JOURNAL OF COMPUTATIONAL PHYSICS **80**, 72–97 (1989)

Vorticity Boundary Conditions and Boundary Vorticity Generation for Two-Dimensional Viscous Incompressible Flows*

CHRISTOPHER R. ANDERSON

Department of Mathematics, University of California, Los Angeles, California 90024

Received April 21, 1987; revised March 4, 1988

In this paper we present boundary conditions appropriate for the vorticity formulation of the two-dimensional incompressible viscous Navier–Stokes equations. These boundary conditions are incorporated into a finite difference scheme and the resulting method is of the "vorticity creation" type; i.e., vorticity is generated at the boundary to ensure that the tangential velocity boundary condition is satisfied. The results of computations with this finite difference method are presented for flow past a circular cylinder. A difference scheme and computational results for a model problem, the Prandtl boundary layer equations describing flow over a semi-infinite flat plate are also presented. © 1989 Academic Press, Inc.

INTRODUCTION

In this paper we present results concerning boundary conditions for the vorticity form of the time-dependent 2-D Navier–Stokes equations. Our primary results are a derivation of appropriate boundary conditions for the vorticity and a description of a finite difference scheme which incorporates these boundary conditions. The finite difference method is used to calculate flow past a circular cylinder. This problem was selected so that our computational results could be judged by a comparison with existing computational and experimental results. The problem of flow past a cylinder is also of interest because of the difficulty which is introduced by the fact that the fluid domain is infinite. In this paper we present a technique for overcoming this difficulty. We also discuss boundary conditions for the vorticity form of the Prandtl boundary layer equations and present a finite difference method for computing solutions of them. We calculate flow past a semi-infinite flat plate and compare the results with the Blasius solution. The reason for considering the Prandtl boundary layer equations is primarily to aid in the exposition of the ideas and techniques used in working with the vorticity form of the Navier–Stokes equations. However, there is independent interest in these equations. We are able to

* Partially supported by NSF Mathematical Sciences Postdoctoral Research Fellowship, MCS 83-11685 and ONR Contracts N00014-82-K-0335 and N00014-86-K-0727.

exhibit a finite difference scheme in which the boundary condition implementation is a direct analog of that used by Chorin in his vortex sheet algorithm [5–7]. We thus find that a technique which has appeared to be of value only for vortex blob methods has an analog which may be useful within the context of finite difference schemes. Our computational results on both the problem of flow past a cylinder and flow past a flat plate indicate that the boundary conditions which we present here can be incorporated into finite difference schemes in such a way that the evolution of vorticity in the fluid and on the boundary is accurately predicted.

Many different techniques for overcoming the difficulty of vorticity boundary conditions have been employed in numerical methods. There appear to be three classes of schemes. The first class are those schemes which exploit the relationship between the vorticity and the stream function on the boundary. (See among others [12, 15].) The second class are those schemes proposed by Quartapelle and Quartapelle and Valz-Gris [16, 17]. In these papers it is shown that in order for the boundary conditions on the velocity to be satisfied, the vorticity should evolve subject to an integral constraint. The schemes presented in the papers essentially consist of ensuring at each time step that the approximate vorticity satisfies this constraint. The third class of schemes are those employed in vortex blob methods and were introduced by Chorin [4–7]. In such methods vorticity is introduced on the boundary in a way which ensures that the boundary conditions on both components of the velocity field are approximately satisfied. Judging from the fact that one can obtain reasonable results with each type of scheme, we conclude, that, although these schemes appear to differ greatly, they must be implicitly satisfying some specific vorticity boundary conditions. One of the motivations of this work was to find a representation for the vorticity boundary conditions which would shed some light on this issue.

The derivation of our vorticity boundary conditions is based on two key observations. The first observation, one contained in the work of Quartapelle and Valz-Gris, is that the boundary conditions on the velocity induce a constraint on the vorticity. The second observation is that the constraint can be ensured by requiring that it be satisfied at an initial time and by requiring that the time derivative of this constraint vanish. The requirement on the time derivative leads to explicit boundary conditions for the vorticity of an integral-differential nature. The derivation of numerical methods which use these boundary conditions can be accomplished by mimicing the derivation of the continuous boundary conditions with discrete operators. This procedure is used to construct a scheme for computing flow past a circular cylinder. Since the vorticity boundary conditions ensure that the vorticity will evolve subject to the appropriate constraint, our numerical method is similar to that which would be obtained by an implementation of Quartapelle and Valz-Gris's technique. However, in our implementation it is immediately apparent that vorticity creation occurs on the boundary as a direct consequence of the requirement that the time derivative of the constraint vanish. Thus, the method exhibits vorticity creation on the boundary and in this respect is similar to those proposed by Chorin. (In the case of the boundary layer equations, there is a direct analogy.)

This suggests a connection between Chorin's techniques and the technique of evolving the vorticity subject to a constraint. We have not been able to clearly see a connection between the schemes which incorporate our boundary conditions and those which might be obtained using the ideas in [12, 15]. We are confident that future investigations will show that the methods are closely related. Other than provide a demonstration of accuracy by a comparison of our computational results with other schemes and experiments, we have not performed an error analysis of our scheme. We are hopeful that our representation of the vorticity boundary conditions will facilitate such an analysis. This representation may also be useful in the analysis of previous numerical schemes-particularly those due to Chorin.

In Section 1 we discuss the origin of the problem with vorticity boundary conditions and then present our derivation of appropriate boundary conditions for the vorticity. Before discussing the numerical implementation of these boundary conditions we discuss, in Section 2, boundary conditions and a finite difference scheme for the Prandtl boundary layer equations. In the remaining two sections we present a finite difference scheme for computing flow past a circular cylinder and discuss results which were obtained with this scheme.

1. Derivation of the Vorticity Boundary Conditions

The equations we are concerned with are the two-dimensional incompressible Navier–Stokes equations,

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\,\mathbf{u} = -\nabla P + \nu\,\Delta\mathbf{u}, \tag{1.1}$$

$$\nabla\cdot\mathbf{u} = 0, \tag{1.2}$$

$$\mathbf{u}(\alpha) = \mathbf{B}(\alpha) \qquad \text{for} \quad \alpha\in\partial\Omega. \tag{1.3}$$

Here $\mathbf{u}$ is the velocity, P the pressure, and ν the viscosity. We assume the fluid is of constant density equal to one. Ω is the region in R^2 with boundary $\partial\Omega$. $\mathbf{B}(\alpha)$ is the velocity on the boundary (often taken to be identically zero).

In the vorticity formulation of (1.1)–(1.3) the velocity field $\mathbf{u}$ is taken to be the sum of a velocity field due to an irrotational flow and a velocity field due to a rotational flow. Let ϕ be the potential for the irrotational flow and Ψ the stream function for the rotational flow. We assume ϕ satisfies

$$\Delta\phi = 0, \qquad \frac{\partial\phi}{\partial\mathbf{n}} = \mathbf{B}\cdot\mathbf{n} \qquad \text{on } \partial\Omega.$$

Here $\mathbf{n}$ denotes the normal to the boundary.

Let ω be the vorticity. By taking the curl of Eq. (1.1) one obtains the following equation for the transport of vorticity:

$$\frac{\partial\omega}{\partial t} + \mathbf{U}\cdot\nabla\omega = \nu\,\Delta\omega. \tag{1.4}$$

Here

$$\mathbf{U} = (\phi_x, \phi_y) + (\Psi_y, -\Psi_x) \tag{1.5}$$

and Ψ, the stream function, is determined from the equation

$$\begin{aligned} \Delta\Psi &= -\omega, \\ \Psi &= 0 \quad \text{on} \quad \partial\Omega; \end{aligned} \tag{1.6}$$

The boundary conditions given above for ϕ and Ψ guarantee that the normal velocity boundary condition is satisfied. To satisfy the tangential velocity condition we must also have

$$\frac{\partial\Psi}{\partial\mathbf{n}} = b(\alpha) \qquad \text{for} \quad \alpha \in \partial\Omega \tag{1.7}$$

where $b(\alpha) = -[\mathbf{B}\cdot\boldsymbol{\tau} - (\phi_x, \phi_y)\cdot\boldsymbol{\tau}]$ and $\boldsymbol{\tau}$ is the unit tangent to $\partial\Omega$.

In the translation of the equations from primitive variables (1.1)–(1.3) to the vorticity form (1.4)–(1.7), boundary conditions for the vorticity are not obtained. Moreover, it appears that to many boundary conditions are given on the stream function. (Both those in (1.6) and (1.7) must be satisfied.) However, a bit of thought reveals that the freedom in choosing the vorticity boundary conditions occurs precisely because the stream function is over determined. For, unless there is some mechanism for manipulating ω in the interior of the domain, the problem which defines Ψ will not, in general, be well posed. It is therefore not surprising that a common thread which runs through all numerical implementations is the manipulation of the vorticity, usually near the boundary, in a way which ensures that both (1.6) and (1.7) are satisfied. Of course, this over determination appears only because of our desire to first compute the vorticity and then secondly the stream function. The boundary conditions (1.6) and (1.7) are quite appropriate when considering the stream function as the primary variable i.e. these boundary conditions are consistent with those necessary for the biharmonic operator occurring in (1.4).

Quartapelle and Quartapelle and Valz-Gris [16, 17] show that one can close the equations for the vorticity by adjoining a constraint which ensures that (1.6) and (1.7) are simultaneously satisfied. The basis for their methods is the following theorem given in [16]:

THEOREM. *A function ξ in Ω is such that $-\xi = \Delta\Psi$, with $\Psi|_\alpha = a(\alpha)$ and $\partial\Psi/\partial\mathbf{n} = b(\alpha)$, if and only if*

$$\int_\Omega \xi\eta \, d\Omega = \int_{\partial\Omega} \left(a(\alpha)\frac{\partial\eta}{\partial\mathbf{n}} - \eta b(\alpha) \right) d\alpha \tag{1.8}$$

for all functions η such that $\Delta\eta = 0$ in Ω.

The numerical methods presented in [16, 17] for solving (1.4)–(1.7) consist of requiring that the approximate solution satisfy (1.8) at every timestep. One of the drawbacks of these methods is the necessity of computing and storing the discrete versions of the harmonic functions used in the implementation of (1.8). In later work [18] it is shown that this problem can be avoided and that the implementation of (1.8) can be reduced to solving a boundary integral equation. That such a reduction is possible suggests that by enforcing the projection condition (1.8) they are implicitly implementing some boundary conditions for the vorticity.

In our derivation of an explicit representation for the boundary conditions we assume the viewpoint, suggested by the work of Quartapelle and Quartapelle and Valz-Gris, that one should consider the evalution of the vorticity as a constrained evolution. However, instead of finding the constraint on the vorticity and adjoining it to the equations as these authors do, we proceed by finding conditions which ensure that as the vorticity evolves the constraint will automatically be satisfied. To do this, we first express the constraint in a different form than that given by (1.8).

We assume that the boundary condition (1.6) is used to determine Ψ and consider the "extra" boundary condition (1.7) as a constraint on the vorticity. To express this constraint we use the Green's function $G(x, s)$ for the domain Ω. This function is the solution of

$$\Delta G(x, s) = \delta(s), \qquad x \in \Omega, \qquad G(\alpha, s) = 0, \qquad \alpha \in \partial\Omega$$

where $\delta(s)$ is a Dirac delta function located at $s \in \Omega$. We have

$$\Psi(x) = -\int_\Omega G(x, s)\, \omega(s, t)\, ds$$

so that condition (1.7) can be written as

$$\frac{\partial}{\partial \mathbf{n}} \int_\Omega G(\alpha, s)\, \omega(s, t)\, ds = -b(\alpha) \qquad \text{for} \quad \alpha \in \partial\Omega. \tag{1.9}$$

To find boundary conditions which ensure that solutions of (1.4) induce a stream function which satisfies both (1.6) and (1.7), we find boundary conditions which guarantee that the derivative with respect to time of the constraint (1.9) vanishes. We require

$$\frac{\partial}{\partial t}\left(\frac{\partial}{\partial \mathbf{n}} \int_\Omega G(\alpha, s)\, \omega(s, t)\, ds + b(\alpha)\right) = 0 \qquad \text{for} \quad \alpha \in \partial\Omega.$$

If we use the fact that ω is a solution of (1.4) and the Green's identity

$$\omega(x) = \int_\Omega G(x, s)\, \Delta\omega(s, t)\, ds - \int_{\partial\Omega} \frac{\partial G}{\partial \mathbf{n}}(x, \alpha)\, \omega(\alpha, t)\, d\alpha,$$

we find

$$\frac{\partial}{\partial t}\frac{\partial}{\partial \mathbf{n}}\int_\Omega G(\alpha, s)\,\omega(s, t)\,ds = 0$$

$$\Leftrightarrow \frac{\partial}{\partial \mathbf{n}}\int_\Omega G(\alpha, s)\frac{\partial \omega}{\partial t}(s, t)\,ds = 0$$

$$\Leftrightarrow \frac{\partial}{\partial \mathbf{n}}\int_\Omega G(\alpha, s)(-\mathbf{U}\cdot\nabla\omega(s, t) + \nu\,\Delta\omega(s, t))\,ds = 0$$

$$\Leftrightarrow \nu\frac{\partial}{\partial \mathbf{n}}\int_\Omega G(\alpha, s)\,\Delta\omega(s, t)\,ds = \frac{\partial}{\partial \mathbf{n}}\int_\Omega G(\alpha, s)[\mathbf{U}\cdot\nabla\omega(s, t)]\,ds$$

$$\Leftrightarrow \frac{\partial \omega}{\partial \mathbf{n}} + \frac{\partial}{\partial \mathbf{n}}\int_{\partial\Omega}\frac{\partial G}{\partial \mathbf{n}}(\alpha, \gamma)\,\omega(\gamma, t)\,d\gamma = \frac{1}{\nu}\frac{\partial}{\partial \mathbf{n}}\int_\Omega G(\alpha, s)[\mathbf{U}\cdot\nabla\omega(s, t)]\,ds. \tag{1.10}$$

This is an integral-differential equation which determines the boundary values. Values satisfying (1.10) and used as data for (1.4) will ensure that the time derivative of (1.9) will be zero. If we assume that the initial vorticity satisfies (1.9), then we will have ensured that this constraint, and hence (1.7), is satisfied for all time.

2. Results for the Prandtl Boundary Layer Equations

In this section we apply the ideas of the previous section to the Prandtl boundary layer equations. This set of equations, which we shall use to describe laminar flow over a half-infinite flat plate, is a simpler set of equations than the 2-D Navier–Stokes equations, and yet when expressed in the vorticity formulation possesses the same problems with vorticity boundary conditions. We shall derive the continuous boundary conditions for the vorticity, and then implement a numerical method using these boundary conditions. Results of computations with this numerical method will be presented. A similar discussion concerning the implementation of numerical methods for the 2-D Navier–Stokes equations will be presented in the next section.

In the vorticity form, the Prandtl boundary equations describing flow over a half-infinite flat plate are

$$\frac{\partial \omega}{\partial t} + \mathbf{u}\cdot\nabla\omega = \nu\frac{\partial^2 \omega}{\partial y^2}, \tag{2.1}$$

$$\omega = -\frac{\partial u}{\partial y}, \tag{2.2}$$

$$u = u_\infty + \int_y^\infty \omega(x, s, t)\,ds, \tag{2.3}$$

$$v = -\int_0^y \frac{\partial u(x, s, t)}{\partial x}\,ds, \tag{2.4}$$

with boundary conditions

$$u = v = 0 \quad \text{at} \quad y = 0, \quad x \geqslant 0, \tag{2.5}$$

$$u = u_\infty \quad \text{at} \quad y = +\infty \qquad \text{and} \qquad (u, v) = (u_\infty, 0) \quad \text{for} \quad x < 0. \tag{2.6}$$

Here (u, v) is the velocity and ω the vorticity. The domain is the quarter plane $0 \leqslant x \leqslant \infty$ and $0 \leqslant y \leqslant \infty$.

As is the case with the Navier–Stokes equations, when one reconstructs the velocity field from the vorticity using (2.3), it is not automatic that the boundary condition on u in (2.5) will be satisfied. Similarly, boundary conditions necessary to close Eq. (2.1) are not given. We proceed, as in Section 1, to derive boundary conditions by expressing the condition on u in (2.5) as a constraint on the vorticity and then setting the time derivative of this constraint equal to zero. Using (2.3), the constraint on the vorticity is

$$u_\infty + \int_0^\infty \omega(x, s, t)\, ds = 0, \qquad x \geqslant 0 \quad \text{and} \quad t \geqslant 0. \tag{2.7}$$

Thus we require

$$\begin{aligned} &\frac{\partial}{\partial t} \int_0^\infty \omega(x, s, t)\, ds = 0 \\ &\Leftrightarrow \int_0^\infty \frac{\partial \omega}{\partial t}(x, s, t)\, ds = 0 \\ &\Leftrightarrow \int_0^\infty -\mathbf{u} \cdot \nabla \omega + v \frac{\partial^2}{\partial y^2}\, ds = 0 \\ &\Leftrightarrow -\frac{\partial \omega}{\partial y}\bigg|_{y=0} = \frac{1}{v} \int_0^\infty [\mathbf{u} \cdot \nabla \omega]\, ds. \end{aligned} \tag{2.8}$$

Now (2.8) is the desired boundary condition and when added to (2.1)–(2.4) closes the equations. We now consider a numerical method for solving these equations which incorporates these boundary conditions.

Our computational domain is the rectangular region described by the points (x, y) such that $0 \leqslant x \leqslant x_\infty$ and $0 \leqslant y \leqslant y_\infty$. The mesh we use is rectangular with widths dx and dy in the x and y directions, respectively. The values of the vorticity and velocity are computed at the grid points $(i\,dx, j\,dy)$ and are designated by $\omega_{i,j}$, $u_{i,j}$, and $v_{i,j}$ with $0 \leqslant i \leqslant m$ and $0 \leqslant j \leqslant n$. To approximate (2.1) we use a one-step explicit method (Euler's method) to advance the solution in time, and approximate the advection term $\mathbf{u} \cdot \nabla \omega$ using a second order upwind differencing scheme due to Colella [8]. (We believe the results are relatively independent of the advection scheme used.) The second derivative term was approximated by central differences.

Our scheme is thus expressed by

$$\frac{\omega_{i,j}^{k+1}-\omega_{i,j}^{k}}{\delta t}=-A_{i,j}(\mathbf{u}^k,\omega^k)+D_y^+D_y^-\omega_{i,j}^k \tag{2.9}$$

where $A_{i,j}(\mathbf{u}^k, \omega^k)$ is the second order approximation to the convective term in (2.1) and

$$D_y^+D_y^-\omega_{i,j}^k=\frac{\omega_{i,j+1}^k-2\omega_{i,j}^k+\omega_{i,j-1}^k}{dy^2}. \tag{2.10}$$

To construct u we use the trapezoidal rule to approximate the integral in (2.3). We assume no vorticity above the line $y=y_\infty$, and hence $u=u_\infty$ for points above this line. For the remaining points $(j\leqslant n)$ we use

$$u_{i,j}=u_\infty+\sum_{p=j}^{n}\frac{(\omega_{i,p}+\omega_{i,p+1})}{2}\,dy; \tag{2.11}$$

v is approximated using

$$v_{i,j}=-\sum_{p=0}^{j}\frac{1}{2}\left(\frac{u_{i+1,p}-u_{i-1,p}}{2\,dx}+\frac{u_{i+1,p+1}-u_{i-1,p+1}}{2\,dx}\right)dy, \tag{2.12}$$

i.e., a trapezoidal rule approximation to (2.4) in which central differences are used to approximate the derivatives of u. At points on the left and right computational boundaries second order one-sided differences are used in (2.12) instead of central differences.

It remains to specify the boundary conditions used for the vorticity. At the left edge of the computational boundary, the inflow side, we assumed that there is no vorticity immediately upstream. At the right edge of the computational boundary we assumed outflow boundary conditions, i.e., the vorticity is unspecified at points just outside of the computational domain. At the top and bottom of the domain we specify the normal derivative of ω. This data was incorporated into the difference stencil using the method of fictitious points. For points on such boundaries, the central difference occurring in (2.10) uses a point just outside the computational domain. This point is eliminated by using the normal derivative boundary condition. For example, at points on the plate, the second derivative term is approximated by

$$D_y^+D_y^-\omega_{i,0}=\frac{2\omega_{i,1}-2\omega_{i,0}}{dy^2}-\frac{2}{dy}\left.\frac{\partial\omega}{\partial y}\right|_{i,0}.$$

For the data at the bottom we used

$$\left.\frac{\partial\omega}{\partial y}\right|_{i,0}=\frac{-1}{\nu}\sum_{p=0}^{n}\left[\frac{A_{i,p+1}(\mathbf{u}^k,\omega^k)+A_{i,p}(\mathbf{u}^k,\omega^k)}{2}\right]dy, \tag{2.13}$$

i.e., a discrete analog of (2.8). At the top we used the boundary condition $\partial\omega/\partial y|_{i,n}=0$. This was chosen because it ensures that the boundary condition $u=0$ at $y=0$ is satisfied exactly by the discrete scheme. This fact can be verified by directly calculating the difference in the velocity at the plate induced by ω^{k+1} and ω^k and employing a discrete version of the arguments used to find the conditions that the time derivative of the constraint (2.7) vanish. An upper boundary condition appears in this discrete derivation because the computational domain is finite.

Our computational results correspond to a parameter selection of $\nu=0.025$, $x_\infty=1.0$, $y_\infty=1.0$, and an onset velocity $u_\infty=1.0$. The values of parameters ν, u_∞, x_∞, and y_∞ were selected to ensure that the majority of the vorticity was confined to the region $y \leqslant y_\infty$ when $0 \leqslant x \leqslant x_\infty$. The use of the boundary condition (2.8) assumes that the initial vorticity induces a velocity field which satisfies all of the boundary conditions. The initial distribution of vorticity used,

$$\omega^0_{i,0}=-2u_\infty/dy, \qquad i=0,\ldots,m,$$
$$\omega^0_{i,j}=0, \qquad i=1,\ldots,m, \qquad 0<j\leqslant n,$$

satisfies this assumption. This choice of initial conditions corresponds to impulsively starting the fluid at time $t=0$ with a velocity u_∞.

A steady state solution of the equations is the Blasius solution [3, 19]. This is a solution which is self-similar with a similarity variable $\eta=y(u_\infty/x\nu)^{1/2}$, i.e., the steady state solution (u, v) is given by

$$u(x,y)=u_b(\eta), \qquad v(x,y)=v_b(\eta)$$

where $(u_b(\eta),\ v_b(\eta))$ is the Blasius solution. Our numerical scheme (2.9) with boundary conditions (2.13) was integrated to steady state. The mesh chosen was uniform in each direction and the timestep was chosen so that $\delta t<\min(dy^2/2\nu,\ dx/u_\infty,\ dy/u_\infty)$. No instability was observed for this choice of parameters. In Fig. 2.1 we present a comparison of the Blasius solution and the computed solution at locations $x=0.25$, $x=0.5$ and $x=0.75$ along the plate. (The exact solution used is that given in [19].) Here $dx=dy=0.025$ and $\delta t=0.005$. The difference between the u velocities of each solution is plotted versus the similarity variable η so that each profile can be compared with the others. The velocity profile which is most in error is that near the leading edge of the plate, $x=0.25$. This is to be expected since the number of points resolving the important features of the velocity field is less at this station than at stations further down the plate. However, the maximum relative error of all the profiles is less than 1%. At the station located at $x=0.5$ the maximum error of the tangential velocity was 4.804%, 1.212%, and 0.242% for mesh widths 0.1, 0.05, and 0.025, respectively, and hence the method appeared to be converging at a rate of at least second order with respect to the spatial discretization.

We conclude this section by discussing the relation between a method introduced by Chorin for solving the Prandtl boundary layer equations and an analogous finite difference scheme. In [5], Chorin uses the method of fractional steps and vorticity boundary creation to solve Eqs. (2.1)–(2.5). The discretization he uses is based on

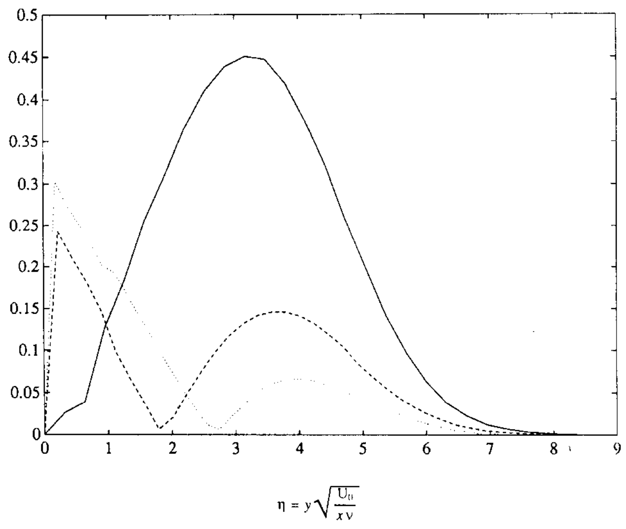

FIG. 2.1. Relative error (in percent) in steady state tangential velocity component for flow past a half infinite flat plate; Error $= (U_{\text{comp}} - U_{\text{Blasius}})/|U_{\text{Blasius}}|$.—, $x = 0.25$; ---, $x = 0.5$; $\cdots$, $x = 0.75$.

computational elements which are segments of vortex sheets. The basic timestep, ignoring the precise implementation details, is as follows: An approximate solution of the inviscid equations is advanced one timestep (Eq. (2.1) without ν). This leads to a vorticity distribution which induces a velocity field which does not satisfy the tangential boundary condition on the plate. If the velocity field at location $i\,dx$ on the plate is u_i, then vortex sheets of cumulative strength $-2u_i$ are created on the boundary. These sheets are then allowed to participate, with the sheets already in the fluid, in a random walk which approximates one step of the evolution of the viscous terms of the equation. If a sheet leaves the computational domain in this latter step, it is deleted. These steps are then repeated to advance the solution in time. (For an elementary discussion of the method see [7].)

Consider now the following fractional step scheme which uses the finite diffeence approximations given previously:

$$\frac{\tilde{\omega}_{i,j}^{k+1} - \omega_{i,j}^{k}}{\delta t} = -A_{i,j}(\mathbf{u}^k, \omega^k), \tag{2.14}$$

$$\frac{\omega_{i,j}^{k+1} - \tilde{\omega}_{i,j}^{k+1}}{\delta t} = \nu D_y^+ D_y^- \tilde{\omega}_{i,j}^{k+1}; \tag{2.15}$$

(2.14) is a discrete approximation to the advection component of the equation, while (2.15) is an approximation to the diffusive component. The normal derivative boundary condition

$$\left.\frac{\partial\tilde{\omega}}{\partial y}\right|_{i,0} = \frac{-1}{\nu}\sum_{p=0}^{n}\left[\frac{A_{i,p+1}(\mathbf{u}^k, \omega^k) + A_{i,p}(\mathbf{u}^k, \omega^k)}{2}\right] dy \tag{2.16}$$

is incorporated into the finite difference stencil occurring in (2.15). The upper boundary condition is $\partial\tilde{\omega}/\partial y|_{i,n}=0$. One can show, as with the explicit scheme given previously, that if the velocity field indiced by ω^k satisfies (2.5) then the velocity induced by ω^{k+1} satisfies (2.5) as well.

The boundary condition (2.16) expresses the fact that there is a flux of vorticity into (out of) the fluid. Our goal is to find the amount of vorticity per timestep that is introduced by this boundary condition so that we may compare this with the amount of vorticity which is introduced at the boundary in Chorin's vortex sheet algorithm.

Given $\tilde{\omega}^{k+1}$, the result of an Euler step using (2.14), we can consider the vorticity ω^{k+1} obtained using (2.15) as being the sum of the solution to two separate problems. The first piece is an approximate solution to the diffusion component of the equation with homogeneous boundary conditions and $\tilde{\omega}^{k+1}$ as initial data. The second piece is an approximate solution of the diffusion component of the equations with normal boundary condition (2.16) but with homogeneous initial data. It is the contribution of this latter piece of the solution which is the amount of vorticity introduced by the boundary condition (2.16). This vorticity $\bar{\omega}$ is that defined by

$$\frac{\bar{\omega}_{i,j}-\omega_{0_{i,j}}}{\delta t}=\nu D_y^+ D_y^- \omega_{0_{i,j}}$$

with initial condition $\omega_{0_{i,j}}=0$ and boundary condition (2.16). Working through the algebra we find that $\bar{\omega}$ is non-zero only at the lower boundary (the points with $j=0$), and is given by

$$\bar{\omega}_{i,0}=\frac{2\,\delta t}{dy}\sum_{p=0}^{n}\left[\frac{A_{i,p+1}(\mathbf{u}^k,\omega^k)+A_{i,p}(\mathbf{u}^k,\omega^k)}{2}\right]dy. \tag{2.17}$$

From (2.14) we have

$$-A_{i,j}=\frac{\tilde{\omega}_{i,j}^{k+1}-\omega_{i,j}^{k}}{\delta t}$$

and using this expression in (2.17) we arrive at

$$\begin{aligned}\bar{\omega}_{i,0}&=\frac{-2}{dy}\sum_{p=0}^{n}\left[\frac{(\tilde{\omega}_{i,p+1}^{k+1}-\omega_{i,p+1}^{k})+(\tilde{\omega}_{i,p}^{k+1}-\omega_{i,p}^{k})}{2}\right]dy\\&=-\frac{2}{dy}\left(u_\infty+\sum_{p=0}^{n}\left[\frac{\tilde{\omega}_{i,p+1}^{k+1}+\tilde{\omega}_{i,p}^{k+1}}{2}\right]dy\right).\end{aligned} \tag{2.18}$$

In this last simplification we assume $\omega_{i,j}^k$ satisfies (2.5) discretely, i.e.,

$$u_\infty+\sum_{p=0}^{n}\frac{(\omega_{i,p+1}^{k}+\omega_{i,p}^{k})}{2}\,dy=0.$$

The expression on the right in Eq. (2.18) is precisely the u velocity at the plate which is induced by $\tilde{\omega}^{k+1}$, i.e., the slip velocity at the plate induced by the vorticity after one step of Euler flow. Thus, again denoting the slip velocity at the point $i\,dx$ as u_i, the amount of vorticity induced on the boundary is $-2u_i/\delta y$. In light of the formula for velocity reconstruction (2.11) this amount of vorticity induces a jump in velocity at the wall of u_i. Thus, a sheet of strength u_i has been generated at the boundary.

With regard to Chorin's method, sheets of cumulative strength $-2u_i$ are generated at every boundary point. (The strength of a sheet is equivalent to the opposite of the jump in velocity which it induces.) However, due to the random walk which these sheets participate in, half of the sheets introduced on the boundary will exit the computational boundary (on average). Hence the cumulative strength of the vortex sheets from the ith boundary point which actually enter the fluid is $-u_i$.

Thus, although Chorin's scheme and the above finite difference method are based on very different discretizations, the strength of the vortex sheet created at each timestep on the boundary is the same. Furthermore, while not a proof, the existence of this finite difference analog of Chorin's method strongly suggests that Chorin's treatment of boundary conditions for the Prandtl equations is just a particular implementation of boundary conditions obtained by requiring the vanishing of the time derivative of the constraint imposed by the no-slip velocity condition at the plate.

3. A Finite Difference Method for Flow Past a Cylinder

In this section we present a finite difference method for calculating flow past a circular cylinder in two dimensions. The construction of this scheme illustrates one possible technique for incorporating the boundary condition (1.10) into a numerical method. There are many other possible discretizations, some of which are computationally more efficient, but we have selected this particular one because of its relative simplicity.

Let ϕ be the potential which corresponds to a uniform flow about the cylinder with an onset velocity parallel to the x axis and of magnitude U_∞ at $x=-\infty$. We use polar coordinates and denote ω as the vorticity and $(u, v)=(\tilde{u}+\phi_r, \tilde{v}+(1/r)\,\phi_\theta)$ as the radial and tangential velocity components, respectively. With this notation Eq. (1.4) in non-dimensional form becomes

$$\frac{\partial\omega}{\partial t}+\left(u, \frac{v}{r}\right)\cdot\left(\frac{\partial\omega}{\partial r}, \frac{\partial\omega}{\partial\theta}\right)=\frac{2}{\mathrm{Re}}\,\Delta\omega. \tag{3.1}$$

The Reynolds number Re is based on the cylinder diameter and is given by $\mathrm{Re}=2r_a U_\infty/\nu$ where ν is the viscosity of the fluid and r_a is the cylinder radius. The velocities $\tilde{u}$ and $\tilde{v}$ are determined from the stream function Ψ by the relations

$$\tilde{u} = \frac{1}{r}\frac{\partial \Psi}{\partial \theta}, \qquad \tilde{v} = -\frac{\partial \Psi}{\partial r}, \tag{3.2}$$

$$\Delta \Psi = -\omega \tag{3.3}$$

$$\Psi = 0 \quad \text{at} \quad r = r_a \qquad \text{and} \qquad \lim_{r \to \infty} \int_0^{2\pi} \frac{\partial \Psi(r, \theta)}{\partial r}\, r\, d\theta = 0. \tag{3.4}$$

This last requirement, the specification of the circulation of the velocity field induced by Ψ at infinity, is necessary for the equation determining Ψ to be well posed.

Equation (3.1) is to be satisfied in the domain Ω described by $r_a < r < \infty$ and $0 \leqslant \theta \leqslant 2\pi$. On the cylinder boundary $\partial\Omega$, we require that $u = 0$ and $v = 0$. If we use the expression for ϕ,

$$\phi = U_\infty \left(r + \frac{r_a^2}{r} \right) \cos(\theta),$$

and set $r = r_a$ then these boundary conditions imply that

$$\tilde{u}(r_a, \theta) = 0, \qquad \tilde{v}(r_a, \theta) = 2U_\infty \sin(\theta), \qquad 0 \leqslant \theta \leqslant 2\pi. \tag{3.5}$$

If we denote by $G(x, y)$ the Green's function corresponding to the problem (3.3)–(3.4) the constraint on the vorticity is

$$\frac{\partial}{\partial \mathbf{n}} \int_\Omega G(\alpha, s)\, \omega(s, t)\, ds = -2U_\infty \sin(\theta), \qquad \alpha = (r_a \cos(\theta), r_a \sin(\theta)), \quad 0 \leqslant \theta \leqslant 2\pi. \tag{3.6}$$

Here $\mathbf{n}$ is the normal pointing into the cylinder. As derived in Section 1, the boundary conditions which will ensure that the time derivative of this constraint vanish are

$$\frac{\partial \omega}{\partial \mathbf{n}} + \frac{\partial}{\partial \mathbf{n}} \int_{\partial\Omega} \frac{\partial G}{\partial \mathbf{n}}(\alpha, \gamma)\, \omega(\gamma, t)\, d\gamma = \frac{\mathrm{Re}}{2} \frac{\partial}{\partial \mathbf{n}} \int_\Omega G(\alpha, s)[\mathbf{U} \cdot \nabla\omega(s, t)]\, ds. \tag{3.7}$$

The finite difference grid used was uniform with mesh width dr and $d\theta$ in the r and θ variables, respectively. Our computational domain is described by an annulus—the set of grid points $i\, dr$, $j\, d\theta$ such that $r_a \leqslant r_a + i\, dr \leqslant r_a + n\, dr = r_b$ and $0 \leqslant j\, d\theta \leqslant m\, d\theta = 2\pi$. Here r_b is the outer or "far field" radius of the computational domain. We denote the vorticity and velocity distribution at time $k\, \delta t$ by ω^k and $\mathbf{u}^k = (u^k, v^k)$, respectively. We used the standard five-point differencing for the Laplacian in polar coordinates, which we designate by Δ^d, and a second order approximation (a modified upwind scheme) to the advection term $\mathbf{u} \cdot \nabla\omega$ due to Colella [8]. We designate this by $A(\mathbf{u}^k, \omega^k)$. (The choice of the approximation method for the advection terms of the equation is not a key element in the derivation of the boundary conditions. We chose this particular one out of the variety available because of its stability characteristics and its lack of numerical diffusion.) We used a first order explicit method for the time integration of the equations. In the interior the scheme is given by

$$\frac{\omega^{k+1}-\omega^k}{\delta t} = -A(\mathbf{u}^k, \omega^k) + \frac{2}{\mathrm{Re}} \Delta^{\mathrm{d}} \omega^k. \tag{3.8}$$

Here the sub-indices (i, j) which refer to the values at the node $(i\,dr,\ j\,d\theta)$ are suppressed. The velocity field $\mathbf{u}$ used in (3.8) is obtained by solving

$$\Delta^{\mathrm{d}} \Psi = -\omega^k \tag{3.9}$$

and then using central difference approximations to (3.2) to obtain approximations to the velocities.

[illegible]

surface of the cylinder and at an artificial computational boundary $r = r_b$. We shall discuss the boundary conditions related to this artificial boundary first.

The effect of a finite computational boundary at $r = r_b$ introduces complications in calculation of (3.8) and in the solution of Laplace's equation (3.9). As regards the treatment of the vorticity transport equation (3.8), we follow the simple procedure of eliminating any vorticity which exists the circular boundary with radius $r_b - dr$ (i.e., at the end of each time step, the vorticity which is in the outer ring of computational nodes—those at $r = r_b$—is set to zero). The sum of vorticity which is deleted is saved, for this quantity is necessary in the consistent calculation of the stream function Ψ.

As regards the calculation of the stream function, we form an approximation to the solution of (3.9) by utilizing a standard fast Poisson solver for the grid points in the computational region $r_a \leqslant r \leqslant r_b$ coupled with "infinite" boundary conditions [2]. The infinite boundary conditions are derived by requiring that the solution in the finite component of the domain $(r_a \leqslant r \leqslant r_b)$ match with an appropriate solution in the infinite component $(r > r_b)$. Such a matching condition can be explicitly obtained because the simple geometry of the problem allows one to compute the infinite component analytically. This procedure, based on ideas from domain decomposition techniques for elliptic problems, is similar to that used successfully by B. Fornberg in his steady state calculations [11]. In the implementation of infinite boundary conditions it is necessary to specify the circulation about the contour $r = r_b$. We set this circulation equal to the total amount of vorticity which has exited the computational domain. With respect to our specific discretization, the amount of circulation accumulated at each timestep c^k is given by

$$c^k = -\sum_j \omega^k_{n,j} r_b\, dr\, d\theta, \qquad 0 \leqslant j\, d\theta \leqslant 2\pi. \tag{3.10}$$

In essence, we are moving the vorticity which has departed from the computational domain out to infinity. As will be seen from the computational results, this simple treatment of the far field boundary conditions—using outflow boundary conditions for the vorticity transport equation and "infinite" boundary conditions for the Laplacian in the stream function calculation—works surprisingly well.

The other boundary conditions which must be dealt with are the conditions on the vorticity at the surface of the cylinder. As in the derivation of the vorticity boundary conditions, we set $\Psi = 0$ on the cylinder boundary and consider the normal derivative boundary condition on Ψ as the constraint to be satisfied. There are two possible paths to follow in the construction of a finite difference scheme which incorporates boundary condition (3.7). One path is to use the discretization of the equations in the interior (3.8) and incorporate a separate dicretization of (3.7) near the boundary to close them. If this path is followed, then it is likely that the boundary conditions corresponding to a discretization of (3.7) will not guarantee that a discrete approximation of the constraint (3.6) will be satisfied exactly at each timestep. However, assuming that the discretizations are done in an intelligent fashion, we expect that over a given time interval (3.6) will be satisfied up to an error which diminishes as the discretization is refined. Unfortunately, such an implementation is not particularly well suited to computations in which a steady state or long time solution is sought. For such solutions, it is desirable to ensure that the error in a discrete approximation to the constraint (3.6) be independent of the time interval. (This ensures that the final solution satisfies the boundary conditions with an acceptable accuracy.) The other implementation path, one which can be used to derive methods suitable for such calculations, starts with a discretization scheme for the interior equations (3.1) and then specific boundary conditions which correspond to such a scheme are obtained by mimicing, with discrete operators, those arguments used to obtain (3.7). The result is a method in which a discrete version of (3.6) is satisfied up to numerical roundoff at every timestep. In our calculations of flow past a cylinder we were interested in computing steady state solutions (in order that our results may be compared to existing computational results) so we chose the latter implementation path. We obtain our boundary conditions on the vorticity by following the derivation of the boundary conditions for the continuous case, but using discrete, rather than continuous operators.

We denote by D_{n} the standard one-sided finite difference approximation to the normal derivative at the cylinder surface:

$$[D_{\mathrm{n}}\omega]_j = \frac{\omega_{0,j} - \omega_{1,j}}{dr}. \qquad (3.11)$$

We denote by Δ_0^{d} the five-point finite difference approximation to the Laplacian in which values on the cylinder boundary are taken to be identically zero. With this notation, the stream function at time $k\,\delta t$ is given by

$$\Psi = -(\Delta_0^d)^{-1}\,\omega^k.$$

A discrete approximation to constraint (3.6) is thus

$$[D_{\mathrm{n}}(\Delta_0^d)^{-1}\,\omega^k]_j = -2U_\infty \sin(j\,d\theta). \qquad (3.12)$$

This condition holds for all points, $(r_a, j\,d\theta)$, on the surface of the cylinder. In this expression, the inverse of the discrete Laplacian in the computational region is computed using "infinite domain" boundary conditions.

In order to ensure that this constraint be satisfied by the numerical scheme, we require that the discrete time difference of the constraint vanish, i.e.,

$$\frac{D_n(\Delta_0^d)^{-1}\,\omega^{k+1} - D_n(\Delta_0^d)^{-1}\,\omega^k}{\delta t} = 0.$$

If one uses the equations which ω satisfies, then this becomes

$$D_n(\Delta_0^d)^{-1}\left(\frac{\omega^{k+1} - \omega^{\prime}}{\delta t}\right) = 0$$

$$\Leftrightarrow D_n(\Delta_0^d)^{-1}\left(-A(u^k, \omega^k) + \frac{2}{\mathrm{Re}}\,\Delta^d\omega^k\right) = 0. \tag{3.13}$$

In this last expression the evaluation of the discrete Laplacian incorporates the yet to be determined vorticity boundary values. In the continuous case, integration by parts is employed and the Laplacian occurring in this last expression is transformed to a functional of the boundary vorticity. We refrain from doing the discrete version of this (because it leads to an expression which is not computationally useful) and instead proceed to show (3.13) can be used to obtain a useful expression for determining the boundary values of ω^k. Assuming (3.12) is satisfied by ω^k, we use (3.13) to determine the vorticity boundary values ω_b^k which will guarantee that ω^{k+1} will satisfy (3.12). We write (3.13) as

$$D_n(\Delta_0^d)^{-1}\left(-A(u^k, \omega^k) + \frac{2}{\mathrm{Re}}\,\Delta_0^d\omega^k + \frac{2}{\mathrm{Re}}\,\Delta^d\omega_b^k\right) = 0. \tag{3.14}$$

Here we are evaluating $(2/\mathrm{Re})\,\Delta^d\omega_b^k$ by extending ω_b^k to be zero in the interior of the domain. The terms involving $A(u^k, \omega^k)$ and $\Delta_0^d\omega^k$ are completely known at time $k\,\delta t$, thus Eq. (3.14) becomes an equation for ω_b^k alone. (We are making the implicit assumption that the approximation to the convective term for the vorticity does not involve boundary values of vorticity.) From this last expression we obtain an equation for ω_b^k,

$$D_n(\Delta_0^d)^{-1}\left(\frac{2}{\mathrm{Re}}\,\Delta^d\omega_b^k\right) = D_n\,(\Delta_0^d)^{-1}\left(A(u^k, \omega^k) - \frac{2}{\mathrm{Re}}\,\Delta_0^d\omega^k\right). \tag{3.15}$$

In our computation, the right-hand side of this equation is computed by evaluating each of the operators in turn, starting with the advection terms and the Laplacian. The discrete system on the left-hand side of (3.15) (the order of which is equal to the number of boundary points) is then solved for the vorticity boundary values. These values are then incorporated into the interior finite difference scheme (3.8).

One may be concerned about the invertibility of the operator on the left-hand side of (3.15). Consider a function p defined on the boundary of the cylinder. The inner product $(p, D_n(\Delta_0^d)^{-1}(\Delta^d p))$ can be cast in the form $(Tp, (\Delta_0^d)^{-1}Tp)$ for a rank n finite difference operator T from R^n to $R^{n \times m}$. Here $(\cdot, \cdot)$ refers to the natural inner product defined on the interior mesh points—products of values times an integration factor $r\,dr\,d\theta$. Also, the action of the discrete Laplacian upon p, $\Delta^d p$, is evaluated assuming that p is extended to be zero at interior points of the domain. From this observation and the result that the finite difference operator $(\Delta_0^d)^{-1}$ (inverted assuming a fixed circulation at $r = r_b$) is negative definite, it follows that the operator in (3.15) is negative definite and hence non-singular.

If one uses a direct method to solve (3.15) then it is necessary to form the operator on the left-hand side explicitly. One can form the matrix representation of (3.15) by considering the operators action upon a basis for ω_b^0. In our particular application of flow past a cylinder, this requires very little work because the operator is represented by a circulant matrix. The matrix can thus be constructed after calculation of the operators action on one basis element. Furthermore, the matrix is diagonalized by the discrete Fourier transform, so the inversion of the resulting matrix equation can be carried out efficiently using the fast Fourier transform. In other situations this will not be possible, but we note that the matrix only depends on the mesh size and does not depend on the physical parameters of the problem. There is some similarity of the operator occurring in (3.15) and that which arises when one employs domain decomposition techniques for solving elliptic boundary value problems [1]. In the latter case iterative methods have proven remarkably efficient and it is quite likely that fast iterative methods for solving (3.15) are possible.

One item to be discussed is the choice of initial conditions. In order that our boundary conditions be effective, we must ensure that the initial vorticity satisfy the constraint (3.12). For the Prandtl boundary layer equations, the choice of initial vorticity was easy to determine. For this problem, the choice of initial conditions is a bit more difficult. We would like an initial vorticity distribution which is non-zero only on the boundary of the cylinder—i.e., a set of initial conditions which corresponds to an impulsively started cylinder. A problem occurs becuase in the discrete equations the vorticity, if confined to the cylinder boundary points alone, does not induce any velocity. Only interior values of vorticity are used to form the right-hand side of the discrete approximation to (3.3)–(3.4). However, a suitable vorticity can be obtained by solving

$$D_n(\Delta_0^d)^{-1}\left(\frac{2}{\mathrm{Re}}\Delta^d\omega_b^0\right) = -2U_\infty \sin(j\,d\theta)$$

for values on the cylinder boundary ω_0^b, and defining

$$\begin{aligned} \omega^0 &= 0 && \text{for} \quad (i\,dr, j\,d\theta) \in \Omega - \partial\Omega, \\ \omega^0 &= \omega_b^0 && \text{for} \quad (i\,dr, j\,d\theta) \in \partial\Omega. \end{aligned}$$

It can be checked that this vorticity distribution, while not satisfying the constraint at time $t=0$ itself, has the property that after the first timestep the approximate vorticity ω^1 will satisfy the constraint. For $k>1$ the use of boundary conditions (3.13) will ensure that the remaining ω^k satisfy the constraint.

We now summarize the basic steps of the algorithm

[0] Construct initial data ω^0.

[I] Given ω^k compute $\tilde{F}=A(u^k,\omega^k)-(2/\mathrm{Re})\,\Delta_0^{\mathrm{d}}\omega^k$. $\mathbf{u}^k$ is determined by solving $\Delta_0^{\mathrm{d}}\Psi^k=-\omega^k$ with appropriate boundary conditions and then differencing the result. In the computation of Ψ^k the circulation at $r=r_b$ is set equal to the opposite of that mass of vorticity which which has exited the computational domain during the previous k steps.

[II] Determine the boundary vorticity ω_b^k by computing the solution to

$$D_{\mathrm{n}}(\Delta_0^{\mathrm{d}})^{-1}\left(\frac{2}{\mathrm{Re}}\,\Delta^{\mathrm{d}}\omega_b^k\right)=D_{\mathrm{n}}(\Delta_0^{\mathrm{d}})^{-1}\,\tilde{F}.$$

[III] Construct ω^{k+1} by using the difference equation

$$\omega^{k+1}=\omega^k+\delta t\left(-A(u^k,\omega^k)+\frac{2}{\mathrm{Re}}\,\Delta^{\mathrm{d}}\omega^k\right)$$

in the interior of the computational domain and using the boundary vorticity computed in step [II]. Remove the vorticity from the outer ring of computational points, those at $r=r_b$, and accumulate the amount of vorticity deleted.

In the next section we will discuss numerical results obtained with this method. As with Prandtl's boundary layer equations, this algorithm is analogous to those used by Chorin. Consider the quantity $\tilde{\omega}$, the result of one step of the Navier–Stokes equations using homogeneous boundary values for the vorticity,

$$\frac{\tilde{\omega}-\omega^k}{\delta t}=\left(-A(u^k,\omega^k)+\frac{2}{\mathrm{Re}}\,\Delta_0^{\mathrm{d}}\omega^k\right).$$

Assuming that constraint (3.12) on the vorticity is satisfied at time $k\,\delta t$, we find that the right-hand side of the equation which determines the vorticity boundary values in step [II] can be expressed in terms of $\tilde{\omega}$, i.e.,

$$\begin{aligned}D_{\mathrm{n}}(\Delta_0^{\mathrm{d}})^{-1}\,\tilde{F}&=D_{\mathrm{n}}(\Delta_0^{\mathrm{d}})^{-1}\left(-A(u^k,\omega^k)+\frac{2}{\mathrm{Re}}\,\Delta_0^{d}\omega^k\right)\\&=-\left[D_{\mathrm{n}}(\Delta_0^{\mathrm{d}})^{-1}\left(\frac{\tilde{\omega}-\omega^k}{\delta t}\right)\right]=-\left[D_{\mathrm{n}}(\Delta_0^{\mathrm{d}})^{-1}\left(\frac{\tilde{\omega}}{\delta t}\right)+\frac{2U_\infty\sin(\theta)}{\delta t}\right].\end{aligned}$$

Thus, as in Chorin's schemes, the right-hand side is proportional to the error (or slip) in tangential velocity at the boundary. (The amount of vorticity actually introduced into the fluid is roughly proportional to the slip, for when the values of ω_b^k are incorporated into the interior scheme the factor δt vanishes, cf. Section 2.)

4. Computational Results

In this section we present some numerical results which were obtained with the method presented in the previous section. Our primary goal in these computations was to assess the accuracy of the approximation of the vorticity at and near the cylinder surface (as this is a quantity which is most dependent on the implementation of vorticity boundary conditions.) For the Reynolds numbers considered here (4–50) the polar grid which we used proved satisfactory. For higher Reynolds numbers in which vorticity dynamics far away from the cylinder surface need to be resolved this grid is not satisfactory. Other grids, such as those successfully used by Fornberg in his steady state calculations [9–11], would be more efficient at resolving these features. Alternatively, it is possible to couple the results of our finite difference scheme near the cylinder surface with a Lagrangian vortex blob method to evolve the vorticity away from the cylinder surface. Such a scheme is currently being implemented and will be discussed elsewhere.

In our calculations we concentrated on two measures of vorticity which have physical significance, the pressure distribution and the coefficient of drag. Both of these quantities can be written as functionals of the boundary vorticity and the normal derivative of the boundary vorticity. For a point θ on the cylinder surface the pressure is given by

$$P(\theta) = P_0 - \nu r_a \int_0^\theta \frac{\partial \omega}{\partial \mathbf{n}}\, d\phi$$

where r_a is the radius of the cylinder. P_0 is the forward stagnation point pressure and the integral is taken from the forward stagnation point clockwise around the cylinder surface. The drag is given by

$$D = \nu \int_0^{2\pi} \left(\omega - \frac{\partial \omega}{\partial \mathbf{n}}\right) \sin\theta\, d\theta.$$

In our calculations we measure the scaled non-dimensional counterpart of these quantities. The drag in this latter case is referred to as the coefficient of drag C_d and is given by $C_d = D/\frac{1}{2}\rho U_0^2 r_a$.

In our calculations $U_0 = 1.0$, $r_a = 1.0$, and $\rho = 1$. The Reynolds number of the flow was varied by changing the coefficient of viscosity. The computations were carried out with a timestep chosen so that the scheme was stable. For the Reynolds numbers considered here, the timestep was limited by the stability restriction

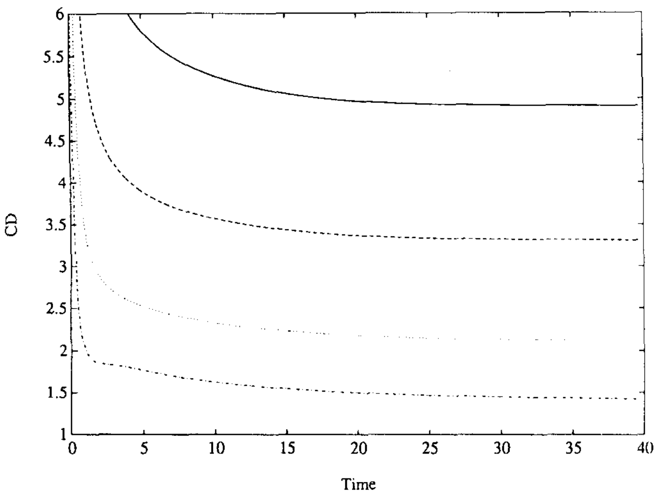

FIG. 4.1. Coefficient of drag C_d, vs time.—, Re = 4; ---, Re = 8; ···, Re = 20; -·-·, Re = 50.

imposed by our use of an explicit scheme to calculate the diffusion of vorticity. (The constraint imposed by the advection scheme was less severe than that for the diffusion scheme.) Thus, in our computations, a timestep δt was initially estimated on the basis of the requirement that $\delta t \leqslant (dr\, d\theta)/4v$. In practice the timestep had to be taken slightly smaller than this estimate—but not exceedingly so.

The change in time of the coefficient of drag for flows with different Reynolds numbers is plotted in Fig. 4.1. The rate depends somewhat on the Reynolds number, the higher the Reynolds number the more rapid the initial convergence to steady state. By time $t = 40$ an approximation to steady state appears to have been reached for each Renolds number. The coefficients of drag at the final times are presented in Table I, along the results of computations by Keller and Takami [13, 14] and the experimental work by Tritton [20]. (We chose these results as representative samples of the great body of results which exist for viscous flow past a cylinder.) The value of the farfield computational boundary r_b was 21. The agreement is generally good, although our results for Reynolds numbers less than 50 appear to be somewhat high. However, we are within the experimental error of the results given by Tritton.

TABLE I

Comparison of the Coefficient of Drag C_d

Re	Present work	Keller and Takami (comp.)	Tritton (exper.)
4	4.9147	4.4282	4.7871
8	3.3165	—	3.3985
20	2.1230	2.0027	2.0225
50	1.4258	1.4182	1.4526

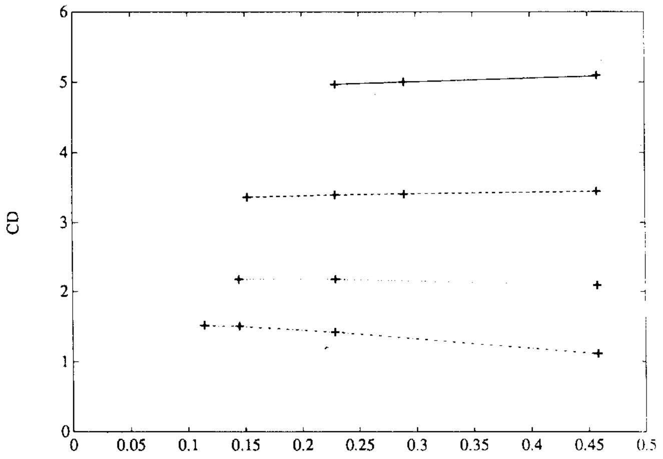

FIG. 4.2. Coefficient of drag vs mesh size (mesh size measured by $\sqrt{dr^2 + d\theta^2}$). —, Re = 4; ---, Re = 8; ···, Re = 20; -·-·, Re = 50.

The effect of numerical parameters upon the drag coefficient is illustrated in Figs. 4.2 and 4.3. In Fig. 4.2, we plot the change in drag with respect to the mesh size for each Reynolds number. In order to reduce the computational labor, the time at which the drag was measured was $t = 20.0$. This is not steady state, and in particular, it is not the same time used to measure the drag which appears in Table I, but sufficiently close to steady state so that the general convergence trend in the drag could be ascertained. These results indicate that for the Reynolds numbers considered, our computational grid was sufficiently fine to evaluate the drag

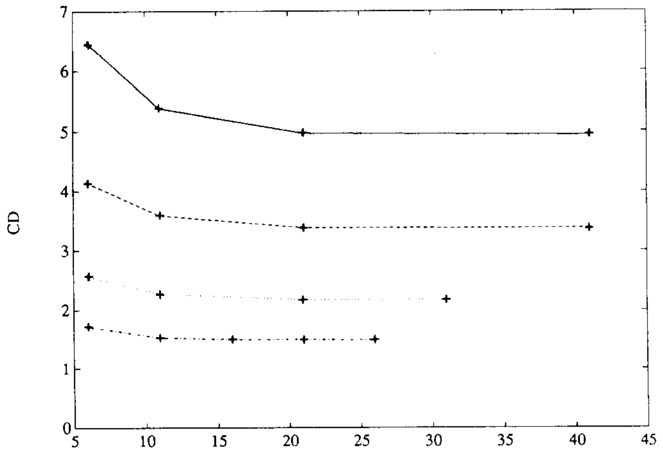

FIG. 4.3. Coefficient of drag vs. outer computational radius. —, Re = 4; ---, Re = 8; ···, Re = 20; -·-·, Re = 50.

coefficient accurately. In Fig. 4.3 the change in drag associated with a change in the far field radius is plotted. We found the effect of the computational radius on the drag a bit surprising. A computational radius of 21 cylinder radii appears to be sufficient to yield accurate drag results—this is not a particularly distant computational radius. Secondly, as the Reynolds number increases, it appears that one need not increase the far field radius. In fact, it appears that one can reduce it. This result is very nice, for it means that calculations of flows at much higher Reynolds numbers may not need a correspondingly larger far field radius. This is the opposite conclusion which might be inferred from the steady state calculations of Fornberg [9–11]. In those calculations it is readily apparent that as the Reynolds number increases, one must increase the far field radius in order to capture the steady state solution.

We were interested in the effect of the far field radius on the pressure distribution. To investigate this, we concentrated upon the effect of this boundary for Reynolds number 50. In Fig. 4.4 we plot the pressure at the cylinder boundary corresponding to an outer computational radius of 6, 16, and 26 computational radii. Fortunately, as an examination of this figure provides, we find that the pressure is not greatly effected by the far-field boundary condition either. The most significant effect of the far-field boundary is to increase the drop from the front to the back of the cylinder. In light of the formula which defines the pressure, it appears that the primary effect of having a close computational radius is to increase the flux of vorticity from the cylinder surface. That this occurs when the computational boundary is close to the cylinder seems plausible, for our computational boundary is a perfect absorber of vorticity and hence "draws" more vorticity away from the cylinder wall. At higher Reynolds, when the flow is non-stationary, it is not clear that a similar mechanism will occur. Further numerical work should clarify the situation in the latter circumstance.

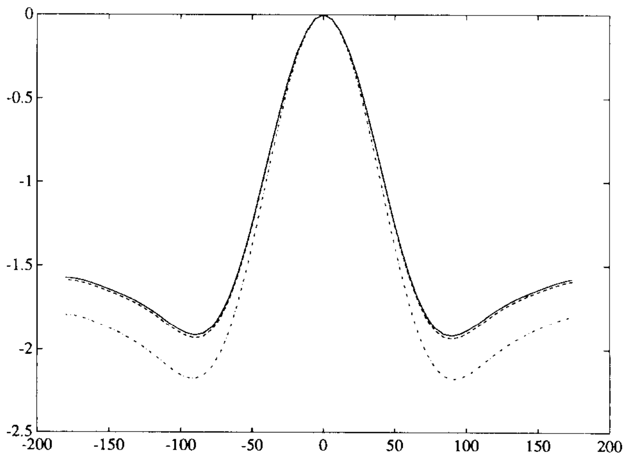

FIG. 4.4. Pressure distribution, $P=(P-P_0)/\frac{1}{2}\rho U_0^2$, at Re = 50.—, outer radius = 26; ---, outer radius = 16; ····-, outer radius = 6.

Lastly, we were interested in the effect of using a first order upwind scheme to approximate the advection term in Eq. (3.1). This interest arises because upwind schemes can be very inaccurate, yet remain popular because of their simplicity and stability properties. We thought that it would be of interest to investigate the quality of the solutions which could be obtained with such schemes. For this purpose we concentrated on flows at $\mathrm{Re} = 50$, and used corner transport upwind [8], a variant of the standard upwind difference scheme which includes the upwind corner points in the difference stencil.

There were significant differences in the results between first and second order upwind. When using the finest grid, the drag calculated at time $t = 20.0$ by the second order scheme was 1.50455, while that obtained with the first order scheme 1.23395—a difference of 17%. The fact that the drag was lower was an unexpected result. The upwind schemes are known for their numerical diffusion and we had thought that the calculations would yield a result corresponding to a more viscous flow, i.e., a higher drag coefficient. However, one must be careful about drawing conclusions about the nature of a numerical solution based on the coefficient of drag alone. In Figs. 4.5a and 4.5b we present vorticity contours obtained with a first and second order upwind scheme. In Fig. 4.5c we present the vorticity contour for a Reynolds number 20 solution (obtained using a second order method.) As is evident from the figures, the first order upwind scheme shares the properties similar to the lower Reynolds number calculation. In particular, there is a lower maximum vorticity on the surface of the cylinder (first order $\mathrm{Re} = 50$; $\omega_{\max} = 6.618$, second order $\mathrm{Re} = 50$; $\omega_{\max} = 7.822$, and second order $\mathrm{Re} = 20$; $\omega_{\max} = 4.673$) and a greater angle between the vorticity contours and a tangent to the cylinder surface. That this qualitative similarity is not reflected in the drag coefficient is due to the fact that

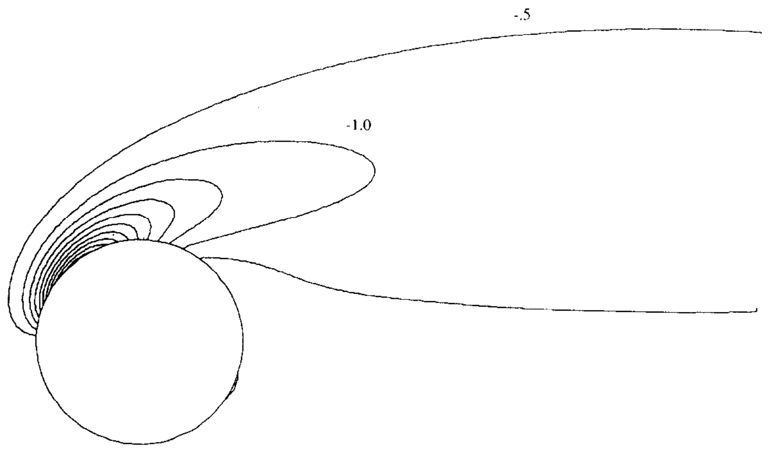

FIG. 4.5a. Vorticity contours for first order upwind, $\mathrm{Re} = 50$, -0.5 to -7.25 (0.5).

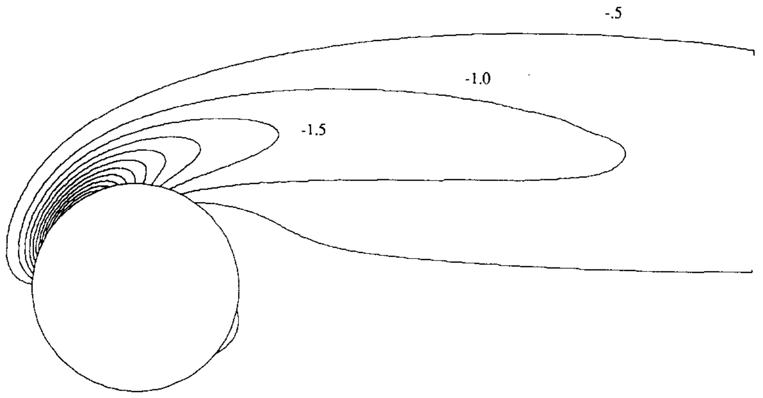

FIG. 4.5b. Vorticity contours for second order upwind, Re = 50, −0.5 to −7.25 (0.5).

this coefficient is the product of a scaling factor and a function of the vorticity, namely, the integral

$$\int_0^{2\pi} \left(\omega - \frac{\partial \omega}{\partial \mathbf{n}} \right) \sin \theta \, d\theta. \tag{4.1}$$

Although the drag coefficient decreases as the Reynolds number increases, the factor (4.1) actually increases with increasing Reynolds number. For example, for the second order scheme, this integral has the approximate values 9.8, 13.2, 21.2, and 37.6 for Reynolds numbers 4, 8, 20, and 50, respectively. For the first order upwind scheme, the value of the integral is about 30.8, reflecting the fact that the computed vorticity field corresponds to a more diffusive calculation. Thus, the coef-

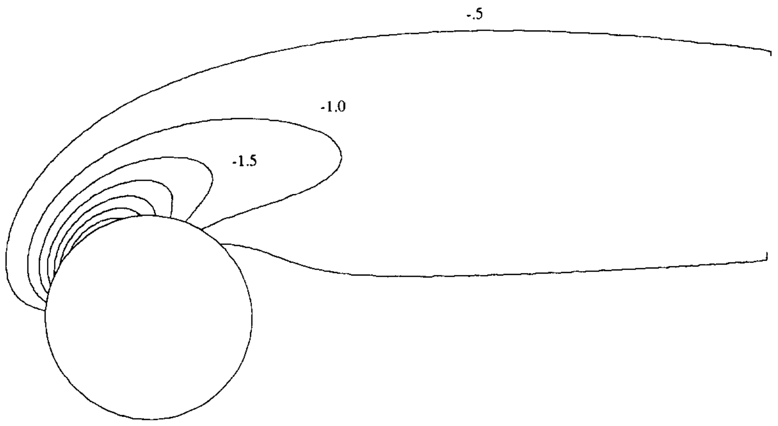

FIG. 4.5c. Vorticity contours for second order upwind, Re = 20, −0.5 to −7.25 (0.5).

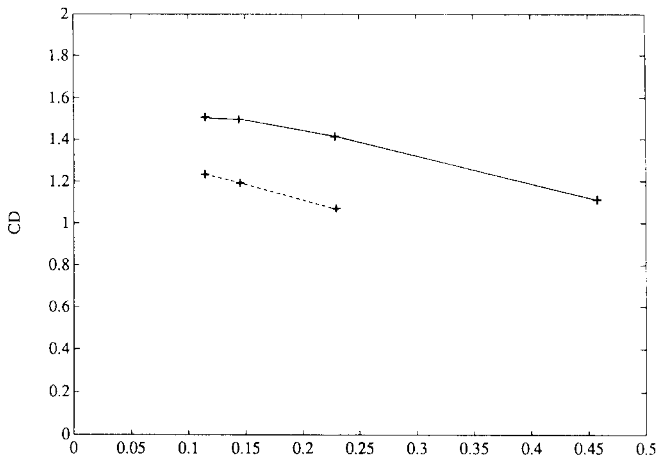

FIG. 4.6. Coefficient of drag vs mesh size (mesh size measured by $\sqrt{dr^2+d\theta^2}$). —, second order upwind; ---, first order upwind.

ficient of drag for a diffusive scheme, being the product of a smaller integral factor times a smaller scaling factor, can be expected to be smaller rather than larger that the correct coefficient. In Fig. 4.6 we plot the coefficient of drag versus a measure of the mesh size, $\sqrt{\delta r^2+\delta\theta^2}$. From this figure we see that at the finest grid level used, the first order upwind scheme has not converged and thus the difference in the drag is understandable. It is clear that a second order scheme is superior and should be used.

5. CONCLUSIONS

In this paper we have given a derivation of appropriate boundary conditions for the vorticity associated with the velocity field of an incompressible, viscous fluid in two dimensions. The key point in the derivation of these boundary conditions is to realize that the evolution of the vorticity is a constrained evolution, and boundary conditions can be determined by requiring that the time derivative of the constraint vanish. We have also presented a method which incorporates these boundary conditions. In the case of the Prandtl boundary layer equations the method which results is analogous to Chorin's vortex sheet method [5] which is of the vorticity creation type. For the complete Navier–Stokes equations, the implementation has similarities to Chorin's original vortex blob method [4] and is formally equivalent to an implementation of the projection method of Quartapelle and Valz-Gris [16, 17]. These observations provide an interpretation of the relation between two apparently different methods. Both methods are designed to ensure that the constraint on the vorticity is satisfied—it is just that the creation algorithms accomplish this by ensuring that the time derivative of the constraint vanish.

Our numerical results for flow past a cylinder indicate that accurate schemes which employ our boundary conditions can be constructed. Moreover, the results indicate that the difficulties associated with an infinite computational domain are not overwhelming, and can be overcome with a straightforward procedure. The accuracy of the scheme we present is satisfactory for low Reynolds numbers (perhaps up to a few hundred), but it is not especially efficient for high Reynolds number flows. Many improvements can be envisioned, for example, using a different grid to resolve the vorticity dynamics downstream from the cylinder or a different time-stepping procedure. One of the benefits of having a concise description of the vorticity boundary conditions is that it facilitates the implementation of these improvements.

ACKNOWLEDGMENTS

The author would like to thank Dr. Claude Greengard for his helpful comments concerning the problems considered in this paper and the reviewers for their helpful comments as well.

REFERENCES

1. C. R. ANDERSON, "On Domain Decomposition," Classic Report 85-09, Department of Computer Science, Stanford University, 1985 (unpublished).
2. C. R. ANDERSON, "The Application of Domain Decomposition to the Solution of Laplace's Equation in Infinite Domains," CAM Report 87-19, Dept. of Mathematics, UCLA, 1987 (unpublished).
3. G. BATCHELOR, *An Introduction to Fluid Mechanics* (Cambridge Univ. Press, Cambridge, 1967), p. 277.
4. A. J. CHORIN, *J. Fluid Mech.* **57**, 758 (1973).
5. A. J. CHORIN, *J. Comput. Phys.* **27**, 428 (1978).
6. A. J. CHORIN, *SIAM J. Sci. Statist. Comput.* **1**, 1 (1980).
7. A. J. CHORIN AND J. MARSDEN, *A Mathematical Introduction to Fluid Mechanics* (Springer-Verlag, New York, 1979), p. 110.
8. P. COLELLA, *J. Comput. Phys.*, to appear.
9. B. FORNBERG, *J. Fluid. Mech.* **98**, 819 (1980).
10. B. FORNBERG, *J. Comp. Phys.* **61**, 297 (1985).
11. B. FORNBERG, *Proceedings, IUTAM Symp. on Boundary-layer Separation, August*, 26-28, 1986 (Springer-Verlag, Berlin/New York).
12. M. GUPTA AND R. MANOHAR, *J. Comput. Phys.* **31**, 265 (1979).
13. H. B. KELLER AND H. TAKAMI, *Phys. Fluids* **12** (part II) suppl., 51 (1969).
14. H. B. KELLER AND H. TAKAMI, in *Numerical Solutions of Nonlinear Differential Equations*, edited by D. Greenspan (Wiley, New York, 1966), p. 115.
15. S. ORSZAG AND M. ISRAELI, *Annu. Rev. Fluid Mech.* **6**, 281 (1974).
16. L. QUARTAPELLE AND F. VALZ-GRIS, *Int. J. Numer. Math. Fluids* **1** , 129 (1981).
17. L. QUARTAPELLE *J. Comput. Phys.* **40**, 453 (1981).
18. L. QUARTAPELLE AND M. NAPOLITANO, *Int. J. Numer. Math. Fluids* **4** , 109 (1984).
19. H. SCHLICHTING, *Boundary Layer Theory* (McGraw-Hill, New York, 1960), p. 139.
20. D. J. TRITTON, *J. Fluid Mech.* **6**, 547 (1959).