UCLA
COMPUTATIONAL AND APPLIED MATHEMATICS

Observations on Vorticity Creation Boundary Conditions

Christopher R. Anderson

September 1988

CAM Report 88-30

Department of Mathematics
University of California, Los Angeles
Los Angeles, CA. 90024-1555

OBSERVATIONS ON VORTICITY CREATION BOUNDARY CONDITIONS

Christopher R. Anderson
Department of Mathematics
University of California
Los Angeles, CA 90024

To appear, Proceedings SIAM Workshop on Vortex Dynamics. April 25-27, 1988, Xerox Training Center, Leesburg, Virginia.

OBSERVATIONS ON VORTICITY CREATION BOUNDARY CONDITIONS

CHRISTOPHER R. ANDERSON*

Abstract. In this paper we discuss some issues related to the problem of vorticity boundary conditions. We outline the origination of the problem and briefly discuss three solution procedures which have been used to overcome it. We primarily focus on one method - that which employs boundary vorticity creation. We give an example which shows the equivalence between creation type boundary conditions and those which exploit the relationship between the vorticity and the stream function on the boundary. We also show how one can obtain greater than first order accuracy in time for methods employing creation type boundary conditions. Lastly, we show by example the problems which arise when the boundary conditions on the vorticity are such that the no-slip condition on the velocity is satisfied only to truncation error rather than to roundoff error.

1. Introduction. There are several different ways of "solving" the problem of vorticity boundary conditions and it is the authors belief that they are all generally equivalent. The ease of implementation of each is different, and, depending on the discretization, certain methods are favored over others. However, there is often value in considering the application of particular techniques in situations in which they have not been previously used and we intend to do this here.

The focus in this paper will be on the vorticity creation approach and we use this approach in the construction of finite difference schemes. The reason for selecting finite differences as the underlying approximation procedure is to demonstrate certain properties of the methods incorporating vorticity creation. The first is that one *can* implement creation boundary conditions in the context of finite differences methods. The approach therefore offers an alternative viewpoint to the methods which are customarily used. Secondly, because of the Lagrangian discretization and the use of a random walk to simulate diffusion, the vortex-blob and vortex-sheet methods [2], [3] are sufficiently complicated that the properties of the creation methods we wish to show would be difficult to demonstrate within that framework.

The problem of vorticity boundary conditions is as follows - consider the two-dimensional Navier-Stokes equations in vorticity form for flow in a domain Ω with solid boundary $\partial\Omega$,

$$\frac{\partial \omega}{\partial t} + \vec{u} \cdot \nabla \omega = \nu \Delta \omega \tag{1}$$

$$u = \frac{\partial \Psi}{\partial y} \quad , \quad v = \frac{-\partial \Psi}{\partial x} \tag{2}$$

where

$$\Delta \Psi = -\omega \tag{3}$$

$$\Psi = 0 \qquad \frac{\partial \Psi}{\partial n} = 0 \qquad \text{on} \quad \partial\Omega. \tag{4}$$

The requirement that $\Psi = 0$ at the boundary ensures the no-flow boundary condition ($\vec{u} \cdot \vec{n} = 0$) and the other condition $\frac{\partial \Psi}{\partial n} = 0$ ensures the no-slip condition ($\vec{u} \cdot \tau = 0$). If one

* Department of Mathematics, UCLA, Los Angeles, California, 90024
Research Supported by ONR Contracts #N00014-86-K-0691, #N00014-86-K-0727 and NSF Grant DM586-57663

wants to keep within the vorticity stream-function framework then there is a fundamental difficulty in computing the solution to (1) - (4). One needs boundary conditions for the vorticity to implement (1), and they are not given, while in (3) - (4) there appear to be too many boundary conditions for the stream function.

Actually, these two difficulties are related. As observed by Quartapelle and Valz-Gris [4] there is a global constraint on the vorticity which will ensure that a solution to (3) - (4) exists. This constraint is that the vorticity must have a certain projection with respect to all harmonic functions defined in the domain, i.e. one must have

$$\int_{\Omega} \omega(\vec{x},t)\eta(\vec{x})d\vec{x} = 0 \tag{5}$$

for all η where η is a function such that $\Delta\eta = 0$ in Ω and $t \geq 0$. One can therefore view the problem of solving equations (1) - (4) as evolving the vorticity subject to this constraint. In appropriate discretizations it just so happens that the number conditions furnished by (5) is equal to the number of unknown boundary values and thus (5) can be used to close the equations. This observation can be used to construct numerical methods. For example in [5] the vorticity is advanced one time step using arbitrary boundary values, and then this approximate solution is projected onto the subspace of vorticity which satisfies (5). We shall refer to this method as the projection method. While this manner of time-stepping guarantees that a discrete version of (1) - (4) is satisfied, the manner in which the boundary values of vorticity are determined is not immediately obvious.

In order to see how the values of vorticity on the boundary are determined, one considers a different procedure for satisfying the constraint. As discussed in [1], if one assumes the initial vorticity satisfies (5) and then asks what are the conditions which ensure that the time derivative of the constraint vanish - then (after incorporating the equations of motion and integration by parts) one can derive boundary conditions of an integral-differential nature. The boundary conditions, suitably interpreted, show that vorticity is created at the boundary in response to slip induced by Euler flow. Thus, this is a way of understanding the boundary conditions which were introduced by Chorin and have been used in conjunction with vortex-blob and vortex-sheet methods [2], [3]. Simply, the methods based on vorticity creation are boundary conditions which guarantee that the time derivative of the constraint (5) vanish as the vorticity evolves subject to (1).

Another method which is used to solve the problem of vorticity boundary conditions which is different than a projection method or a creation type method, is that which we call the "boundary vorticity-stream function method". (See for example [6].) In this technique the relationship between the stream function and the vorticity (3) and the boundary values (4) are exploited to determine values of the vorticity. If one thinks of determining the stream function given the vorticity as running (3) forward, then this procedure is running (3) backwards to get the vorticity. This method is popular with those using finite difference techniques.

It is not clear at the outset that the three methods are all identical. They certainly have the same flavor - manipulating the vorticity in order to satisfy the boundary conditions on the velocity, but they differ in their method of manipulation. As is explained in [1], the goal of the projection methods and creation methods is to ensure (5) is satisfied, so, in principle, if one assumes an appropriate choice of implementation, both approaches should lead to equivalent schemes. (However, an appropriate discretization may not be easy to find, as for example in the case of designing a projection method when using a vortex-blob discretization.) The relationship between the boundary vorticity-stream function method and the projection or creation methods is still a bit of a puzzle. However, in light of the fact that both the creation schemes and the boundary vorticity-stream function methods introduce vorticity on the boundary, it seems possible that the techniques are related. In some instances we can see this, and in the first section of this paper we will, for a model problem, show the exact equivalence between a creation type method and a boundary vorticity-stream function method. This is a very simple model problem - Stokes flow in the upper half plane - and so this is not a proof of general equivalence. I am presenting this example primarily to convince those who use the boundary vorticity-stream function method that there may be some merit to considering the problem of vorticity boundary conditions within the framework of the vorticity creation approach or the projection approach (since, after all, one obtains the same answer). The reason for considering other viewpoints is that it may allow one to resolve certain difficulties which one encounters with the boundary-vorticity stream function method, i.e. the determination of explict methods which have high order spatial accuracy. Alternatively, I present the example to help convince those who use the creation or projection approach that the boundary vorticity-stream function method may also be worth considering.

Originally the vortex creation type boundary conditions were derived by considering a fractional step scheme for solving the Navier-Stokes or Prandtl equations. One step consists of advancing the inviscid component of the equations and the second step consists of advancing the viscous component. The second step is accompanied by vorticity creation which is introduced in order to satisfy the no-slip condition on the velocity at the boundary. However, this viewpoint is encumbering, in that it appears there is a limitation on the time accuracy which can be achieved because of the fractional-step nature of the scheme. If one considers such creation boundary conditions as being derived from the principle that they are ensuring that the time derivative of the constraint (5) vanish, then this difficulty disappears and one can easily see how to derive high-order accurate methods in time. In section 2 we present the formal argument, and then demonstrate the feasibility of the approach by showing the computational results of a higher order (in time) finite difference method for the Prandtl equations.

Lastly, in the implementation of vorticity boundary conditions of the creation type one can devise methods which have the property that they satisfy the no-slip condition on the velocity only to truncation error rather than to round-off error. This difference, while seemingly small, may have important implications for the use of the schemes. In particular, we find that one is suitable for finding steady-state solutions and the other not. We show the results of a computational method for the Prandtl equations which demonstrate the effect of not satisfying the no-slip condition exactly.

2. Presentation of Equivalent Schemes. In this section we shall show by way of example the equivalence between two different methods of implementing vorticity boundary conditions. The first scheme, described to the author by H.O. Kreiss, is of the boundary vorticity-stream function type. The second is a vorticity creation type, and could be considered a finite difference implementation of boundary conditions which are used for vortex-blob methods. What we find interesting about this example is that although the methods differ greatly in their derivation and implementation they are indeed algebraically equivalent and give identical results.

The equations to be solved concern Stokes flow in the upper half-plane in which the vorticity is a function independent of x. Thus if $\omega(x,y,t)$ is the vorticity, then we are assuming that $\omega(x,y,t) = \omega(y,t)$. The equations to be solved are

$$\frac{\partial \omega}{\partial t} = \nu \frac{\partial^2 \omega}{\partial y^2} \qquad \text{for} \quad y > 0 \tag{6}$$

$$\frac{\partial^2 \Psi}{\partial y^2} = -\omega \qquad \text{for} \quad y > 0 \tag{7}$$

(8) $$\Psi = 0 \qquad \frac{\partial \Psi}{\partial y} = 0 \qquad \text{at} \quad y = 0$$

$$\frac{\partial \Psi}{\partial y} = 0 \qquad \text{at} \quad y = \infty$$

The velocity field corresponding to the vorticity is given by

(9) $$u = \frac{\partial \Psi}{\partial y} \qquad v = -\frac{\partial \Psi}{\partial x}$$

We assume that the initial vorticity distribution induces a velocity field which satisfies both the no-slip and no-flow condition on the boundary $y = 0$, i.e. if we solve

(10) $$\frac{\partial^2 \Psi}{\partial y^2} = -\omega(y,t) \qquad \Psi = 0 \quad \text{at} \quad y = 0 \qquad \frac{\partial \Psi}{\partial y} = 0 \quad \text{at} \quad y = \infty$$

at $t = 0$, then

(11) $$\frac{\partial \Psi(y,t)}{\partial y} = 0 \quad \text{at} \quad y = 0$$

at $t = 0$ also.

It can be verified that if we solve (6) with the boundary condition on the vorticity $\frac{\partial \omega}{\partial y} = 0$ then (10) and (11) will hold for all $t > 0$ as well. The manner in which we show the two different methods to be equivalent is to show they both implement a finite difference procedure for solving (6) with this Neumann boundary condition.

For the computational methods to be described, consider the finite difference grid as in Figure 1. We assume a uniform mesh width h in each direction and that the index j in the y-direction extends from 0 to M where M is large. (The fact that M is finite is not particularly important for this discussion and could be eliminated, but this would make the discussion a little more difficult to present.)

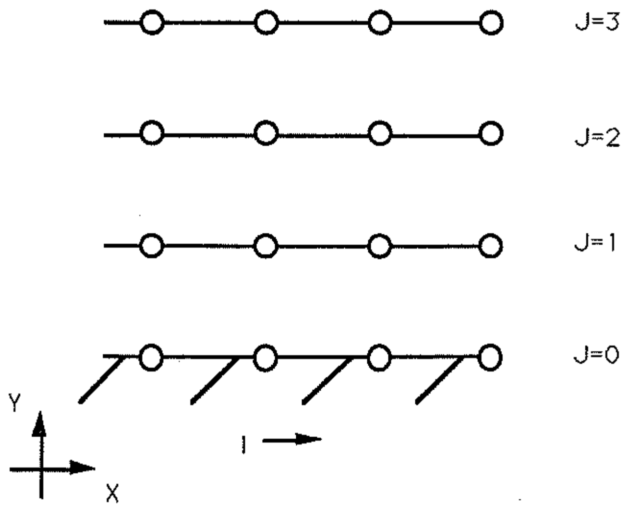

Figure 1

Since the solution is independent of the x- coordinate, we will only subscript the computational variables with the index j, and we will use superscripts to denote the time level.

We assume the standard second order 3-point difference approximation is used for the discretization of the Laplacian and that the integration method in time is forward Euler. We designate the discrete Laplacian by Δ. In the boundary vorticity-stream function implementation, one uses the boundary conditions on the stream function $\Psi = 0$ and $\dfrac{\partial \Psi}{\partial y} = 0$ to conclude that $\Psi_0^n = 0$ and $\Psi_1^n = 0$ for all $n > 0$, i.e. the stream function vanishes at $y = 0$ and one mesh width in ($j = 1$). If the vorticity at time $n\delta t$, ω^n, is known for $j = 2, 3, 4, \ldots$ we show how to obtain the vorticity at time $(n+1)\delta t$, ω^{n+1}.

(i) Solve

$$\Delta \Psi^n = -\omega^n \qquad \Psi_1^n = 0 \quad \text{and} \quad \frac{\Psi_M^n - \Psi_{M-1}^n}{h} = 0$$

to obtain Ψ^n for $j = 1, 2, 3, \ldots$.

(ii) Set

$$\omega_1^n = -\frac{\Psi_2^n - 2\Psi_1^n + \Psi_0^n}{h^2}$$

(iii) Obtain ω^{n+1} by computing an approximate solution to (6) using

$$\omega_j^{n+1} = \omega_j^n + \delta t \nu \Delta \omega_j^n \qquad \text{for} \quad j = 2, 3, 4, \ldots$$

with the Laplacian computed using boundary values ω_1^n from (ii).

The complete algorithm begins with a specification of the initial distribution of vorticity followed with a repetition of steps (i)-(iii) to determine the vorticity at all later times. We note that steps (i)-(ii) are essentially done in order to evaluate ω_1^n - a quantity necessary to evolve the approximation to (6). We now show how steps (i)-(iii) are equivalent to solving (6) with Neumann boundary conditions applied at $j = 1$.

Assuming ω^n is given for $j = 1, 2, 3, \ldots$ we find an expression for ω_1^{n+1}. We have

$$\begin{aligned}
\omega_1^{n+1} &= -\frac{\Psi_2^{n+1} - 2\Psi_1^{n+1} + \Psi_0^{n+1}}{h^2} \\
&= \frac{(\Delta^{-1}\omega^{n+1})_2}{h^2} \\
&= \frac{(\Delta^{-1}[\omega_j^n + \delta t \nu \Delta \omega_j^n])_2}{h^2} \\
&= \frac{(\Delta^{-1}\omega_j^n)_2 + \delta t \nu (\Delta^{-1}\Delta\omega_j^n)_2}{h^2} \\
&= -\frac{\Psi_2}{h^2} + \delta t \nu \frac{(\omega_2^n - \omega_1^n)}{h^2} \\
&= -\frac{\Psi_2^n - 2\Psi_1^n + \Psi_0^n}{h^2} + \delta t \nu \frac{(\omega_2^n - \omega_1^n)}{h^2} \\
&= \omega_1^n + \delta t \nu \frac{(\omega_2^n - \omega_1^n)}{h^2}
\end{aligned}$$

Here we have used extensively that $\Psi_0^n = 0$ and $\Psi_1^n = 0$ for all n. There is some tricky business in the evaluation of $\Delta^{-1}\Delta\omega_j^n$, since the Laplacian being inverted has homogeneous values (because it corresponds to the equation for the stream function), and the Laplacian of the vorticity involves the boundary values of vorticity. To be a bit more explicit we have

$$\Delta^{-1}\Delta\omega_j^n = \begin{bmatrix} & & \\ & A & \\ & & \end{bmatrix}^{-1} \left\{ \begin{bmatrix} & & \\ & A & \\ & & \end{bmatrix} \begin{bmatrix} \omega_2^n \\ \omega_3^n \\ \omega_4^n \\ \vdots \end{bmatrix} + \begin{bmatrix} \frac{\omega_1}{h^2} \\ 0 \\ 0 \\ \vdots \end{bmatrix} \right\}$$

where

$$\begin{bmatrix} & & \\ & A & \\ & & \end{bmatrix} = \begin{bmatrix} -\frac{2}{h^2} & \frac{1}{h^2} & & & \\ \frac{1}{h^2} & -\frac{2}{h^2} & \frac{1}{h^2} & & \\ & \frac{1}{h^2} & -\frac{2}{h^2} & \frac{1}{h^2} & \\ & & & & \ddots \\ & & & & \end{bmatrix}$$

From this last expression, one can deduce that the component for $j = 2$ will be $\omega_2^n - \omega_1^n$.

The equation which governs the values of ω_1,

$$\omega_1^{n+1} = \omega_1^n + \delta t\nu\frac{(\omega_2^n - \omega_1^n)}{h^2}$$

is the approximate solution of (6) with the boundary condition $\dfrac{\partial\omega}{\partial y} = 0$ incorporated by setting $\omega_0^n = \omega_1^n$.

We now discuss a vorticity creation type scheme for computing the solution to these same equations. Assume that the vorticity at time $n\delta t$ induces a stream function which satisfies all boundary conditions, i.e. if we solve

$$\Delta\Psi^n = -\omega^n \qquad \Psi_0^n = 0 \quad \text{and} \quad \frac{\Psi_M^n - \Psi_{M-1}^n}{h} = 0$$

then $\dfrac{\Psi_1^n - \Psi_0^n}{h} = 0$ as well. We now describe how to advance the solution one timestep.

(i) Advance the vorticity with a finite difference approximation to (6) using

$$\tilde{\omega}^{n+1} = \omega^n + \nu\delta t\Delta\omega^n$$

with homogeneous boundary values in the evaluation of the discrete Laplacian for points next to the boundary.

(ii) Calculate the slip on the boundary induced by $\tilde{\omega}^{n+1}$. First solve

$$\Delta\tilde{\Psi}^{n+1} = -\tilde{\omega}^n \qquad \tilde{\Psi}_0^{n+1} = 0 \quad \text{and} \quad \frac{\tilde{\Psi}_M^{n+1} - \tilde{\Psi}_{M-1}^{n+1}}{h} = 0$$

and then set the boundary values of vorticity to be proportional to the slip,

$$\omega_0^n = -\left(\frac{h}{\nu\delta t}\right)\left(\frac{\tilde{\Psi}_1^{n+1} - \tilde{\Psi}_0^{n+1}}{h}\right).$$

(iii) Diffuse the "created" vorticity and construct the solution at $(n+1)\delta t$ by combining this vorticity with $\tilde{\omega}^{n+1}$,

$$\omega^{n+1} = \begin{cases} \tilde{\omega}^{n+1} & \text{for} \quad j = 2, 3, 4 \ldots \\ \tilde{\omega}_1^{n+1} + \delta t \nu \dfrac{\omega_0^n}{h^2} & \text{for} \quad j = 1. \end{cases}$$

It will be shown for this method, like the one presented previously, that the vorticity satisfies a discrete heat equation with Neumann boundary conditions. We have

$$\begin{aligned} \omega_1^{n+1} &= \tilde{\omega}_1^{n+1} + \delta t \nu \frac{\omega_0^n}{h^2} \\ &= \omega_1^n + \frac{\delta t \nu}{h^2}(-2\omega_1^n + \omega_2^n) - \frac{\tilde{\Psi}_1^{n+1}}{h^2}. \end{aligned} \tag{12}$$

Now using the formula for $\tilde{\Psi}^{n+1}$ we find

$$\begin{aligned} \tilde{\Psi}^{n+1} &= -\Delta^{-1}(\omega_n + \delta t \nu \Delta \omega^n) \\ &= \Psi^n - \delta t \nu \Delta^{-1}(\Delta \omega^n) \\ &= \Psi^n - \delta t \nu \omega^n. \end{aligned} \tag{13}$$

Here it is understood that in the term $\Delta\omega^n$ the Laplacian considered is one which uses homogeneous boundary values. By assumption $\Psi_1^n = 0$ so the first component of the above equation in conjunction with (12) yields

$$\begin{aligned} \omega_1^{n+1} &= \omega_1^n + \frac{\delta t \nu}{h^2}(-2\omega_1^n + \omega_2^n) + \delta t \nu \omega_1^n \\ &= \omega_1^n + \frac{\delta t \nu}{h^2}(\omega_2^n - \omega_1^n). \end{aligned} \tag{14}$$

Thus, the vorticity evolves in accordance with a discrete heat equation with Neumann boundary conditions applied at $j = 1$.

This demonstration depended on the fact that the vorticity ω^n induced a stream function such that $\Psi_0 = 0$ and $\Psi_1^n = 0$. In order to continue the argument by induction we must show that $\Psi_1^{n+1} = 0$ holds. This fact follows from the definition of Ψ^{n+1} and the relation (14). Specifically,

$$\begin{aligned} \Psi_1^{n+1} &= (\Delta^{-1}\omega^{n+1})_1 \\ &= (\Delta^{-1}[\omega^n + \frac{\delta t \nu}{h^2}\Delta_N \omega^n])_1 \\ &= \Psi_1^n + (\Delta^{-1}[(\delta t \nu \Delta \omega^n) + \frac{\delta t \nu}{h^2}\omega_1^n])_1 \\ &= \delta t \nu \omega_1^n - \delta t \nu \omega_1^n = 0 \end{aligned} \tag{15}$$

In this last demonstration we have used the notation Δ_N to represent the discrete second derivative approximation which uses homogeneous Neumann boundary conditions.

The fact that each method satisfies the same algebraic equations, an approximation to (6) with Neumann boundary conditions establish that the two methods are equivalent. It is interesting to note that the creation scheme provides values of vorticity on the boundary ω_0^n while the other method doesn't. This suggests that the vorticity creation methods may be particularly useful if one is interested in knowing the values of the vorticity on the boundary. Work is in progress to exploit this property of vorticity creation methods to obtain reliable estimates of the forces acting on solid boundaries. Also, the implementation of the boundary conditions of the vorticity-stream function type depended on being able to invert the boundary conditions $\Psi = 0$ and $\dfrac{\partial\Psi}{\partial y} = 0$ to obtain $\Psi_0 = 0$ and $\Psi_1 = 0$. This inversion is easy with first order differences, but is difficult when higher order differences are used which incorporate more interior points than just Ψ_1.

3. Higher Order Accuracy In Time. As mentioned in the introduction, viewing the vorticity creation boundary conditions as arising from a fractional step procedure is limiting in that it appears the methods in which they are incorporated can only be first order accurate in time. If one considers these creation type schemes from a viewpoint of ensuring that the time derivative of the integral constraint on the vorticity is satisfied, then the technique for deriving higher order methods is easily seen. In order to demonstrate this fact we use the setting of the Prandtl equations. These equations, used to describe flow close to flat boundaries, form a reasonable model problem because when written in the vorticity form the difficulty with the boundary conditions is of the same type as that for the full Navier-Stokes equations. We now present the Prandtl equations and show how one can derive higher order methods in time for their solution. We then discuss a finite difference implementation of this approach and present some computational results.

Let $\vec{u} = (u, v)$ represent the velocity field above a half-infinite flat plate which extends from $x = 0$ to $x = \infty$. In the Prandtl equations the vorticity is given by $\omega = -u_x$, and the equation of evolution for the vorticity is

$$\frac{\partial\omega}{\partial t} + \vec{u}\cdot\nabla\omega = \nu\frac{\partial^2\omega}{\partial y^2} \tag{16}$$

$$u = v = 0 \qquad \text{at} \quad y = 0 \tag{17}$$

$$u = U_0 \qquad \text{at} \quad y = \infty \tag{18}$$

$$u = U_0 + \int_y^\infty \omega(x,s,t)ds \qquad v = -\frac{\partial}{\partial x}\int_0^y u(x,s,t)ds \tag{19}$$

If ω is arbitrary then it follows from (19) that $u = U_0$ at $y = \infty$ and $v = 0$ at $y = 0$. What is not automatically satisfied is the no-slip condition on the plate $u = 0$ at $y = 0$. Thus, as is the case in the Navier-Stokes equations the game is to evolve the vorticity in such a way that this latter no-slip condition at $y = 0$ is satisfied as well. We now derive boundary conditions for the vorticity which will guarantee the no-slip condition.

The integral condition which the vorticity must satisfy in order that the no-slip condition is satisfied is

$$\int_0^\infty \omega(x,s,t)ds = -U_0. \tag{20}$$

This is a condition analogous to (5) for the full Navier-Stokes equation. To ensure that this hold for all time, we assume that the initial vorticity $\omega(x, y, 0)$ satisfies (20) and evolve the vorticity such that the time derivative of (20) is zero. Taking the time derivative under the integral sign and using the equations of motion followed by integration by parts, one obtains

$$\begin{aligned}\int_0^\infty \left(\frac{\partial \omega}{\partial t}\right)ds &= \int_0^\infty \left(-\vec{u}\cdot\nabla\omega + \nu\frac{\partial^2\omega}{\partial y^2}\right)ds \\ &= \int_0^\infty (-\vec{u}\cdot\nabla\omega)ds + \nu\frac{\partial\omega}{\partial y}\Big|_0^\infty\end{aligned}$$

Setting this time derivative to zero gives the boundary condition

$$\frac{\partial\omega}{\partial y}\Big|_{y=0} = \int_0^\infty (-\vec{u}\cdot\nabla\omega)ds. \tag{21}$$

Thus, the equations one solves if one is using boundary conditions of the vorticity creation type are

$$\frac{\partial\omega}{\partial t} = -\vec{u}\cdot\nabla\omega + \nu\frac{\partial^2\omega}{\partial y^2}\omega \tag{22}$$

with boundary conditions

$$\frac{\partial\omega}{\partial y}\Big|_{y=0} = \int_0^\infty (-\vec{u}\cdot\nabla\omega)ds. \tag{23}$$

Reference [1] gives some of the details for understanding the correspondence between methods which incorporate these conditions and the boundary conditions introduced by Chorin in [3].

To derive higher order methods, it is useful to consider the equations written in a more general form. If we denote by ω the values of vorticity in the interior of the domain and the normal derivative of vorticity at $y = 0$ by ω_N, then formally we write (22)-(23) as

$$\frac{\partial\omega}{\partial t} = \mathrm{F}(\omega,\omega_N) \qquad \omega_N = \mathrm{H}\omega$$

or

$$\frac{\partial\omega}{\partial t} = \mathrm{F}(\omega,\mathrm{H}\omega). \tag{24}$$

Here H is a non-linear operator which determines the boundary values in terms of the interior vorticity.

We now just consider (24) as an ordinary differential equation and choose whatever scheme we like. Of course the implementation may not be so simple because of the presence of H, but we at least have a procedure for suggesting methods. One time-stepping scheme which does work out are Runge-Kutta schemes. If we consider time discrete and the spatial derivatives continuous one such method is the following,

$$\omega^* = \omega^n + \delta t\mathrm{F}(\omega^n,\mathrm{H}(\omega^n)) \tag{25}$$

$$\omega^{n+1} = \omega^n + \frac{\delta t}{2}[\mathrm{F}(\omega^n,\mathrm{H}(\omega^n)) + \mathrm{F}(\omega^*,\mathrm{H}(\omega^*))] \tag{26}$$

In this scheme we would use the following expression for $\mathrm{F}(\omega^n,\mathrm{H}(\omega^n))$,

$$F(\omega^n, H(\omega^n)) = -\vec{u}^n \cdot \nabla\omega^n + \nu\frac{\partial^2\omega^n}{\partial y^2}$$

with the boundary condition

$$\frac{\partial\omega}{\partial y}\,|_{y=0} = \int_0^\infty (-\vec{u}^n \cdot \nabla\omega^n)ds$$

used to close the second derivative operator. Similarly for $F(\omega^*, H(\omega^*))$,

$$F(\omega^*, H(\omega^*)) = -\vec{u}^* \cdot \nabla\omega^* + \nu\frac{\partial^2\omega^*}{\partial y^2}$$

with boundary condition

$$\frac{\partial\omega^*}{\partial y}\,|_{y=0} = \int_0^\infty (-\vec{u}^* \cdot \nabla\omega^*)ds$$

and $\vec{u}^*$ is the velocity field constructed from ω^*.

If one considers finite difference schemes to approximate the spatial operators then the implementation of the higher order method is straight forward - one just forms approximation of the terms in the equations in the natural way. The implementation within the context of the vortex blob method or vortex sheet methods is a bit more difficult. One immediate problem is the fact that the approximation of diffusion is typically done using a random walk technique and this is of relatively low order accuracy in time. If one has a more accurate approximation of the diffusion process then it is still not clear how to implement the method - the Lagrangian approximation used makes it difficult to evaluate the term $\vec{u} \cdot \nabla\omega$. One possible way to obtain the pieces necessary to form the second order approximation without having to resort to evaluating $\vec{u} \cdot \nabla\omega$ explicitly is the following: we express the solution at time $(n+1)\delta t$ as

$$\omega^{n+1} = \frac{1}{2}\omega^n + \frac{1}{2}\left[\omega^* + \delta t F(\omega^*, H(\omega^*))\right] \tag{27}$$

ω^* is computed via one step of forward Euler starting with ω^n,

$$\omega^* = \omega^n + \delta t(-\vec{u}^n \cdot \nabla\omega^n + \nu\frac{\partial^2\omega^n}{\partial y^2})$$

with a Neumann boundary condition

$$\frac{\partial\omega^n}{\partial y}\,|_{y=0} = \int_0^\infty (-\vec{u}^n \cdot \nabla\omega^n)ds.$$

The second term on the right of (27) is the vorticity distribution after one forward Euler step approximation to (22)-(23) starting from ω^* with a similar Neumann boundary condition based on ω^*. The solution at time $(n+1)\delta t$ is therefore the sum of ω^n and a vorticity distribution which can be obtained by performing two Euler steps in succession. Since each of these can be carried out within a Lagrangian framework, in principle a higher order method (in time) could be obtained.

In order to test our procedure for developing higher order methods we incorporated these ideas in a finite difference scheme for (16) -(19). The discretization scheme is similar to that described in [1]. The advection term is discretized using a second order centered difference scheme (for simplicity and accuracy) of the form

$$\vec{u} \cdot \nabla \omega \approx \frac{1}{2}[\mathrm{D}_0^x(u\omega) + u\mathrm{D}_0^x\omega] + \frac{1}{2}[\mathrm{D}_0^y(v\omega) + v\mathrm{D}_0^y\omega] = \mathrm{A}_{i,j}(\omega) \tag{28}$$

where D_0 is the central difference operator and the superscript refers to the direction the difference is taken. The second difference operator is discretized using the standard 3-point second order approximation. The following approximations are used for the reconstruction of the velocity from the vorticity,

(29). $$u_{i,j} = U_0 + \sum_{k=j}^{M}(\omega_{i,k} + \omega_{i,k+1})\frac{h}{2}$$

$$v_{i,j} = \sum_{k=1}^{j}(\mathrm{D}_0^y u_{i,j} + \mathrm{D}_0^y u_{i,j})\frac{h}{2} \tag{30}$$

The boundary condition was approximated by

$$\frac{\partial \omega}{\partial y} \approx \sum_{k=j}^{M} -(\mathrm{A}_{i,k} + \mathrm{A}_{i,k+1})\frac{h}{2} \tag{31}$$

where $\mathrm{A}_{i,k}$ is the approximation to the advection term.

δt	$\omega(.25, 0)$	Rate	$\omega(.25, .25)$	Rate
2.5×10^{-2}	-5.97395×10^{-1}	*	-7.04753	*
1.25×10^{-2}	-5.89219×10^{-1}	*	-7.19023	*
6.25×10^{-3}	-5.84495×10^{-1}	.8654	-7.26641	.9366
3.125×10^{-3}	-5.82107×10^{-1}	.9891	-7.30585	.9658

Table 1
Rate Of Convergence for First Order Scheme

δt	$\omega(.25, 0)$	Rate	$\omega(.25, .25)$	Rate
2.5×10^{-2}	-5.86852×10^{-1}	*	-7.39724	*
1.25×10^{-2}	-5.88144×10^{-1}	*	-7.73545	*
6.25×10^{-3}	-5.88013×10^{-1}	2.074	-7.73408	3.315
3.125×10^{-3}	-5.79827×10^{-1}	2.089	-7.73466	2.283

Table 1
Rate Of Convergence for Second Order Scheme

The computational domain was a region $x = h$ to $x = 1.0$ and $y = 0$ to $y = 1.0$. We used $h = .05$, $\nu = .025$, $U_0 = 1.0$. Euler's method and the second order Runge-Kutta method described above were used to advance the solution in time. To verify the accuracy of the schemes, the values of the vorticity at (.25,0.0) (on the plate) and (.25,.25) (in the boundary layer) were monitored. These quantities were evaluated for different time steps and by using Aitken extrapolation a rate of convergence was estimated. In Table 1 and

Table 2 we give the values of the vorticity at these two locations at time t=0.25. In the column just to the right of the values of vorticity is the estimate of the rate of convergence corresponding to those values.

These results demonstrate that it is possible to implement higher order methods and use vorticity creation type boundary conditions. One pleasant fact is that the boundary values of the vorticity are also be determined with second order accuracy in time.

4. Satisfaction of Boundary Conditions. In this section we discuss some of the problems which arise when one does not satisfy the boundary conditions on the velocity to the precision of roundoff error, but only to the precision of the truncation error of the underlying numerical scheme. The manner in which the problem arises is well described in the setting of the model problem from the last section - the Prandtl boundary layer equations. In these equations we have the evolution equations for the vorticity in the interior of the domain

$$\frac{\partial \omega}{\partial t} + \vec{u} \cdot \nabla \omega = \nu \frac{\partial^2 \omega}{\partial y^2} \tag{32}$$

and a constraint

$$\int_0^\infty \omega(x,s,t)ds = -U_0. \tag{33}$$

If we assume that the initial vorticity satisfies this constraint then a boundary condition which, if incorporated in (32), will guarantee (33) for $t > 0$ is

$$\frac{\partial \omega}{\partial y}|_{y=0} = \int_0^\infty (-\vec{u} \cdot \nabla \omega)ds. \tag{34}$$

This boundary condition works because it ensures that the time derivative of the constraint vanishes.

Given a discretization procedure for the evolution of the vorticity in the interior and a method for reconstructing the velocity from the vorticity, there is a discrete equivalent to the constraint (33). If this discrete constraint is satisfied, then the discrete vorticity will induce a discrete velocity which satisfies the boundary conditions exactly (i.e. to roundoff). In the design of boundary conditions of the vorticity creation type, one has a choice of approximating (34) using any convenient consistant discretization, or approximating (34) using a discretization which is based on the other approximations made in the problem and one which ensures that the discrete constraint (33) is satisfied exactly. In the first case the implementation will most likely only ensure that the discrete constraint is satisfied to the order of the truncation error of the approximations used in the boundary condition. This will cause velocities on the order of the truncation error to occur on the boundary. On the other hand, the boundary conditions which ensure that the discrete constraint is satisfied to roundoff at every time step will provide vorticity distributions which induce velocities on the order of roundoff at the boundary. (These vorticity boundary conditions are derived by following arguments similar to those used to derive the continuous ones, but using the discrete operators associated with the approximations of the interior equations and the reconstruction of the velocity from the vorticity.) The difference between "exact" and "truncation error" boundary conditions seems minor but it can lead to methods with very different behavior.

It is clear that if one is integrating the equations up to a fixed time and the solution of the equations is smooth then using "exact" boundary conditions or "truncation error" boundary conditions should lead to the same solution. Differences in the methods may occur when the solution is not smooth or when one is interested in the long time behavior of the solutions.

In order to gain some insight into these questions we considered two finite difference schemes for the Prandtl boundary layer equations. The first was that based on the finite difference approximations given in the last section. As discussed in [1] it can be shown that the discrete equivalent of (33) is satisfied at every time-step. The second scheme we used differed from the one just mentioned in that Simpson's (as opposed to the trapezoidal rule in (29)) was used in the construction of the u velocity from the vorticity. Simpson's rule is fourth order accurate and the change should yield a method of comparable accuracy, and possibly one which is more accurate. However, in this modified scheme the vorticity boundary condition (31), while being a natural discretization of the continous boundary condition (34), did not guarantee that the discrete constraint (now based on Simpson's rule) was satisfied to roundoff.

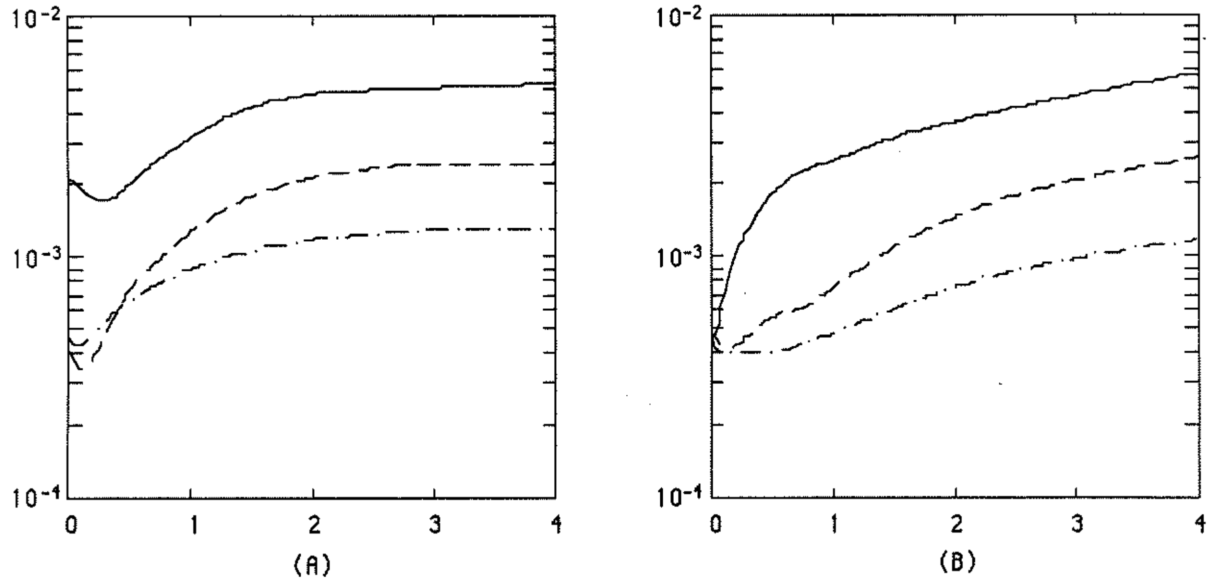

Figure 2

L^2 Error in U Velocity Profile at X=1.0 vs. Time

(A) : Satisfaction of Discrete Boundary Conditions to Roundoff

(B) : Satisfaction of Discrete Boundary Conditions to Trunc. Error

——— : $h = 0.1$, – – – – : $h = 0.05$, – · – · – : $h = 0.025$

In the first experiment we started close to a steady state profile (furnished by the Blasius solution) and observed the capability of each of the schemes to latch onto the steady state solution and remain there. To avoid the problem with the singularity of the solution near the leading edge of the plate we used a domain which did not include the point $x = 0$. The computational domain was from $x = .5$ to $x = 1.5$ and $y = 0.0$ to $y = 1.0$. We used $\nu = 0.025$ and $U_0 = 1.0$. In Figure 1(A) and 1(B) we plot the discrete L^2 error in the u velocity profile at the station $x = 1.0$ for three different mesh widths, $h = 0.1$, $h = .05$ and $h = 0.025$. The time steps for all of the solutions was $\delta t = 0.0025$ and the solution was advanced using forward Euler. Figure 1(A) corresponds to the original method in which the discrete constraint is satisfied at every time step. As can be seen, after an initial transient period the method converges to a steady solution and the errors do not increase in time. During the complete run the magnitude of the velocity at the boundary was on the order of roundoff (10^{-6}). In Figure 1(B) we see the method based on Simpson's rule does not latch onto a steady state solution. Although the error in the profiles is less for this method up to a time between $t = 3.0$ and $t = 4.0$, the errors definitely grow in time. Further computational results showed the errors didn't stop growing. Typically the errors grew until the velocities in the computed solution caused the advection scheme to go unstable. In figure 2 we show the L^2 error in the slip velocity along the plate from $x = 0.5$ to $x = 1.5$ for the method which used Simpson's rule. As expect the errors grew in time with the rate which diminished as the truncation error of the spatial discretization diminished.

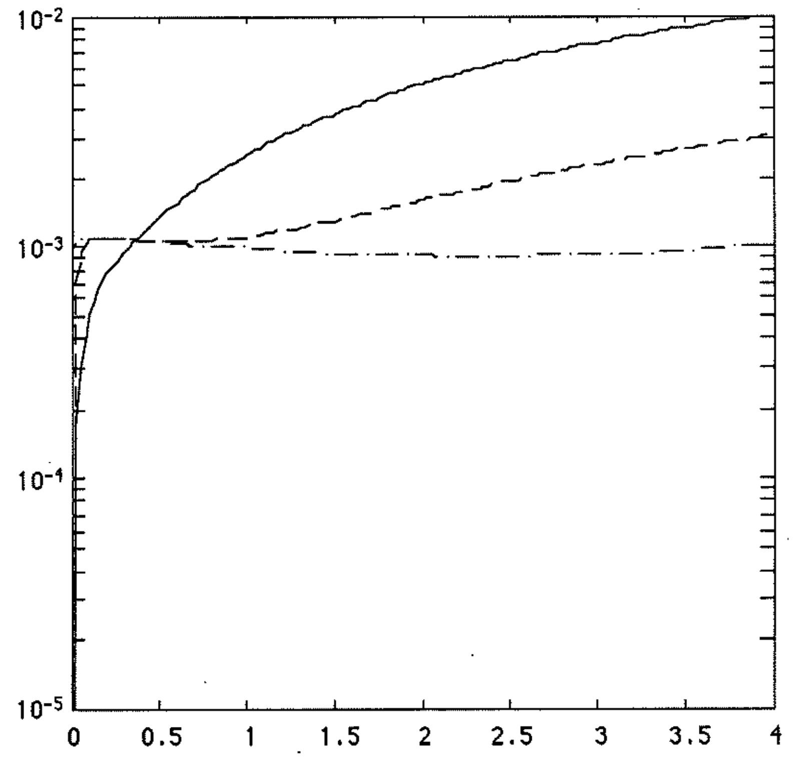

Figure 3

L^2 Norm of U Velocity Along Plate (y=0) vs. Time

Satisfaction of Discrete Boundary Conditions to Trunc. Error

——— : $h = 0.1$, – – – – : $h = 0.05$, – · – · – : $h = 0.025$

In the second experiment we wanted to test if both methods were capable of finding the Blasius profile beginning from an impulsively started plate. The computational domain was from $x = h$ to $x = 1.0$ and $y = 0.0$ to $y = 1.0$. As before $\nu = 0.025$ and $U_0 = 1.0$. In Figure 3 we plot the L^2 error in the u velocity profile at $x = .5$ as a function of time. The lower curves correspond to different mesh sizes used in the original scheme and the upper curves correspond to different mesh sizes with the scheme which employed Simpson's rule. The original method, that in which the constraint was satisfied exactly, had no difficulty in computing a steady solution. The other method did not work at all. The solution never converged to a steady state. The solutions obtained with the two methods did agree for a very short time, but it is not clear that interval of time would increase as the mesh size was

[6] M. Israeli and S.A. Orszag, *Numerical Simulation of Viscous Incompressible Flows*, Ann. Rev. of Fluid Mechs., 6, pg. 281, 1974.