# A Pseudospectral Matrix Element Method for Solution of Three-Dimensional Incompressible Flows and Its Parallel Implementation

HWAR C. KU, RICHARD S. HIRSH, THOMAS D. TAYLOR, AND ALLAN P. ROSENBERG*

*Johns Hopkins University Applied Physics Laboratory,*
*Johns Hopkins Road, Laurel, Maryland 20707*

Received March 11, 1988; revised October 3, 1988

A new pseudospectral matrix element (PSME) method employing the primitive variable formulation of the Navier–Stokes equations was used to simulate 3-dimensional time-dependent driven cavity flow at a Reynolds number of 3200 with an aspect ratio of 3 in the spanwise direction as well as 3-dimensional flow over a backward step. The new method only requires functions which are $c^0$ continuous across the interface between two adjacent elements. It also ensures that the continuity equation is satisfied everywhere, in the interior (including the inter-element points) and on the boundary. The resulting complex geometry for flow over a backward step can be divided into a number of overlapping subdomains by a domain decomposition, of simpler geometry with patched grid points, in which the solution is more easily obtained. With an iterative procedure between subdomains, the complete solution is found by the Schwarz alternating procedure (SAP). With an eigenfunction expansion for the pressure, storage requirements for the 3D inversion step, $O(N^6)$, are reduced to $O(N^3)$ if the inverse of pressure equation is not stored. The parallel implementation of the three most time-consuming procedures: (i) computing the partial derivatives of scalar fields in terms of dotproduct; (ii) transforming between eigenfunction space and physical space for the pressure in terms of matrix multiplications; and (iii) performing the forward and backward sweeps of an LU decomposition to solve for the pressure have been efficiently performed on a parallel computer. The numerical results have reproduced dynamic longitudinal Taylor–Görtler-like (TGL) vortices in qualitative agreement with the experimental results of Koseff *et al.* and also indicated other 3-dimensional effects on the flow development. Computational results for both 2-and 3-dimensional flow over a backward-facing step at different Reynolds numbers are also presented in this paper. No pronounced 3-dimensional effects are observed for Reynolds numbers up to 450 except in the boundary layer along the spanwise direction. © 1989 Academic Press, Inc.

## 1. INTRODUCTION

In the authors' earlier paper [1] a pseudospectral matrix (PSM) method for solution of the 3-dimensional incompressible Navier–Stokes equations was presented to solve a 3D driven cavity flow with Reynolds number of 100, 400, and

[*] Also at Sachs/Freeman Associates.

0021-9991/89 $3.00

1000. None of these simulations exhibited the phenomenon of TGL vortices, which requires a Reynolds number of at least 1200 according to the experimental results of Koseff *et al.* [2]. The interesting task of finding such vortices in 3-dimensional flow at a Reynolds number 3200 has been first successfully investigated by a finite difference modified Quick scheme by Freitas *et al.* [3]. Their numerical results at time 20 min are in good agreement with those found by the experiment, but they do not show the time development of the TGL vortices. Takemoto and Nakamura [4] used a third-order upwind difference scheme to well reproduce the TGL vortices in a cubic driven cavity flow but only at an early stage about 7 min. Here we approach the problem raised by the experiment by extending the techniques in our earlier paper. We will use the same aspect ratio as the experiment and show the time development of TGL vortices up to 20 min.

Flow over a backward step is always presented as the standard test problem for different numerical schemes. Some investigators [5, 6] assumed that the fully developed velocity profile is established right at the step and some considered the upstream effect before the step. Recently Armaly *et al.* [7] have performed a careful experimental study of such a flow for a wide range of Reynolds numbers. Although great care was taken to ensure that the initial inlet flow was 2-dimensional, in addition to the well-known entrance effects they also found that as the Reynolds number increased above 450, 3-dimensional effects became gradually apparent downstream from the step. These results motivated us to do numerical experiments on 3-dimensional flow over a backward step.

In addition to the strict time step size constraints, $O(1/N^4)$ for the viscous term and $O(1/N^2)$ for the convective term, the global PSM method (single element of our earlier paper [1]) is not appropriate for problems with stiff gradients in the interior of the domain because the well-known Gibbs' oscillations happen. In order to overcome these problems, a $c^0$ pseudospectral matrix element (PSME) method which combines the desired features of domain decomposition [8, 9] and a larger time step size has been developed to solve the incompressible Navier–Stokes equations in primitive variable form. The PSME method offers the same advantage over the $c^1$ method (flux continuity required) as the spectral element derived from a variational principle [10] in that it requires only the continuity of functions at the inter-element points; a continuous first derivative is implicitly imposed through integration by parts.

As pointed out by Kopriva [11] in the 1-dimensional inviscid case, with $N$ points and $K$ subdomains, the time step increases to $\Delta t \sim O(K/N^2)$ and the effort of matrix multiplication decreases to $O(N^2/K)$ rather than $O(N^2)$ as for a single element. What is more important, the sparsity of the resulting operators, nearly a 1D block tri-diagonal matrix in the pressure equation, tremendously speeds the LU factorization in contrast to the dense matrix structure resulting from the global method.

The more complex geometry of the backward-facing step has led us to a more elaborate domain decomposition technique than the multiple elements used to handle driven cavity flow. The solution of boundary-value problems for complex geometries has been successfully implemented by exchanging data among the different zones (or subdomains), i.e., solving the problem on each subdomain separately and then updating the boundary values on the overlapped interface. The data exchange can been done in a straightforward manner; this is what is usually called SAP [12]. Fuchs [13] pointed out that one major difficulty with the SAP when solving the incompressible Navier–Stokes equations in streamfunction-vorticity form for a rigid body confined in a flow region is that the value of the streamfunction is unknown on the surface of the rigid body. In primitive variables the question arises of how the boundary conditions for the pressure are handled on the iteratively solved subdomains even with the prescribed velocity components. Instead of directly imposing conditions on both the pressure and its first derivatives at the edge of each overlapped region [14], the consistent use of the continuity equation to generate the subdomain pressure boundary conditions (including the overlapped interfaces) should be preferable. This is mainly because the former definitely creates nondivergence-free velocity fields along the imaginary overlapped interfaces while the latter still guarantees differential mass conservation there.

In recent years computer designers, driven by the quest for speed and stimulated by the developing capability of single chips, have built machines of great complexity and increasing diversity. Producing efficient code for any of these machines is difficult and often involves special features, e.g., compiler directives and special hand-coded assembly language routines, that make migration of the code to a different machine difficult. The algorithms used in this paper can be cleanly described using the rather minimal set of vector and parallel concepts proposed for Fortran 8x. Specifically, we use the ability to do dotproducts and matrix multiplication between subsets of arrays as well as the ability to specify that it is safe to run some loops containing subroutines concurrently. The Alliant FX/Fortran compiler does a tolerable job of optimizing our program, and we expect that compilers for machines with quite different architectures will be able to do so as well.

For the purpose of high-speed computation, the issue of how to map a numerical algorithm for the solution of each subdomain onto the specific architecture of parallel machines so as to achieve highly parallel performance remains an open question. There are roughly two approaches to this problem: (1) distribute processors proportional to the amount of cpu time required per subdomain or (2) allocate all the processors to a single subdomain and do different subdomains in sequence. On massively parallel systems without shared memory the first approach is more natural, but on shared memory systems with relatively few processors both are feasible. In this case, one problem with the first approach is the requirement of a compiler smart enough to anticipate the cpu time difference among subdomains; otherwise some processors with their jobs done earlier have to sit idle until other processors finish their jobs. On the contrary, there is no data communication constraint for the second one and it becomes more attractive if the computation of each subdomain can be highly parallelizeable. In this paper, the authors would like to use this concept to apply the PSME method to the solution of incompressible flows in complex geometries.

The paper has eight sections: the first being the Introduction. Section 2 provides general information on the calculation of derivatives by the PSME method and comparisons between the $c^0$ and $c^1$ methods for a standard convection-diffusion test case. Section 3 describes the physical problems and appropriate boundary conditions and Section 4 formulates the Navier–Stokes equations in primitive variable form by the PSME method and outlines the solution techniques. In Section 5 the Schwarz alternating procedure used in flow over a backward step is explained. Section 6 gives some details on parallel implementation of Navier–Stokes equations and provides timings on the Alliant FX/8 series of multi-processors. Section 7 presents the numerical results on dynamic longitudinal TGL vortices as well as flow over a backward step and the last section provides our conclusions.

## 2. Calculation of Derivatives by PSME Method

The spatial domain is divided into $NE$ elements, each of which has $N+1$ collocations which are scaled translations of the $N+1$ standard collocations $x_j = \cos \pi(j-1)/N$ $(1 \leqslant j \leqslant N+1)$ in the interval $[-1, 1]$. Explicitly the points for the $e$th element are $x_j = \frac{1}{2}[a^e + b^e + (b^e - a^e) \cos \pi(j-1)/N]$ $(1 \leqslant j \leqslant N+1;$ $1 \leqslant e \leqslant NE)$. The derivatives of a function $f(x)$ in the interior of element $e$ can be discretized as

$$f^e_x(x_i) = \frac{1}{L^e} \sum_{m=1}^{N+1} \widehat{GX}^{(1)}_{i,m} f^e_m \tag{1a}$$

$$f^e_{xx}(x_i) = \frac{1}{(L^e)^2} \sum_{m=1}^{N+1} \widehat{GX}^{(2)}_{i,m} f^e_m = \frac{1}{L^e} \sum_{m=1}^{N+1} \widehat{GX}^{(1)}_{i,m} (f^e_x)_m. \tag{1b}$$

Here $L^e$ is the length of the $e$th element defined on the interval $[a^e, b^e]$, $f^e_m \equiv f^e(x_m)$, and $\widehat{\mathbf{GX}}^{(1)}$, $\widehat{\mathbf{GX}}^{(2)}$ are the invariant derivative matrices based on the domain $[0, 1]$ with $\widehat{\mathbf{GX}}^{(q)} = 2^q \mathbf{T} \mathbf{R}^{(q)} \hat{\mathbf{T}}$, $q = 1, 2$, and $\mathbf{R}^{(2)} = \mathbf{R}^{(1)} \mathbf{R}^{(1)}$, where $\mathbf{T}$, $\mathbf{R}^{(1)}$, $\hat{\mathbf{T}}$ are $(N+1) \times (N+1)$ matrices with elements

$$T_{i,j} = \cos \frac{\pi(i-1)(j-1)}{N} \tag{2a}$$

$$R^{(1)}_{i,j} = \begin{cases} 0, & i \geqslant j \text{ or } i+j \text{ is even} \\ 2(j-1)/C_i, & \text{otherwise } (C_1 = 2,\ C_i = 1 \text{ for } i \geqslant 2) \end{cases} \tag{2b}$$

and

$$\hat{T}_{i,j} = \frac{2}{N \bar{C}_i \bar{C}_j} \cos \frac{\pi(i-1)(j-1)}{N}$$
$$(\bar{C}_1 = \bar{C}_{N+1} = 2,\ \bar{C}_i = 1 \text{ for } 2 \leqslant i \leqslant N). \tag{2c}$$

The interfacial derivatives at the inter-element points, are approximated by weighting of the derivatives from each side, according to the relations

$$f_x\,|_{\text{interface}}=\alpha f^{e}_{x}+\beta f^{e+1}_{x} \tag{3a}$$

$$f_{xx}\,|_{\text{interface}}=\alpha f^{e}_{xx}+\beta f^{e+1}_{xx}, \qquad \alpha+\beta=1,\ 1\leqslant e\leqslant NE-1. \tag{3b}$$

Choosing $\alpha$ and $\beta$ to be their respective fractions of the total length of two adjacent elements, Eqs. (3), in view of Eqs. (1), now become

$$f_x\,|_{\text{interface}}=\frac{1}{L^e+L^{e+1}}\sum_{m=1}^{N+1}\left(\widehat{GX}^{(1)}_{N+1,m}f^{e}_{m}+\widehat{GX}^{(1)}_{1,m}f^{e+1}_{m}\right) \tag{4a}$$

$$f_{xx\ \text{interface}}=\frac{1}{L^e+L^{e+1}}\sum_{m=1}^{N+1}\left(\widehat{GX}^{(1)}_{N+1,m}(f^{e}_{x})_m+\widehat{GX}^{(1)}_{1,m}(f^{e+1}_{x})_m\right). \tag{4b}$$

In Eq. (4a), $c^0$ continuity is explicitly assumed whenever the calculation of interface values of the first derivative is required. However, $c^0$ continuity is only implicitly assumed for the second derivative calculation. We can expand Eq. (4b) using Eq. (1b) and impose $c^1$ continuity to get

$$f_{xx}\,|_{\text{interface}}=\frac{1}{L^e+L^{e+1}}\left[\sum_{m=1}^{N+1}\left(BX^{*}_{N+1,m}f^{e}_{m}+BX^{**}_{1,m}f^{e+1}_{m}\right)\right.$$
$$\left.+\left(\widehat{GX}^{(1)}_{N+1,N+1}+\widehat{GX}^{(1)}_{1,1}\right)f_x\,|_{\text{interface}}\right], \tag{5}$$

where it has been implicitly assumed that $f^{e}_{x}(x_{N+1})=f^{e+1}_{x}(x_1)=f_x\,|_{\text{interface}}$. As will be shown shortly, the explicit value of $f_x\,|_{\text{interface}}$ is never needed. This requirement is met by the finite element method employing variational or Galerkin procedures with trial functions that are $c^0$ across element boundaries, i.e., flux (first derivative) continuity is intrinsically satisfied through integration by parts. This is exactly the same for the PSME method because the second term in the bracket of Eq. (5) is automatically cancelled out since, $\widehat{GX}^{(1)}_{1,1}=(2N^2+1)/3=-\widehat{GX}^{(1)}_{N+1,N+1}$. The elements of the modified matrices **BX***, **BX**** are

$$BX^{*}_{N+1,m}=\frac{1}{L^e}\sum_{n=1}^{N}\widehat{GX}^{(1)}_{N+1,n}\widehat{GX}^{(1)}_{n,m} \tag{6a}$$

$$BX^{**}_{1,m}=\frac{1}{L^{e+1}}\sum_{n=2}^{N+1}\widehat{GX}^{(1)}_{1,n}\widehat{GX}^{(1)}_{n,m}. \tag{6b}$$

The proposed PSME method together with a $c^1$ [15, 16] method has been tested on a standard 1-dimension convection-diffusion problem

$$\frac{d^2f}{dx^2}-\text{Pe}\,\frac{df}{dx}=0 \tag{7}$$

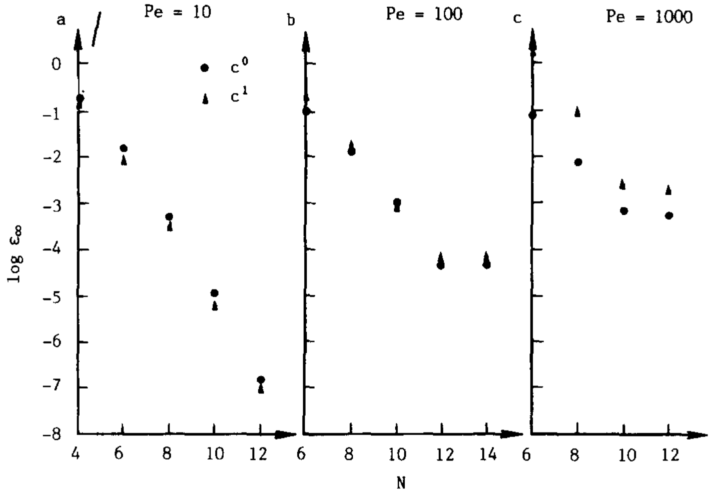


FIG. 1. A plot of the error for two element approximation at the inter-element position (a) $x=0.5$, (b) $x=0.9$, and (c) $x=0.99$. Here $\varepsilon_\infty = \|f_{\text{numerical}} - f_{\text{exact}}\|_\infty$, and $N$ is the number of points per element.

with boundary conditions

$$f(0)=1 \tag{8a}$$

$$f(1)=0 \tag{8a}$$

which has the exact solution $f(x) = (\exp \mathrm{Pe} - \exp \mathrm{Pe}\, x)/(\exp \mathrm{Pe} - 1)$. With the inter-element position at $x=0.5$ for $\mathrm{Pe}=10$, the two element approximation for both the $c^0$ and $c^1$ methods exhibit exponential convergence as the number of points per element is increased as indicated in Fig. 1a. However, at $\mathrm{Pe}=100$ and 1000 with the inter-element position at $x=0.9$ and 0.99, Figs. 1b and 1c show that the $c^0$ method is superior to the $c^1$ method in resolving the steep change of function, typical of high Pe number, because the $c^1$ method yields the well-known Gibbs' oscillations.

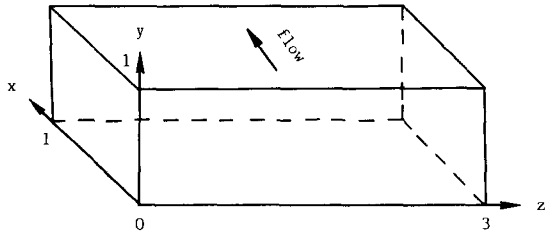


FIG. 2. Three-dimensional cavity flow configuration and coordinate system.

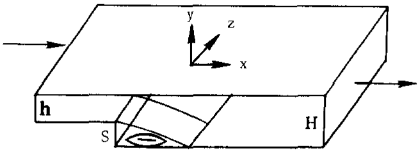


FIG. 3. Three-dimensional configuration for flow over a backward step.

## 3. GOVERNING EQUATIONS

The 3-dimensional cavity flow in a rectangular box (shown in Fig. 2) with an aspect ratio of 3 in the spanwise direction as well as 3-dimensional flow over a backward step (shown in Fig. 3) with an expansion ratio 2 in the vertical direction and an aspect ratio of 8 in the spanwise direction can be represented in a primitive variable formulation. In cartesian coordinates, the time-dependent Navier–Stokes equations in dimensionless form can be written as

$$\frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial x_j} = -\frac{\partial p}{\partial x_i} + \frac{1}{\text{Re}} \frac{\partial^2 u_i}{\partial x_j^2} \tag{9a}$$

$$\frac{\partial u_i}{\partial x_i} = 0. \tag{9b}$$

Here $u$, $v$, $w$ are the velocity components in the horizontal (flow), vertical, and spanwise directions, respectively, and Re is the Reynolds number based on the horizontal length of the rectangular box and top moving plate velocity in driven cavity flow, while in flow over a backward step the Ryenolds number is defined by the maximum velocity of inflow and the step height.

### 3.1. *Driven Cavity Flow*

Equations (9) are only solved for half of the domain due to the experimental results [2] showing symmetry about the $xy$ plane at $z = 1.5$, i.e., $u$, $v$, $p$ are symmetric ($\partial u/\partial z = 0$, $\partial v/\partial z = 0$, $\partial p/\partial z = 0$) and $w$ is anti-symmetric ($w = 0$) with respect to the central plane. Initially all velocity components are zero; at time $t > 0$, the top plate suddenly starts translating with a uniform velocity.

### 3.2. *Flow over a Backward Step*

Similarly, only half of the domain needs to be solved according to the experimental results [7] of symmetry about the $xy$ plane at $z = 0$. The boundary conditions are given by

$$v = w = 0, \qquad u = 4(y - y^2) \qquad \text{at } x = -2 \tag{10a}$$

$$\frac{\partial^2 u}{\partial x^2} = \frac{\partial^2 v}{\partial x^2} = \frac{\partial^2 w}{\partial x^2} = 0 \qquad \text{at } x = l \tag{10b}$$

$$u = v = w = 0 \qquad \text{at } y = 1 \tag{10c}$$

$$u = v = w = 0 \qquad \text{at } y = 0 \text{ for } x \leqslant 0 \tag{10d}$$

$$u = v = w = 0 \qquad \text{at } y = -1 \text{ for } x \geqslant 0 \tag{10e}$$

$$u = v = w = 0 \qquad \text{at } z = 4 \tag{10f}$$

$$\frac{\partial u}{\partial z} = \frac{\partial v}{\partial z} = \frac{\partial p}{\partial z} = 0, \qquad w = 0, \qquad \text{at } z = 0. \tag{10g}$$

Appropriate outflow boundary conditions are required when calculations are performed in open domains without making a fully developed flow profile at far downstream, and these conditions should have little influence on the upstream flow development. This is usually the case when the upstream continuously generates the disturbance which will be propagated into the downstream.

Note that different downstream boundary conditions are possible, for instance, in the 2-dimensional case.

1. $u$, $v$ both prescribed
2. $\partial u/\partial x = 0$, $v = 0$
3. $\partial v/\partial x = 0$, $\partial^2 u/\partial x^2 = 0$
4. $\partial^2 u/\partial x^2 = 0$, $\partial^2 v/\partial x^2 = 0$.

Conditions (1) and (2) seem too restrictive to be applied on the truncated domain, while conditions (3) and (4) yield the least effect upon the upstream flow development. Conditions (1) and (2) are the only appropriate downstream boundary conditions for fully developed flow at infinity or far away from the step.

## 4. Primitive Variable Formulation

The method applied to solve the Navier–Stokes equation is Chorin's [17] splitting technique. According to this scheme, the equations of motion, in tensor form, are

$$\frac{\partial u_i}{\partial t} + \frac{\partial p}{\partial x_i} = F_i, \tag{11}$$

where $F_i = -u_j\, \partial u_i/\partial x_j + 1/\mathrm{Re}\ \partial^2 u_i/\partial x_j^2$.

The first step is to split the velocity into a sum of predicted and corrected values. The predicted velocity is determined by time integration of the momentum equations without the pressure term

$$\bar{u}_i^{n+1} = u_i^n + \Delta t F_i^n. \tag{12}$$

The second step is to develop the pressure and corrected velocity fields that satisfy the continuity equation by using the relationships

$$u_i^{n+1} = \bar{u}_i^{n+1} - \Delta t \frac{\partial p}{\partial x_i} \tag{13a}$$

$$\frac{\partial u_i^{n+1}}{\partial x_i} = 0. \tag{13b}$$

The explicit scheme just outlined is spectrally accurate in space but only first-order accurate in time. It can be modified to second-order accuracy in time by treating the viscous term by a Crank–Nicholson scheme and the convective term by an Adams–Bashforth scheme as described in [8]. Thus, Eqs. (11)–(13) are rewritten as

$$\frac{\partial u_i}{\partial t} + \frac{\partial p}{\partial x_i} = F_i + \frac{1}{\text{Re}} \frac{\partial^2 u_i}{\partial x_j^2}, \tag{14}$$

where $F_i = -u_j \, \partial u_i / \partial x_j$. 1st step:

$$\bar{u}_i^{n+1} = u_i^n + \Delta t \left( \frac{3}{2} F_i^n - \frac{1}{2} F_i^{n-1} + \frac{1}{2\text{Re}} \frac{\partial^2 u_i^n}{\partial x_j^2} \right); \tag{15}$$

2nd step:

$$u_i^{n+1,k} = \bar{u}_i^{n+1} - \Delta t \frac{\partial p^k}{\partial x_i} + \frac{\Delta t}{2\,\text{Re}} \frac{\partial^2 u_i^{n+1,k}}{\partial x_j^2} \tag{16a}$$

$$u_i^{n+1,k+1} = u_i^{n+1,k} - \Delta t \frac{\partial p'}{\partial x_i} \tag{16b}$$

$$\frac{\partial u_i^{n+1,k+1}}{\partial x_i} = 0; \tag{16c}$$

where $p^{k+1} = p^k + p'$ and $k$ is the iteration number. This approach is effective at low Reynolds numbers because the gain in time step size more than offsets the costs of iterating the coupled momentum and pressure equations.

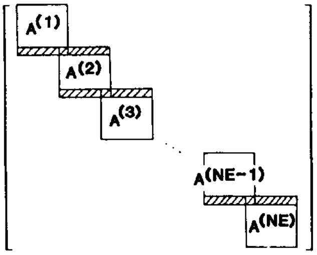
For the current paper, we have used both methods to simulate the TGL vortices at the Reynolds number Re = 3200. It was found that generally 3–4 iteration counts were required for Stokes solution, but the time step size only increases by approximately a factor of 2, so the entire rest of the paper will be focused on the simple method and its implementation on a parallel computer. This also applies to our discussion of 3-dimensional flow over a backward step.

In order to set up a simple and efficient matrix operation for derivatives by the PSME method, a differentiation operator with global-type structure can be constructed which combines each local element derivative. To do this, Eqs. (1a) and (4a), representing the first derivatives, are cast into the form

$$\mathbf{f}' = \mathbf{G}^{(1)}\mathbf{f}, \tag{17}$$

where $\mathbf{G}^{(1)}$ has the diagonal form

$$\mathbf{G}^{(1)} = \tag{18}$$



The hatched area in Eq. (18) arises from the use of Eq. (4a) at the element–element interface while the non-overlapped area is simply Eq. (1a) in the interior of each element. The blocks $\mathbf{A}^{(n)}$, $n = 1, ..., NE$ are of size $(N+1)^2$ with a one point overlap at the corners.

In an analogous manner, the second derivative in Eqs. (1b) and (4b) can be written in the form

$$\mathbf{f}'' = \mathbf{G}^{(2)}\mathbf{f}, \tag{19}$$

where $\mathbf{G}^{(2)}$ has the same diagonal form as that of $\mathbf{G}^{(1)}$.

Using the subscripts $i$, $j$, $k$ for the $x$, $y$, and $z$ directions, the discretization of Eq. (13a) takes the form

$$u^{n+1}_{i,j,k} = \bar{u}^{n+1}_{i,j,k} - \Delta t \sum_{m=m_1(i)}^{m_2(i)} GX^{(1)}_{i,m}\, p_{m,j,k} \tag{20a}$$

$$v^{n+1}_{i,j,k} = \bar{v}^{n+1}_{i,j,k} - \Delta t \sum_{l=l_1(j)}^{l_2(j)} GY^{(1)}_{j,l}\, p_{i,l,k} \tag{20b}$$

$$w^{n+1}_{i,j,k} = \bar{w}^{n+1}_{i,j,k} - \Delta t \sum_{n=n_1(k)}^{n_2(k)} GZ^{(1)}_{k,n}\, p_{i,j,n}. \tag{20c}$$

Here $m_1(i)$, $l_1(j)$, $n_1(k)$ are the column indices for the first nonzero elements in rows $i$, $j$, $k$ of their respective partial derivative matrices $\mathbf{GX}^{(1)}$, $\mathbf{GY}^{(1)}$, and $\mathbf{GZ}^{(1)}$, while $m_2(i)$, $l_2(j)$, $n_2(k)$ are the column indices of the last nonzero elements in those rows.

Taking the divergence of Eq. (13a), applying the continuity equation (13b) throughout the whole domain, and incorporating prescribed velocity boundary conditions generates a pressure Poisson equation in the interior and supplemental equations at the boundaries. In the interior,

$$\sum_{m=m_1(i)}^{m_2(i)} BX_{i,m}\, p_{m,j,k} + \sum_{l=l_1(j)}^{l_2(j)} BY_{j,l}\, p_{i,l,k} + \sum_{n=n_1(k)}^{n_2(k)} BZ_{k,n}\, p_{i,j,n} = S_{i,j,k} \tag{21}$$

for $2 \leqslant i \leqslant NX$, $2 \leqslant j \leqslant NY$, $2 \leqslant k \leqslant NZ$, where $NX+1$, $NY+1$, $NZ+1$ are the overall number of points in the $x$, $y$, $z$ directions, respectively. $S_{i,j,k}$ is the source term corresponding to the divergence of the predicted velocity from Eq. (13a). The supplemental pressure equations at the boundaries have the form

$$\sum_{m=m_1(i)}^{m_2(i)} BX_{i,m} p_{m,j,k} = S_{i,j,k}, \qquad i = 1,\ NX+1 \tag{22a}$$

$$\sum_{l=l_1(j)}^{l_2(j)} BY_{j,l} p_{i,l,k} = S_{i,j,k}, \qquad j = 1,\ NY+1 \tag{23b}$$

$$\sum_{n=n_1(k)}^{n_2(k)} BZ_{k,n} p_{i,j,n} = S_{i,j,k}, \qquad k = 1,\ NZ+1. \tag{22c}$$

The coefficients for the matrices **BX**, **BY**, **BZ** are

$$BX_{i,m} = \sum_{p=2}^{NX} GX^{(1)}_{i,p} GX^{(1)}_{p,m} \tag{23a}$$

$$BY_{j,l} = \sum_{q=2}^{NY} GY^{(1)}_{j,q} GY^{(1)}_{q,l} \tag{23b}$$

$$BZ_{k,n} = \sum_{r=2}^{NZ} GZ^{(1)}_{k,r} GZ^{(1)}_{r,n}. \tag{23c}$$

Equations (20)–(23) are the general formulas for the 3-dimensional pressure Poisson equations. Note that when special boundary conditions apply, the derivative matrices $\mathbf{GX}^{(1)}$, $\mathbf{GY}^{(1)}$, $\mathbf{GZ}^{(1)}$ can be modified in compliance with the (i) symmetry (the first derivative of the function is zero); (ii) antisymmetry (the function and its second derivative are zero), and (iii) inflection (the second derivative of the function equals zero and changes sign at the inflection point) boundary conditions. The range of the index associated with each derivative matrix and its coefficients will be changed accordingly.

Following an eigenfunction expansion technique (described in detail in Ref. [1]), we first diagonalize the reduced derivative operators **BX***, **BZ***, which differ from the original operators **BX**, **BZ** by having absorbed the unknown pressure boundary terms expressed in terms of interior pressure field variables. This is done so they have an expansion in real eigenvalues

$$\mathbf{EX}^{-1}\mathbf{BX}^{*}\mathbf{EX} = \mathbf{\Lambda} \tag{24a}$$

$$\mathbf{EZ}^{-1}\mathbf{BZ}^{*}\mathbf{EZ} = \chi \tag{24b}$$

where $\mathbf{\Lambda}$ is a diagonal matrix of dimension $NX-1$ with diagonal elements $\alpha_i$ which are the eigenvalues of the matrix $\mathbf{BX}^*$ and $\chi$ is a diagonal matrix of dimension $NZ-1$ with diagonal elements $\beta_k$ which are the eigenvalues of the matrix $\mathbf{BZ}^*$. The column vectors of the matrices **EX**, **EZ** are the corresponding eigenvectors associated with each eigenvalue.

If the pressure is expanded in a series of the eigenfunctions such that

$$\mathbf{P} = \mathbf{EX}\ \hat{\mathbf{P}}\ \mathbf{EZ}^{\mathrm{T}} \tag{25a}$$

and the source term similarly expanded such that

$$\mathbf{S}^* = \mathbf{EX}\ \hat{\mathbf{S}}^*\ \mathbf{EZ}^{\mathrm{T}}, \tag{25b}$$

where the source term $\mathbf{S}^*$ is the original **S** combined with known boundary source terms to form the operators $\mathbf{BX}^*$, $\mathbf{BZ}^*$, then the original 3-dimensional pressure equation is reduced to a simple 1-dimensional equation for each $i = 2, ..., NX$ and $k = 2, ..., NZ$,

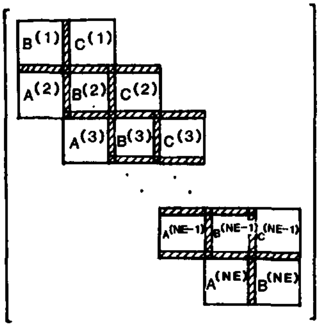
$$\sum_{l=l_1(j)}^{l_2(j)} BY_{j,l}\,\hat{p}_{i,l,k} + (\alpha_i + \beta_k)\,\hat{p}_{i,j,k} = \hat{S}^*_{i,j,k}, \qquad 2 \leqslant j \leqslant NY \tag{26a}$$

and

$$\sum_{l=l_1(j)}^{l_2(j)} BY_{j,l}\,\hat{p}_{i,l,k} = \hat{S}^*_{i,j,k}, \qquad j = 1,\ NY+1. \tag{26b}$$

The overall solution for the pressure can be obtained through the linear superposition of each eigenvalue and its associated eigenvectors. The matrix **BY** is nearly a block tri-diagonal matrix with the partitioned form

**BY** =



(27)

where $\mathbf{A}^{(n)}$, $\mathbf{B}^{(n)}$, $\mathbf{C}^{(n)}$, $n = 1, ..., NE$, are $(N+1)^2$ matrices with a single point inter-overlapped shaded region and the order of the square matrix **BY** equals $NE \times N + 1$ (i.e., $NY + 1$).

Solving the bånd matrix using the eigenfunction expansion technique on a parallel computer will be dealt with in Section 6.

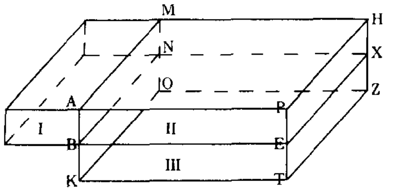


FIG. 4. Three-dimensional configuration of domain decomposition.

## 5. Schwarz Alternating Procedure

The SAP for iterative solution of the incompressible Navier–Stokes equations in primitive variable form for the 3-dimensional flow over a backward step is summarized as follows (see Fig. 4):

1. First assume $u^{n+1}$, $v^{n+1}$, $w^{n+1}$ on $\square ABMN$. Usually $u^n$, $v^n$, $w^n$ would be a good initial guess.

2. Solve domain $\mathrm{II} \cup \mathrm{III}$ employing boundary conditions derived from the continuity of velocity field on $\square ABMN$, i.e., $\partial u/\partial x = -\partial v/\partial y - \partial w/\partial z$ where the pressure Poisson equation is solved by an eigenfunction expansion technique.

3. With the solutions of $u^{n+1}$, $v^{n+1}$, $w^{n+1}$ on $\square NBEX$ from step (2), solve domain $\mathrm{I} \cup \mathrm{II}$ employing the same type boundary conditions as above for the pressure on $\square NBEX$, i.e., $\partial v/\partial y = -\partial u/\partial x - \partial w/\partial z$, to update $u^{n+1}$, $v^{n+1}$, $w^{n+1}$ on $\square ABMN$.

4. Repeat steps (2) and (3) until the converge tolerance has been met for $u^{n+1}$, $v^{n+1}$, $w^{n+1}$ along $\square ABMN$, $\square NBEX$.

The novel features of this iteration scheme are : (i) for each iteration the divergence-free condition is exactly satisfied everywhere except at the boundary points in the direction where the inflow carries singular information; (ii) for every time step only a few iterations (usually 2–4) are required to reach the converged solution for velocity components $u$, $v$, $w$; (iii) consistent mass conservation holds along $\overline{BN}$ despite a singular value for the vorticity; and (iv) within each subdomain the eigenfunction expansion technique is still available to decompose the original 3-dimensional problem into a simple 1-dimensional matrix operator.

## 6. Parallel Implementation of N-S Equations

It is easy to use parallelism to solve problems which can be decomposed into completely independent subproblems, but becomes progressively more difficult to do efficiently as dependencies between the subproblems increase. In this section we will reduce the ideas of the previous sections to parallelizable Fortran code. We assume the compiler can detect and the target machine exploit the inherent parallelism of an outer loop containing an inner loop with data dependencies which allow the outer loop to be distributed across processors. The Alliant FX series will then attempt to vectorize the inner loop, and we have tried to write our inner loops so this is possible. It is not, of course, necessary for exploiting the outer loop parallelism. We make use of two intrinsic functions from the proposed Fortran 8x standard—*matmul* and *dotproduct*. This is not necessary, but it simplifies the programming significantly in some places.

In our version of the time splitting approach three steps account for most of the run time. These are computing the partial derivatives involved in updating the predicted velocity, transforming back and forth from physical to eigenfunction space, and solving the sequence of reduced 1-dimensional pressure equations in eigenfunction space. Each will be discussed separately below.

### 6.1. *Predicted Velocity: Vector Dot*

Updating the predicted velocity according to Eq. (12) requires that the first and second partial derivatives be computed in each of three directions for each component of the velocity at each point in space. Letting $i$, $j$, $k$ be the indices of the spatial coordinate, $\phi$ be any of the velocity components, and recalling the structure of the derivative matrices specified in Eqs. (17), (18), and (19), we can rewrite Eq. (12) as

for each $i=2, ..., NX$, $j=2, ..., NY$, $k=2, ..., NZ$,

$$\begin{aligned}\tilde{\phi}^{n+1}_{i,j,k} &= \phi^n_{i,j,k}\\ &\quad + \Delta t\left[\text{dotproduct}\left(\frac{1}{\text{Re}}GX^{(2)}_{i,m_1(i):m_2(i)} - u^n_{i,j,k}GX^{(1)}_{i,m_1(i):m_2(i)}, \phi^n_{m_1(i):m_2(i),j,k}\right)\right.\\ &\quad + \text{dotproduct}\left(\frac{1}{\text{Re}}GY^{(2)}_{j,l_1(j):l_2(j)} - v^n_{i,j,k}GY^{(1)}_{j,l_1(j):l_2(j)}, \phi^n_{i,l_1(j):l_2(j),k}\right)\\ &\quad \left. + \text{dotproduct}\left(\frac{1}{\text{Re}}GZ^{(2)}_{k,n_1(k):n_2(k)} - w^n_{i,j,k}GZ^{(1)}_{k,n_1(k):n_2(k)}, \phi^n_{i,j,n_1(k):n_2(k)}\right)\right].\end{aligned}$$

Here the three dotproduct terms correspond to the three spatial directions. Each loop instantation of a dotproduct involves one row of a matrix having the structure shown in Eq. (18) and an expression such as $m_1(i): m_2(i)$ specifies the index range for the dotproduct. The matrices are sufficiently sparse that it is worthwhile to perform the dotproduct only on the nonzero elements. Since dotproducts are linear in each factor, we use one dotproduct to compute both the first and second partials. Changing subscripts to arguments, the "for each" statements to do loops, and using "*" to denote multiplication converts the above expression to a syntactically correct Fortran 8x fragment.

Although computing partial derivatives is not the most time-consuming part of the calculation, it is the hardest to do efficiently. The fundamental difficulty is that computing derivatives involves operations which are local in the topology of 3-dimensional space, but will not be local in either the Fortran representation of a

3-dimensional array or its realization in machine memory. The problem is compounded on machines, such as the Alliant, which have a memory cache. One way to ameliorate the problem is to partition a 3-dimensional array, such as a velocity component field, into smaller but still 3-dimensional blocks each of which will fit into a cache. Operations on such arrays are then performed block by block. Recognizing when it is efficient to do this and generating appropriate code clearly pose serious challenges for a compiler. A discussion of how to block some basic linear algebra using as primitive the operation $\mathbf{C} \rightarrow \mathbf{C} + \mathbf{AB}$, where $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ are 2-dimensional arrays, can be found in [18]. Some of these simpler cases have been implemented in the Alliant compiler. It does a tolerable but far from perfect job on our derivative computation problem. We could probably improve its performance by hand-blocking our arrays at a cost of increased programming effort and reduced portability. It would be interesting to see what the Fortran compilers for machines with a great number of processors could do with this problem.

### 6.2. *Tensor Product: Matrix Multiply*

Converting the pressure and source term from eigenfunction space back to physical space in Eqs. (25a) and (25b) requires a series of matrix multiplications. This is also true for the reverse conversions from physical to eigenfunction space. To implement this we use the Fortran 8x intrinsic function *matmul*. For example, Eq. (25a) becomes, for each $j = 1, ..., NY + 1$,

$$p_{:,j,:} = \text{matmul}[\mathbf{EX}, \text{matmul}(\hat{p}_{:,j,:}, \mathbf{EZ}^{\mathrm{T}})],$$

where ":" indicates the full range including the unnecessary indices for boundary pressure terms. This is only for convenience, and the specific index range would be permitted.

### 6.3. *Pressure Equations: LU Decomposition*

Using Chorin's splitting scheme, our techniques lead to a separable problem for the finite dimensional analogue of the Poisson equation for the pressure. With an eigenfunction expansion in two directions, the resulting sequence of 1-dimensional problems is solved by LU factorization. This reduces our storage requirements for the inverse from $O(N^6)$ to $O(N^4)$. If the inverse is not stored, storage requirements for the inversion step are reduced to $O(N^2)$. By paying careful attention to the nearly block tri-diagonal structure of the reduced 1-dimensional operator, Eq. (26a), where the eigenvalues $\alpha_i$, $\beta_k$ appeared on the diagonal line with the exclusion of two boundary points $y = 0$ and 1, an efficient implementation of both the forward and backward substitution of LU decomposition has been developed to match the architecture of parallel processors. Without loss of generality, we assume storage for the inverse matrix of pressure $O(N^4)$ beyond the physical memory of the machine itself. Although forward sweep is automatically parallelizable, storage of $O(N^3)$ instead of $O(N^2)$ is still required in order to execute the backward substitution in parallel, so that the overall memory required for pressure solution is of the same order of magnitude as for the field variables.

The pressure matrix structure as sketched in Eq. (27) can be interpreted as an algebraic equation

$$\mathbf{A}\mathbf{x} = \mathbf{b}. \tag{28}$$

The discussion of the solution of Eq. (28) is split into two steps.

6.3.1. *Forward sweep.* In the process of row–column reduction, rows run in a concurrent mode and columns run in a vector mode. All off-diagonal zero coefficients can be screened out without any manipulation. The scalar multiples making a

used later for the reduction of source term **b** in Eq. (28). The speedup is proportional to the number of processors applied to this step except for an offset caused by data addressing.

6.3.2. *Backward substitution.* The eigenfunction expansion is crucial to performing the backward substitution in parallel because it makes Eq. (26) independent for different values of the eigenvalue indices $\alpha_i$, $\beta_k$.

The following Fortran code is part of the complete solution for the pressure and illustrates the essential point. First we reduce the source term **b** by using the elements stored in the lower triangular part of **A** during the forward sweep

```
do i = 2, NP
  i1 = i - 1
  do n = 2, NUM
    b(i, n) = b(i, n) - dotproduct(A(i, 1:i1, n), b(1:i1, n))
  end do
end do
```

where *NUM* is the total number of points in one of the directions with the eigenvalues; *NP* is the total number of points for Eq. (27). Next we perform the backward substitution as

```
do n = 2, NUM
  b(NP, n) = b(NP, n)/A(NP, NP, n)
end do
do k = 1, NP - 1
  i = NP - k
  ind = iref(i)
  m = NP - k + 1
  do n = 2, NUM
    b(i, n) = (b(i, n) - dotproduct(A(i, m:ind, n), b(m:ind, n)))/A(i, i, n)
  end do
end do
```

TABLE I

Cpu Time in Seconds per Time Step

| Number of processors | 1[a] | 1 | 2 | 3 | 4 | 8 |
|---|---|---|---|---|---|---|
| | 132.0 | 80.3 | 42.86 | 30.48 | 24.42 | 14.97 |
| Speedup[b] | | 1 | 1.87 | 2.63 | 3.29 | 5.37 |

[a] Code without any optimization.
[b] Compared to one processor.

where $iref(i)$ is the index of nonzero upper diagonal elements with respect to each row index $i$. The innermost loop performs both source term reduction and backward substitution on a given row. This is done for all the (independent) matrices labeled by eigenvalues before the row index is changed.

6.3.3. *Timings.* Table I shows some timings for implementation of our program for the solution of 3-dimensional driven cavity flow with $NX = 42$ (7 elements), $NY = 36$ (single element) and $NZ = 48$ (8 elements with equal length), running on an Alliant FX/8-series (eight processors) computer.

The method used to compute the solution of the Navier–Stokes equations in parallel performance is very important for the technique of domain decomposition with SAP. The basic idea is that in alternately performing steps (2) and (3) of Section 5 all the processors are used to update whichever subdomain $\mathrm{II} \cup \mathrm{III}$ or $\mathrm{I} \cup \mathrm{II}$ is current. Updating such a geometrically simple region can be performed efficiently in parallel as we sketched earlier on driven cavity flow.

Tables II and III show some timings for the implementation of our program for the solution of both 2- and 3-dimensional flow over a backward step at Reynolds number 375 (four iterations of SAP) with $NX = 60$ (10 elements), $NY = 36$ (6 elements) at downstream, while $NX = 12$ (2 elements), $NY = 18$ (3 elements) at upstream, and $NZ = 48$ (single element), running on the same computer.

Note that using eight processors speeds the computation by factors of 5.37 and 5.5 for the 3-dimensional driven cavity flow and flow over a backward step, respectively. It is worth emphasizing that our program is written entirely in terms of the proposed Fortran 8x standard and can be run on any machine for which a compiler implementing the standard exists.

TABLE II

Cpu Time in Seconds per Time Step (2D)

| Number of processors | 1 | 2 | 3 | 4 | 8 |
|---|---|---|---|---|---|
| | 4.07 | 2.10 | 1.48 | 1.18 | 0.74 |
| Speedup[a] | 1 | 1.94 | 2.75 | 3.45 | 5.52 |

[a] Compared to one processor.

TABLE III

Cpu Time in Seconds per Time Step (3D)

| Number of processors | 1 | 2 | 3 | 4 | 8 |
|---|---|---|---|---|---|
| | 235.5 | 127.7 | 90.1 | 73.5 | 43.5 |
| Speedup[a] | 1 | 1.84 | 2.61 | 3.20 | 5.42 |

[a] Compared to one processor.

## 7. Results and Discussion

### 7.1 *Driven Cavity Flow*

This subsection discusses the results we have obtained for the 3-dimensional driven cavity flow. In the present case, as mentioned in Section 6, 43, 37, and 49 grid points were used in the $x$, $y$, and $z$ directions, respectively. For $\mathrm{Re} = 3200$ integration of the time-dependent 3-dimensional momentum equation was performed from time zero to $t = 166.67$ (real time 20 min) with $\Delta t = 0.0036$, which is larger than that of the global method.

In order to exhibit the dynamic behavior of TGL vortices, the computed velocity fields are displayed in vector plots at time 10, 15, and 20 min, respectively. The $xy$ plane velocity vector plots at the central plane $z = 1.5$ are sketched in Figs. 5a, 6a, 7a. Despite the prominent boundary effect caused by the side wall in the spanwise direction, primary vortices appearing on both sides of the $xy$ plane at the lower corners still can be observed, but the intensity change as a function of time may imply some periodic relationship. At the same time, the 2-dimensional secondary vortex generated at the left upper corner no longer exists in the 3-dimensional case. This is probably due to the fact that the energy confined in a 2D domain will split a part in the third direction when the 3D problem arises.

Recall from [1] that, for the $yz$ plane flow patterns at $\mathrm{Re} = 1000$ in the cubic driven cavity flow, no TGL vortices were found near the downstream wall while a single pair of upper and lower corner recirculating flows was seen at the middle plane $x = 0.5$, as well as a nearly upward flow close to the upstream wall and a downward flow near the downstream wall appeared. Figures 5d, 6d, and 7d show the well-developed flow pattern visualized at plane $x = 0.765$, where between 3–4 pairs of TGL vortices plus one corner pocket recirculating flow near the bottom surface were formed. These computed results demonstrate the temporal variation in corner vortex and TGL vortex strength and size due to spatial location. In Figs. 5c, 6c, and 7c at the mid-plane $x = 0.5$, less obvious TGL vortices are formed when the position is far away from the downstream well. The presence of TGL vortices is explained by Koseff *et al.* [2] as the formation of the corner vortex inducing a rotational effect which propagates out from the side wall toward the center, and in conjunction with primary cell circulation results in the unstable interface. All the computational results are in qualitative agreement with those found by the experiment of [2]. A firework-like flow pattern near the upstream wall, seen in

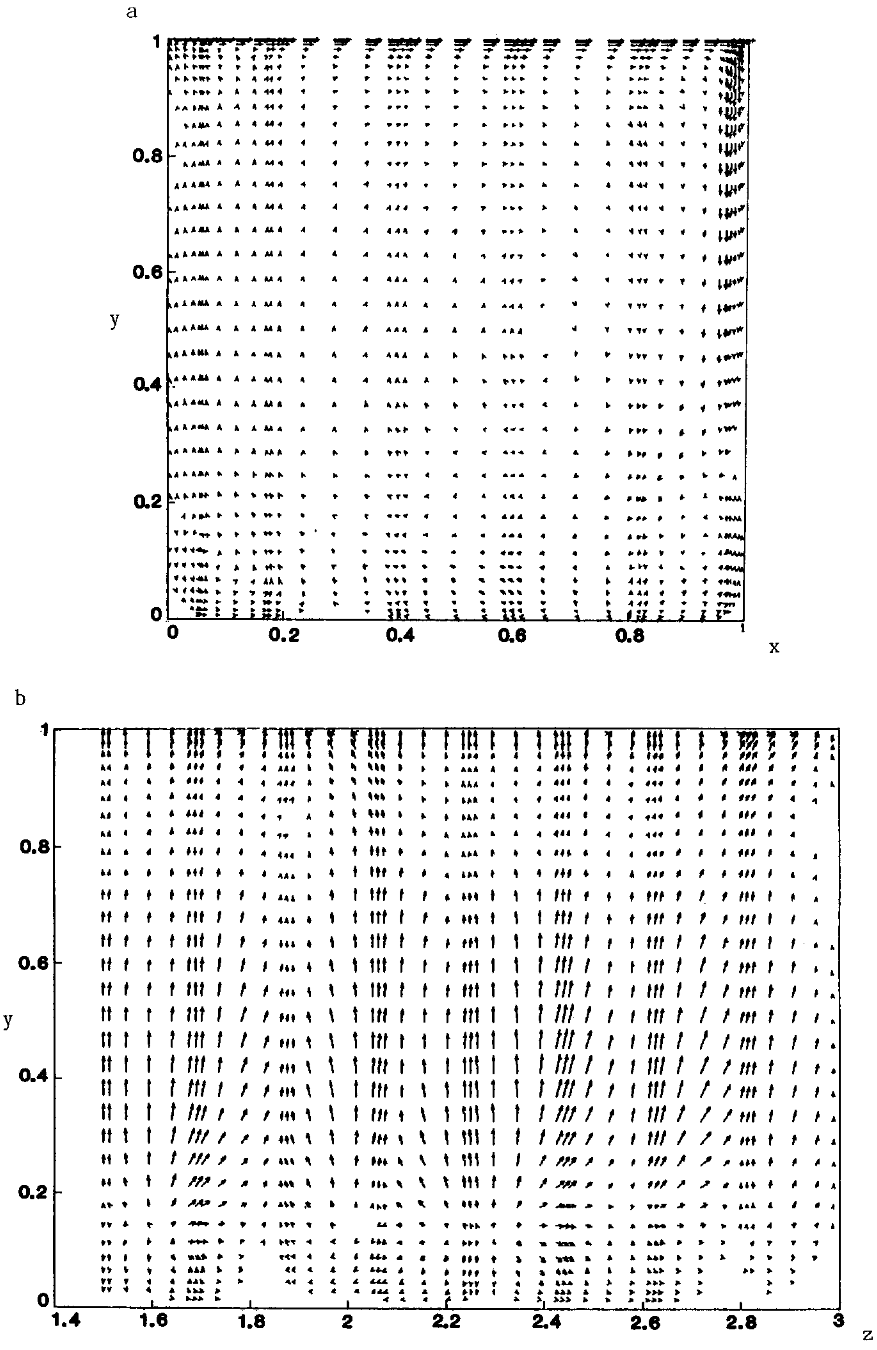


FIG. 5. Flow direction vectors at time 10 min for Re = 3200 in the (a) $z = 1.5$ plane, (b) $x = 0.03$ plane, (c) $x = 0.5$ plane, and (d) $x = 0.765$ plane.

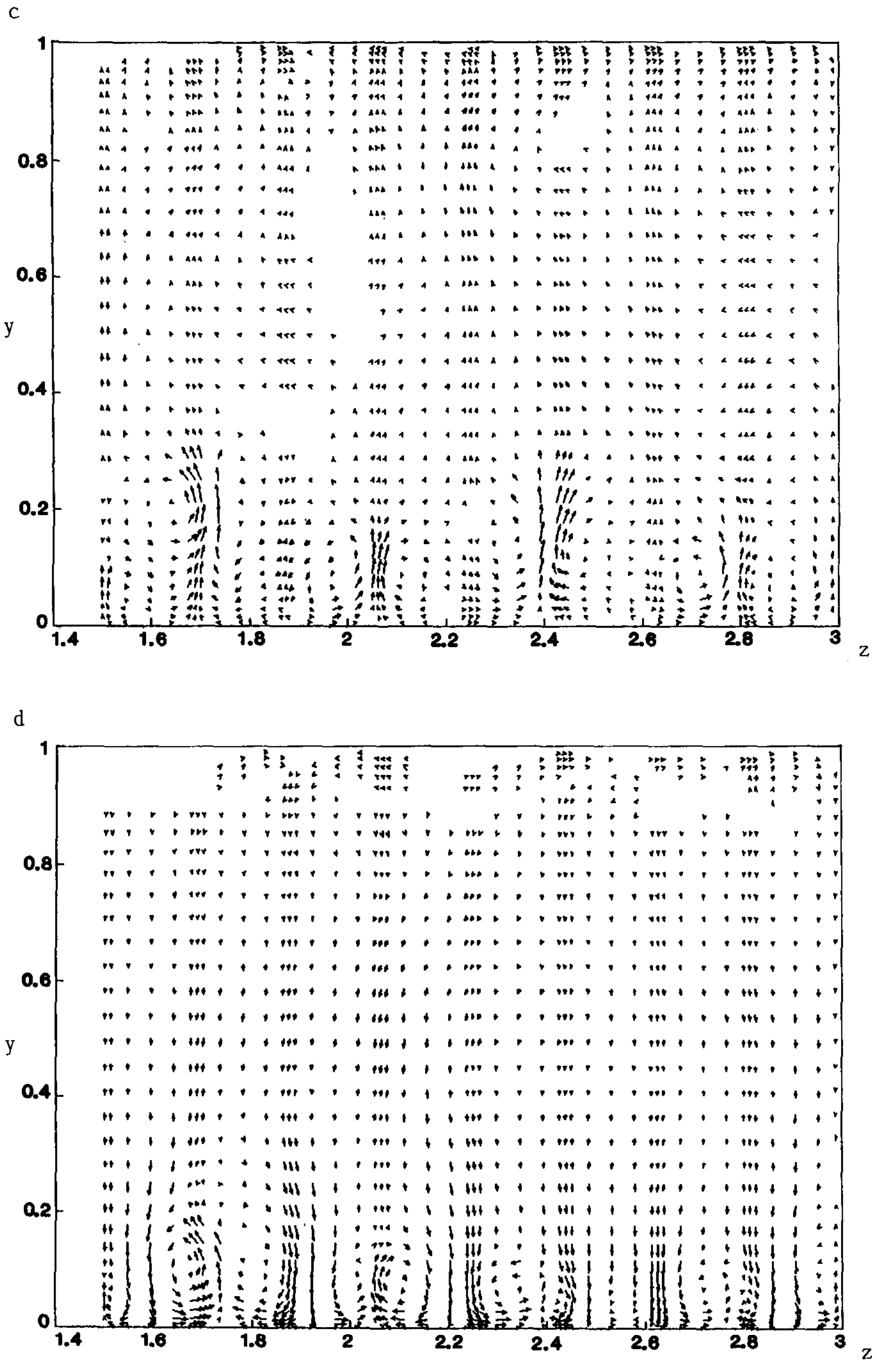


FIGURE 5—*Continued*

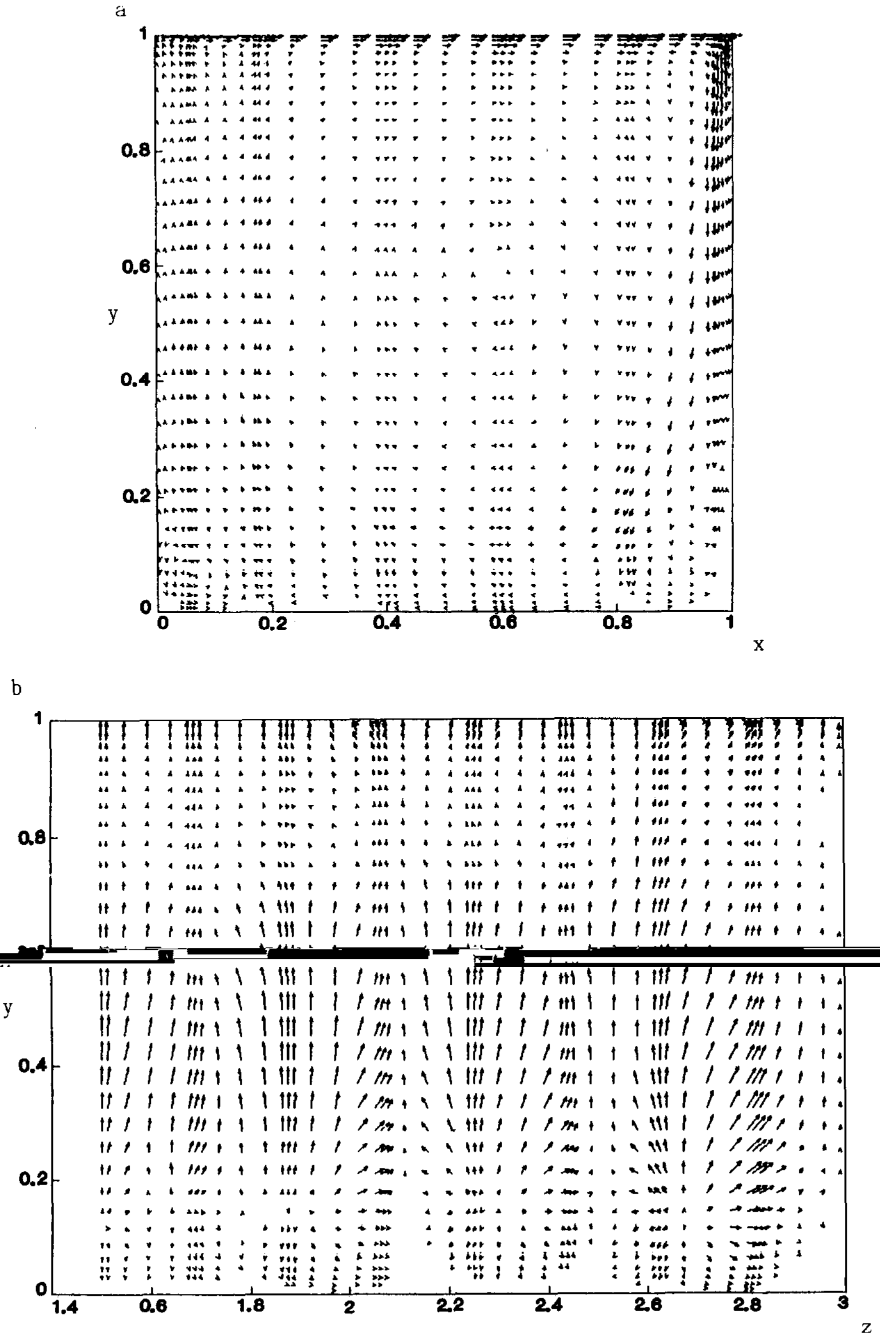


FIG. 6. Flow direction vectors at time 15 min for Re = 3200 in the (a) $z = 1.5$ plane, (b) $x = 0.03$ plane, (c) $x = 0.5$ plane, and (d) $x = 0.765$ plane.

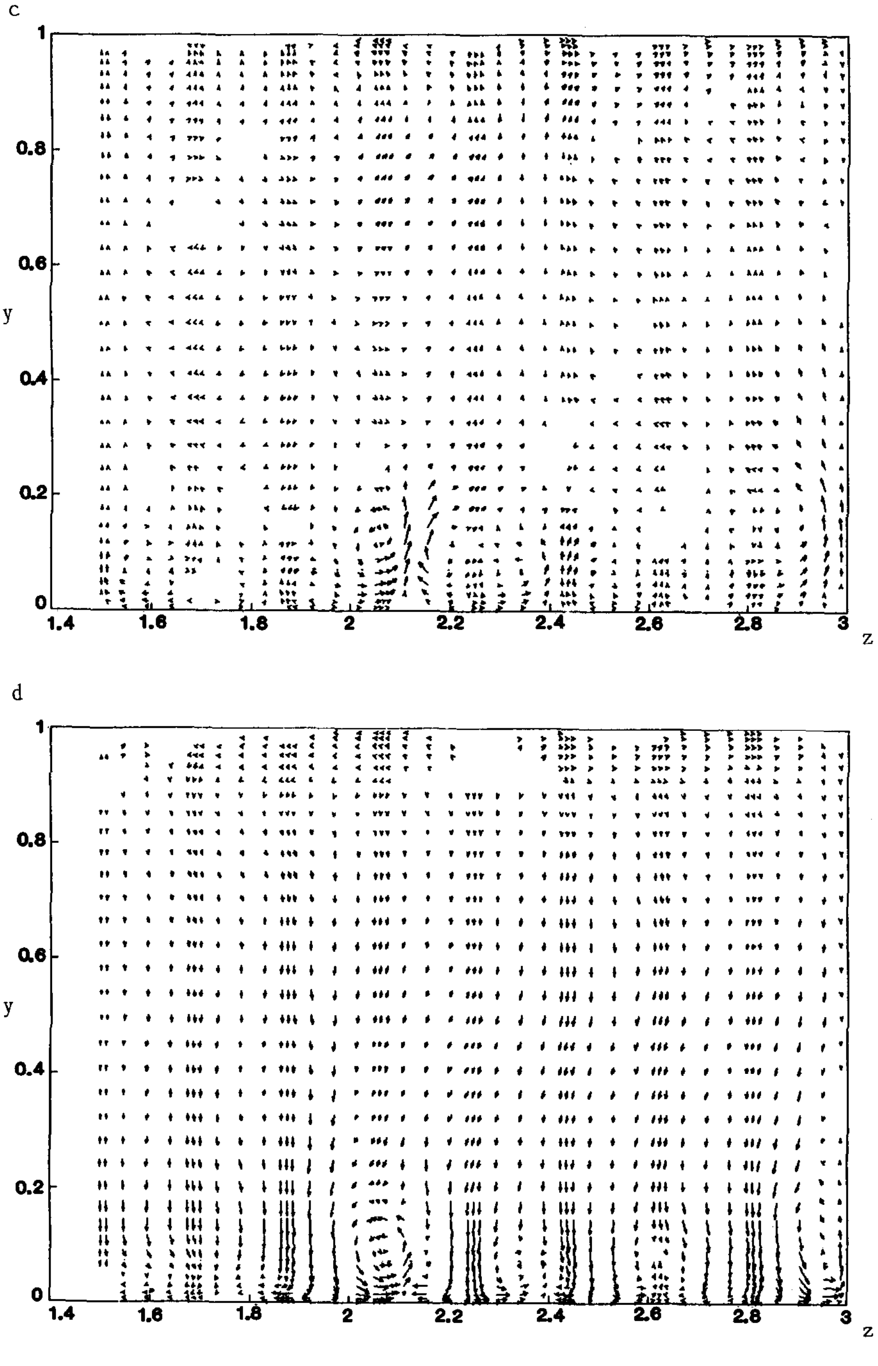


FIGURE 6—*Continued*

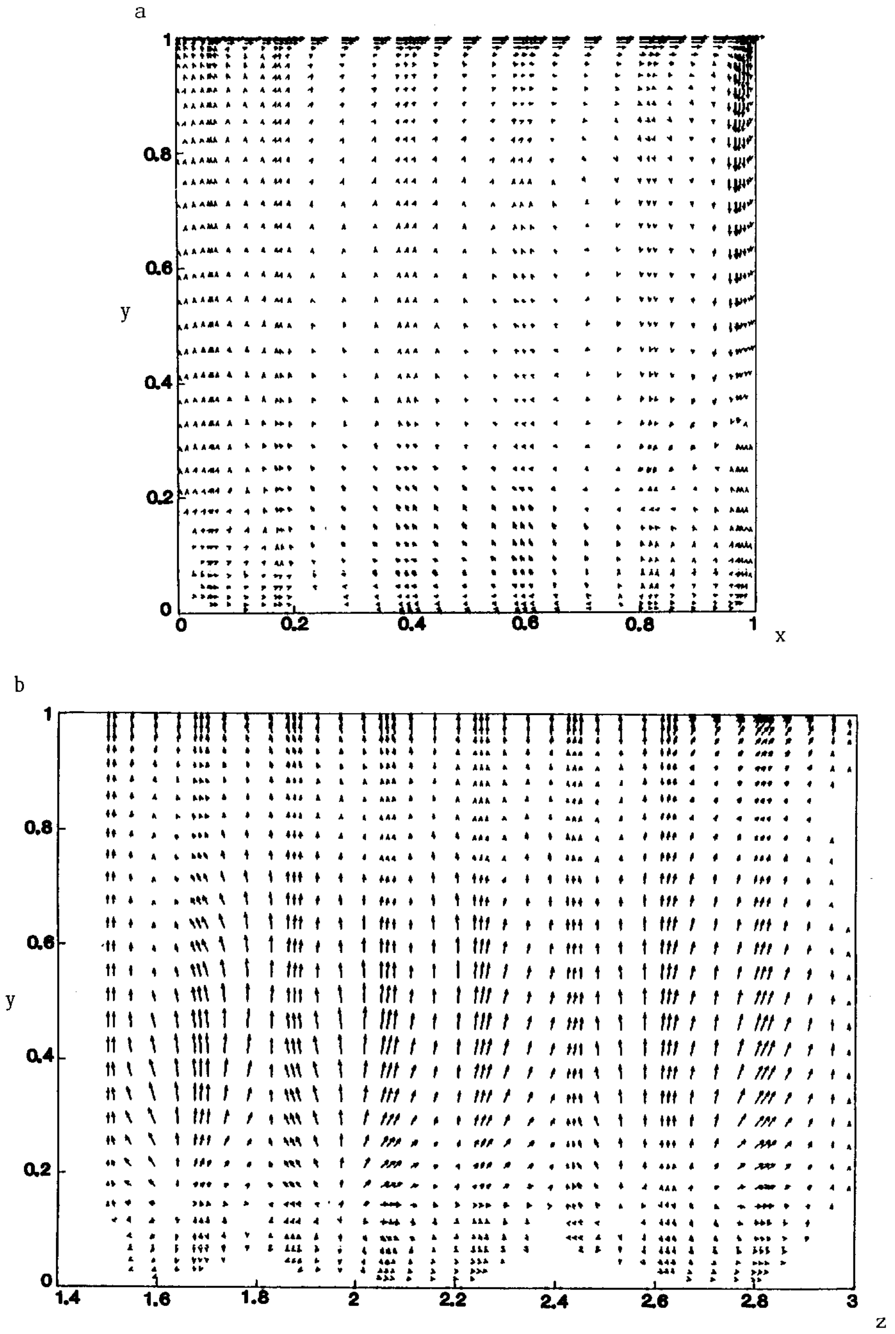


FIG. 7. Flow direction vectors at time 20 min for Re = 3200 in the (a) $z = 1.5$ plane, (b) $x = 0.03$ plane, (c) $x = 0.5$ plane, and (d) $x = 0.765$ plane.

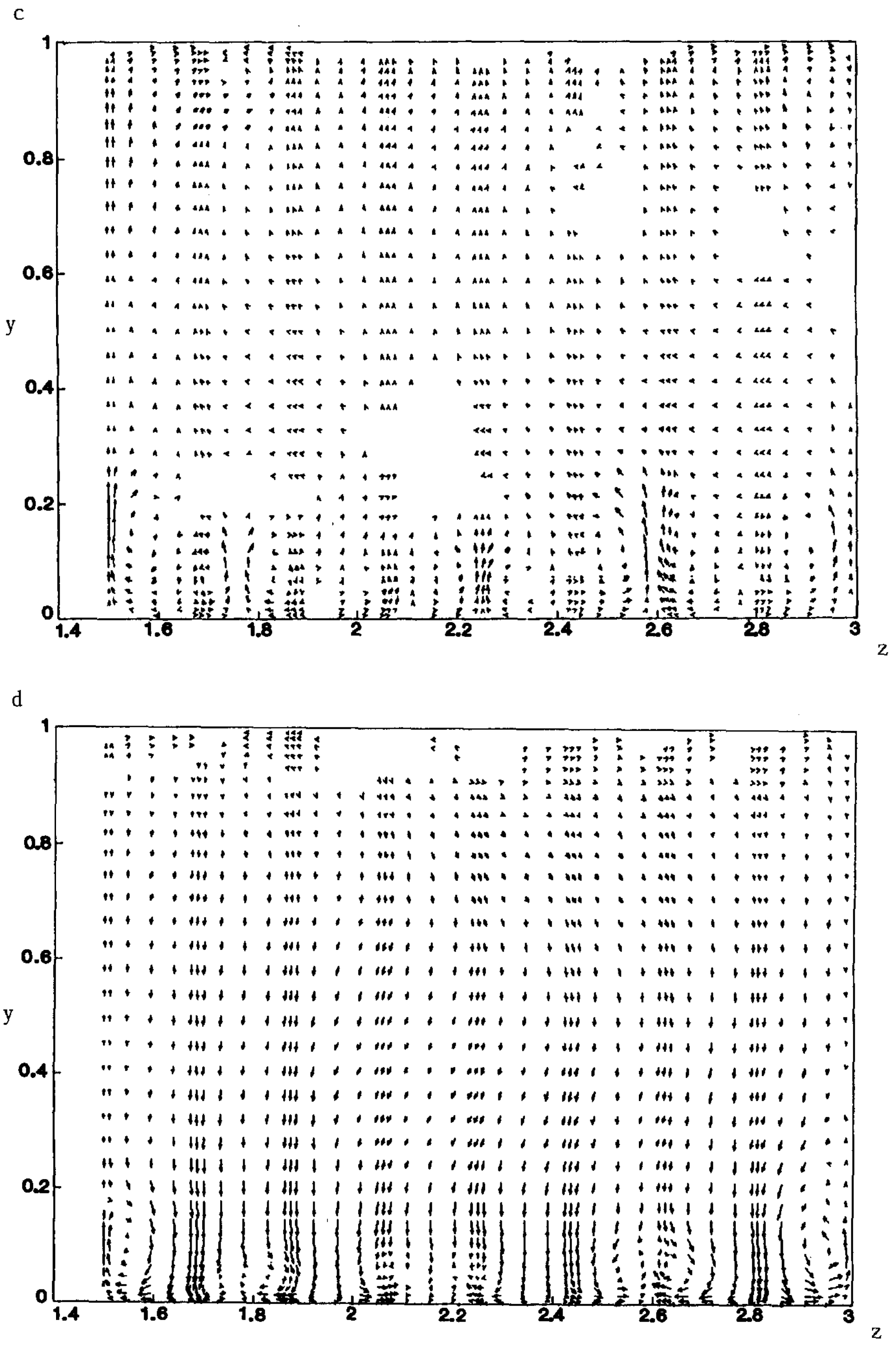


FIGURE 7—*Continued*

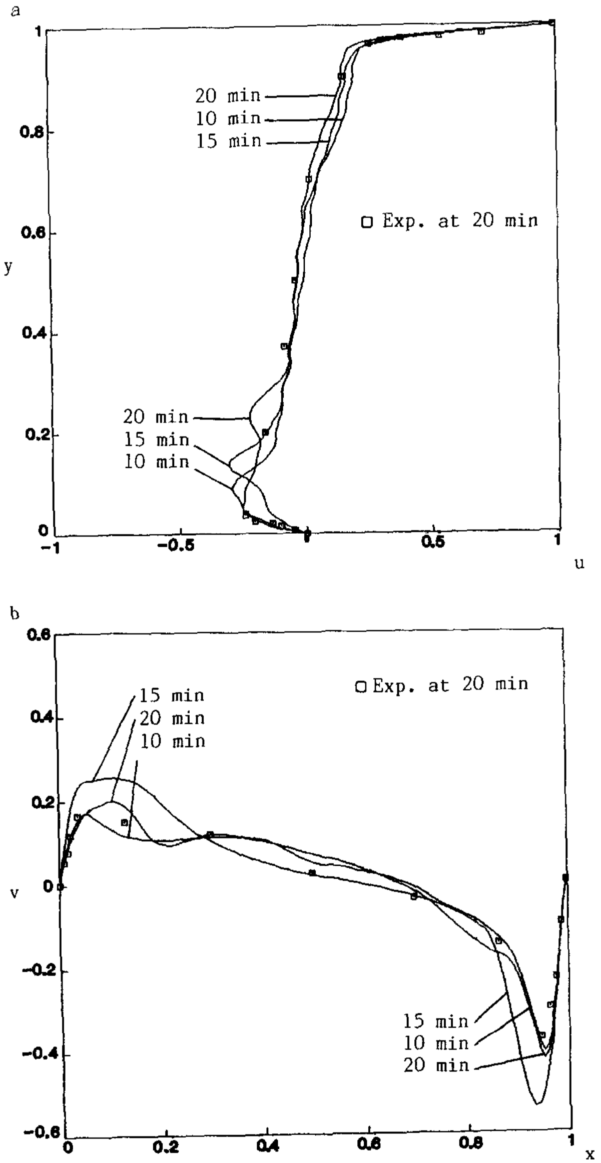


FIG. 8. Three-dimensional cavity velocity profiles for Re = 3200 on (a) vertical centerline and (b) horizontal centerline.

Figs. 5b, 6b, and 7b at plane $x = 0.03$, is continuously entrained by the top moving plate. This is the same as the low Reynolds number in the cubic driven cavity flow problem.

Figures 8 compares the numerical results of time evolution of velocity profiles for the $u$ component on the vertical centerline and $v$ component on the horizontal centerline at the symmerty plane $z = 1.5$ together with experimental results shown on the same plot. Despite some deviation from the experimental results, qualitatively a good agreement still can be observed. These results, moreover, provide information on the dynamic behavior of TGL vortices.

### 7.2. *Flow over a Backward Step*

This subsection presents the results obtained by applying the present scheme, domain decomposition with SAP, to the 2- and 3-dimensional flow over a backward step. First, we will examine the effect of our downstream boundary conditions, the second derivative of velocity components vanishing in the streamwise direction, upon upstream flow development. Figure 9 shows that the reattachment length at Re = 150 with cut-off downstream position $x/S = 19$ (8 elements), 10.5 (5 elements), and 7.5 (4 elements) agrees very well with those found by Dirichlet boundary condition applied at infinity by the same method. Even the minimum streamfunction, i.e., $\psi_{\min} = -0.04421$, $-0.04434$, $-0.04425$ for $x/S = 19$, 10.5, and 7.5, respectively, exhibits the same accuracy when compared to [19], $\psi_{\min} = -0.0436$ by using 33 * 297 grid points. Figure 10 provides streamline plots for the range of Reynolds number between Re = 75 and Re = 450. As expected, with

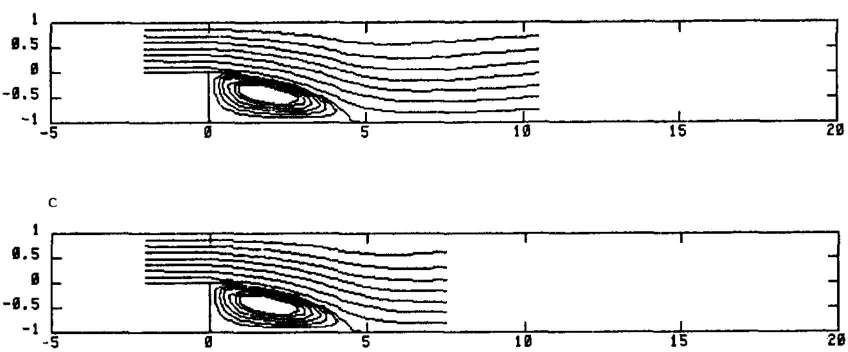


FIG. 9. Flow over a step, Re = 150, with downstream conditions at $x/S =$: (1) 19.0, (b) 10.5, (c) 7.5.

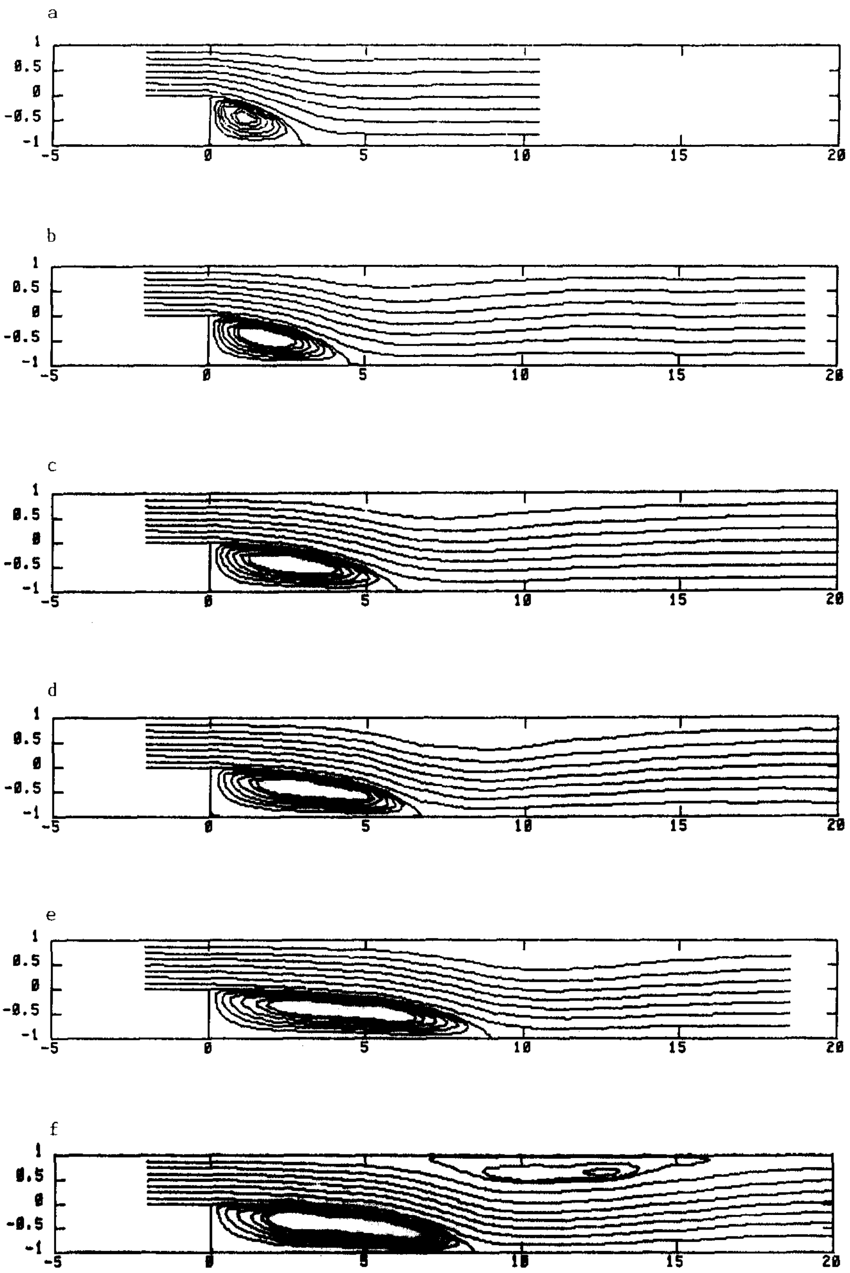


FIG. 10. Streamlines for flow over a step with Re =: (a) 75, (b) 150, (c) 225, (d) 300, (e) 375, (f) 450.

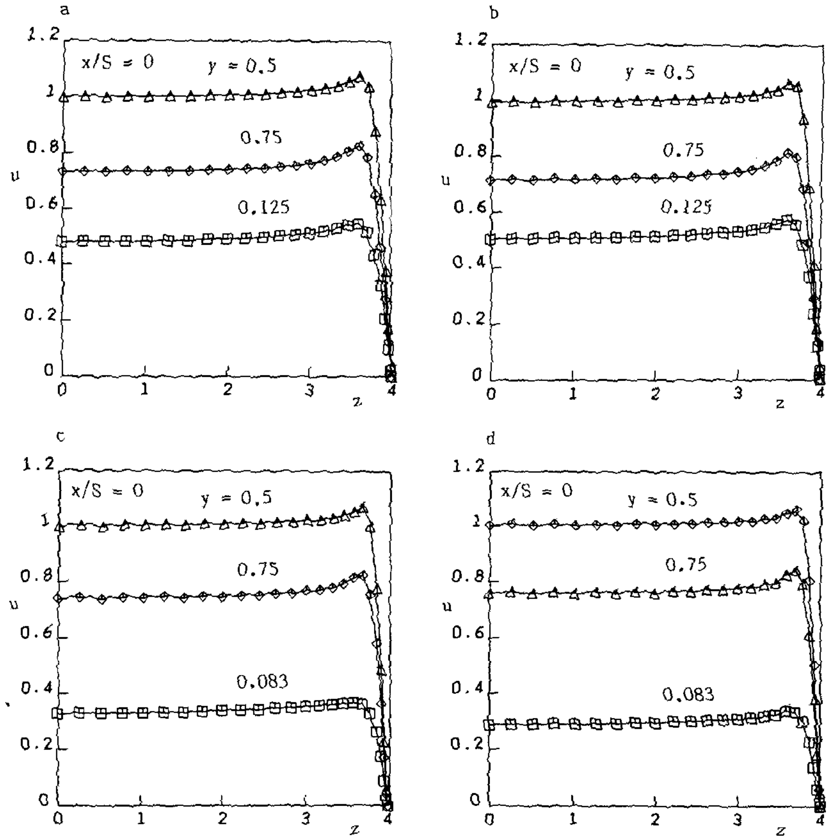


FIG. 11. Velocity profiles of the inlet flow along the transverse direction with Re =: (a) 225, (b) 300, (c) 375, (d) 450.

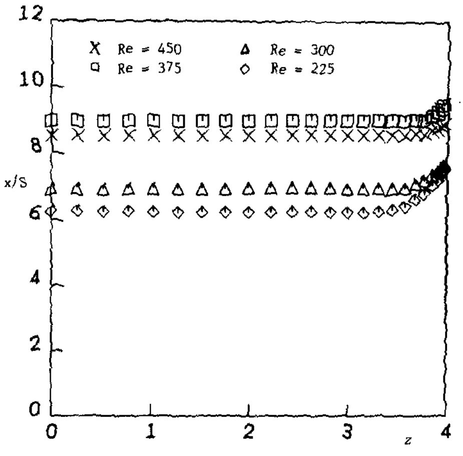


FIG. 12. Spanwise location of reattachment line with different Reynolds numbers.

increasing Reynolds number, the downstream reattachment length of flow development increases up to Re = 375, while at Re = 450 the reattachment length slightly decreases and the secondary vortex appears at the upper wall.

As Armaly *et al.* [7] have pointed out, the deviation of the 2-dimensional flow calculation from experimental results is due to the 3-dimensionality of the experimental flow observed above Reynolds number 450. From the numerical results of 3-dimensional flow, the velocity profiles of inlet flow along the transverse direction are given in Fig. 11. As expected, this shows 2-dimensional inlet flow except for the overshooting phenomena within the thin boundary layers due to the well-known entrance effect. The 2-dimensionality of the flow, except in the boundary layers, for Reynolds number, Re = 225, 300, 375, is demonstrated by the plot of the spanwise location of the reattachment line shown in Fig. 12. The same figure

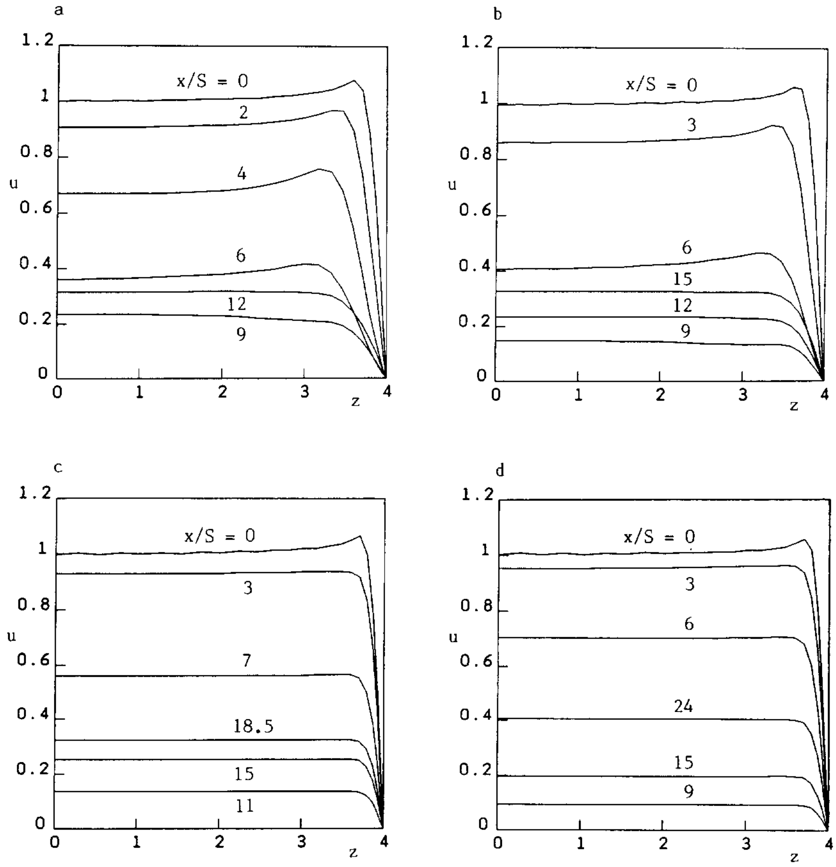


FIG. 13. Downstream spanwise velocity profiles with Re =: (a) 225, (b) 300, (c) 375, (d) 450 at vertical position $y = 0.5$.

also provides the data for Re = 450 where the 3-dimensionality of the flow gradually develops, i.e., the reattachment length along the symmetry plane is slightly higher than 2D. Downstream spanwise velocity profiles are plotted in Figs. 13 and 14 at vertical positions $y = 0.5$ and $-0.5$, respectively. All profiles outside the boundary layers remain 2-dimensional for all the indicated Reynolds numbers. Figure 15 sketches downstream velocity profiles for both 2- and 3-dimensional flow along the symmetry $xy$ plane at $z = 0$. It indicates that the deviation between 2- and 3-dimensional flow is not pronounced for Reynolds numbers up to 375 except the downstream region close to the step at Re = 450.

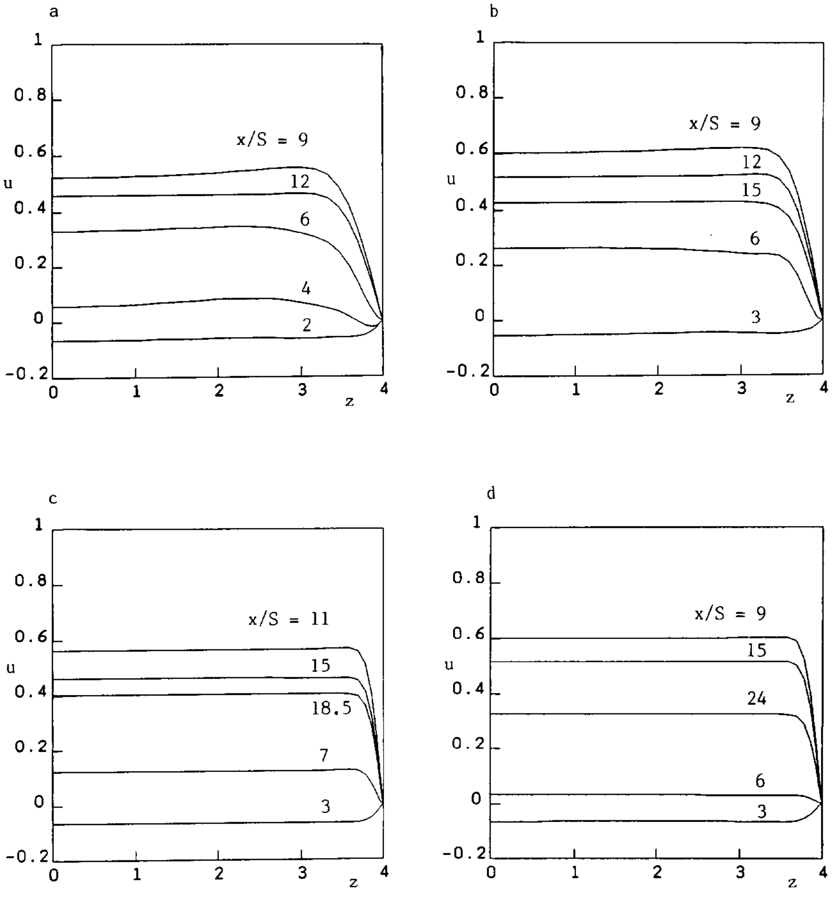


FIG. 14. Downstream spanwise velocity profiles with Re =: (a) 225, (b) 300, (c) 375, (d) 450 at vertical position $y = -0.5$.

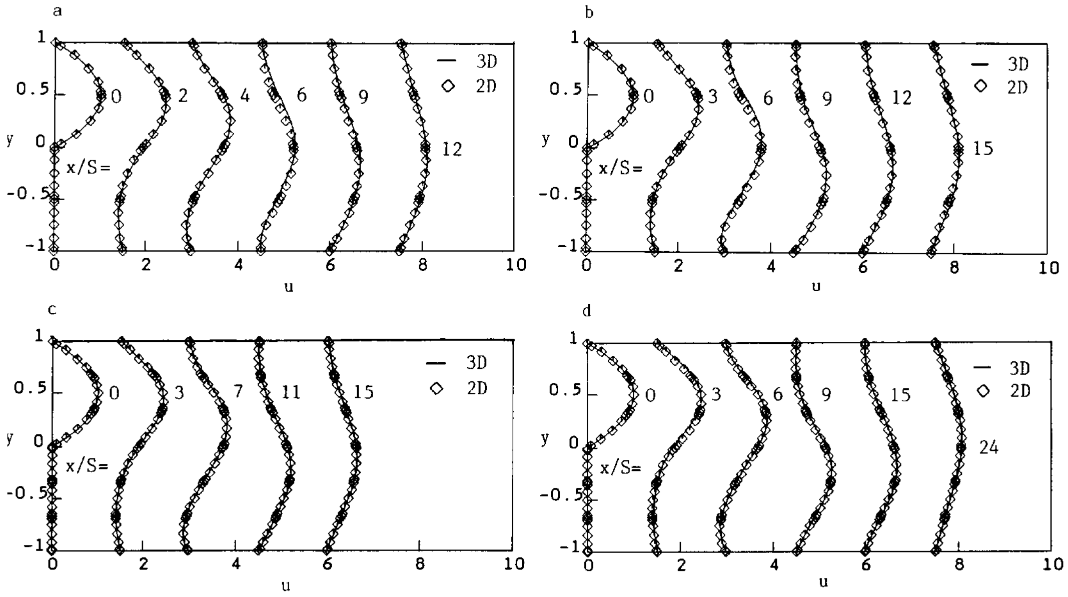


FIG. 15. Downstream velocity profile plots for 3-dimensional flow with Re =: (a) 225, (b) 300, (c) 375, (d) 450 along the symmetry plane $z = 0$.

## 8. CONCLUSIONS

The 3-dimensional Navier–Stokes equations have been solved by Chebyshev pseudospectral matrix element method employing a primitive variable formulation of splitting technique to simulate the dynamic TGL vortices in driven cavity flow. In the solution approach, the continuity equation is satisfied everywhere, in the interior (including inter-element points) and on the boundaries. The numerical test case indicates that the $c^0$ PSME method is superior to the $c^1$ PSME method when applied to the problem with steep gradients.

Domain decomposition with SAP has been used to simulate flow over a backward step. In the solution approach, the complex geometry is divided into a few subdomains, each of simple geometry, where the pressure solution can be easily obtained. With the continuity equation as the boundary conditions at the overlapped interfaces, an eigenfunction expansion technique can be applied to each subdomain to give the direct and fast solution of the 3-dimensional pressure Poisson equations in 1-dimensional form so that the parallel implementation of each subdomain could be easily achieved.

The key feature of the work presented is that with an eigenfunction expansion in two directions, the resulting 3-dimensional direct matrix inversion for the pressure Poisson equations is reduced to a set of simple 1-dimensional problems for which the data dependency occurring in the backward substitution of LU decomposition is eliminated. This makes possible efficient parallel performance of both the forward and backward sweep. The other time-consuming parts of the calculation can be reduced to dotproducts and matrix multiplications—basic linear algebra operations defined in the proposed Fortran 8x standard and efficiently implementable on wide classes of vector, parallel, and vector-parallel machines. The main results shown here were obtained on an Alliant FX/8 vector-parallel mini-supercomputer with two processors.

Numerical results computed by the PSME method have shown dynamic lateral TGL vortices in driven cavity flow which are in accord with experiments. For flow over a backward step, numerical results are in good agreement with the 2-dimensional experimental results. For Reynolds numbers up to 450, we found that 3-dimensional effects are confined to the boundary layer region in the spanwise direction.

## Acknowledgments

The authors would like to thank Mr. John Dutton of Alliant Computer System Corporation for his assistance in getting timings on the four and eight processor machines. This work was partially supported by the Office of Naval Research under the Contract N00039-87-C-5301.

## References

1. H. C. Ku, R. S. Hirsh, and T. D. Taylor, *J. Comput. Phys.* **70**, 439 (1987).
2. J. R. Koseff and R. L. Street, *J. Fluid Eng. Amer. Soc. Mech. Eng.* **106**, 385 (1984).
3. C. J. Freitas, R. L. Street, A. N. Findikakis, and J. R. Koseff, *Int. J. Numer. Methods Fluids* **5**, 561 (1985).
4. Y. Takemoto and Y. Nakamura, in *Proceedings, Tenth Int. Numer. Methods in Fluid Dynamics*, Beijing, 1987, edited by Z. Youlan (Springer-Verlag, Berlin, 1987), p. 594.
5. J. Kim and P. Moin, *J. Comput. Phys.* **59**, 308 (1985).
6. S. E. Rogers, J. L. Chang, and D. Kwak, *J. Comput. Phys.* **73**, 364 (1987).
7. B. F. Armaly, F. Durst, J. C. F. Pereira, and B. Schönung, *J. Fluid Mech.* **127**, 473 (1983).
8. H. C. Ku, R. S. Hirsh, and T. D. Taylor, in *Proceedings, Seventh GAMM Conference on Numerical Methods in Fluid Mechanics*, Louvain-la-Neuve, 1987, edited by M. Deville (Vieweg, Braunschweig, 1987), p. 151.
9. H. C. Ku, R. S. Hirsh, and T. D. Taylor, in *Proceedings, Eighth Computing Methods in Applied Sci. & Eng.* (INRIA, Versailles, 1987), p. 167.
10. A. T. Patera, *J. Comput. Phys.* **54**, 468 (1984).
11. D. Kopriva, NASA Langley Research Center, ICASE Report No. 86-28 (unpublished).
12. L. W. Ehrlich, *SIAM J. Sci. Stat. Comput.* **7**, 989 (1986).
13. L. Fuchs, in *Proceedings, Seventh GAMM Conference on Numerical Methods in Fluid Mechanics*, Louvain-la-Neuve, 1987, edited by M. Deville (Vieweg, Braunschweig, 1987), p. 96.
14. Y. Morchoisne, "Pseudospectral Space-Time Calculations of Incompressible Viscous Flow," AIAA 19th Aerospace Sciences Meeting, St. Louis, AIAA paper No. 81-0109, 1981 (unpublished).
15. H. C. Ku and D. T. Hatziavramidis, *J. Comput. Phys.* **56**, 495 (1984).
16. D. Gottlieb and L. Lustman, *SIAM Numer. Anal.* **20**, 909 (1983).
17. A. J. Chorin, *Math. Comput.* **22**, 745 (1968).
18. K. Gallivan, W. Jalby, and U. Meier, *SIAM Sci. Stat. Comput.* **8**, 1079 (1987).
19. Y. D. Schkalle and F. Thiele, *Notes on Numerical Fluid Mech.*, Bièvres, 1983, edited by K. Morgan, J. Periaux, and F. Thomasset (Vieweg, Braunschweig, 1984), p. 372.