},$$

where $z_{i,j}^m$ is the $i+(j-1)(I-1)$ component of $\mathbf{z}^m$. This is a natural ordering of the expansion coefficients, since it associates the (i, j) coefficient with the nullvector for $\mathbf{A}$ that represents a flow through the four cells around the intersection of grid lines in the upper right-hand corner of the (i, j) cell. Both indexings for the expansion coefficients will be implicitly used in the next three equations, with the left-hand sides of the equations using the single subscript l, and the right-hand sides using the double subscripts $\{i, j\}$. The premultiplication of the discrete momentum equations by $\mathbf{C}^{\mathrm{T}}$ can easily be analysed locally throughout the grid, or by rows of the resulting product. If a row of $\mathbf{C}^{\mathrm{T}}$ corresponds to the null vector with coefficient $z_{i,j}$, then the corresponding component equation of the premultiplied momentum equations is just the weighted sum of the discrete momentum equations for the primitive velocity components $u_{i,j}$, $u_{i,j+1}$, $v_{i,j}$, and $v_{i+1,j}$. The discrete velocity components can now be replaced by the appropriate components of $\mathbf{Cz}^{n+1}$, $\mathbf{Cz}^n$, and $\mathbf{Cz}^{n-1}$, as given by (21a)–(21b) below, and the details of transforming Eqs. (15) and (17) into Eqs. (18) and (19) become apparant. These calculations are straightforward but messy algebra and are included only in summary because of their length. With these conventions, and at time t_m,

$$\mathbf{C}^{\mathrm{T}}\mathbf{Cz}^m = \mathrm{La}(\mathbf{z}^m),$$

where La is the conventional 5-point centered difference approximation to the Laplacian,

$$\mathbf{C}^{\mathrm{T}}\,\mathrm{df}(\mathbf{C}\mathbf{z}^m)=\frac{1}{\mathrm{Re}}\,\mathrm{Bi}(\mathbf{z}^m),$$

where Bi is the conventional 13-point centered difference approximation to the biharmonic operator, and

$$\mathbf{C}^{\mathrm{T}}\,\mathrm{cv}(\mathbf{C}\mathbf{z}^m)=\delta_x(\delta_y(\mathbf{z}^m)\,\mathrm{La}(\mathbf{z}^m))-\delta_y(\delta_x(\mathbf{z}^m)\,\mathrm{La}(\mathbf{z}^m)),$$

where δ_x and δ_y are the conventional centered difference operators

$$\delta_x(\mathbf{z}^m)_{i,j}=\frac{z^m_{i+1,j}-z^m_{i-1,j}}{2\,\Delta x}\qquad\text{and}\qquad\delta_y(\mathbf{z}^m)_{i,j}=\frac{z^m_{i,j+1}-z^m_{i,j-1}}{2\,\Delta y}.$$

The actual equation for $\mathbf{C}^{\mathrm{T}}\,\mathrm{cv}(\mathbf{C}\mathbf{z}^m)$ is given in an Appendix. With this notation, we may write (15) as

$$\begin{aligned}&\mathrm{La}(\mathbf{z}^{n+1})-\frac{\Delta t}{2\,\mathrm{Re}}\,\mathrm{Bi}(\mathbf{z}^{n+1})\\&\quad+\frac{\Delta t}{2}\,[\delta_x(\delta_y(\mathbf{z}^{n+1})\,\mathrm{La}(\mathbf{z}^{n+1}))-\delta_y(\delta_x(\mathbf{z}^{n+1})\,\mathrm{La}(\mathbf{z}^{n+1}))]\\&\quad=\mathrm{La}(\mathbf{z}^n)+\frac{\Delta t}{2\,\mathrm{Re}}\,\mathrm{Bi}(\mathbf{z}^n)-\frac{\Delta t}{2}\,[\delta_x(\delta_y(\mathbf{z}^n)\,\mathrm{La}(\mathbf{z}^n))-\delta_y(\delta_x(\mathbf{z}^n)\,\mathrm{La}(\mathbf{z}^n))],\end{aligned}\tag{18}$$

and we may write (17) as

$$\begin{aligned}&\mathrm{La}(\mathbf{z}^{n+1})-\frac{\Delta t}{2\,\mathrm{Re}}\,\mathrm{Bi}(\mathbf{z}^{n+1})\\&\quad=\mathrm{La}(\mathbf{z}^n)+\frac{\Delta t}{2\,\mathrm{Re}}\,\mathrm{Bi}(\mathbf{z}^n)-\frac{3\,\Delta t}{2}\,[\delta_x(\delta_y(\mathbf{z}^n)\,\mathrm{La}(\mathbf{z}^n))-\delta_y(\delta_x(\mathbf{z}^n)\,\mathrm{La}(\mathbf{z}^n))]\\&\quad+\frac{\Delta t}{2}\,[\delta_x(\delta_y(\mathbf{z}^{n-1})\,\mathrm{La}(\mathbf{z}^{n-1}))-\delta_y(\delta_x(\mathbf{z}^{n-1})\,\mathrm{La}(\mathbf{z}^{n-1}))].\end{aligned}\tag{19}$$

Let us take $\mathbf{z}^m$ to be a discrete approximation at time t_m of a scalar function ψ defined on Ω, where the discretization $\mathbf{z}^m$ is defined at the $(I-1)(J-1)$ points of intersection of the grid lines between the mesh cells in the staggered grid. Note that the approximation

$$\mathbf{z}^m\approx\psi(t_m)$$

is defined at the same discrete times as the primitive variable solution, but that it is defined on a different spatial grid. Now notice that the linear terms in (18) and

(19) represent a conventional centered space approximation for the linear terms in the fourth-order stream function equation

$$\frac{\partial \Delta\psi}{\partial t}=\frac{1}{\mathrm{Re}}\Delta^2\psi+\frac{\partial\psi}{\partial x}\Delta\frac{\partial\psi}{\partial y}-\frac{\partial\psi}{\partial y}\Delta\frac{\partial\psi}{\partial x}. \tag{20}$$

Taylor series expansions of these two algorithms show that they are in fact second-order centered space discretizations of the fourth-order streamfunction equation (20). The usual streamfunction boundary conditions for this problem are $\psi=0$ on the entire boundary $\partial\Omega$, and $\partial\psi/\partial\eta=0$ on the walls while $\partial\psi/\partial\eta=1$ on the lid. These boundary conditions are consistent with the modification of algorithms (18) or (19) in cells near the boundary that results from using conventional primitive variable exterior cell velocity boundary treatments. Consider also the recovery of the velocity components in (13). From Fig. 2a we see that $u_{i,j}$ on the right-hand face of the (i, j) cell has flow contributions only from the nullvectors $\tilde{\mathbf{c}}_l$ for $l=i+(j-1)(I-1)$ when $j\leqslant J-2$, and for $l=i+(j-2)(I-1)$ when $2\leqslant j$, with

$$u_{i,j}=\frac{1}{\Delta y}(z_{i,j}-z_{i,j-1})\approx\frac{\partial\psi}{\partial y}. \tag{21a}$$

Similarly, $v_{i,j}$ on the top face of the cell is

$$v_{i,j}=\frac{1}{\Delta x}(z_{i-1,j}-z_{i,j})\approx-\frac{\partial\psi}{\partial z}. \tag{21b}$$

If $\mathbf{z}^m$ is interpreted as a discrete vector field perpendicular to the (x, y) plane, then the entire discrete velocity field is $\tilde{\mathbf{u}}^m=\mathbf{C}\mathbf{z}^m$, so that premultiplication by $\mathbf{C}$ may be interpreted as taking the discrete curl of $\mathbf{z}^m$. Recall that a discretization was initially formulated in terms of the primitive variables and then premultiplied by $\mathbf{C}^{\mathrm{T}}$ in (15) or (17), and by (12) this may be interpreted as taking the discrete curl of the discrete momentum equation, where the resulting product only has nonzero components perpendicular to the (x, y) plane. We may therefore interpret $\mathbf{z}^m$ as defining a discrete streamfunction at time t_m on the $(I-1)\times(J-1)$ points of intersection of the grid lines between the mesh cells in the staggered grid. For the primitive variable discretization and nullvectors of $\mathbf{A}$ that we have chosen, the discrete manipulations of the FDG method are the exact analog of the manipulations of the continuum Navier–Stokes equations that are used to derive the fourth-order streamfunction equation.

When a staggered grid is used, the resulting staggered velocity components must be interpolated in order to obtain all of the velocity components at the same grid point. Let us average the x velocity components in the y direction and the y velocity components in the x direction, with

$$U^m_{i,j}=\tfrac{1}{2}(u^m_{i,j+1}+u^m_{i,j}) \qquad \text{and} \qquad V^m_{i,j}=\tfrac{1}{2}(v^m_{i+1,j}+v^m_{i,j}),$$

for $1 \leqslant i \leqslant I-1$ and $1 \leqslant j \leqslant J-1$ at time t_m. For the usual centered difference operators defined above,

$$\delta_x(U^m_{i,j}) + \delta_y(V^m_{i,j}) = 0,$$

so that in this sense the interpolated solution is exactly divergence free throughout the interior mesh and at every time t_m. The interpolated velocity solution $(U^m_{i,j}, V^m_{i,j})$ may also be obtained directly from the discrete streamfunction solution as

$$U^m_{i,j} = \frac{1}{2\,\Delta y}(z^m_{i,j+1} - z^m_{i,j-1}) \approx \frac{\partial\psi}{\partial y}$$

and

$$V^m_{i,j} = -\frac{1}{2\,\Delta y}(z^m_{i+1,j} - z^m_{i-1,j}) \approx -\frac{\partial\psi}{\partial x},$$

where the derivatives are evaluated at the intersection of the grid lines at the upper right corner of the (i, j) cell. We can formulate a primitive variable discretization using the advantages for mass balance of a staggered grid with velocity components defined on cell faces, then use the FDG method to obtain a related streamfunction algorithm that automatically creates a streamfunction boundary treatment from the primitive variable boundary treatment, and finally recover a discretely divergence free velocity solution with all components defined at every grid point. Both types of mesh may be used where they are convenient, and a pressure solution never has to be considered unless it is desired.

There are two possible views of the FDG and dual variable methods. If attention is focused on the primitive variable discretization, then these methods may be viewed as variable reduction techniques that can lead to a more efficient solution of the primitive variable algorithm equations. If attention is focused on interpreting the application of the FDG and dual variable methods, then for 2-dimensional problems these methods may be viewed as techniques for developing streamfunction algorithms that are based upon the underlying primitive variable discretization and its properties. If appropriate choices are made in the discretization, then the FDG and dual variable methods will produce an algorithm for the fourth-order streamfunction equation which will give a discrete primitive variable solution with characteristics inherent in the primitive variable discretization. In their discussion of the FDG method for steady problems Stephens *et al.* [19] suggested that a discrete streamfunction interpretation is possible for the coefficient vector **z**. Amit *et al.* [1] point out that their dual variable algorithm does not lead to a discretization of the fourth-order streamfunction equation (20), but that (21a)–(21b) do hold for their method, so their dual variables may be intepreted as a discrete streamfunction. The streamfunction interpretation of the coefficient vectors depends upon the choice of the basis vectors for the nullspace of **A**, and the choice of the primitive variable discretization. If a choice of the basis vectors for D_h^0 is made so that (21a)–(21b) does not apply, then the discrete streamfunction interpretation of the expansion coefficients $\mathbf{z}^m$ would be unreasonable. If a basis for the nullspace of the discrete divergence operator is chosen so that (21a)–(21b) does hold, then the streamfunction interpretation is reasonable even if the algorithm for $\mathbf{z}^{n+1}$ is not recognizable as a discretization of the streamfunction equation (20). There is a nonsingular transformation between any two bases for the nullspace of the discrete divergence operator, so that the expansion coefficients with respect to any basis will uniquely determine the expansion coefficients with respect to any other basis. In this sense, an expansion coefficient solution with respect to any basis may be interpreted as a discrete streamfunction. For 3-dimensional problems the FDG and dual variable methods are related to the formulation of algorithms for the velocity potential.

5. Solving General Flow Problems

The analysis of the divergence constraint is the crucial issue for applying the FDG method to general problems, and results in the decomposition of the discrete solution vector $\tilde{\mathbf{u}}^m$ at time t_m as the sum of two discretely divergence free vectors $\tilde{\mathbf{u}}^m = \tilde{\mathbf{w}}_p^m + \tilde{\mathbf{w}}_0^m$, with $\tilde{\mathbf{w}}_p^m$ assuming any boundary values imposed on the solution, and with $\tilde{\mathbf{w}}_0^m$ equal to zero where boundary values are prescribed. If $\mathbf{A}$ is the matrix representation of the discrete divergence operator for some choice of grid and primitive variable discretization, then the issues are to find a basis for the nullspace [illegible] be found only once, but the particular solution $\mathbf{w}_p$ might need to be found at every time step if the boundary data changes with time. The continuous solution at any time must satisfy

$$\int_\Omega \nabla \cdot \mathbf{u} \, dv = 0. \tag{22}$$

If we interpret the separate continuity constraint equations as applying to each grid cell or at each grid point, and if an appropriate conservative discretization is used, then the discrete analog of (22) is that the weighted sum over the grid of the separate continuity constraint equations is zero. For problems in which the discrete boundary velocities are all specified, at least the last row of $\mathbf{A}$ is a linear combination of the previous rows, so that $\mathbf{A}$ is not of full row rank, and the prescribed boundary data must be consistent in the sense that the last component of $\tilde{\mathbf{b}}$ is the same linear combination of the previous components. The consistency of the boundary data is a discrete analog of (1e). Amit *et al.* [1] have proven that for general flow problems and grids $\mathbf{A}$ has full row rank if the discrete boundary velocities are not prescribed on at least part of the boundary. The problems of finding a particular solution and a basis for Null[A] when $\mathbf{A}$ has full row rank also occurs in structural analysis, and four techniques based on the LU and QR decompositions have been presented in Kaneko, Lawo, and Thierauf [10] for the solution of these problems. These techniques are independent from the dimension of the problem, from the grid used for the discretization, and from the finite difference approximations that are used. For a general problem the discrete continuity equation may be written as

$$\mathbf{A}\mathbf{w} = \mathbf{b}, \tag{23}$$

where $\mathbf{A}$ is an $M \times N$ matrix, $\mathbf{w}$ is $N \times 1$, and $\mathbf{b}$ is $M \times 1$. We will assume that $M < N$. If $\mathbf{A}$ is not of full row rank, then $\text{Rank}[\mathbf{A}] = M - \delta$, so that

$$d = \dim(\text{Null}[\mathbf{A}]) = N - M + \delta.$$

Since there are δ dependent equations in the linear constraint system, there is an $M \times M$ permutation matrix $\mathbf{P}$, and an $M \times (M - \delta)$ matrix $\mathbf{R}$ such that

$$\mathbf{A} = \mathbf{P}\begin{pmatrix}\mathbf{A}_1\\ \mathbf{A}_2\end{pmatrix} = \mathbf{P}\mathbf{R}\mathbf{A}_1 \qquad \text{and} \qquad \mathbf{b} = \mathbf{P}\begin{pmatrix}\mathbf{b}_1\\ \mathbf{b}_2\end{pmatrix} = \mathbf{P}\mathbf{R}\mathbf{b}_1,$$

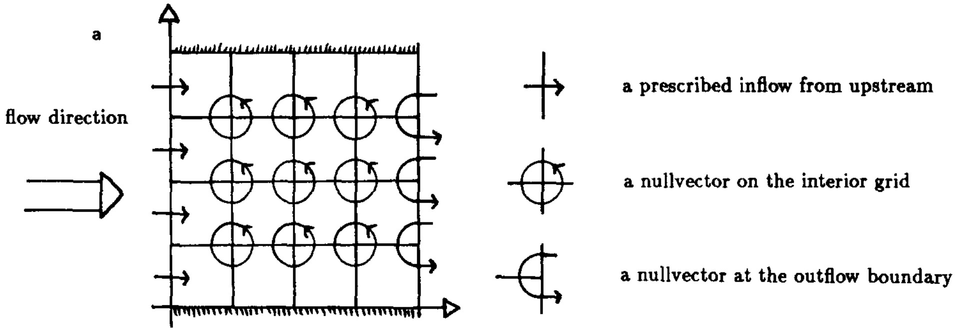
where $\mathbf{A}_1$ is $(M - \delta) \times N$, $\mathbf{A}_2$ is $\delta \times N$, $\mathbf{b}_1$ is $(M - \delta) \times 1$, $\mathbf{b}_2$ is $\delta \times 1$, and where $\mathbf{A}_1$ has full row rank. The methods of Kaneko *et al.* [10] may now be used on $\mathbf{A}_1$, and the results can be adapted to $\mathbf{A}$. The dual variable method (see Amit *et al.* [1] and Hall [8]) uses network theory algorithms to solve these problems.

A basis for the nullspace of $\mathbf{A}$ and a particular solution to (23) can both be found even for quite complicated problems by the methods of Section 3 that are based on mass balance considerations. We will present simple examples of how to deal with two specific problem types using a staggered grid. Figure 3(a) presents a 4×4 syaggered grid for a typical throughflow computation. We will use the same indexing as in Section 3. The u velocity for $\mathrm{i} = 0$ and $j = 1, 2, 3, 4$ are specified as the given inflow, the u velocity for $i = 1, 2, 3, 4$ and $j = 1, 2, 3, 4$, and the v velocity for $i = 1, 2, 3, 4$ and $j = 1, 2, 3$ are all undetermined. The v velocity for $j = 0$ or $j = 4$ and $i = 1, 2, 3, 4$ are all set to 0 as wall velocities. There are 28 unknown velocities and 16 unknown pressure values. The discrete divergence matrix $\mathbf{A}$ is 16×28, and $h\mathbf{A}$ is given in Fig. 3(b), where $h = \Delta x = \Delta y$. In this case the data vector $\tilde{\mathbf{b}}$ from the prescribed inflow has transpose

$$\tilde{\mathbf{b}}^{\mathrm{T}} = \frac{1}{h}(u_{0,1}, 0, 0, 0, u_{0,2}, 0, 0, 0, u_{0,3}, 0, 0, 0, u_{0,4}, 0, 0, 0),$$

where $\{u_{0,j}\}_{j=1}^{4}$ are velocity components that are normal to the inflow boundary. The equations $\mathbf{A}\tilde{\mathbf{w}}_p = \tilde{\mathbf{b}}$ add up to

$$\frac{1}{h}\sum_{j=1}^{4} u_{4,j} = \frac{1}{h}\sum_{j=1}^{4} u_{0,j} \qquad \text{or} \qquad \frac{1}{h}\left(\sum_{j=1}^{4} u_{4,j} - \sum_{j=1}^{4} u_{0,j}\right) = 0,$$


a
flow direction
a prescribed inflow from upstream
a nullvector on the interior grid
a nullvector at the outflow boundary

b

$$\begin{array}{c|cccccccccccccccccccccccccccc}
i & 1 & 1 & 2 & 2 & 3 & 3 & 4 & 4 & 1 & 1 & 2 & 2 & 3 & 3 & 4 & 4 & 1 & 1 & 2 & 2 & 3 & 3 & 4 & 4 & 1 & 2 & 3 & 4 \\
j & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 4 & 4 & 4 & 4 \\
\text{velocity component} & u & v & u & v & u & v & u & v & u & v & u & v & u & v & u & v & u & v & u & v & u & v & u & v & u & u & u & u \\
\hline
 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 & -1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 & 0 & 0 & -1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 & 0 & 0 & 0 & 0 & -1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & -1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & -1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & -1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & -1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & -1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & -1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 \\
 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 1 & 0 \\
 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 1
\end{array}$$

FIG. 3. (a) A 4×4 grid for a throughflow problem. (b) Components of **A** for a 4×4 grid with throughflow, scaled by h.

and this is just the discrete mass balance equation for flow in and out of the entire computational domain. In this case **A** has full row rank (see Amit *et al.* [1]), and the nullspace D_h^0 of **A** has dimension

$$d = \dim(\text{Null}[\mathbf{A}]) = 28 - 16 = 12.$$

Nine linearly independent nullvectors for **A** can be found just as in Section 3, and are illustrated in Fig. 3(a) as the flows around the grid intersections for $1 \leqslant i \leqslant 3$ and $1 \leqslant j \leqslant 3$. The remaining three nullvectors are illustrated in Fig. 3(a) as the semicircular flows in and out of the downstream boundary around the grid intersections for $i = 4$ and $1 \leqslant j \leqslant 3$. These last three nullvectors multiplied by h have transposes

$$h\tilde{\mathbf{c}}_{4,1}^{\mathrm{T}} = (0, 0, 0, 0, 0, 0, 1, -1, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0),$$

$$h\tilde{\mathbf{c}}_{4,2}^{\mathrm{T}} = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, -1, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0),$$

$$h\tilde{\mathbf{c}}_{4,3}^{\mathrm{T}} = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, -1, 0, 0, 0, -1).$$

One particular solution is easy to find, with the inflow being washed straight downstream and out, with transpose scaled by h

$$h\tilde{\mathbf{w}}_p^{\mathrm{T}} = (a, 0, a, 0, a, 0, a, 0, b, 0, b, 0, b, 0, b, 0, c, 0, c, 0, c, 0, c, 0, d, d, d, d),$$

where $a = u_{0,1}$, $b = u_{0,2}$, $c = u_{0,3}$, and $d = u_{0,4}$. Another particular solution can be obtained by introducing a flow along the inflow boundary cell layer and up to the upper wall, then down the wall surface, and out the upper outflow boundary cell. If a, b, c, and d are the prescribed flows across the inflow boundary, then this other particular scaled solution is

$$h\tilde{\mathbf{w}}_p^{\mathrm{T}} = (0, a, 0, 0, 0, 0, 0, 0, 0, a \\ + b, 0, 0, 0, 0, 0, 0, 0, a + b + c, 0, 0, 0, 0, 0, 0, e, e, e, e),$$

where $e = a + b + c + d$. The particular solutions merely state that what goes into the grid must come out at each time step or that mass must be discretely globally conserved. Independently from the choice of nullvectors and a particular solution, the momentum equation algorithm for interior cells must be modified near the boundary to incorporate whatever boundary treatments are being used. After discrete equations that incorporate boundary treatments have been formulated for each of the unknown velocity components on the grid, and with the use of a particular solution such as those above, the FDG method can be employed to expand the solution with respect to the nullvectors obtained above and to derive equations for the expansion coefficients as in (15) and (17) from Section 3.

Figure 4 illustrates a portion of a MAC grid for a problem with an obstacle. The illustrated flow $\tilde{\mathbf{c}}$ around the obstacle is discretely divergence free, and must be included as an independent vector in the basis for the nullspace of the matrix representation $\mathbf{A}$ of the discrete divergence operator. Let us assume that this grid is a portion of the $I \times J$ grid for a driven cavity problem which is otherwise the same as in Section 3, and with the same discretization as in Section 3. With this assumption there are $(I-1)J + I(J-1) - 4$ unknown velocity components, there are $IJ - 1$ unknown pressure values, and there are $(I-1)(J-1) - 4$ discretely divergence free vectors that are defined around cell corners as in Section 3. The matrix $\mathbf{A}$ is $(IJ-1) \times (2IJ - I - J - 4)$ with row rank $IJ - 2$, so that $d = \dim(\mathrm{null}[\mathbf{A}]) = (I-1)(J-1) - 3$. The additional nullvector $\tilde{\mathbf{c}}$ will complete the basis of D_h^0 for this problem. Note that there are only $d - 1$ components for the discrete streamfunction $\mathbf{z}^m$ as defined above in Section 3. The extra nullvector $\tilde{\mathbf{c}}$ will have an expansion coefficient which may be considered as a streamfunction

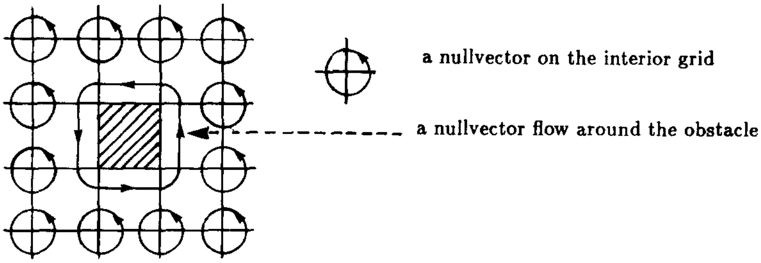

a nullvector on the interior grid
a nullvector flow around the obstacle

FIG. 4. A 4 × 4 grid segment for flow with an obstacle.

value on the surface of the obstacle. It is possible to solve this problem without introducing the nullvector $\tilde{\mathbf{c}}$ and its expansion coefficient into the FDG method. If ψ_0 is the known value of ψ on any bounding surface, and if ψ_s is the unknown [illegible] the surface of the embedded obstacle, then

$$\psi_s - \psi_0 = \int_\gamma u\,dy - v\,dx = 0,$$

where the line integral is along any path γ through the flow domain from the bounding surface where ψ_0 is known to the embedded obstacle (as in Batchelor [2]). The discrete analog of this can be used to calculate the value of the discrete streamfunction on the surface of the embedded obstacle, and this value would then be the expansion coefficient for the extra nullvector $\tilde{\mathbf{c}}$. The FDG method may now proceed with the expansion and equation derivation with just the usual nullvectors for **A** in **C** as above in Section 3. The advantage of not using the extra nullvector $\tilde{\mathbf{c}}$ in the FDG process is that the bandwidth and general from of the equations derived for the expansion coefficients will not be disturbed by the introduction of an expansion vector that flows through relatively many computational cells. More general problems with obstacles embedded in the flow can be handled in a similar fashion.

6. Numerical Examples

In this section we will present the results of numerical calculations obtained with the Crank–Nicolson Adams–Bashforth algorithm from Eq. (17). The coefficient matrix for the linear terms at time t^{n+1} is constant in time and was factored before time stepping with a LINPACK banded LU decomposition subroutine. A LINPACK banded backward solution subroutine was used to solve the linear system for each time step. The reported calculations are for the asymptotic steady state results in a square driven cavity at Re = 400, 1000, and 3200, and for the time evolution or vortex dynamics at Re = 1000 in a square driven cavity and in a rectangular driven cavity with aspect ratio 2. The cavity lid is always the upper surface, the lid is always impulsively started, and the lid always moves from the left to the right. The calculations in the square driven cavity are all with $\Delta t = 0.01$ on a grid with 64 by 64 cells, and the calculations in the rectangular driven cavity are with $\Delta t = 0.025$ on a grid with 40 by 80 cells. The criterion for convergence to steady state was generally taken to be

$$\frac{\|\tilde{\mathbf{u}}^{n+1} - \tilde{\mathbf{u}}^n\|_1}{\|\tilde{\mathbf{u}}^{n+1}\|_1} \leqslant 5.0 \times 10^{-8},$$

where $\|\tilde{\mathbf{u}}\|_1$ is the L_1 norm obtained as the sum of the absolute value of all velocity components on the interior grid multiplied by the cell size. The only exception was

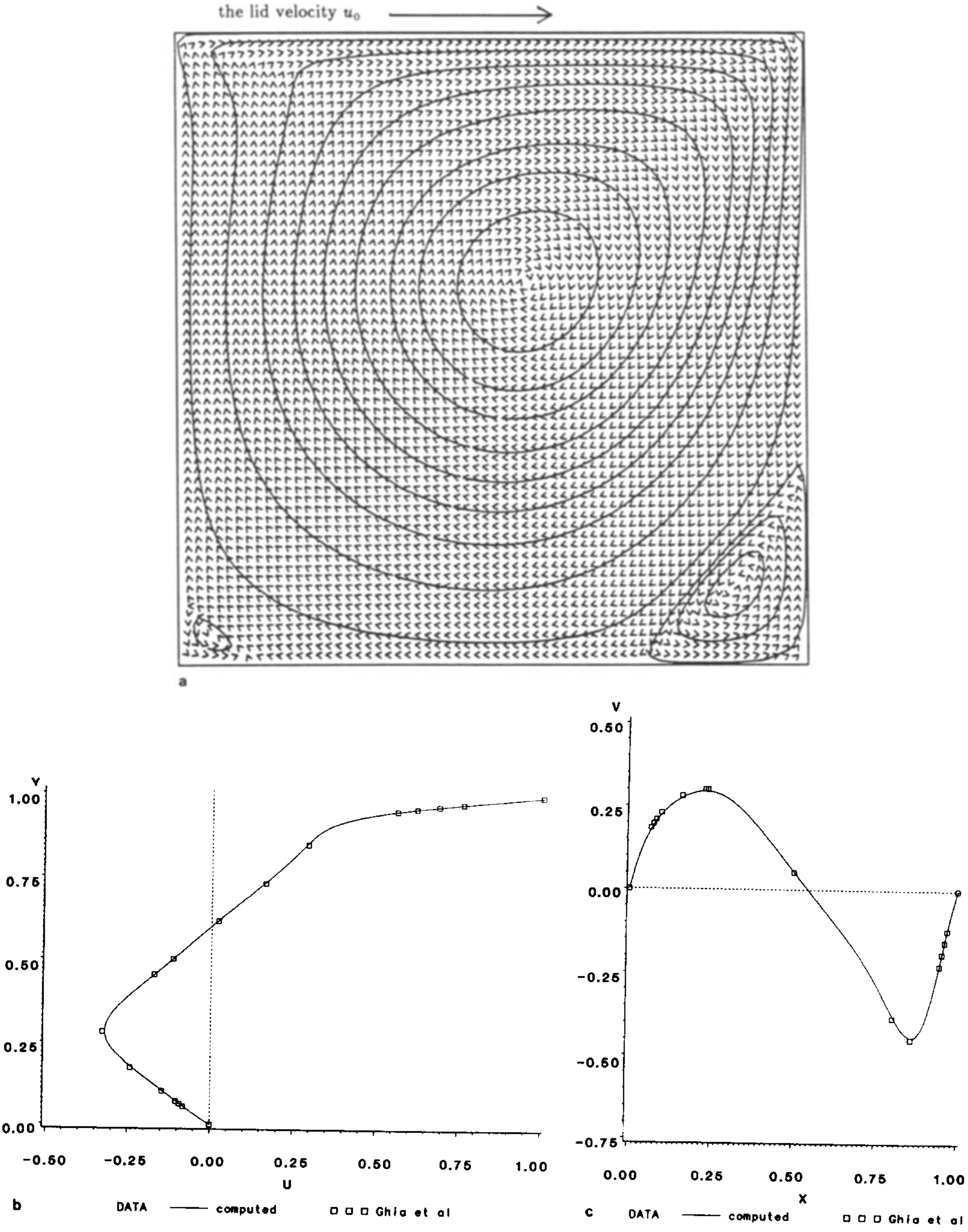


FIG. 5. Asymptotic steady state flow, 64 × 64 grid, streamfunction contours with directional vectors, u at $x = 0.5$, and v at $y = 0.5$: (a)–(c) Re = 400, $t = 46.66$; (d)–(f) Re = 1000, $t = 99.94$; (g)–(i) Re = 3200, $t = 250.00$.

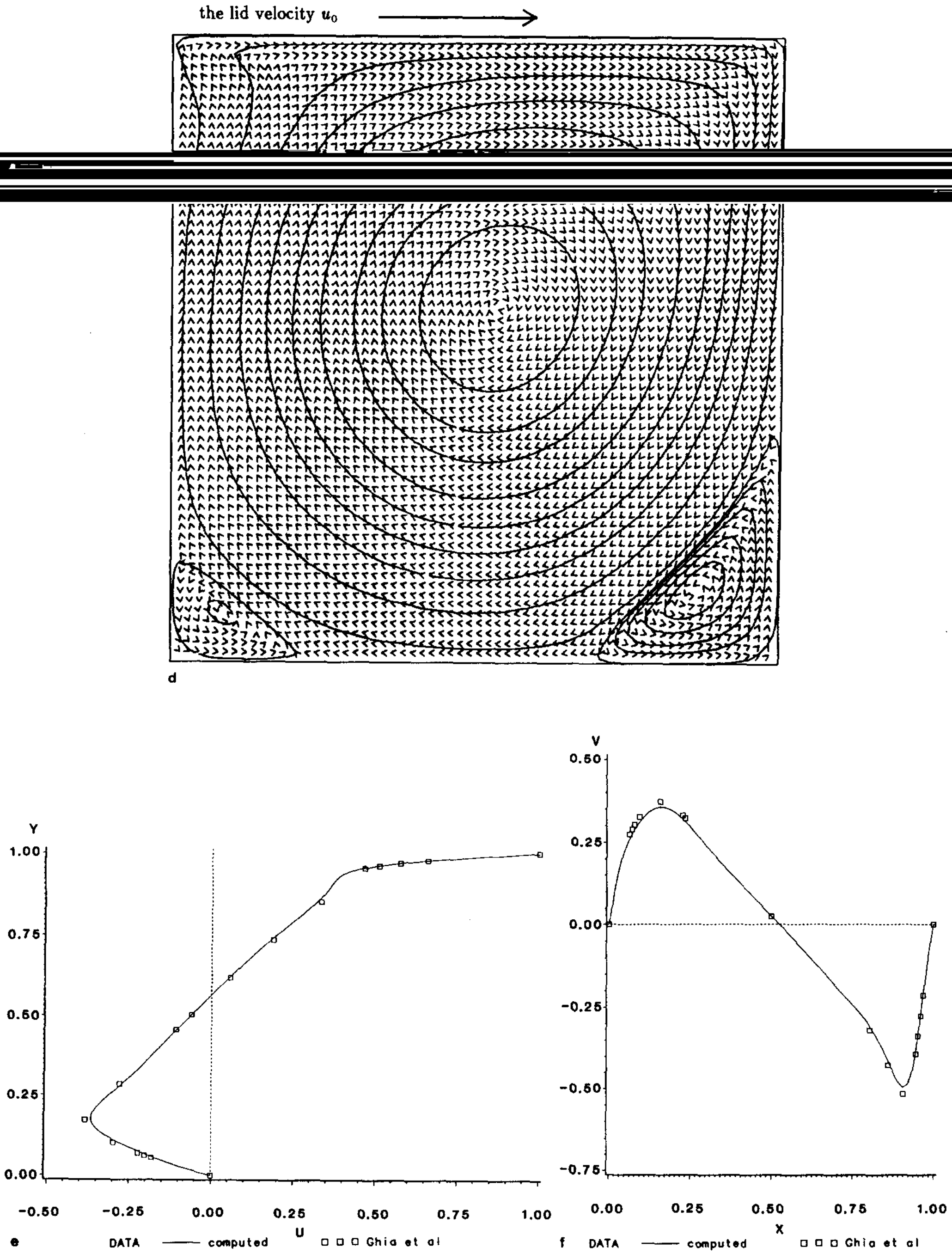


FIG. 5—*Continued*

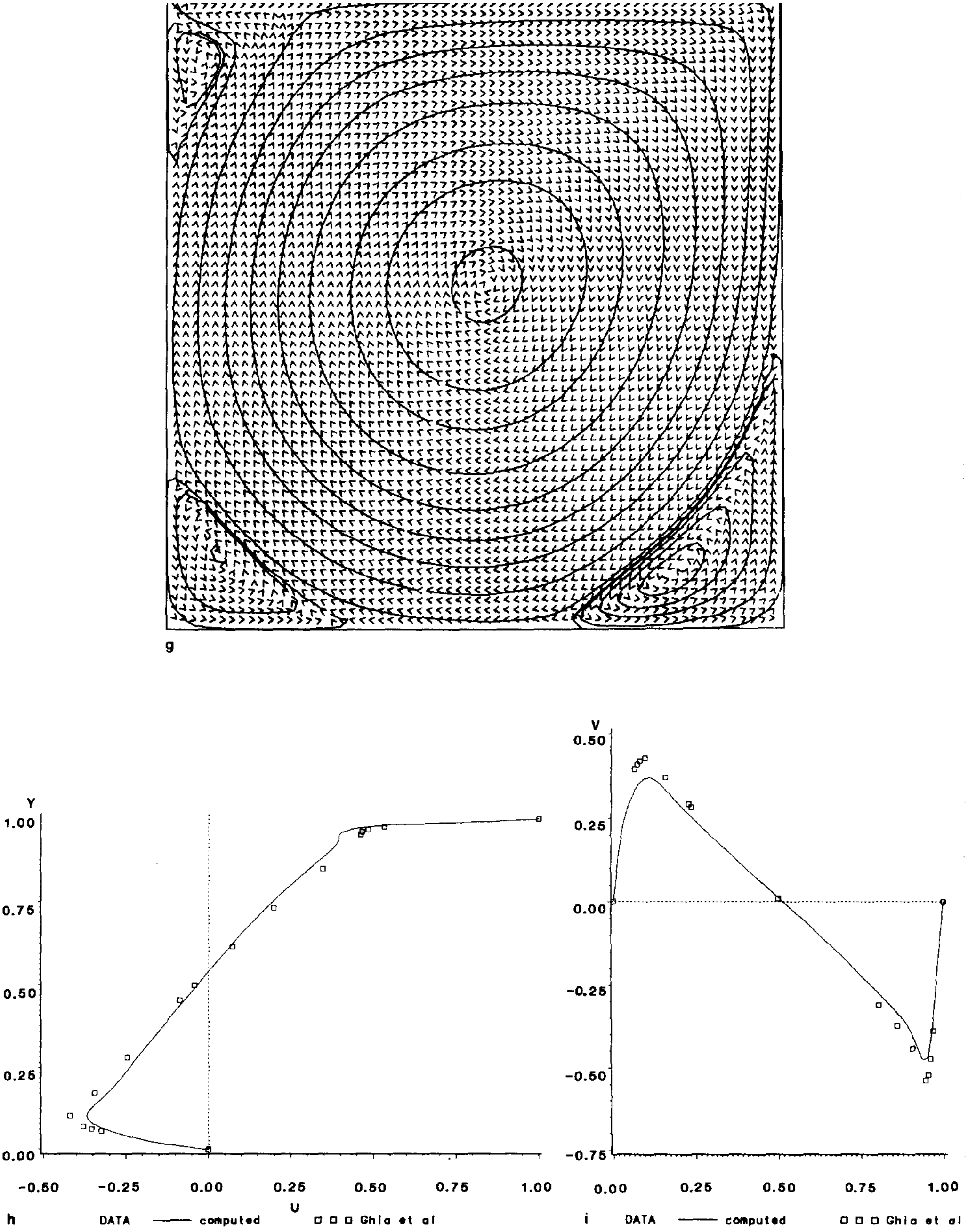


FIG. 5—*Continued*

for the square cavity computation at Re = 3200, where convergence was taken at $t = 250$, with

$$\frac{\|\tilde{\mathbf{u}}^{n+1} - \tilde{\mathbf{u}}^{n}\|_1}{\|\tilde{\mathbf{u}}^{n+1}\|_1} \leqslant 2.74 \times 10^{-7}.$$

The reported computational results for the asymptotic steady state cases are compared with the published data in Ghia, Ghia, and Shin [5], while the reported computational results for the time evolution cases are compared with the vortex dynamics published in Gustafsen and Halasi [6, 7].

The asymptotic results for the square driven cavity are given in Figs. 5(a)–(c) for Re = 400 at $t = 46.63$, in Figs. 5(d)–(f) for Re = 1000 at $t = 99.94$, and in Figs. 5(g)–(i) for Re = 3200 at $t = 250$. Figures 5(a), (d), and (g) give streamfunction contours overlayed with directional vector plots for the velocity fields. These three plots all agree qualitatively with published results and show the primary central vortex, secondary vortices in both of the two lower corners, and a secondary vortex in the upper left corner for Re = 3200. The 64 by 64 grid resolution is too coarse for the resolution of any additional vortices in the corner vortex cascades. Figures 5(b), (e), and (h) give the profile of the x velocity component u as a function of y for $x = 0.5$, and Figures 5(c), (f), and (i) give the profile of the y velocity component v as a function of x for $y = 0.5$. The results from Ghia *et al.* [5] for a grid with 128 by 128 cells are overlayed on these plots. The local extremes for the x velocity component u near the bottom wall differ from the results in Ghia *et al.* [5] by 2.2% at Re = 400, by 4.6% et Re = 1000, and by 12.5% at Re = 3200. The local extremes for the y velocity component v near the downstream wall on the right differ from the results in Ghia *et al.* [5] by 1.6% at Re = 400, by 3.9% at Re = 1000, and by 11.9% at Re = 3200. The velocity profiles are in excellent agreement at Re = 400, they are in very good agreement at Re = 1000, and they are in reasonable agreement at Re = 3200. The local extremes for the streamfunction in the primary vortex are $\psi = -0.1198$ at Re = 400, $\psi = -0.11359$ at Re = 1000, and $\psi = -0.10646$ at Re = 3200. These differ from the published results in Ghia *et al.* [5] by 1.7% at Re = 400, by 3.7% at Re = 1000, and by 11.6% at Re = 3200. The local extremes for the streamfunction in the lower downstream secondary vortex are $\psi = 0.0005749$ at Re = 400, $\psi = 0.001892$ at Re = 1000, and $\psi = 0.003637$ at Re = 3200. These differ from the published results in Ghia *et al.* [5] by 10.5% at Re = 400, by 8.1% at Re = 1000, and by 15.8% at Re = 3200. The strengths of the primary vortices agree with the published data as well as the local velocity extremes, while the strengths of the main secondary vortices are in only fair agreement with the published data from finer grids. The overall agreement with the published data in Ghia *et al.* [5] is excellent at Re = 400, very good at Re = 1000, and reasonable at Re = 3200. The computations for Re = 3200 were stopped with a relative L_1 norm change in the flow field that is five times larger than for Re = 400 and Re = 1000. This relatively premature end of the calculations for Re = 3200 will have made a contribution to the error for the asymptotic results in this case.

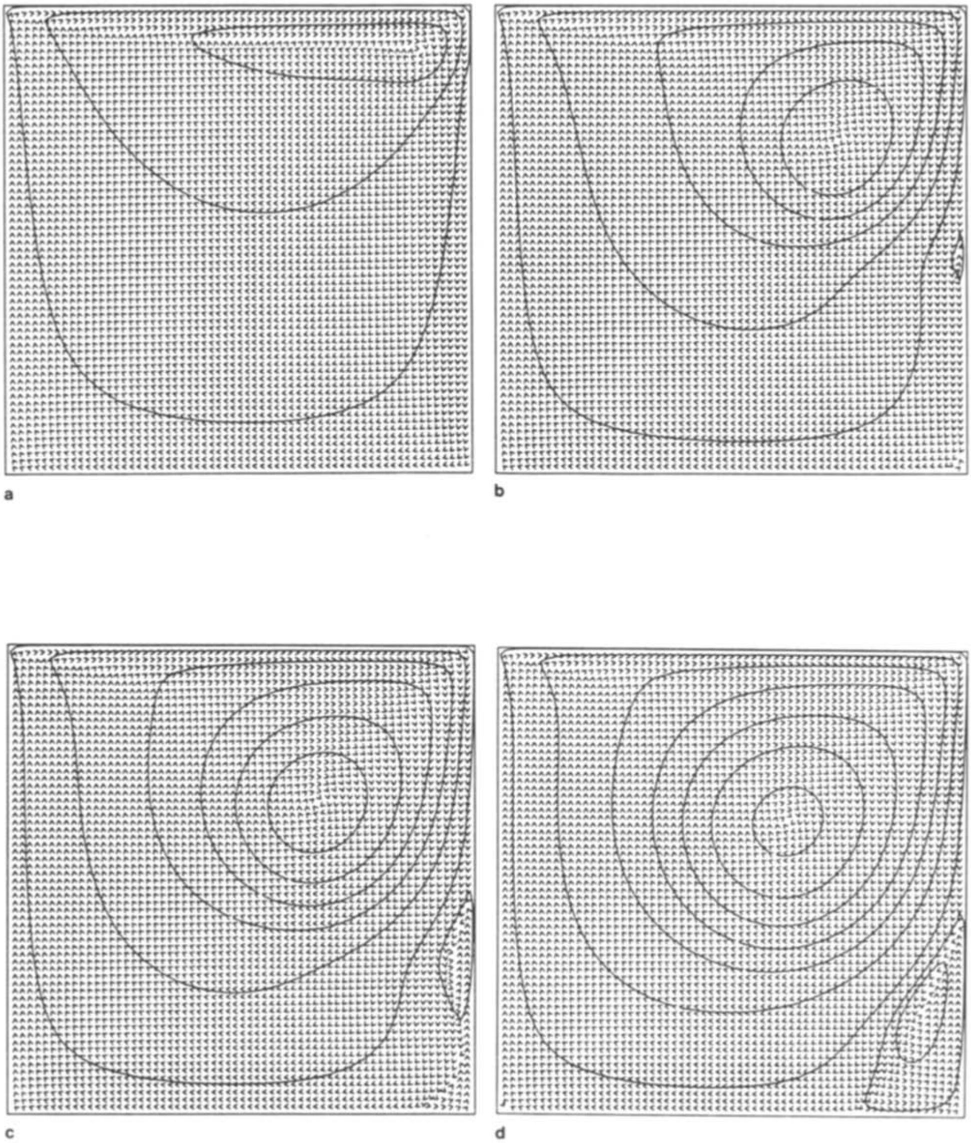


FIG. 6. Time evolution of the square cavity, Re = 1000, 64 × 64 grid, streamfunction contours overlayed with directional vectors: (a)–(f) $t = 1.0$, 4.0, 5.8, 8.0, 16.0, 32.0, respectively.

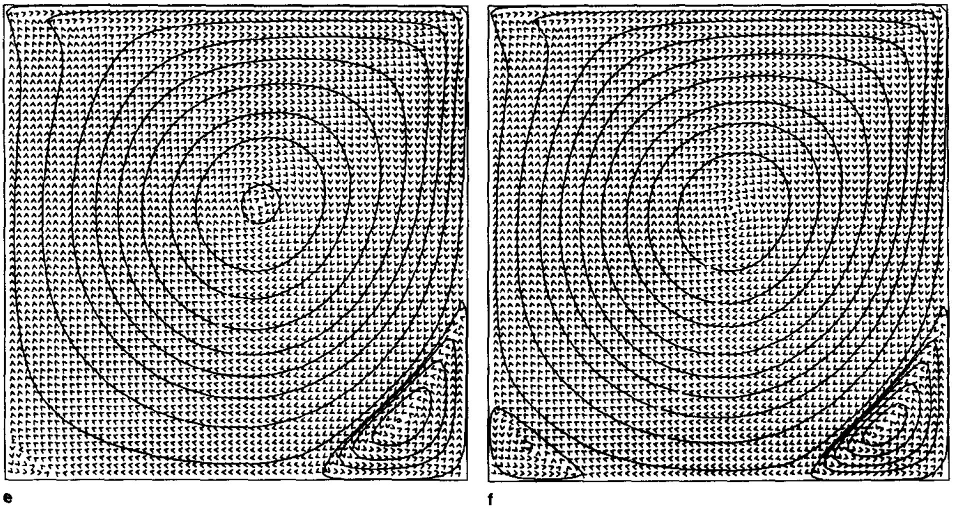


FIG. 6—*Continued*

The time evolution of the driven cavity for Re = 1000 with a 64 by 64 grid is presented in Figs. 6(a)–(f). These figures show streamfunction contour plots overlayed with directional vectors from the velocity field. The interesting feature of these figures is the development in time of the vortex dynamics in the cavity. In Fig. 6(a) at $t = 1.0$ the outer streamfunction contour has begun to lift away from the downstream wall close to its top near the moving lid. By $t = 4.0$ in Fig. 6(b) the negative streamfunction contours around the primary vortex are very distorted by the development of a relatively weak positive streamfunction peak on and around a small recirculation bubble that is located halfway down the right wall. Note that the flow separates above this small recirculation and then reattaches below it, while [redacted] culation on the right wall and reattachment on the bottom wall. Note that the positive streamfunction peak is still above the corner. At $t = 8.0$ in Fig. 6(d) the recirculation on the wall has moved almost entirely into the corner, although it is still underdeveloped along the bottom wall. Note that a distinct recirculation has started in the lower left corner. At $t = 16.0$ in Fig. 6(e) the general form of the primary vortex and the two secondary vortices in the lower corners has been established. The development from $t = 16.0$ to $t = 32.0$ in Fig. 6(f) and the asymptotic state at $t = 99.94$ in Fig. 5(d) is just a strengthening of the vortices that are already present by $t = 16.0$. In the driven cavity there is no bubble type recirculation on the right-hand wall for Re = 400, but the general pattern at Re = 1000 and higher is for the secondary vortex in the lower right corner to develop from a separate eddy on the downstream wall that coalesces with a weaker corner recirculation. The bubble type recirculation on the right hand wall begins to appear between $t = 3.6$ and 3.8 for Re = 1000, and between $t = 2.0$ and $t = 3.0$ for Re = 3200. As Reynolds number increases, the distinct bubble recirculation on the right-hand wall begins earlier and becomes stronger before it coalesces with the corner recirculation on the lower right. The secondary vortices in the lower corners of the square driven cavity begin to appear at about $t = 1.5$ on both sides at Re = 400, at about $t = 3.2$ on the right side and $t = 7.2$ on the left side at Re = 1000, and at about $t = 6.5$ on the right side and $t = 11.5$ on the left side at Re = 3200. The occurrance of the secondary vortices in the lower corners is delayed by the development of the bubble recirculation on the right-hand wall, with a greater delay for the recirculation in the lower left-hand corner. The recirculation in the lower right corner inherits much of its initial strength from the recirculation bubble that descends along the wall to join with the corner recirculation. This pattern is consistent with the vortex dynamics for deeper cavity flows that have been reported by Gustafson and Halasi [6, 7]. The appearance of the recirculating bubble in the middle of the wall on the right for Re = 1000 in the early stage of flow development for the square driven cavity has been reproduced at about $t = 4.0$ using the nonlinear Crank–Nicolson method of Soh and Goodrich [18].

The time evolution of the driven cavity with aspect ratio 2 for Re = 1000 with a grid having 40 by 80 cells is presented in Figs. 7(a)–(j). This problem with the same grid resolution has previously been presented in Gustafson and Halasi [7]. Figures 7(a)–(j) show streamfunction contour plots overlayed with directional vectors from the velocity field. The deeper cavity gives rise to more interesting dynamics, and the streamfunction contours with directional vector overlays show the dramatic extent of the effects due to the secondary vortex that begins to develop along the downstream wall. Figure 7(a) shows a nearly symmetric flow at $t = 0.1$. Figure 7(b) shows the early strengthening of the primary vortex and the early distortion of its outer streamfunction contours along the downstream wall near the lid. Figure 7(c)–(d) show the presence of a recirculating vortex on the downstream wall with the development of a relatively weak positive streamfunction peak. From $t = 6.0$ to $t = 8.0$ in Fig. 7(e) there is the development of two new and distinct recirculating vortices in the lower corners. By $t = 10.0$ in Fig. 7(f) all three of the secondary recirculating vortices have joined together in one massive recirculating region along the boundary from the center of the downstream wall down to and along the entire lower wall. By $t = 12.0$ in Fig. 7(g) there is a dramatic breakout of this secondary vortex from the lower wall and deeply into the lower half of the cavity. By $t = 16.0$ in Fig. 7(h) this secondary vortex has developed to the point where it fills the lower half of the cavity. By $t = 28.0$ in Figs. 7(i) there are tertiary vortices developing in both of the lower corners, and the general pattern for the asymptotic flow field is present. From this point on the development of the flow field in the cavity merely stengthens the vortices that are present until the asymptotic result is reached by $t = 165.675$ in Fig. 7(j). These vortex dynamics are in agreement with those observed by Gustafson and Halasi [7].

The computational results that we have reported are in very good agreement with the published results for the problems that have been treated, within the

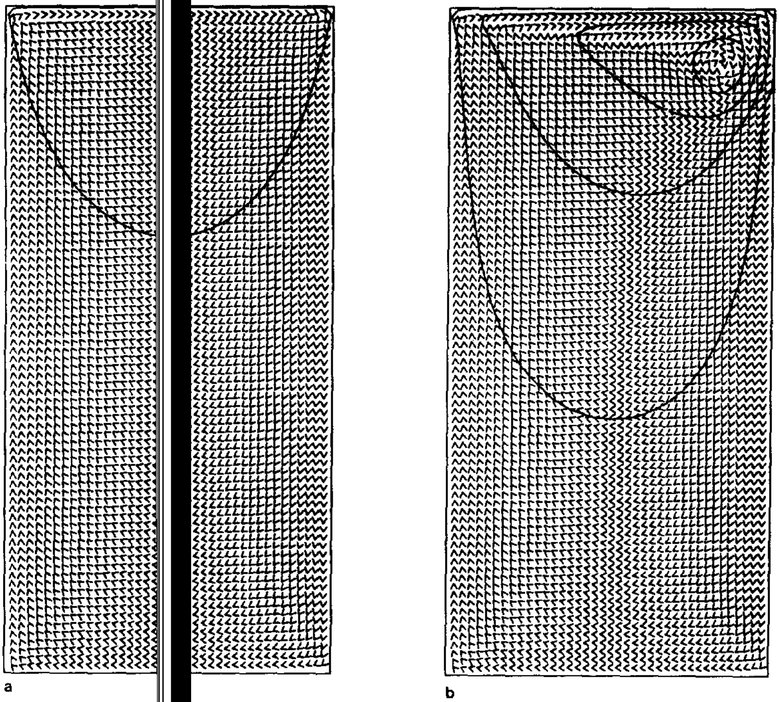


FIG. 7. Time evoluti vith aspect ratio 2, Re = 1000, 40 × 80 grid, streamfunction contours over-
layed with directional ve s: (a)–(j) $t = 0.1$, 2.0, 4.0, 6.0, 8.0, 10.0, 12.0, 16.0, 28.0, 165.675, respectively.

limitations of the g resolution that we have used. Accuracy in both space and
time has been den strated. Additional driven cavity calculations have been
performed with the e algorithm for Re from 100 to 100,000, using grids having
from 8 by 8 to 64 b 4 cells, with time step size approximately equal to the spatial
step size and indepe nt from the Reynolds number. The mesh Reynolds number
for these calculatio anges up to 5000. In all cases the resulting flow fields have
shown no spatial os tions at any time, even though central differencing has been
used for all linear d nonlinear terms in the equations. This computational
experience demonst s the robustness of the algorithm across a wide range of
Reynolds numbers. ne of the higher Reynolds number flows have been shown
because the 64 by 6 rid maximum is too coarse for reasonable spatial resolution
above Re = 3200. If accept the results at Re = 1000 or Re = 3200 with a 64 by
64 grid as having g spatial resolution, and if we scale the mesh by the boundary
layer thickness or 1/ e, then for Re = 50,000 we would need from 250 to 450 cells
along a side for ade te grid resolution. The computer code that we have used to
obtain the results p nted in this paper is based on algorithm (17) and (19) and
uses a LINPACK b ed LU decomposition subroutine to solve

$$\mathrm{La}(\mathbf{z}^{n+} \quad - \frac{\Delta t}{2\,\mathrm{Re}}\,\mathrm{Bi}(\mathbf{z}^{n+1}) = \text{source terms from } t_n \text{ and } t_{n-1}$$

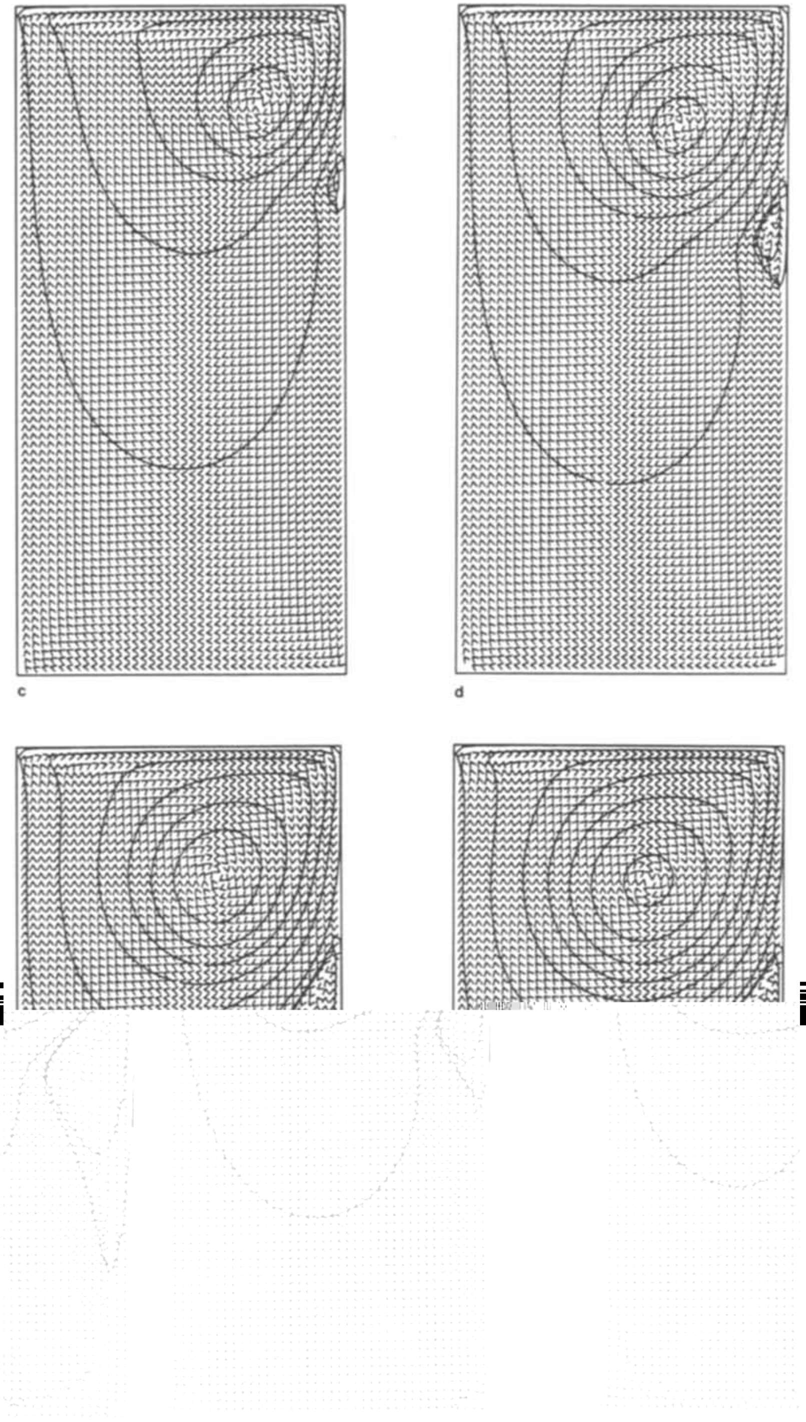


FIG. 7—*Continued*

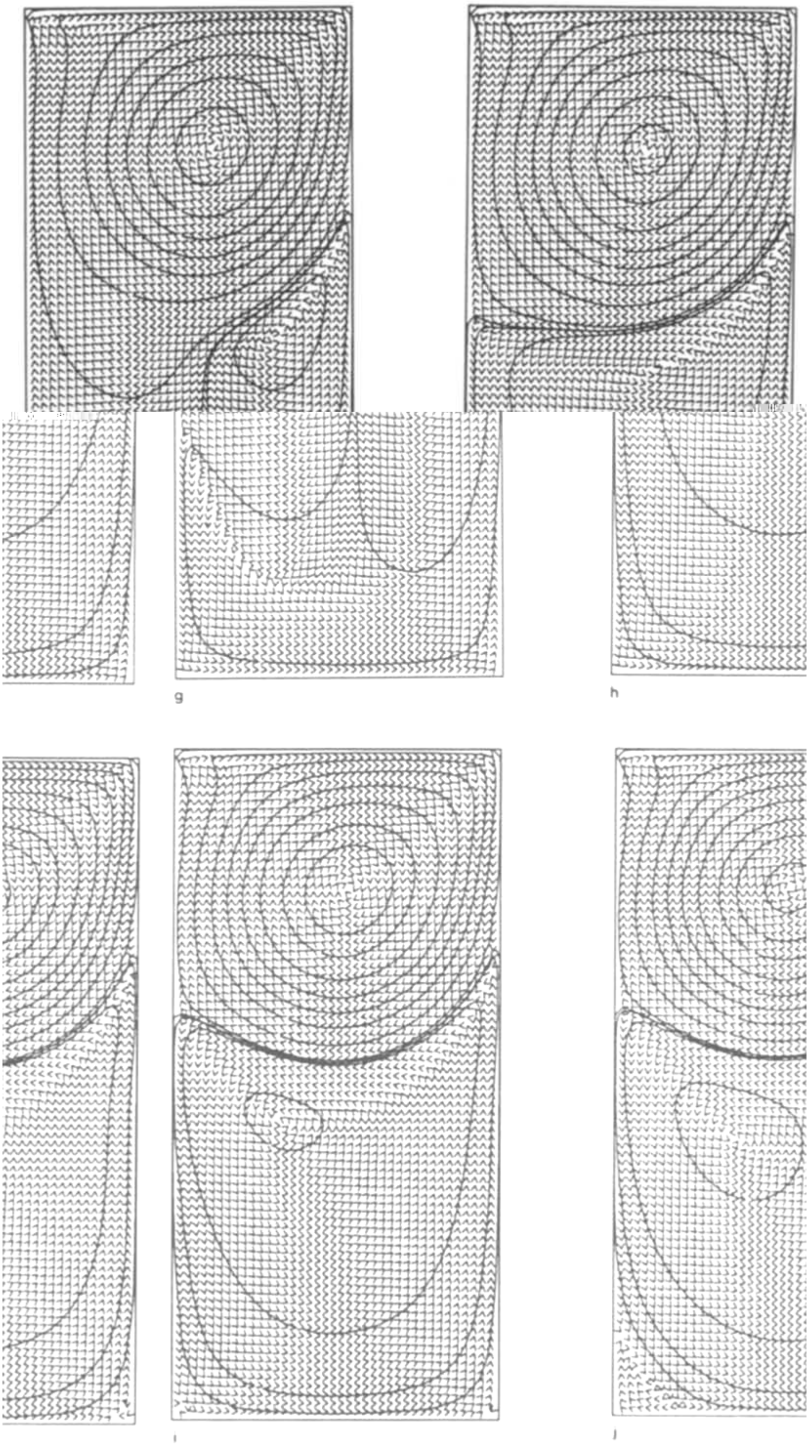


FIG. 7—*Continued*

by back substitution at each time step. This solver requires storage on the order of the cube of the number of grid cells along the cavity side. The memory constraint of the system that was used for the calculation of the reported numerical results limits the grid resolution to a 64 by 64 grid with this solver. The algorithm based on (17) and (19) is effective, but the ancillary solver needs to be improved in order to efficiently obtain fine mesh resolutions. A new solution procedure is being implemented with algorithm (17) and (19), but with a much more efficient solver for the associated linear equation problem at each time step. Results with higher resolution and at higher Reynolds numbers will be presented with the refined algorithm.

7. Summary

The FDG method was extended to time dependent incompressible Navier–Stokes equations with a continuous time formulation that is valid in two or three space dimensions. A detailed example of algorithm development was given for the driven cavity using a staggered grid and central differencing for the primitive variable scheme. It was shown how mass balance in the primitive variable formulation can be used to solve the essential problems associated with applying the FDG method for general problems in two dimensions. The use of the FDG method with this underlying discretization in two dimensions was shown to be the discrete analog of the continuum manipulations that lead to the fourth-order streamfunction equation. This method of algorithm development can use a staggered grid and mass balance for a primitive variable formulation, with the FDG method leading to a reduction in the number of variables, while the discrete streamfunction interpretation of the derived expansion variables gives a primitive variable solution that is discretely divergence free and that has all velocity components defined at the same point. Asymptotic and time evolution results obtained with a Crank–Nicolson Adams–Bashforth algorithm for $\mathrm{Re} = 400$, 1000, and 3200 were shown to be in good agreement with published data. The dramatic evolution of secondary vortices from bubble recirculations starting on the wall was shown for $\mathrm{Re} = 1000$. Computational experience was reported for Re up to 100,000 with mesh Reynolds number up to 5000 and without any spatial oscillations, even though central differencing was used for all space derivatives.

APPENDIX: The Nonlinear Streamfunction Terms

To see the detailed structure of the convection terms in (18) and (19), we will look more closely at $\mathbf{C}^{\mathrm{T}}\,\mathrm{cv}(\mathbf{C}\tilde{\mathbf{z}}^m)$. It will be convenient to reindex the components of $\mathbf{z}^m$ at time t_m by the cells with which they are associated in the natural mesh order as discussed above. Let $z^m_{i,j}$ be the $i+(j-1)(I-1)$ component of $\mathbf{z}^m$, for $1 \leqslant i \leqslant I-1$ and $1 \leqslant j \leqslant J-1$. If $h = 1/\Delta x = 1/\Delta y$, then with these conventions, we may write the convection terms at time t_m as

$$\frac{\Delta t}{2}\mathbf{C}^{\mathrm{T}}\,\mathrm{cv}(\mathbf{C}\mathbf{z}^m)$$

$$=\frac{\Delta t}{2h^4}\left(z^m_{i+1,j}z^m_{i+1,j+1}-z^m_{i+1,j+1}z^m_{i,j+1}+z^m_{i,j+1}z^m_{i-1,j+1}-z^m_{i-1,j+1}z^m_{i-1,j}\right.$$

$$\left.+z^m_{i-1,j}z^m_{i-1,j-1}-z^m_{i-1,j-1}z^m_{i,j-1}+z^m_{i,j-1}z^m_{i+1,j-1}-z^m_{i+1,j-1}z^m_{i+1,j}\right)$$

$$+\frac{\Delta t}{8h^4}\left(z^m_{i+1,j+1}z^m_{i,j+2}-z^m_{i,j+2}z^m_{i-1,j+1}+z^m_{i-1,j+1}z^m_{i-2,j}-z^m_{i-2,j}z^m_{i-1,j-1}\right.$$

$$\left.+z^m_{i-1,j-1}z^m_{i,j-2}-z^m_{i,j-2}z^m_{i+1,j-1}+z^m_{i+1,j-1}z^m_{i+2,j}-z^m_{i+2,j}z^m_{i+1,j+1}\right).$$

REFERENCES

[redacted]

3. A. J. CHORIN, *J. Math. Comput.* **22**, 745 (1968).
4. J. FROMM, *Methods Comput. Phys.* **3**, 345 (1964).
5. U. GHIA, K. N. GHIA, AND C. T. SHIN, *J. Comput. Phys.* **48**, 387 (1982).
6. K. GUSTAFSON AND K. HALASI, *J. Comput. Phys.* **64**, 279 (1986).
7. K. GUSTAFSON AND K. HALASI, *J. Comput. Phys.* **70**, 271 (1987).
8. C. A. HALL, *SIAM J. Algebraic Discrete Methods* **6**, 220 (1985).
9. F. H. HARLOW AND J. E. WELCH, *Phys. Fluids* **8**, 2182 (1965).
10. I. KANEKO, M. LAWO, AND G. THIERAUF, *Int. J. Num. Methods Eng.* **18**, 1469 (1982).
11. J. KIM AND P. MOIN, *J. Comput. Phys.* **59**, 308 (1985).
12. O. A. LADYZHENSKAYA, *The Mathematical Theory of Viscous Incompressible Flow* (Gordon & Breach, New York, 1963).
13. P. MOIN AND J. KIM, *J. Comput. Phys.* **35**, 381 (1980).
14. J. M. ORTEGA AND W. C. RHEINBOLDT, *Iterative Solution of Nonlinear Equations in Several Variables* (Academic Press, New York, 1970).
15. W. C. RHEINBOLDT, *Methods for Solving Systems of Nonlinear Equations* (SIAM, Philadephia, 1974).
16. P. J. ROACHE, *Computational Fluid Dynamics* (Hermosa, Albuqueque NM 1972).
17. P. J. ROACHE AND M. A. ELLIS, *Comput. Fluid* **3**, 305 (1975).
18. W. Y. SOH AND J. W. GOODRICH, *J. Comput. Phys.* **79**, 113 (1988).
19. A. B. STEPHENS, J. B. BELL, J. M. SOLOMON, AND L. B. HACKERMAN, *J. Comput. Phys.* **53**, 152 (1984).
20. R. TEMAM, *Navier–Stokes Equations* (North-Holland, Amsterdam, 1977).
21. J. E. WELCH, F. H. HARLOW, J. P. SHANNON, AND B. J. DALY, "The MAC Method," Los Alamos Scientific Laboratory Report No. LA-3425, 1965 (unpublished).
22. G. P. WILLIAMS, *J. Fluid Mech.* **37**, 727 (1969).

Note: The following references have been added at the suggestion of the reviewers.

23. R. AMIT, C. CULLEN, C. HALL, G. MESINA, AND T. PORSCHING, in *Proceedings, Third International Conference on Finite Elements in Flow Problems, Banff (Canada), June 1980.*
24. U. BULGARELLI, G. GRAZIANI, D. MANSUTTI, AND R. PIVA, in *Proceedings, Sixth GAMM Conference on Numerical Methods in Fluid Mechanics*, edited by D. Rues and W. Kordulla (Vieweg, Braunschweig, 1986), pp. 31–38.
25. J. Burkardt, C. Hall, and T. Porsching, *SIAM J. Algebraic Discreto Methods* **7**, 476 (1986).
26. C. HALL, J. PETERSON, T. PORSCHING, AND F. SLEDGE, *Int. J. Num. Meth. Eng.* **21**, 883 (1985).