INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS, VOL. 11, 515–539 (1990)

SOLUTION TECHNIQUES FOR THE VORTICITY–STREAMFUNCTION FORMULATION OF TWO-DIMENSIONAL UNSTEADY INCOMPRESSIBLE FLOWS

T. E. TEZDUYAR, J. LIOU, D. K. GANJOO AND M. BEHR

Department of Aerospace Engineering and Mechanics, and Minnesota Supercomputer Institute, University of Minnesota, Minneapolis, MN 55455 U.S.A.

SUMMARY

A review of our solution techniques for the vorticity–streamfunction formulation of two-dimensional incompressible flows is presented. While both the viscous and inviscid cases are considered, the derivation of the proper finite element formulations for multiply connected domains is emphasized. In all formulations associated with the vorticity transport equation, the streamline upwind/Petrov–Galerkin method is used. The adaptive implicit–explicit and grouped element-by-element solution strategies are employed to maximize the computational efficiency. The solutions obtained in all test cases compare well with solutions from previously published investigations. The convergence and benchmark studies performed in this paper show that the solution techniques presented are accurate, reliable and efficient.

KEY WORDS Vorticity–streamfunction Unsteady incompressible flows

1. INTRODUCTION

In this paper we present a review of our solution techniques for the vorticity–streamfunction formulation of two-dimensional incompressible flows. We consider both the viscous and inviscid cases.

The advantages in using the vorticity–streamfunction formulation for two-dimensional computations are well known. The difficulties in this formulation are associated with the convection term in the vorticity transport equation, lack of boundary condition for the vorticity at no-slip boundaries, and determination of the value of the streamfunction at the internal boundaries for multiply connected domains.

For viscous flow problems, at no-slip boundaries corresponding to solid surfaces the variational equations needed to determine the value of the vorticity at such boundaries can be derived from the Poisson equation for the streamfunction.[1] This subject was also addressed in References 2–5. In problems involving multiply connected domains, an additional variational equation is needed for each internal boundary to determine the value of the streamfunction at that boundary. These additional equations can be derived by integrating the equation of motion in the velocity–pressue formulation along each internal boundary and combining the result with the variational formulation of the vorticity transport equation.[1] Formulations of this type can also be found, for steady problems, in References 6–8.

For inviscid flows there is no need for boundary conditions for the vorticity at solid surfaces. However, in the case of multiply connected domains the value of the streamfunction at the internal

0271–2091/90/130515–25$12.50

Received 6 February 1989
Revised 14 April 1989

boundaries must still be determined as part of the overall solution. The additional equations needed can, again, be derived by integrating the equation of motion in the velocity–pressure formulation along each internal boundary and combining the result with the variational formulation of the Poisson equation for the streamfunction.[9] Of course, whether the flow is viscous or inviscid, these difficulties related to solid boundaries do not exist for computations based on velocity–pressure formulations.[10, 11]

In all finite element formulations corresponding to the vorticity transport equation we employ the streamline upwind/Petrov–Galerkin (SUPG) method. The vorticity transport equation involves convection terms which become dominant as the Reynolds number increases. It is well known that, owing to such dominant convection terms, especially in the presence of sharp layers in the solution, regular (Bubnov–) Galerkin finite element and classical centred finite difference methods lead to spurious oscillations in the solution. The SUPG scheme minimizes such oscillations, yet introduces minimal numerical diffusion. Schemes of this type have been successfully applied to various fluid dynamics and convection–diffusion–reaction problems.[12, 13]

In the formulations described above, implicit time integration of the spatially discretized vorticity transport equation and spatial discretization of the Poisson equation for the streamfunction lead to linear equation systems with large global matrices. For most problems of practical interest, management of these large matrices places a heavy demand on the CPU time and memory resources of the computational environment. Sometimes this demand can become too heavy even for the largest and fastest computers of our time. The two alternative solution strategies reviewed in this paper are the adaptive implicit–explicit (AIE) solution scheme and the grouped element-by-element (GEBE) iteration method.

The implicit–explicit algorithm proposed by Hughes and Liu[14] for solid mechanics and heat transfer problems involves static allocation of the implicit and explicit elements based on stability and accuracy considerations. In Tezduyar and Liou[15] the same static approach was applied to fluid mechanics problems and the AIE scheme was presented in its preliminary stage.

The AIE method is based on dynamic grouping of the elements into the implicit and explicit subsets as dictated by the element level stability and accuracy considerations. For this purpose the algorithm continuously monitors, for all elements, the Courant number and the level of variations in the solution. In this approach we can have implicit elements only where they are needed; elsewhere in the domain computations can be performed explicitly, and this of course results in substantial reductions in the computational cost involved.

The dynamic grouping involved in the AIE scheme does not have to be based on the distinction between implicitly and explicitly treated elements. The concept can be extended to other cases in which the letters 'I' and 'E' in 'AIE' refer to any two procedures; and the reason for favouring one procedure over another could be based on any factor, such as the cost efficiency, the type of spatial or temporal discretizations, the differential equations used for modelling, etc. Examples of such AIE approaches can be found in Reference 16.

The GEBE method is a variation of the element-by-element (EBE) iteration method and is based on arrangement of the elements into groups with no inter-element coupling within each group. In the EBE iteration method the preconditioning matrix is chosen to be sequential product of the element level matrices. Consequently the need for the formation, storage and factorization of large global matrices is eliminated. Element level matrices can be either stored or recomputed. In the case where the element level matrices are stored, the storage need is still only linearly proportional to the number of elements. The EBE implementations in computational fluid dynamics were first reported, in the context of compressible Euler equations, by Hughes *et al.*[17] Applications to convection–diffusion equations and incompressible flows can be seen in Tezduyar *et al.*[18] Hughes and Ferencz[19] proposed EBE schemes which are based on Crout and

Gauss–Seidel factorizations. The EBE method was also employed in conjunction with the GMRES method[20] by Hughes and Shakib.[21]

The GEBE method has some common ground with the operator-splitting and domain decomposition methods.[22] The GEBE preconditioner is a sequential product of the element group matrices with the condition that no two elements in the same group can be adjacent. This grouping is achieved by a simple algorithm which is applicable to arbitrary meshes.[23] In this form, vectorization and parallel implementation of the EBE method become very clear. To minimize the overhead associated with synchronization in parallel computations we try to minimize the number of groups.

The governing equations and the finite element formulation are given in Sections 2 and 3. The AIE and GEBE methods are described in Sections 4 and 5. We present our numerical tests and conclusions in Sections 6 and 7.

2. VORTICITY–STREAMFUNCTION FORMULATION OF TWO-DIMENSIONAL INCOMPRESSIBLE FLOWS

Consider a two-dimensional spatial domain Ω and a time interval $(0, T)$ with the spatial and temporal co-ordinates denoted by $\mathbf{x} \in \bar{\Omega}$ and $t \in [0, T]$, where a superposed bar indicates the set closure. The boundary Γ of the domain Ω consists of an external boundary Γ_0 and q internal boundaries denoted by Γ_k, $k = 1, 2, \ldots, q$, i.e.

$$\Gamma = \bigcup_{k=0}^{q} \Gamma_k. \tag{1}$$

We start with the following incompressible flow equations:

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} + \frac{1}{\rho} \nabla p - \nu \nabla^2 \mathbf{u} = 0 \quad \text{on } \Omega \times (0, T), \tag{2}$$

$$\nabla \cdot \mathbf{u} = 0 \quad \text{on } \Omega \times (0, T), \tag{3}$$

where $\mathbf{u}$, p, ρ and ν are the velocity, pressure, density and kinematic viscosity respectively. The incompressibility constraint (3) can be satisfied automatically by defining a streamfunction ψ such that

$$\mathbf{u} = \nabla^{\perp} \psi = \left\{ \frac{\partial \psi}{\partial x_2}, -\frac{\partial \psi}{\partial x_1} \right\}. \tag{4}$$

A convection–diffusion equation for the scalar vorticity

$$\omega = \frac{\partial u_2}{\partial x_1} - \frac{\partial u_1}{\partial x_2} \tag{5}$$

can be used in place of equation (2); this equation for the vorticity is obtained by taking the curl of (2):

$$\frac{\partial \omega}{\partial t} + \nabla^{\perp} \psi \cdot \nabla \omega - \nu \nabla^2 \omega = 0 \quad \text{on } \Omega \times (0, T). \tag{6}$$

Furthermore, by eliminating $\mathbf{u}$ between equations (4) and (5) we can obtain a Poisson equation involving the vorticity and the streamfunction:

$$\nabla^2 \psi + \omega = 0 \quad \text{on } \Omega \times (0, T). \tag{7}$$

Remarks

1. In the case of inviscid flows equation (6) reduces to a hyperbolic equation.
2. For viscous flows the internal boundaries (Γ_k, $k=1,2,\ldots,q$) represent obstacles with no-slip surfaces, whereas for inviscid flows these boundaries represent impermeable obstacles with friction-free surfaces.

On the boundaries the tangential and normal components of the velocity can be expressed as the normal and tangential derivatives of the streamfunction:

$$u_\tau = \boldsymbol{\tau}\cdot\mathbf{u} = \frac{\partial\psi}{\partial n}, \tag{8a}$$

$$u_n = \mathbf{n}\cdot\mathbf{u} = -\frac{\partial\psi}{\partial\tau}, \tag{8b}$$

where $\boldsymbol{\tau}$ and $\mathbf{n}$ are the unit tangential and normal vectors.

We assume that the external boundary Γ_0 admits the following decomposition with respect to the type of boundary condition specified for ψ:

$$\Gamma_0 = \overline{\Gamma_{\mathscr{g}} \cup \Gamma_{\mathscr{G}} \cup \Gamma_{\mathscr{h}}}. \tag{9a}$$

$$\varnothing = \Gamma_{\mathscr{g}} \cap \Gamma_{\mathscr{G}}, \qquad \varnothing = \Gamma_{\mathscr{G}} \cap \Gamma_{\mathscr{h}}, \qquad \varnothing = \Gamma_{\mathscr{h}} \cap \Gamma_{\mathscr{g}}. \tag{9b}$$

A similar decomposition can be made with respect to ω:

$$\Gamma_0 = \overline{\Gamma_{\tilde{\mathscr{g}}} \cup \Gamma_{\mathscr{G}} \cup \Gamma_{\tilde{\mathscr{h}}}}, \tag{10a}$$

$$\varnothing = \Gamma_{\tilde{\mathscr{g}}} \cap \Gamma_{\mathscr{G}}, \qquad \varnothing = \Gamma_{\mathscr{G}} \cap \Gamma_{\tilde{\mathscr{h}}}, \qquad \varnothing = \Gamma_{\tilde{\mathscr{h}}} \cap \Gamma_{\tilde{\mathscr{g}}}. \tag{10b}$$

On the basis of these decompositions, the boundary conditions for the external boundary are summarized in Table I.

Several comments can be made about these boundary conditions, and the interested reader can find those in References 1 and 9.

We will assume that at all internal boundaries the normal component of the velocity is zero, i.e.

$$-\frac{\partial\psi}{\partial\tau} = 0 \qquad \text{on } \Gamma_k \times (0,T), \quad k=1,2,\ldots,q. \tag{11}$$

Furthermore, we will assume that for viscous flows the tangential velocity at an internal boundary is known; therefore we can write

$$\frac{\partial\psi}{\partial n} = (u_\tau)_k \qquad \text{on } \Gamma_k \times (0,T), \quad k=1,2,\ldots,q, \tag{12}$$

Table I. Boundary conditions for the external boundary Γ_0

Boundary	Condition
$\Gamma_{\mathscr{g}}$	$\psi = \mathscr{g}$
$\Gamma_{\tilde{\mathscr{g}}}$	$\omega = \tilde{\mathscr{g}}$
$\Gamma_{\mathscr{G}}$	$\psi = \mathscr{G}$, $\partial\psi/\partial n = \mathscr{H}$
$\Gamma_{\mathscr{h}}$	$\partial\psi/\partial n = \mathscr{h}$
$\Gamma_{\tilde{\mathscr{h}}}$	$\nu\,\partial\omega/\partial n = \tilde{\mathscr{h}}$*

* This condition vanishes for inviscid flows.

where $(u_\tau)_k$ is the given tangential velocity for the internal boundary k.

From equation (11) we know that the value of the streamfunction is invariant along each internal boundary; however, we do not know what that value is and therefore we need to determine it as part of the overall solution. Whether the flow is viscous or inviscid, to determine such unknown values of the streamfunction we first need to write equation (2) along each internal boundary, i.e.

$$\frac{\partial u_\tau}{\partial t}+u_\tau\frac{\partial u_\tau}{\partial \tau}+\frac{1}{\rho}\frac{\partial p}{\partial \tau}+\nu\frac{\partial \omega}{\partial n}=0 \quad \text{on } \Gamma_k\times(0,T), \quad k=1,2,\ldots,q. \tag{13}$$

$$\frac{\partial}{\partial t}\int_{\Gamma_k}u_\tau \mathrm{d}\Gamma+\int_{\Gamma_k}\frac{\partial}{\partial \tau}\left(\frac{1}{2}u_\tau^2+\frac{p}{\rho}\right)\mathrm{d}\Gamma+\nu\int_{\Gamma_k}\frac{\partial \omega}{\partial n}\mathrm{d}\Gamma=0 \quad \text{on } (0,T), \quad k=1,2,\ldots,q. \tag{14}$$

Assuming that $\frac{1}{2}u_\tau^2+p/\rho$ is single-valued, the second integral on the left-hand side of equation (14) vanishes; consequently we get

$$\frac{\partial}{\partial t}\int_{\Gamma_k}u_\tau \mathrm{d}\Gamma+\nu\int_{\Gamma_k}\frac{\partial \omega}{\partial n}\mathrm{d}\Gamma=0 \quad \text{on } (0,T), \quad k=1,2, \;..,q. \tag{15}$$

From this equation we can obtain the additional equation needed to determine the unknown value of the streamfunction at an internal boundary. For *viscous flows*, assuming that $(u_\tau)_k$ is invariant along Γ_k, equation (15) leads to

$$\nu\int_{\Gamma_k}\frac{\partial \omega}{\partial n}\mathrm{d}\Gamma=-S_k\frac{\partial (u_\tau)_k}{\partial t} \quad \text{on } (0,T), \quad k=1,2,\ldots,q, \tag{16}$$

where S_k is the length of the boundary Γ_k. For *inviscid flows*, from (15) we get

$$\int_{\Gamma_k}u_\tau \mathrm{d}\Gamma=(\text{constant})_k \quad \text{on } (0,T), \quad k=1,2,\ldots,q. \tag{17}$$

For time-dependent problems the constants in (17) are evaluated from the given initial condition, whereas for steady state problems they must be given.

Finally, the initial condition for this problem consists of specification of the initial distribution of the vorticity, i.e.

$$\omega(\mathbf{x},0)=\omega_0(\mathbf{x}) \quad \text{on } \Omega, \tag{18}$$

where $\omega_0(\mathbf{x})$ is a given function.

3. THE FINITE ELEMENT FORMULATION

Let $\mathscr{E}$ denote the set of all elements resulting from the finite element discretization of the computational domain Ω into subdomains Ω^e, $e=1,2,\ldots,n_{\mathrm{el}}$, such that

$$\bar{\Omega}=\bigcup_{e=1}^{n_{\mathrm{el}}}\bar{\Omega}^e, \qquad \varnothing=\bigcap_{e=1}^{n_{\mathrm{el}}}\Omega^e, \tag{19}$$

where n_{el} is the number of elements. Let Γ^e denote the boundary of Ω^e. Let $\mathscr{E}_{\mathscr{G}}$ and $\mathscr{E}_k$, $k=1,2,\ldots,q$, denote the set of elements adjacent to the boundaries $\Gamma_{\mathscr{G}}$ and Γ_k, $k=1,2,\ldots,q$, i.e.

$$\mathscr{E}_{\mathscr{G}}=\{\Omega^e\,|\,\Omega^e\in\mathscr{E},\ \Gamma^e\cap\Gamma_{\mathscr{G}}\neq\varnothing\}, \tag{20}$$

$$\mathscr{E}_k=\{\Omega^e\,|\,\Omega^e\in\mathscr{E},\ \Gamma^e\cap\Gamma_k\neq\varnothing\}, \quad k=1,2,\ldots,q. \tag{21}$$

We associate to $\mathscr{E}$ the following finite-dimensional space:

$$H^{1h}=\{\phi^h \mid \phi^h\in C^0(\bar{\Omega}),\ \phi^h|_{\Omega^e}\in P^1\,\forall\Omega^e\in\mathscr{E}\}. \tag{22}$$

The solution spaces for the streamfunction and vorticity, which are respectively denoted by S^h and $\tilde{S}^h$, are defined as follows:

$$S^h=\{\psi^h \mid \psi^h\in H^{1h},\ \psi^h\doteq g \text{ on } \Gamma_g,\ \psi^h\doteq\mathscr{G} \text{ on } \Gamma_{\mathscr{G}},\ \partial\psi^h/\partial\tau\doteq 0 \text{ on } \Gamma_k,\ k=1,2,\ldots,q\}, \tag{23}$$

$$\tilde{S}^h=\{\omega^h \mid \omega^h\in H^{1h},\ \omega^h\doteq\tilde{g} \text{ on } \Gamma_{\tilde{g}}\}. \tag{24}$$

The weighting function spaces needed are listed below:

$$V^h_*=\{w^h \mid w^h\in H^{1h},\ w^h\doteq 0 \text{ on } \Gamma_g\cup\Gamma_{\mathscr{G}}\bigcup_{k=1}^{q}\Gamma_k\}, \tag{25}$$

$$\tilde{V}^h_*=\{w^h \mid w^h\in H^{1h},\ w^h\doteq 0 \text{ on } \Gamma_{\tilde{g}}\cup\Gamma_{\mathscr{G}}\bigcup_{k=1}^{q}\Gamma_k\}, \tag{26}$$

$$V^h_{\mathscr{G}}=\{w^h \mid w^h\in H^{1h},\ w^h|_{\Omega^e}=0\,\forall\Omega^e\notin\mathscr{E}_{\mathscr{G}}\}, \tag{27}$$

$$V^h_k=\{w^h \mid w^h\in H^{1h},\ w^h|_{\Omega^e}=0\,\forall\Omega^e\notin\mathscr{E}_k\},\qquad k=1,2,\ldots,q, \tag{28}$$

$$V^h_{k\mathbf{R}}=\{w^h \mid w^h\in V^h_k,\frac{\partial w^h}{\partial\tau}\doteq 0 \text{ on } \Gamma_k\},\qquad k=1,2,\ldots,q. \tag{29}$$

The finite element formulation associated with equation (6) is given as follows: find $\psi^h\in S^h$ and $\omega^h\in\tilde{S}^h$ such that

$$\int_\Omega w^h\left(\frac{\partial\omega^h}{\partial t}+\nabla^\perp\psi^h\cdot\nabla\omega^h\right)\mathrm{d}\Omega+\int_\Omega \nu\nabla w^h\cdot\nabla\omega^h\mathrm{d}\Omega+\sum_{e\in\mathscr{E}}\int_{\Omega^e}\delta^h\left(\frac{\partial\omega^h}{\partial t}+\nabla^\perp\psi^h\cdot\nabla\omega^h-\nu\nabla^2\omega^h\right)\mathrm{d}\Omega$$
$$=\int_{\Gamma_{\tilde{h}}} w^h\tilde{h}\,\mathrm{d}\Gamma\quad\forall w^h\in\tilde{V}^h_*\langle\bigcup_{k=1}^{q}V^h_k\rangle, \tag{30}$$

where δ^h is a $C^{-1}(\Omega)$ *Petrov–Galerkin* supplement to the weighting function w^h. In this formulation the terms inside '$\langle\ \rangle$' exist only for inviscid flows.

Remarks

1. The term δ^h needs to act only in the element interiors and therefore is allowed to be discontinuous across element boundaries. Setting this term to zero reduces the formulation to a (Bubnov–) *Galerkin* one.
2. Various Petrov–Galerkin procedures, particularly those based on the streamline upwind/Petrov–Galerkin (SUPG) formulations, have been successfully applied to a wide range of fluid flow problems. For examples and discussion on the subject see References 12 and 24.
3. For the precise definition of δ^h we refer the interested reader to References 1 and 6.

The discrete variational formulation association with (7) is stated as follows: find $\psi^h\in S^h$ and $\omega^h\in\tilde{S}^h$ such that

$$\int_\Omega\nabla w^h\cdot\nabla\psi^h\mathrm{d}\Omega-\int_\Omega w^h\omega^h\mathrm{d}\Omega=\int_{\Gamma_h}w^h h\,\mathrm{d}\Gamma\quad\forall w^h\in V^h_* \tag{31}$$

and

$$\int_{\Omega} \nabla w^h \cdot \nabla \psi^h \, \mathrm{d}\Omega - \int_{\Omega} w^h \omega^h \, \mathrm{d}\Omega = \int_{\Gamma_{\mathscr{h}}} w^h \mathscr{h} \, \mathrm{d}\Gamma + \int_{\Gamma_{\mathscr{g}}} w^h \mathscr{H} \, \mathrm{d}\Gamma \langle\!\langle + \sum_{k=1}^{q} \int_{\Gamma_k} w^h (u_\tau)_k \, \mathrm{d}\Gamma \rangle\!\rangle$$

$$\forall w^h \in V^h_{\mathscr{g}} \langle\!\langle \bigcup_{k=1}^{q} V^h_k \rangle\!\rangle, \tag{32}$$

where the terms inside '⟪ ⟫' exist only for viscous flows.

For *viscous flows*, to derive the additional variational formulations needed to determine the values of the streamfunction at the internal boundaries, we use equation (6) and the function spaces given by (29). These variational formulations are stated as follows: find $\psi^h \in S^h$ and $\omega^h \in \tilde{S}^h$ such that

$$\int_{\Omega} w^h \left(\frac{\partial \omega^h}{\partial t} + \nabla^{\perp} \psi^h \cdot \nabla \omega^h \right) \mathrm{d}\Omega + \int_{\Omega} \nu \nabla w^h \cdot \nabla \omega^h \mathrm{d}\Omega + \sum_{e \in \mathscr{E}_k} \int_{\Omega^e} \delta^h \left(\frac{\partial \omega^h}{\partial t} + \nabla^{\perp} \psi^h \cdot \nabla \omega^h - \nu \nabla^2 \omega^h \right) \mathrm{d}\Omega$$

$$= \int_{\Gamma_k} w^h \nu \frac{\partial \omega^h}{\partial n} \quad \forall w^h \in V^h_{k\mathrm{R}}, \quad k = 1, 2, \ldots, q. \tag{33}$$

Because w^h is invariant along Γ_k, the right-hand side of (33) can be obtained from (16); therefore we get

$$\int_{\Omega} w^h \left(\frac{\partial \omega^h}{\partial t} + \nabla^{\perp} \psi^h \cdot \nabla \omega^h \right) \mathrm{d}\Omega + \int_{\Omega} \nu \nabla w^h \cdot \nabla \omega^h \mathrm{d}\Omega + \sum_{e \in \mathscr{E}_k} \int_{\Omega^e} \delta^h \left(\frac{\partial \omega^h}{\partial t} + \nabla^{\perp} \psi^h \cdot \nabla \omega^h - \nu \nabla^2 \omega^h \right) \mathrm{d}\Omega$$

$$= -w^h(\Gamma_k) S_k \frac{\partial (u_\tau)_k}{\partial t} \quad \forall w^h \in V^h_{k\mathrm{R}}, \quad k = 1, 2, \ldots, q. \tag{34}$$

For *inviscid flows*, the additional variational formulations needed are obtained by using the same function space given by (29) but in conjunction with equation (7). This leads to the following finite element formulation: find $\psi^h \in S^h$ and $\omega^h \in \tilde{S}^h$ such that

$$\int_{\Omega} \nabla w^h \cdot \nabla \psi^h \, \mathrm{d}\Omega - \int_{\Omega} w^h \omega^h \, \mathrm{d}\Omega = \int_{\Gamma_k} w^h u_\tau \, \mathrm{d}\Gamma \qquad \forall w^h \in V^h_{k\mathrm{R}}, \quad k = 1, 2, \ldots, q. \tag{35}$$

Again because w^h is invariant along Γ_k we can rewrite (35) as

$$\int_{\Omega} \nabla w^h \cdot \nabla \psi^h \, \mathrm{d}\Omega - \int_{\Omega} w^h \omega^h \, \mathrm{d}\Omega = w^h(\Gamma_k) \int_{\Gamma_k} u_\tau \, \mathrm{d}\Gamma \qquad \forall w^h \in V^h_{k\mathrm{R}}, \quad k = 1, 2, \ldots, q. \tag{36}$$

and by using (17) we get

$$\int_{\Omega} \nabla w^h \cdot \nabla \psi^h \, \mathrm{d}\Omega - \int_{\Omega} w^h \omega^h \, \mathrm{d}\Omega = w^h(\Gamma_k) \, (\text{constant})_k \quad \forall w^h \in V^h_{k\mathrm{R}}, \quad k = 1, 2, \ldots, q. \tag{37}$$

Now we have the complete set of equations needed to solve for all the unknowns of the problem whether the flow is viscous or inviscid. For further details of the spatial discretization and the description of the temporal discretization procedure see References 1 and 9.

The fully discretized equations can be solved in various ways. In this work we solve them either in their fully coupled form or by employing a block iteration technique. Solving them in their fully coupled form is quite straightforward and needs no explanation. The block iteration method, which is described in detail in Reference 1, requires, at every iteration, solution of mainly two linear equation systems. These equation systems, after hiding all the subscripts and superscripts denoting the time step and iteration count, can be written as follows:

$$\bar{\mathbf{M}}\Delta\mathbf{v}=\mathbf{R}, \tag{38}$$

$$\mathbf{K}\,\mathbf{d}=\mathbf{F}, \tag{39}$$

where the vectors **v** and **d** contain the unknown nodal values of the vorticity and streamfunction respectively. The matrices $\bar{\mathbf{M}}$ and **K** are formed by assembling their element level contributors, i.e.

$$\bar{\mathbf{M}}=\sum_{e\in\mathscr{E}}\bar{\mathbf{M}}^e, \tag{40a}$$

$$\mathbf{K}=\sum_{e\in\mathscr{E}}\mathbf{K}^e. \tag{40b}$$

The right-hand-side vectors **R** and **F**, which are known at a given iteration step, are also formed by the assembly of their element level contributors.

Remarks

1. The matrix $\bar{\mathbf{M}}^e$ has the same dimensions as the global matrix $\bar{\mathbf{M}}$ but only very few non-zero entries (e.g. 4 × 4 for a two-dimensional quadrilateral element with a scalar unknown).
2. The matrix **K** is symmetric and positive-definite. A particular version of the Petrov–Galerkin method employed results in $\bar{\mathbf{M}}$ also being symmetric and positive-definite;[15] however, we will assume that in general $\bar{\mathbf{M}}$ is not symmetric and positive-definite.

4. THE ADAPTIVE IMPLICIT–EXPLICIT (AIE) METHOD

Let $\mathscr{E}_{\mathrm{I}}$ and $\mathscr{E}_{\mathrm{E}}$ be the subsets of $\mathscr{E}$ corresponding to the implicit and explicit elements, respectively, such that

$$\mathscr{E}=\mathscr{E}_{\mathrm{I}}\cup\mathscr{E}_{\mathrm{E}},\qquad \varnothing=\mathscr{E}_{\mathrm{I}}\cap\mathscr{E}_{\mathrm{E}}. \tag{41}$$

Consequently, from equation (40a) we get

$$\bar{\mathbf{M}}=\sum_{e\in\mathscr{E}_{\mathrm{I}}}\bar{\mathbf{M}}^e+\sum_{e\in\mathscr{E}_{\mathrm{E}}}\bar{\mathbf{M}}^e. \tag{42}$$

The AIE scheme is based on modifying $\bar{\mathbf{M}}$ by replacing $\bar{\mathbf{M}}^e(\forall e\in\mathscr{E}_{\mathrm{E}})$ with its lumped mass matrix part,

$$\bar{\mathbf{M}}=\sum_{e\in\mathscr{E}_{\mathrm{I}}}\bar{\mathbf{M}}^e+\sum_{e\in\mathscr{E}_{\mathrm{E}}}\bar{\mathbf{M}}^e_{\mathrm{L}}. \tag{43}$$

The grouping given by (41) is achieved dynamically (adaptively) on the basis of stability and accuracy considerations.

The stability criterion is in terms of the element Courant number $C_{\Delta t}$ which is defined as

$$C_{\Delta t}=\frac{\|\mathbf{u}\|\,\Delta t}{h}, \tag{44}$$

where h is the element length.[1] Any element with its Courant number greater than the stability limit of the explicit method should belong to the implicit group $\mathscr{E}_{\mathrm{I}}$.

For the accuracy criterion we use a quantity which we expect to be a fair representative of the dimensionless wave number. On the basis of the observations from our numerical tests we select the accuracy criterion as

$$\sigma^e_{\mathscr{E}} = \frac{\sigma^e}{\sigma^e_{\mathrm{s}}}, \tag{45a}$$

where

$$\sigma^e = j^e(\phi)\left(\left|\frac{\partial f_1}{\partial x_1}\right| + \left|\frac{\partial f_2}{\partial x_2}\right|\right)^e, \tag{45b}$$

$$\sigma^e_{\mathrm{s}} = \max_{e \in \mathscr{E}}\left(\left|\frac{\partial f_1}{\partial x_1}\right| + \left|\frac{\partial f_2}{\partial x_2}\right|\right)^e. \tag{45c}$$

Here ϕ is a generic symbol for the dependent variable, while f_1 and f_2 are the fluxes corresponding to directions x_1 and x_2. The dimensionless element level 'jump' $j^e(\phi)$ is defined as

$$j^e(\phi) = \frac{\max_a \phi^e_a - \min_a \phi^e_a}{\phi_{\max} - \phi_{\min}}, \tag{45d}$$

where a is the element node number and ϕ^e_a is the value of the dependent variable at node a of the element e. The instantaneous maximum and minimum values of ϕ are denoted by $\phi_{\max}$ and $\phi_{\min}$. This definition of $j^e(\phi)$ represents the magnitude of the variation of the solution across an element. The elements with $\sigma^e_{\mathscr{E}}$ greater than a predetermined value belong to group $\mathscr{E}_{\mathrm{I}}$.

Borrowing from the adaptive mesh refinement techniques,[25] we also experimented with the accuracy indicator in terms of the $\mathscr{L}^2$-norm of the residual r:

$$\sigma^e_{\mathscr{E}} = \frac{\|r\|^0_{\Omega^e} - \min_e(\|r\|^0_{\Omega^e})}{\max_e(\|r\|^0_{\Omega^e}) - \min_e(\|r\|^0_{\Omega^e})}, \tag{46a}$$

with

$$\|r\|^0_{\Omega^e} = \left(\int_{\Omega^e} r^2 \mathrm{d}\Omega\right)^{1/2}. \tag{46b}$$

Note that all expressions are based on the 'previous solution'. The 'previous solution' can come from the previous time step, non-linear iteration or pseudo-time-step iteration, whichever is appropriate.

The implementation of the AIE scheme is straightforward. A global search is performed over the entire set of elements. With this search, on the basis of the two critical parameters given by (44) and (45a) (or (46a)), the elements are grouped into the subsets $\mathscr{E}_{\mathrm{I}}$ and $\mathscr{E}_{\mathrm{E}}$. Compared to having all the elements treated implicitly, this grouping results in substantial savings in CPU time and memory.

For the solution of (39) we use a preconditioned conjugate gradient method.[26] In conjunction with the AIE scheme employed for equation (38), we define our preconditioning matrix $\mathbf{P}$ to be

$$\mathbf{P} = \sum_{e \in \mathscr{E}_{\mathrm{I}}} \mathbf{K}^e + \sum_{e \in \mathscr{E}_{\mathrm{E}}} \mathrm{diag}(\mathbf{K}^e), \tag{47}$$

where $\mathbf{K}^e$ is the element contribution matrix of $\mathbf{K}$. If $\mathscr{E}_{\mathrm{E}} = \varnothing$ then $\mathbf{P} = \mathbf{K}$, and the solution technique becomes a direct one. If $\mathscr{E}_{\mathrm{I}} = \varnothing$ then the method becomes a Jacobi iteration.[26] Other definitions for **P** are of course possible; the one given by (47) is simple to implement and reflects the expectation that there may be a correlation between the solution procedures for (38) and (39).

Remarks

1. It is important to note that the grouping does not have to be only with respect to implicit and explicit treatments. In the acronym 'AIE', the letter 'I' can refer to the 'I-elements' subject to some 'I-procedure', while the letter 'E' can refer to the 'E-elements' subject to some other 'E-procedure'. We can employ the 'I-procedure' where it is needed/desired and the 'E-procedure' elsewhere. For further discussions and comments as well as other possible 'I/E procedures' see Reference 16.
2. In the AIE approach one can have a high degree of refinement throughout the mesh and raise the implicit flag only for those elements which need to be treated implicitly.
3. The savings in CPU time and memory can be maximized by performing, as often as desired, an equation renumbering at the implicit zones to obtain optimal bandwidths. Bandwidth optimizers are already available for finite element applications, especially in the area of structural mechanics.

5. THE GROUPED ELEMENT-BY-ELEMENT (GEBE) ITERATION METHOD

The linear equation systems (38) and (39) are both in the form

$$\mathbf{A}\,\mathbf{x} = \mathbf{b}. \tag{48}$$

The iterative solution techniques used to solve (48) are described in this section.

Element grouping

The elements are arranged into 'parallelizable' groups in such a way that

$$\mathscr{E} = \bigcup_{K=1}^{N_{\mathrm{pg}}} \mathscr{E}_K, \tag{49a}$$

$$\varnothing = \mathscr{E}_{\mathrm{J}} \cap \mathscr{E}_{\mathrm{K}} \quad \text{for } J \neq K, \tag{49b}$$

and such that no two elements within a group are neighbours (being neighbours is defined as having at least one common node). The number of such groups is denoted by N_{pg}. On the basis of this grouping, the matrix **A** can be written as

$$\mathbf{A} = \sum_{K=1}^{N_{\mathrm{pg}}} \mathbf{A}_K, \tag{50}$$

where the 'group matrices' are defined as

$$\mathbf{A}_K = \sum_{e \in \mathscr{E}_K} \mathbf{A}^e, \quad K = 1, 2, \ldots, N_{\mathrm{pg}}. \tag{51}$$

Remarks

1. Because there is no inter-element coupling within each group, computations performed in element-by-element fashion (such as operations performed on a group matrix) do not depend on the ordering of the elements.

2. In parallel computations, before we can start with a new element group we first have to finish with the one that we have already started with. To minimize the overhead associated with this synchronization, we try to minimize the number of groups. A simple element grouping algorithm which serves this purpose is described in Reference 23.
3. To increase the vector efficiency of the computations performed, within each group the elements are processed in packets of n_{ep} elements, where n_{ep} is the optimum packet size for a given vector environment (e.g. 64).

GEBE-preconditioned iterative solution method

Equation (48) is solved with a preconditioned iteration method which is built on solving

$$\mathbf{P}\,\Delta\mathbf{y}_m=\mathbf{r}_m \tag{52}$$

for $\Delta\mathbf{y}_m$, where $\mathbf{P}$ is the preconditioning matrix and $\mathbf{r}_m$ is the residual vector defined as

$$\mathbf{r}_m=\mathbf{b}-\mathbf{A}\,\mathbf{x}_m. \tag{53}$$

The form of the scaling matrix $\mathbf{W}$ depends on the properties of $\mathbf{A}$. If $\mathbf{A}$ is symmetric and positive-definite, one obvious choice is

$$\mathbf{W}=\operatorname{diag}\mathbf{A}. \tag{54a}$$

The alternative choice

$$\mathbf{W}=\operatorname{lump}\mathbf{M} \tag{54b}$$

gives a symmetric and positive-definite scaling matrix even if $\mathbf{A}$ is not so. Although the mass matrix comes up usually in the context of time-dependent problems, for scaling purposes it can be introduced also in the context of steady-state problems. In reality, the context in which we use this scaling matrix does not require it to be symmetric and positive-definite; it is sufficient that $\mathbf{W}$ is symmetric, and of course non-singular.

Two-pass GEBE preconditioner (2P-GEBE)

In this method the preconditioning matrix is defined as

$$\mathbf{P}=\prod_{K=1}^{N_{\mathrm{pg}}}(\mathbf{E}_K\mathbf{W}^{-1})\,\mathbf{W}\prod_{K=N_{\mathrm{pg}}}^{1}(\mathbf{W}^{-1}\mathbf{E}_K), \tag{55}$$

where

$$\mathbf{E}_K=\mathbf{W}+\tfrac{1}{2}\mathbf{B}_K,\qquad K=1,2,\ldots,N_{\mathrm{pg}}, \tag{56}$$

with

$$\mathbf{B}_K=\mathbf{A}_K-\mathbf{W}_K,\qquad K=1,2,\ldots,N_{\mathrm{pg}}, \tag{57a}$$

or

$$\mathbf{B}_K=\mathbf{A}_K,\qquad K=1,2,\ldots,N_{\mathrm{pg}}. \tag{57b}$$

The choice given by (57a) leads to 'Winget regularization'. For comments on the relationship between the GEBE and three regular EBE methods see Reference 23.

In the next two subsections we describe the GEBE versions of the Crout- and Gauss–Seidel-factorization-based EBE preconditioners.[19]

Crout-factorization-based GEBE preconditioner (Crout-GEBE)

Based on the Crout factorization

$$\mathbf{W}+\mathbf{B}_K=\mathbf{L}_K\mathbf{D}_K\mathbf{U}_K, \qquad K=1,2,\ldots,N_{\mathrm{pg}}, \tag{58}$$

the preconditioning matrix is defined as

$$\mathbf{P}=\left(\prod_{K=1}^{N_{\mathrm{pg}}}\mathbf{L}_K\right)\mathbf{W}\prod_{K=1}^{N_{\mathrm{pg}}}(\mathbf{W}^{-1}\mathbf{D}_K)\prod_{K=N_{\mathrm{pg}}}^{1}\mathbf{U}_K. \tag{59}$$

Gauss–Seidel-factorization-based GEBE preconditioner (GS–GEBE)

Consider the following lower triangular–diagonal–upper triangular decomposition:

$$\mathbf{A}_K=\mathbf{A}_K^{\mathrm{L}}+\mathbf{A}_K^{\mathrm{D}}+\mathbf{A}_K^{\mathrm{U}}, \qquad K=1,2,\ldots,N_{\mathrm{pg}}. \tag{60}$$

Based on this decomposition the preconditioning matrix is given as

$$\mathbf{P}=\prod_{K=1}^{N_{\mathrm{pg}}}(\mathbf{I}+\mathbf{A}_K^{\mathrm{L}}\mathbf{W}^{-1})\mathbf{W}\prod_{K=N_{\mathrm{pg}}}^{1}(\mathbf{I}+\mathbf{W}^{-1}\mathbf{A}_K^{\mathrm{U}}). \tag{61}$$

Updating the vector $\mathbf{x}_m$

If the matrix $\mathbf{A}$ is symmetric and positive-definite we employ a preconditioned conjugate gradient method with the preconditioners as defined in the previous subsections.

If the matrix $\mathbf{A}$ is not symmetric and positive-definite then the vector $\mathbf{x}_m$ is updated as follows:

$$\mathbf{x}_{m+1}=\mathbf{x}_m+s\Delta\mathbf{y}_m, \tag{62}$$

where the search parameter s is determined by the formula

$$s=\frac{(\mathbf{A}\Delta\mathbf{y}_m)\cdot r_m}{\|\mathbf{A}\Delta\mathbf{y}_m\|^2}, \tag{63}$$

which is derived by minimizing $\|\mathbf{r}_{m+1}\|^2$ with respect to s.

For further details and discussions on the GEBE algorithms as well as comments on the vectorization and parallel processing of these algorithms see Reference 23.

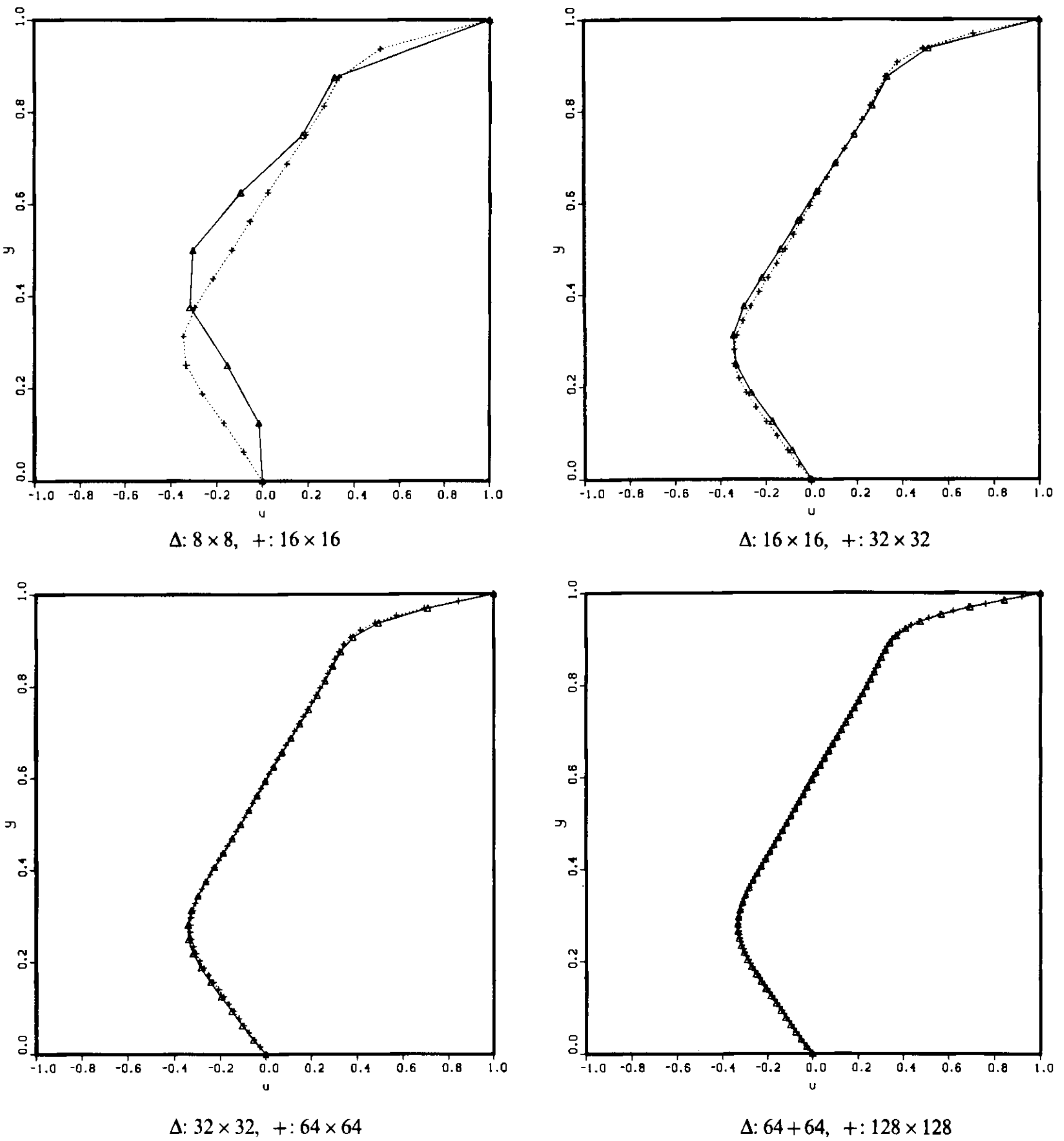
6. NUMERICAL EXAMPLES

We have tested the fully coupled and block iterative forms of the formulation together with the AIE and GEBE methods.

Convergence and benchmark studies

Numerous researchers have performed computations for the driven cavity flow problem. We chose this problem for our convergence (based on successive mesh refinement) and benchmark (based on the CPU time and memory requirement) studies. In this test the lid of the cavity has unit velocity; based on this velocity and the dimension of the cavity the Reynolds number is 400.

In the convergence study uniform meshes were used; the number of elements are 8 × 8, 16 × 16, 32 × 32, 64 × 64 and 128 × 128. We employed the GEBE solution method in conjunction with the block iteration form. The steady-state solution obtained with a given mesh was supplied (by interpolation) as the initial condition for the mesh with one level of higher refinement. Figure 1



Δ: 8 × 8, +: 16 × 16

Δ: 16 × 16, +: 32 × 32

Δ: 32 × 32, +: 64 × 64

Δ: 64+64, +: 128 × 128

Figure 1. Driven cavity flow at Reynolds number 400: convergence test based on successive mesh refinement; steady state value of u_1 along the vertical centreline

shows the steady-state value of u_1 along the vertical centre line. As far as this particular quantity is concerned, it is clear that the method converges very rapidly; the values for the 16 × 16 mesh are sufficiently close to those for the 128 × 128 mesh. Table II shows the maximum value of the streamfunction obtained with various meshes, together with the values found in the literature.

For the benchmark study based on the CPU time and memory requirements, we compared the fully coupled, block iteration, AIE block iteration and GEBE block iteration formulations. We

Table II. Driven cavity flow at Reynolds number 400: maximum value of the streamfunction obtained by various computations

8 × 8	0·110
16 × 16	0·112
32 × 32	0·118
64 × 64	0·115
128 × 128	0·114
Reference 2	0·118
Reference 27	0·121
Reference 28	0·108
Reference 29	0·102

Table III. Driven cavity flow at Reynolds number 400: comparisons based on the CPU time and memory needed for the coefficient matrices (a uniform 64 × 64 mesh was used)

	Coupled	Block-iteration	AIE block iteration	EBE block iteration
CPU time	1·0	0·4692	0·2614	0·1097
Memory	1·0	0·2701	0·0702	0·0873

employed a uniform 64 × 64 mesh and set the time step to a value that results in a maximum element Courant number of 3·0. Table III shows the comparisons based on the CPU time and memory needed for the coefficient matrices. It should be noted that the maximum element Courant number can be pushed up to 8·0 without loss of stability if the fully coupled formulation is used. Regardless of the number of iterations, the explicit method does not converge in this test. In the GEBE iteration method we chose to store the element level matrices. Alternatively, we could have chosen to recompute these matrices as they were needed; this would have resulted in a reduction in memory but an increase in CPU time. The results obtained by the various methods are indistinguishable from each other. Figures 2 and 3 show the results obtained by the AIE block iteration method at various time steps.

Flow past a bank of cylinders

Motivated by the study carried out by Joseph and his team[30] we have tested our fully coupled formulation on simulation of flow past a bank of cylinders. According to Schlichting,[31] when the solidity (i.e. the ratio of the obstructed area to the total area) is high enough, interference between the wakes behind each cylinder occurs. A related experimental investigation of flow past a pair of cylinders by Kim and Durbin[32] indicates that the wakes flop between two asymmetric states provided the gap between the cylinders is less than one diameter.

In our test we chose a solidity of 0·66 and consider a two-cylinder unit. The finite element mesh shown in Figure 4 consists of 4800 elements and 4979 nodes. In this preliminary study we impose symmetry conditions at the top and bottom computational boundaries (we also plan to experiment with periodic conditions in the very near future). At the upstream boundary both the normal and tangential derivatives of the streamfunction are specified to give a uniform freestream

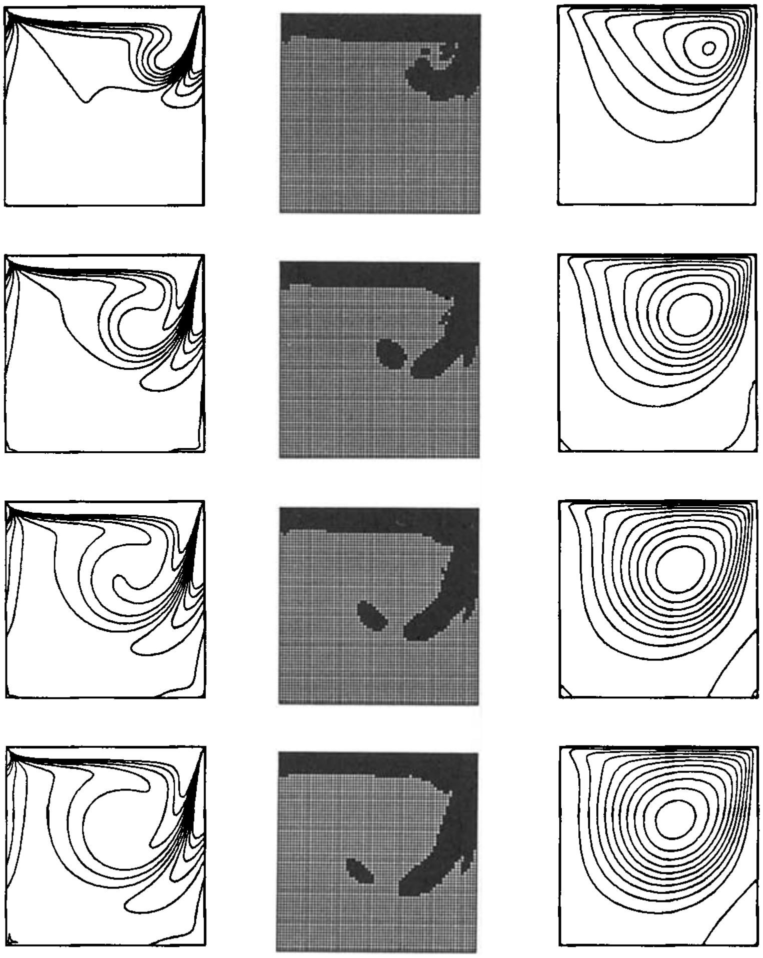


Figure 2. Driven cavity flow at Reynolds number 400: solution obtained by the AIE method; vorticity, distribution of the implicit elements, and streamfunction at $t=2{\cdot}34$, $4{\cdot}69$, $7{\cdot}03$ and $9{\cdot}38$

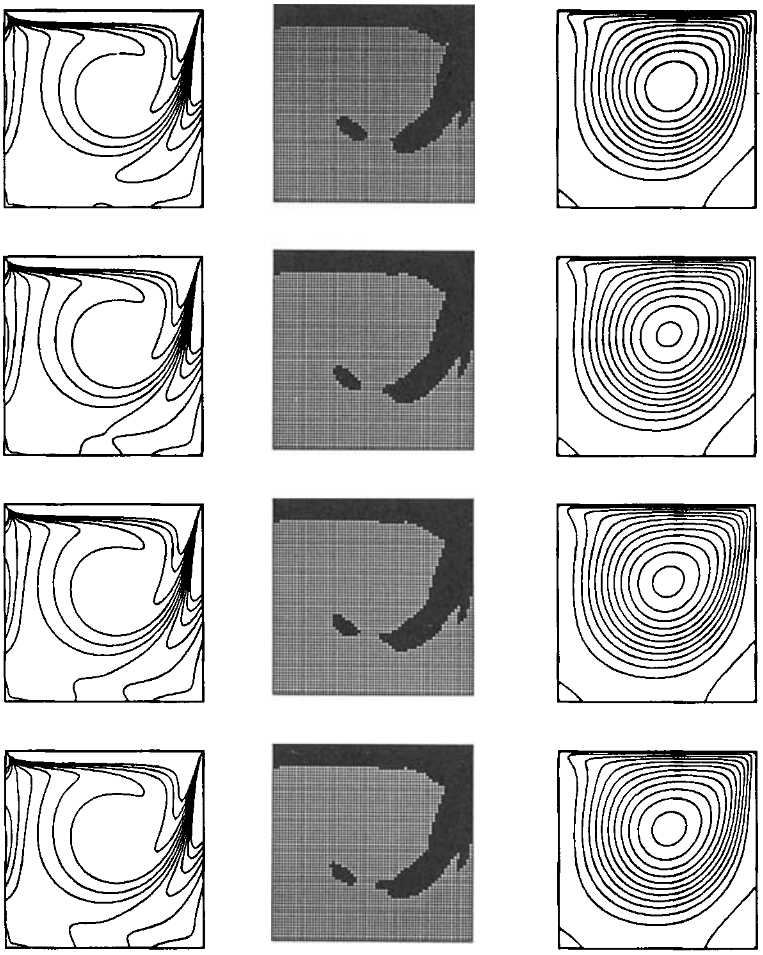


Figure 3. Driven cavity flow at Reynolds number 400: solution obtained by the AIE method; vorticity, distribution of the implicit elements, and streamfunction at $t = 11{\cdot}72$, $14{\cdot}06$, $16{\cdot}41$ and $18{\cdot}75$

velocity. At the downstream boundary the normal derivative of both the vorticity and streamfunction is specified to be zero. The Reynolds number based on the freestream velocity and the cylinder diameter is 100.

The symmetric solutions at $t = 5$ and $t = 50$ are shown in Figures 5 and 6. The steady-state solution is achieved long before $t = 50$. At $t = 51$, to break the symmetry an external perturbation is applied for a duration of only one time step. Figure 7 shows the solution at $t = 80$, at which time the asymmetry becomes quite visible. The solutions at $t = 100$ and $t = 200$ are shown in Figures 8 and 9.

Flow past two flat plates

Analysis of wakes behind flat plates has various applications, including the wake of a ribbon parachute and airflow through the slotted dive brakes of an aircraft.[33, 34] The problem of wake interference behind two flat plates was examined experimentally and numerically by Hayashi

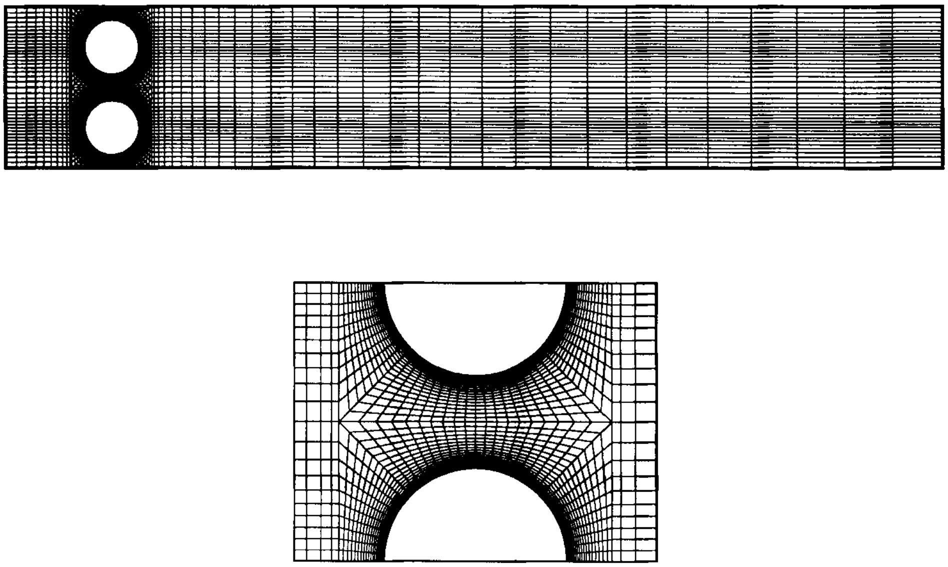


Figure 4. Flow past a bank of cylinders at Reynolds number 100: the finite element mesh with 4800 elements and 4979 nodes

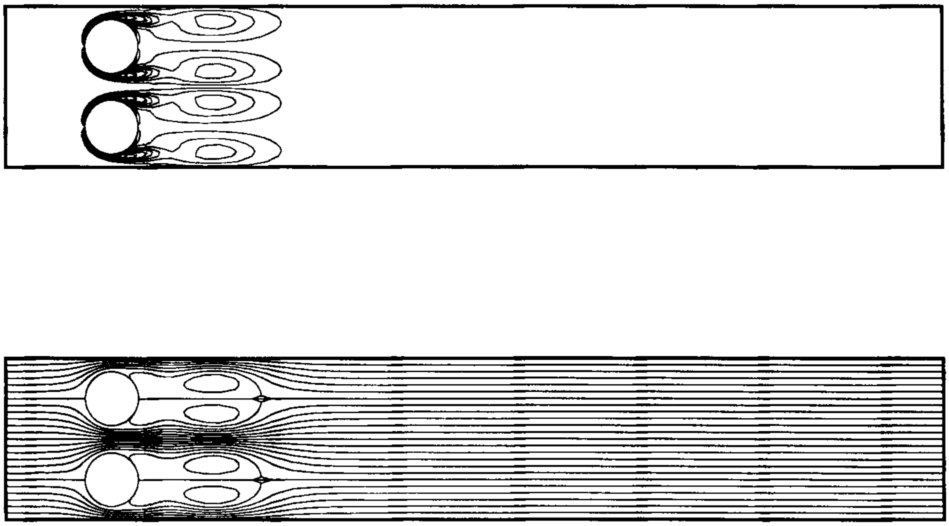


Figure 5. Flow past a bank of cylinders at Reynolds number 100: vorticity and streamfunction at $t = 5$

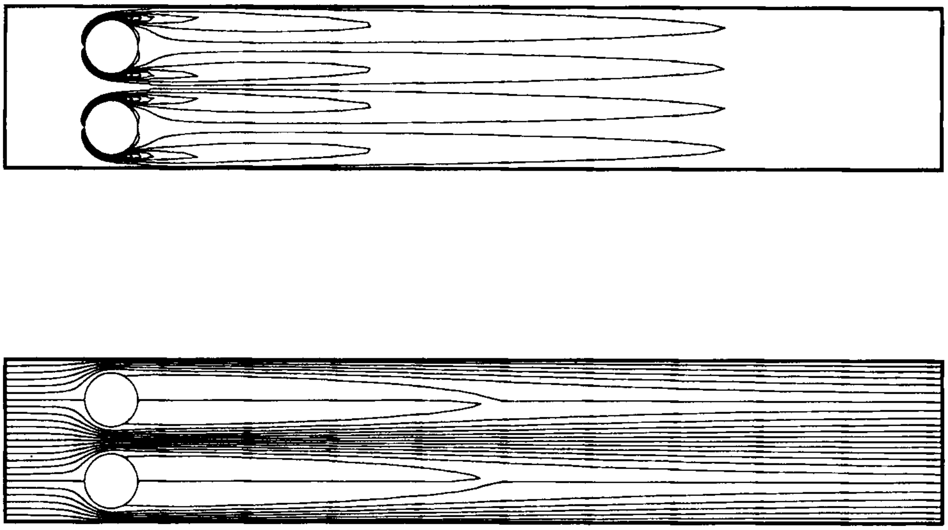


Figure 6. Flow past a bank of cylinders at Reynolds number 100: vorticity and streamfunction at $t = 50$

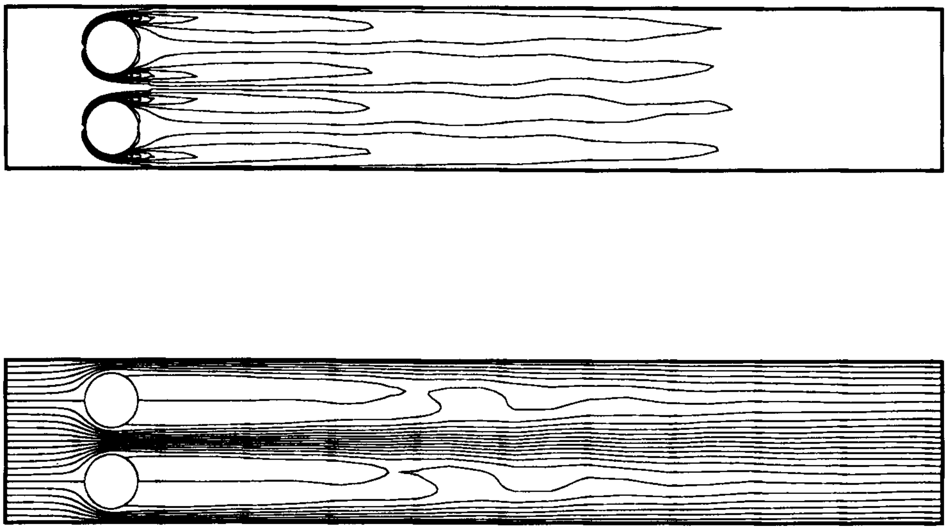


Figure 7. Flow past a bank of cylinders at Reynolds number 100: vorticity and streamfunction at $t = 80$

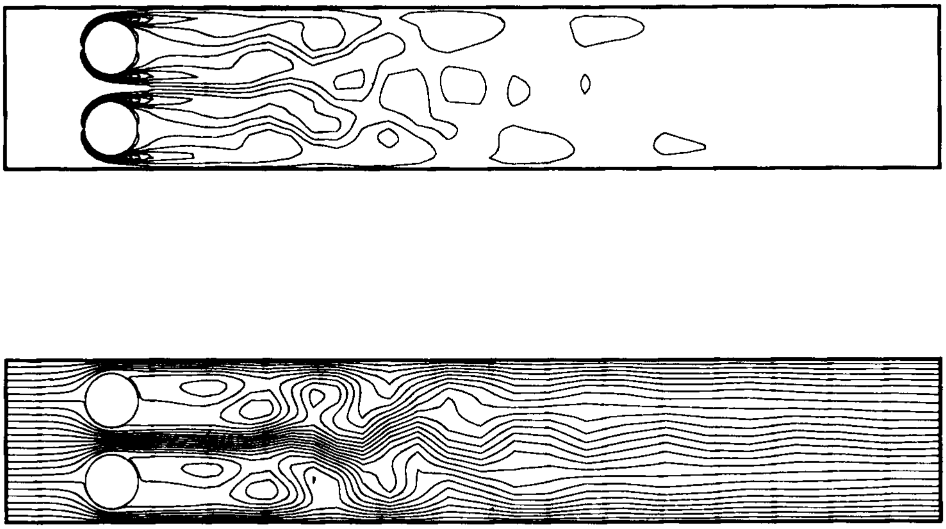


Figure 8. Flow past a bank of cylinders at Reynolds number 100: vorticity and streamfunction at $t = 100$

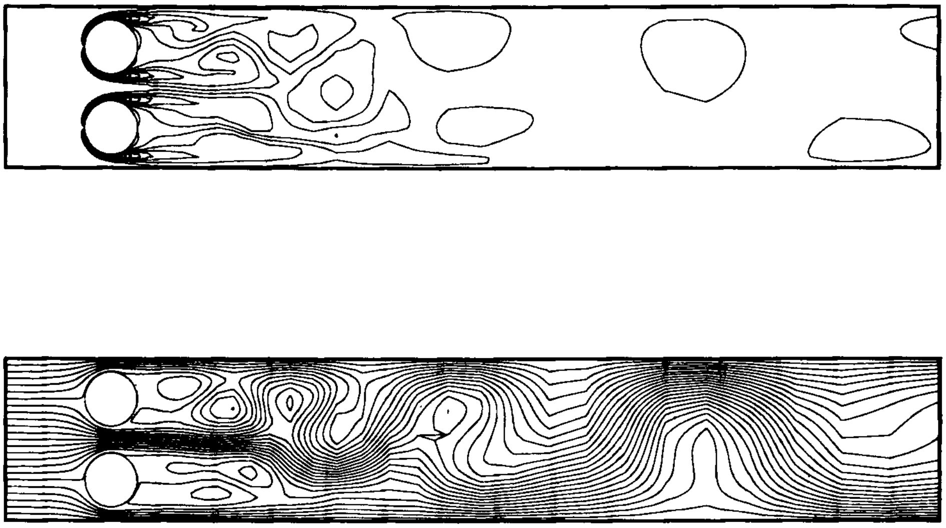


Figure 9. Flow past a bank of cylinders at Reynolds number 100: vorticity and streamfunction at $t = 200$

et al.[35, 36] The numerical results presented in References 35 and 36 are obtained by finite difference calculations. The experimental flow visualizations performed by Higuchi *et al.*[33, 34] include the single- and double-plate configurations at relatively high Reynolds numbers. We borrowed two flow visualization pictures (see Figure 10) from Reference 33 only to give an idea about the physical problem.

In our test we consider a two-plate unit. The length of each plate is 1·0 and the gap between the plates is 0·5. The plate thickness is 0·25. A uniform freestream velocity with a magnitude of 1·0

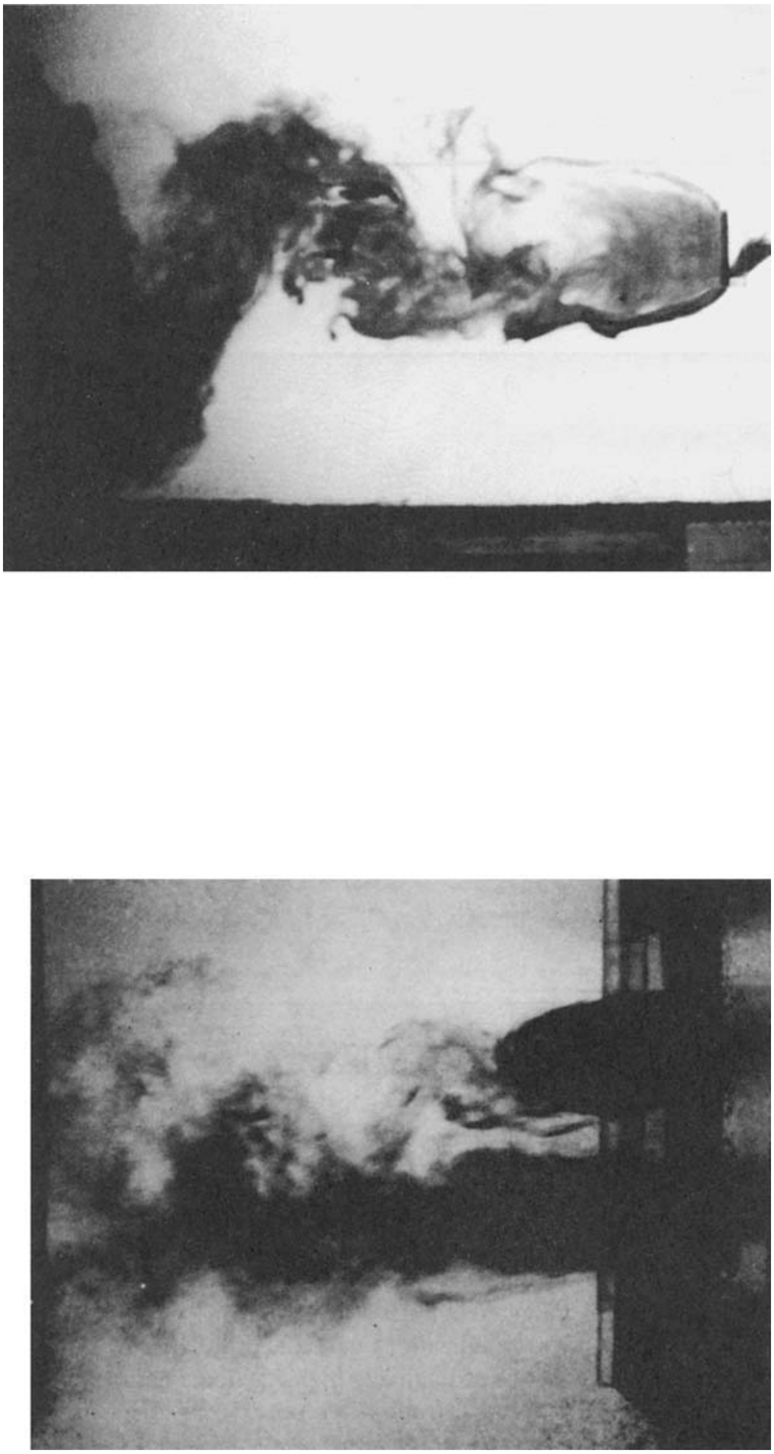


Figure 10. Flow visualizations in water for a single plate and for two plates (with gap 1·0) at Reynolds number 1400 (courtesy of H. Higuchi and H. J. Kim[33])

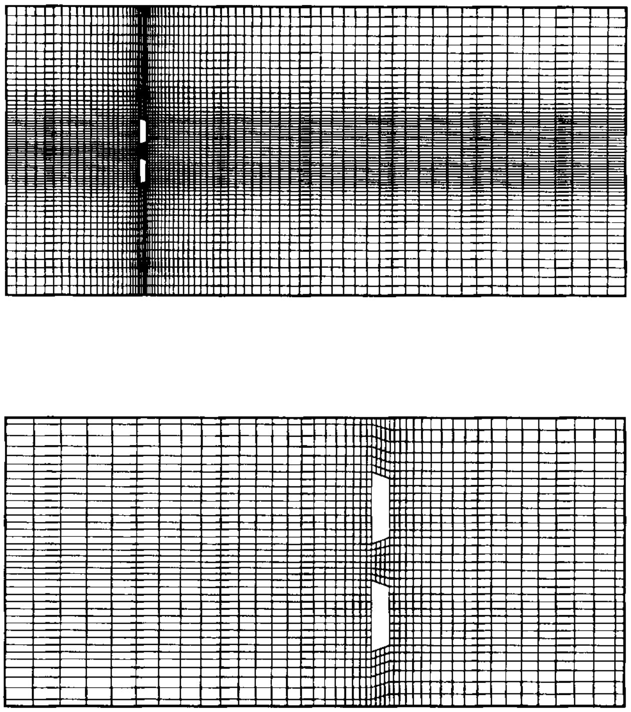


Figure 11. Flow past two plates at Reynolds number 50: the finite element mesh with 4374 elements and 4534 nodes

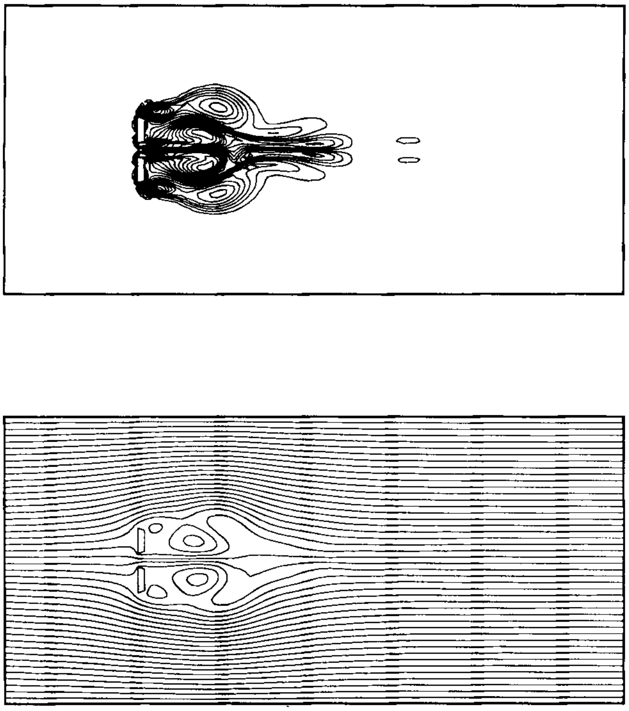


Figure 12. Flow past two plates at Reynolds number 50: vorticity and streamfunction at $t = 10{\cdot}0$

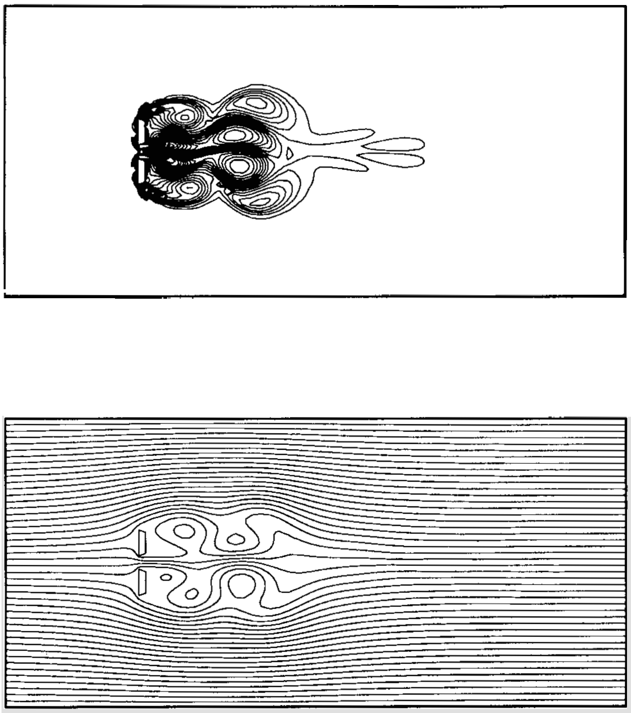


Figure 13. Flow past two plates at Reynolds number 50: vorticity and streamfunction at $t=12{\cdot}5$

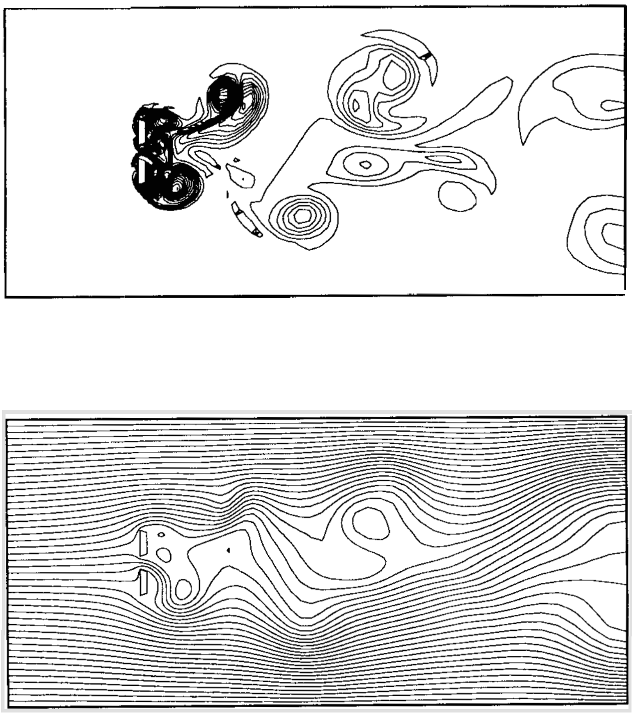


Figure 14. Flow past two plates at Reynolds number 50: vorticity and streamfunction at $t=145{\cdot}0$

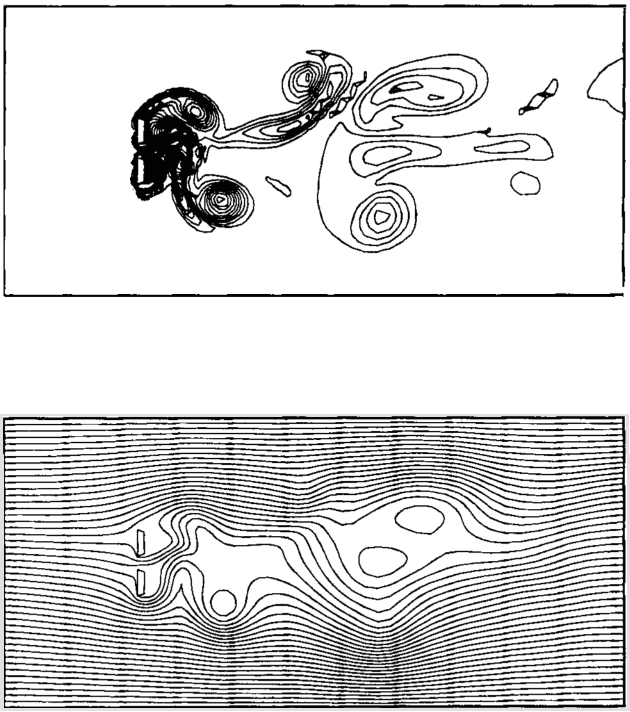


Figure 15. Flow past two plates at Reynolds number 50: vorticity and streamfunction at $t = 147{\cdot}5$

results in a Reynolds number of 50 based on the gap length. A non-uniform mesh consisting of 4374 elements and 4534 nodes was employed (see Figure 11).

Initially the flowfield develops symmetrically as shown in Figure 12. Transition to an asymmetric flow pattern occurs at $t \approx 12$. Following this symmetry breaking, vortex shedding occurs (see Figures 13–15). These preliminary results compare well qualitatively with the results obtained by Hayashi *et al.*[35] on an 11,528-node grid.

7. CONCLUDING REMARKS

We have presented a review of our solution techniques for the vorticity–streamfunction formulation of two-dimensional incompressible flows. We have emphasized the derivation of the proper finite element formulations for multiply connected domains. These formulations cover both the viscous and inviscid cases. In all finite element formulations corresponding to the vorticity transport equation we employed the streamline upwind/Petrov–Galerkin procedure. This procedure minimizes the spurious oscillations encountered in convection-dominated problems, yet introduces minimal numerical diffusion.

To make our computations efficient we use two solution strategies: the adaptive implicit–explicit (AIE) scheme and the grouped element-by-element (GEBE) iteration method. Such strategies become crucial for large-scale computations.

We have performed several numerical tests. In all cases the results obtained compared well with the previously published results. The convergence study (based on successive mesh refinement) for one of the test problems shows that the formulations presented are accurate and reliable. The benchmark computations indicate that the AIE and GEBE strategies result in substantial reductions in the computational cost involved.

ACKNOWLEDGEMENTS

This research was sponsored by NASA–Johnson Space Center under contract NAS-9-17892 and by NSF under grant MSM-8796352.

REFERENCES

1. T. E. Tezduyar, R. Glowinski and J. Liou, 'Petrov–Galerkin methods on multiply-connected domains for the vorticity–streamfunction formulation of the incompressible Navier–Stokes equations', *Int. j. numer. methods fluids*, **8**, 1269–1290 (1988).
2.' A. Campion-Renson and M. J. Crochet, 'On the stream function–vorticity finite element solutions of Navier–Stokes equations', *Int. j. numer. methods eng.* **12**, 1809–1818 (1978).
3. R. Glowinski and O. Pironneau, 'Numerical methods for the first biharmonic equation and for the two-dimensional stokes problem', *SIAM Rev.* **21**, 167–212 (1979).
4. M. O. Bristeau, R. Glowinski and J. Periaux, 'Numerical methods for the Navier–Stokes equations: applications to the simulation of compressible and incompressible viscous flows', *Computer Physics Report, University of Houston, Research Report UH/MD-*4, *February* 1987.
5. M. F. Peeters, W. G. Habashi and E. G. Dueck, 'Finite element stream function–vorticity solutions of the incompressible Navier–*Stokes equations*', *Int. j. numer. methods fluids*, **7**, 17–27 (1987).
6. V. Girault and P. A. Raviart, *Finite Element Approximation of the Navier–Stokes Equations*, Springer-Verlag, New York, 1979.
7. A. Mizukami, 'Short communications: a stream function–vorticity finite element formulation for Navier–Stokes equations in multi-connected domain', *Int. j. numer. methods eng.*, **19**, 1403–1420 (1983).
8. M. D. Gunzburger and R. A. Nicolaides, 'Algorithmic and theoretical results on computation of incompressible viscous flows by finite element methods', *Comput. Fluids*, **13**, 361–373 (1985).
9. T. E. Tezduyar, 'Finite element formulation for the vorticity–stream function form of the incompressible Euler equations on multiply-connected domains', *Computer Methods in Applied Mechanics and Engineering*, UMSI *88/94*, University of Minnesota Supercomputer Institute, October 1988.
10. P. M. Gresho and R. L. Sani, 'On pressure boundary conditions for the incompressible Navier–Stokes equations', R. H. Gallagher, R. Glowinski, P. M. Gresho, J. T. Oden and O. C. Zienkiewicz (eds), *Finite Elements in Fluids, Vol. 7*, Wiley, New York, 1987, pp. 123–157.
11. P. M. Gresho and S. T. Chan, 'Semi-consistent mass matrix techniques for solving the incompressible Navier–Stokes equations', *Lawrence Livermore National Laboratory, Preprint UCRL-99503*, 1988.
12. A. N. Brooks and T. J. R. Hughes, 'Streamline upwind/Petrov–Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier–Stokes equations', *Comput. Methods Appl. Mech. Eng.* **32**, 199–259 (1982).
13. T. E. Tezduyar and T. J. R. Hughes, 'Finite element formulations for convection dominated flows with particular emphasis on the compressible Euler equations', *Proc. AIAA 21st Aerospace Science Meeting*, Reno, NV, January 1983, AIAA Paper 83–0125.
14. T. J. R. Hughes and W. K. Liu. 'Implicit–explicit finite elements in transient analysis: stability theory', *J. Appl. Mech.* **45**, 371–374 (1978).
15. T. E. Tezduyar and J. Liou, 'Element-by-element and implicit-explicit finite element formulations for computational fluid dynamics, in R. Glowinski, G. H. Golub, G. A. Meurant and J. Periaux (eds), *Proc. 1st Int. Symp. on Domain Decomposition Methods for Partial Differential Equations*, SIAM, Philadelphia, PA, 1988, pp. 281–300.
16. T. E. Tezduyar and J. Liou, 'Adaptive implicit–explicit finite element algorithms for fluid mechanics problems', in T. E. Tezduyar and T. J. R. Hughes (eds), *Recent Developments in Computational Fluid Dynamics, AMD Vol. 95*, ASME, New York, 1988, pp. 163–184.
17. T. J. R. Hughes, J. Winget, I. Levit and T. E. Tezduyar, 'New alternating direction procedures in finite element analysis based upon EBE approximation factorizations', in S. N. Atluri and N. Perrone (eds), *Computer Methods for Nonlinear Solids and Mechanics, AMD Vol. 54*, ASME, New York, 1983, pp. 75–110.
18. T. E. Tezduyar, J. Liou, T. Nguyen and S. Poole, 'Adaptive implicit–explicit and parallel element-by-element iteration schemes', in T. F. Chan, R. Glowinski, J. Periaux and O. B. Widlund (eds), *Domain Decomposition Methods*, SIAM, Philadelphia, PA, 1989, pp. 443–463.
19. T. J. R. Hughes and R. M. Ferencz, 'Fully vectorized EBE preconditioners for nonlinear solid mechanics: applications to large-scale three-dimensional continuum, shell and contact/impact problems', in R. Glowinski, G. H. Golub, G. A. Meurant and J. Periaux (eds), *Proc. 1st Int. Symp. on Domain Decomposition Methods for Partial Differential Equations*, SIAM, Philadelphia, PA, 1988, pp. 261–280.

20. Y. Saad and M. H. Schultz, GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems', *Research Report YALEU/DCS/RR-254*, 1983.
21. T. J. R. Hughes and F. Shakib, private communications, 1988.
22. R. Glowinski, Q. V. Dinh and J. Periaux, 'Domain decomposition methods for nonlinear problems in fluid dynamics', *Comput. Methods Appl. Mech. Eng.* **40**, 27–109 (1983).
23. T. E. Tezduyar and J. Liou, 'Grouped element-by-element iteration schemes for incompressible flow computations', *Computer Physics Communications*, *UMSI 88/95*, University of Minnesota Supercomputer Institute, October 1988.
24. T. J. R. Hughes, 'Recent progress in the development and understanding of SUPG methods with special reference to the compressible Euler and Navier–Stokes equations', in R. H. Gallagher, R. Glowinski, P. M. Gresho, J. T. Oden and O. C. Zienkiewicz (eds), *Finite Elements in Fluids*, *Vol. 7*, Wiley, New York, 1987, pp. 273–287.
25. G. F. Carey and J. T. Oden, *Finite Elements: Computational Aspects*, *Vol. 3*, Prentice-Hall, Englewood Cliffs, NJ, 1984.
26. R. Glowinski, *Numerical Methods for Nonlinear Variational Problems*, Springer-Verlag, New York, 1984.
27. M. D. Olson and S. Y. Tuann, 'Primitive variables versus stream function finite element solutions of the Navier–Stokes equations', *Finite Element Methods in Flow Problems*, S. Margherita Ligure, 1976, pp. 57–68.
28. S. Ozawa, 'Numerical studies of steady flow in a two dimensional square cavity at high Reynolds numbers', *J. Phys. Soc. Japan*, **38**, 889–895 (1975).
29. O. C. Burggraf, 'Analytical and numerical studies of the structure of steady separated flows,' *J. Fluid Mech.*, **24**, 113–151 (1966).
30. P. Singh, P. Caussignac, A. Fortes, D. D. Joseph and T. Lundgren, 'Stability of Periodic arrays of cylinders across the stream by direct simulation of steady flow', *J. Fluid Mech.*, in the press.
31. H. Schlichting, *Boundary-Layer Theory*, 7th edn., McGraw-Hill, New York, 1979, pp. 744–745.
32. H. J. Kim and P. A. Durbin, 'Investigation of the flow between a pair of circular cylinders in the flopping regime', *J. Fluid Mech.*, **196**, 431–448 (1988).
33. H. Higuchi and H. J. Kim, 'Flow visualizations and mean pressure and velocity measurements behind the grid models', *Sandia Report 48–2215*, Department of Aerospace Engineering and Mechanics, University of Minnesota, April 1986.
34. H. Higuchi and F. Takahashi, 'Flow past two-dimensional parachute models,' *J. Aircraft*; also *AIAA Paper 88–2524*, 1988.
35. M. Hayashi, A. Sakurai and Y. Ohya, 'Numerical and experimental studies of wake interference of two normal flat plates at low Reynolds numbers', *Mem. Fac. Eng., Kyushu Univ.*, **43**, 165–177 (1983).
36. M. Hayashi, A. Sakurai and Y. Ohya, 'Wake interference of a row of normal flat plates arranged side by side', *J. Fluid Mech.*, **164**, 1–25 (1986).