JOURNAL OF COMPUTATIONAL PHYSICS 121, 29–50 (1995)

On the Rokhlin-Greengard Method with Vortex Blobs for Problems Posed in All Space or Periodic in One Direction

JOHN T. HAMILTON AND GEORGE MAJDA

Department of Mathematics, The Ohio State University, 231 West 18th Avenue, Columbus, Ohio 43210-1174

Received April 4, 1994; revised December 19, 1994

In this paper we consider the Rokhlin–Greengard (R-G) fast multipole algorithm when used to evaluate vortex blob interactions in a two-dimensional fluid. We use exact solutions of the incompressible Euler equations to demonstrate that the R-G algorithm can compute vortex blob interactions accurately. However, we also show that the structure of vortex blobs forces a practical limitation on the highest (finest) bisection level one can use in the R-G algorithm, a restriction which does not apply when point vortices are used. If this maximum bisection level is exceeded, then the accuracy of the R-G algorithm with blobs may be significantly reduced. A similar constraint should hold in three dimensions. We also extend the R-G algorithm with blobs to problems which are periodic in one spatial dimension and unbounded in the other, and we document the performance of the resulting algorithm using some exact periodic solutions of the incompressible Euler equations. © 1995 Academic Press, Inc.

1. INTRODUCTION

Vortex methods are a well-known class of numerical methods for solving the incompressible Euler and Navier–Stokes equations. They are particularly useful when the vorticity distribution in a fluid is concentrated in a relatively small region of the flow or when the vorticity is not smooth. A vortex sheet is an example of a flow which has both the above characteristics.

Vortex methods were first used by Rosenhead [37] to try to simulate the roll-up of a vortex sheet. The modern seminal paper on vortex methods is Chorin [17], and the subsequent development of the theory of vortex methods as well as applications of these methods has been rapid. Theoretical justification of the vortex blob method for inviscid fluids can be found in [23, 24, 12, 13, 7, 18, 10]. Vortex methods and their applications have been the subject of survey articles including [32, 33, 38], and they have been the main theme of several conferences, including [9, 16, 22, 8, 11].

An important practical limitation on the use of vortex methods is their high operation count. A vortex method using n point vortex particles (or blobs) requires $\mathbb{O}(n^2)$ floating point operations (see the first paragraph in Section 3). This situation changed dramatically with the development of fast vortex methods which reduce this operation count to either $\mathbb{O}(n \log n)$ or $\mathbb{O}(n)$ operations. Fast vortex methods in two and three spatial dimensions are described in [4, 21, 20, 41, 5, 6, 19, 27, 15, 1, 2]. In this paper we focus on the Rokhlin–Greengard (henceforth, R-G) fast multipole method [21, 20] with vortex blobs in two dimensions, although some of our results should also apply to its three-dimensional implementation and to other multipole-like methods (see [6, 19]).

In its original form, the R-G algorithm applies to point vortex calculations in two dimensions. Some investigators [5, 6, 15] have done timing studies with the R-G vortex blob method and concluded that the use of vortex blobs does not significantly increase the running time of the R-G algorithm. In this paper we use an exact solution of the two-dimensional Euler equations to demonstrate that, due to the structure of vortex blobs, there is a maximum bisection level for the R-G multipole method with the property that the accuracy of the multipole method with blobs may deteriorate significantly when this level is exceeded. Therefore, the structure of vortex blobs induces a practical limitation on the maximum bisection level if the velocity field evaluated by the R-G algorithm is to accurately approximate the velocity field obtained through evaluation by the direct $\mathbb{O}(n^2)$ algorithm. A similar result should hold for three-dimensional implementations of the R-G algorithm with blobs or any other multipole-like method.

In this paper we also show how to extend the R-G algorithm with vortex blobs to two-dimensional problems which are periodic in one spatial direction and unbounded in the other direction. We document the performance of the resulting algorithm using some exact solutions of the incompressible Euler equations.

This paper is organized in the following way: In Section 2 we briefly review the vortex blob method and introduce some notation. In Section 3 we review the details of the R-G algorithm which are essential for our presentation. We then show how to accurately incorporate vortex blobs into this algorithm and obtain a practical formula which relates the maximum bisection level to the blob size (and other method parameters) and which indicates when the R-G algorithm with blobs will produce results which are essentially identical to those obtained with the $\mathbb{O}(n^2)$ algorithm. We conclude Section 3 by showing how

0021-9991/95 $12.00
Copyright © 1995 by Academic Press, Inc.
All rights of reproduction in any form reserved.

to extend the R-G algorithm with blobs to two-dimensional problems which are periodic in one spatial direction and unbounded in the other direction. In Section 4 we use an exact solution of the incompressible Euler equations in all of space to determine the accuracy of the R-G algorithm with blobs and the acceptable bisection level for the R-G algorithm. In Section 5 we consider two different exact solutions of the two-dimensional Euler equations where the support of the vorticity is periodic in one spatial direction but not in the second direction. We use these explicit solutions to document the performance of our periodic extension of the R-G method with blobs. We include a summary and some conclusions in Section 6.

2. THE VORTEX METHOD

The flow of a homogeneous incompressible inviscid fluid is determined by the Euler equations

$$\frac{Du}{Dt} \equiv \frac{\partial u}{\partial t} + (u \cdot \nabla)u = -\frac{1}{\rho}\nabla p, \tag{2.1}$$

subject to the incompressibility condition

$$\nabla \cdot u = 0. \tag{2.2}$$

Here $u = u(z, t)$ is the velocity, p is the pressure, and ρ is the fluid density. In this paper we are concerned only with flows in two spatial dimensions, so $z = (x, y)$, $u = (u_1, u_2)$, and the development below is specialized to this case. One may think of the vectors z and u as elements of $\mathbf{R}^2$ or of $\mathbf{C}$, and we shall use whichever viewpoint seems more convenient at a particular time. Thus, we shall write either $z = (x, y)$ or $z = x + iy$; the appropriate interpretation will be clear from the context.

The vorticity form of (2.1), obtained by taking the curl of (2.1), is given by

$$\frac{D\omega}{Dt} \equiv \frac{\partial \omega}{\partial t} + u \cdot \nabla\omega = 0. \tag{2.3}$$

Here $\omega = \partial u_2/\partial x - \partial u_1/\partial y$ is the only component of the vorticity vector $\omega = \nabla \times u$ which is not identically zero. As the material derivative D/Dt gives the change in a fluid parameter as it moves with the flow, (2.3) shows that the vorticity of a fluid element is preserved by the flow.

Vortex methods provide a numerical technique for solving the Euler equations. They are derived and analyzed in many references, including [23, 13, 7]. Consequently, we will present only a brief description of the method here.

Application of vortex methods in two dimensions is based on two primary results. First, Eq. (2.3) implies that the vorticity associated with a fluid element is constant along the solution curves of the *particle trajectory equations*,

$$\frac{d}{dt} z(t) = u(z(t), t). \tag{2.4}$$

Second, the kinematic relationships

$$\begin{pmatrix} -u_2 \\ u_1 \end{pmatrix} = \nabla\Psi \quad \text{with } \Delta\Psi \equiv \frac{\partial^2\Psi}{\partial x^2} + \frac{\partial^2\Psi}{\partial y^2} = -\omega, \tag{2.5}$$

where Ψ denotes a stream function, show that the velocity field can be computed from the vorticity field. In a two-dimensional space without boundaries this relationship is given explicitly by the Biot–Savart Law,

$$\begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = \int K(z - \tilde{z})\omega(\tilde{z})\, d\tilde{\mathbf{z}} = (K * \omega)(z), \tag{2.6}$$

where the kernel $K(z)$ is given by

$$K(z) = -\frac{1}{2\pi|z|^2}\begin{pmatrix} y \\ -x \end{pmatrix}. \tag{2.7}$$

To obtain a vortex method, superpose a grid parallel to the x- and y-axes with spacing h in each direction, and let $z_i(t)$ denote the location at time t of a fluid element originally located at the grid node ζ_i. (Here the subscript $i = (i_1, i_2)$ is a multi-index characterizing the points of the grid used in the discretization.) Approximate (2.4), (2.6) by discretizing the integral in (2.6), that is, by replacing the integral in (2.6) by a sum over vortices with strength ω_i located at $z_i(t)$. This procedure produces the approximate particle trajectory equations

$$\frac{d}{dt} z_i(t) = u_h(z_i(t), t) \quad \text{with } z_i(0) = \zeta_i, \tag{2.8}$$

where u_h is an approximate velocity field.

In the point vortex method the approximate vorticity field ω_h is given by

$$\omega_h(z) = \sum_j \omega_j \delta(z - z_j(t)), \tag{2.9}$$

where $\delta(z)$ denotes the Dirac delta distribution, and the approximate velocity field is given by

$$u_h(z) = \sum_j \omega_j K(z - z_j). \tag{2.10}$$

In each of these equations the term with $j = i$ is omitted if $z = z_i$.

In the vortex blob method

$$\omega_h(z) = \sum_j \omega_j \psi_\delta(z - z_j(t)) \tag{2.11}$$

and

$$u_h(z) = \sum_j \omega_j K_\delta(z - z_j(t)) \tag{2.12}$$

where $K_\delta = K * \psi_\delta$ is the convolution of the singular kernel K with an appropriate mollifier $\psi_\delta = \delta^{-2}\psi(z/\delta)$. Various choices for ψ have been proposed. For example, kernels mollified by rational functions can be found in [34]. We will use a function developed by Beale and Majda [14] which satisfies, in addition to $\int \psi = 1$, the conditions:

$$\text{(i)} \quad |D^\beta \psi(z)| \le C_{\beta j}(1 + |z|^2)^{-j} \tag{2.13a}$$

for some constant $C_{\beta j}$ for every multi-index β and every integer j, and

$$\text{(ii)} \quad \int z^\beta \psi(z)\, d\mathbf{z} = 0 \tag{2.13b}$$

for every β satisfying $1 \le |\beta| \le \hat{p} - 1$ for some integer $\hat{p}$.

Condition (i) requires that ψ be smooth and rapidly decreasing, while (ii) requires that a certain number of moments vanish. (The condition $\int \psi = 1$ specifies the zeroth moment.) Beale and Majda [12, 13] and Anderson and Greengard [7] have shown that for smooth initial conditions vortex blob methods based on a function ψ satisfying these conditions will converge, provided the shape parameter δ is chosen to satisfy $\delta = h^q$, where q is fixed with $0 < q < 1$ (so as h is less than 1, δ is somewhat larger than h). With this value of δ, the rate of convergence is $\delta^{\hat{p}} = h^{\hat{p}q}$ for the discrete particle trajectories and velocities.

We have performed our computations with the Beale–Majda fourth-order kernel [14], obtained by taking ψ as a linear combination of two Gaussians with different scalings,

$$\psi^{(4)}(r) = \frac{1}{\pi}(2e^{-r^2} - e^{-r^2/2}), \tag{2.14}$$

where $r = |z|$. This yields the fourth-order kernel

$$K_\delta^{(4)}(z) = -\frac{1}{2\pi r^2}(1 - 2e^{-r^2/\delta^2} + e^{-r^2/2\delta^2}) \begin{pmatrix} y \\ -x \end{pmatrix}. \tag{2.15}$$

We shall refer to this kernel hereafter simply as K_δ. Note that $K_\delta \approx K$ if $e^{-r^2/2\delta^2} \ll 1$, so the latter condition allows us to determine when two blobs see each other as point vortices. This will be important in the application of the fast particle algorithm in Section 3.

3. A FAST ALGORITHM FOR COMPUTATION OF VELOCITIES

In the calculation of velocities in Section 2, whether by (2.10) for point vortices or (2.12) for vortex blobs, to evaluate the velocity of the ith particle we must sum the velocities induced on that particle by each of the other particles. Thus, if there are n particles in all, the calculation of the velocity of each particle requires work of $\mathbb{O}(n)$ floating point operations, and the work required for the calculation of the velocities of all n particles will be $\mathbb{O}(n^2)$. For large values of n a method which requires fewer than $\mathbb{O}(n^2)$ floating point operations is needed to keep computational times within reasonable bounds, and considerable attention has been given to the development of such methods. One such method is the *fast multipole algorithm* developed by Rokhlin and Greengard [20, 21] which provides, under suitable conditions, a method which requires $\mathbb{O}(n)$ floating point operations and which yields velocities which agree, to within a predetermined accuracy, with those obtained by direct calculations.

The R-G algorithm is designed for calculations on ensembles of particles which interact *via* a Coulombic type law. In two dimensions, if r is the distance between the interacting particles, this means an interaction which may be described by a potential varying as $\log r$ or by a force or an induced velocity varying as $1/r$. Here the particles are located at the points z_j and z_k in the complex plane, and $r = |z_j - z_k|$. Thus, the algorithm is designed for the calculation of a potential field

$$V(z) = -\sum_{j=1}^{n} e_j \log|z - z_j| = \sum_{j=1}^{n} e_j \Re(-\log(z - z_j))$$

appropriate to a collection of point charges of strengths e_j located, respectively, at the points z_j. (Here $\Re$ indicates that the real part is to be taken.) The force at a point z is given by

$$F(z) = -\nabla V(z) = \sum_j \frac{e_j}{|z - z_j|^2}(z - z_j) = \sum_j \frac{e_j}{z^* - z_j^*}, \tag{3.1}$$

where * indicates the complex conjugate and where the term with $j = i$ is omitted from the summation if $z = z_i$. Note that from (2.7) the kernel $K(z)$ may be written as $K(z) = i/(2\pi z^*)$. Hence, from (2.10),

$$u(z) = \frac{i}{2\pi}\sum_j \frac{\omega_j}{z^* - z_j^*}, \tag{3.2}$$

so computation of velocities in the fluid dynamics problem is just (3.1) with e_j replaced by $i\omega_j/(2\pi)$.

We will first review the application of the R-G algorithm to an ensemble of n point vortices. Next, we point out certain restrictions which are necessary in the application of the method to an ensemble of vortex blobs. We then conclude this section

with a description of the extension of the method to problems periodic in one spatial dimension.

A. *Ensemble of Point Vortices*

We start by describing the R-G algorithm, including details only as necessary for the current work. See Chapters 1 and 2 of [20] for more detail. The idea of the algorithm is to break the calculations of the velocity field into two parts. Write (2.10) as

$$u(z) = u_h^{(n)} + u_h^{(f)} = \sum_{z_j \text{ near } z} K(z - z_j)\omega_j + \sum_{z_j \text{ far from } z} K(z - z_j)\omega_j, \tag{3.3}$$

where the meaning of "near" and "far from" will be explained below. The first summation in (3.3) is calculated directly, using the kernel (2.7). The R-G algorithm replaces the second summation of (3.3) by a Taylor series whose region of convergence includes the point z. By retaining a sufficient number of terms in this series, the contribution of the second summation in (3.3) to the velocity field can be computed to any desired accuracy consistent with the floating point precision.

Assume that our planar ensemble of point vortices, or particles, lies in a bounded region of space. Enclose the ensemble in a square, which we take here to have side 1. This is the *computational domain* and the only box at level zero. Bisect the computational domain both horizontally and vertically to obtain four squares of side $\frac{1}{2}$, these being the four boxes of level 1. These are the *children* of the level 0 box, and the level 0 box is the *parent* of these four boxes. Continue this process as many times as desired, obtaining at level l a partition of the computational box into 4^l squares, each of side $s_l = 2^{-l}$. By *higher level* is meant a level with a smaller value of l, and by *lower level* a level with a larger value of l. Thus, the lowest level in this box structure corresponds to the largest value of l, denoted by $l_{\max}$, or to the level with the finest mesh. For any box the collection of *nearest neighbour boxes* is the set of all other boxes at the same level which are adjacent to the given box, with two boxes being considered adjacent if their boundaries have any point in common. The *interaction list* for a given box, B, is the collection of boxes at the same level as the box B whose parents are nearest neighbours of the parent of B but which are not themselves nearest neighbours of B. See Fig. 1, where the boxes labeled "N" are nearest neighbours of the box labeled "B," and the boxes labeled "I" comprise the interaction list of box "B".

Now we can define the ideas of "near" and "far from" which appear in Eq. (3.3). Each particle of the ensemble is contained in a certain box at the lowest level (precautions being taken to assign a particle which lies on a boundary to a unique box). For a particle in a given box B at the lowest level, the particles "near" that particle are the other particles in the same box and all particles in the boxes which are nearest neighbours of B. All other particles are "far from" the given particle.

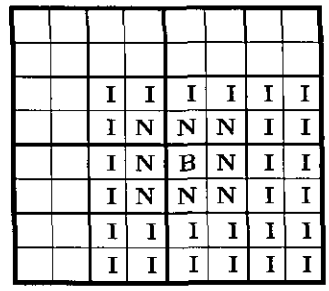

FIG. 1. Relation of a box B to its nearest neighbour boxes (N) and the boxes on its interaction list (I). If the outer border encloses the computational domain, this partitioning has $l_{\max} = 3$.

Thus, for a particle in the lowest level box B the velocity induced by all other particles in that box and by the particles in the boxes which are nearest neighbours of B is obtained by direct calculations. (This is the term $u_h^{(n)}$ of (3.3).) The velocity induced by all other particles (the term $u_h^{(f)}$ of (3.3)) is calculated by the Taylor series produced by the R-G algorithm.

The steps of the algorithm may be summarized as follows, where steps (1) and (2) are known as the *upwards pass* and steps (3) through (5) are known as the *downwards pass*:

(1) For each box B at the lowest level construct a Laurent series which converges at each point outside some circle containing B to the velocity induced at that point by the particles inside box B.

(2A) For each box $\tilde{B}$ at the next higher level use the Laurent series constructed in (1) for the children of $\tilde{B}$ to construct a Laurent series for $\tilde{B}$ which converges at each point outside some circle containing $\tilde{B}$ to the velocity induced at that point by the particles inside box $\tilde{B}$.

(2B) Repeat (2A) for each higher level, finishing with the single box at level 0.

(3) The circles of steps (1) and (2) are chosen so that all boxes other than the box in question and its nearest neighbours lie completely outside the circle. Hence the Laurent series for a given box converges to the velocity induced by particles inside that box at all points outside that box and its nearest neighbours. Assume that we have, at a given level l, a Taylor series about the center of each box which gives the velocity induced by all particles outside that box and its nearest neighbours. (This series is vacuous at levels 0 and 1.) Use this series to construct, for each child of a box at level l, a Taylor series which converges to the velocity induced by particles other than those in that level $l + 1$ box, its nearest neighbours and the boxes on its interaction list.

(4) Now add to the Taylor series generated by (3) the contributions from the boxes on the interaction list of the box at level $l + 1$ for which we are calculating the Taylor series. Each of these contributions is obtained by using the known Laurent series for the box on the interaction list to construct a

Taylor series around the center of the level $l+1$ box which has, at points in the level $l+1$ box in question, a sum equal to that of the original Laurent series.

(5) Repeat steps (3) and (4) for each lower level down to the lowest level.

Now we develop the ideas above in more detail. First consider a box B at the lowest level, with center z_0, and suppose that there is a particle in this box at z_j with vorticity ω_j. Let $\mathcal{C}_R$ be a circle centered at z_0 and of radius large enough to include all of B. In particular $R > |z_0 - z_j|$ for each particle contained in B. We may take the complex conjugate of (3.2) to obtain

$$[u(z)]^* = -\frac{i}{2\pi}\sum_j \frac{\omega_j}{z - z_j}, \tag{3.4}$$

where $[u(z)]^*$ is called the *conjugate velocity*. (It is the meromorphic function $[u^*(z)]$ that is actually computed by the algorithm, with the velocity $u(z)$ being recovered at the end.) Then if z is any point outside $\mathcal{C}_R$, we can describe the velocity at z induced by the particle at z_j by a Laurent series in $z - z_0$ simply by expanding

$$\frac{1}{z - z_j} = \frac{1}{(z - z_0)\left(1 - \dfrac{z_j - z_0}{z - z_0}\right)} = \frac{1}{z - z_0}\sum_{k=0}^{\infty}\left(\frac{z_j - z_0}{z - z_0}\right)^k. \tag{3.5}$$

For each particle in the box B use (3.5) to convert the corresponding term of (3.4) to a Laurent series about z_0, add these series, each multiplied by the appropriate ω_j, and multiply the resulting sum by $(-i/2\pi)$. The result is a Laurent series centered at z_0 which represents, at points outside $\mathcal{C}_R$, the conjugate velocity induced by all particles in B. By retaining a sufficient number of terms of the last series, we thus can obtain the conjugate velocity induced by the particles in B at an arbitrary point z outside $\mathcal{C}_R$ to any specified accuracy. No further consideration of the individual particles is required. The series for the conjugate velocity represents an analytic function and will, therefore, have a Taylor series representation in any circle inside its region of convergence.

Move to the next higher level and consider the box $\tilde{B}$ at that level which is the parent of box B. Let $\tilde{z}_0$ be the center of $\tilde{B}$, and let $\mathcal{C}_{\tilde{R}}$ be a circle centered at $\tilde{z}_0$ and of radius $\tilde{R}$ which is sufficiently large so that $\mathcal{C}_{\tilde{R}}$ is contained in $\mathcal{C}_{\tilde{R}}$. For a point z such that $|z - \tilde{z}_0| > \tilde{R}$ we can, by computations similar to those used to obtain (3.5), rewrite the Laurent series centered at z_0 as a new Laurent series centered at $\tilde{z}_0$ and convergent outside $\mathcal{C}_{\tilde{R}}$. (This process is referred to by Rokhlin and Greengard as "translation of a multipole expansion," as their series for the potentials include a logarithmic term as well as the Laurent series.) In the same way, translate the Laurent series for the other three children of $\tilde{B}$ and combine these four translated series to yield a single Laurent series, centered at $\tilde{z}_0$, which yields, for all points outside $\mathcal{C}_{\tilde{R}}$, the conjugate velocity induced by all particles inside the box $\tilde{B}$. Continuing this process, we eventually obtain a collection of Laurent series, one associated with each box at each level and each of which can be used for points outside the corresponding circle of convergence to compute the conjugate velocity induced by all particles inside the associated box. Using the notation of Rokhlin and Greengard, we denote by $\Psi_{l,i}(z)$ the Laurent series associated with box $B_{l,i}$, the ith box at level l (where $1 \le i \le 4^l$).

Now we reverse the process, using the Laurent series constructed above to produce for each box a Taylor series $\Phi_{l,i}(z)$ which yields, at each point within the box $B_{l,i}$, the conjugate velocity induced at that point by all particles other than those contained in $B_{l,i}$ itself and its eight (or fewer) nearest neighbour boxes.

Start at level 2 (boxes at levels 0 and 1 have nothing but nearest neighbour boxes). Recall that the side of a box at level l is $s_l = 2^{-l}$. The inner radius R_l of the region of convergence for the Laurent series corresponding to the boxes at level l is $\le s_l/\sqrt{2}$, so it satisfies $R_l < 1.5 s_l$. Hence all boxes other than the box in question and its nearest neighbours are completely contained in the region of convergence. Thus the velocity at a point z inside box $B_{2,j}$ induced by all particles outside of box $B_{2,j}$ and its nearest neighbours is given by $\sum_i' \Psi_{2,i}(z)$, where the prime on $\sum$ indicates that the sum is to be taken over all level 2 boxes except for $B_{2,j}$ and its nearest neighbours. As this series converges for all points inside $B_{2,j}$, it can be converted to a Taylor series centered at the center of $B_{2,j}$. We thus find the series $\Phi_{2,j}(z)$ for each box $B_{2,j}$ of level 2. (These calculations lead to expressions of the type given by (3.6), (3.7), and (3.8), below.)

Suppose that we know the series $\Phi_{l,i}(z)$ for each of the boxes of level l. The computation of the Taylor series for the boxes of level $l+1$ proceeds in two steps. For a given box $B_{l+1,j}$ first translate the Taylor series for the parent of $B_{l+1,j}$ to obtain a Taylor series $\tilde{\Phi}_{l+1,j}(z)$ around the center of $B_{l+1,j}$. This is the series we want, except for the fact that it does not include the contributions of the boxes on the interaction list of $B_{l+1,j}$. (Examine Fig. 1 to see why this is true.) Accordingly, we use the Laurent series for the boxes on the interaction list for $B_{l+1,j}$ to obtain their contributions to the Taylor series for box $B_{l+1,j}$ and add these contributions to the series $\tilde{\Phi}_{l+1,j}(z)$ to obtain the Taylor series $\Phi_{l+1,j}(z)$. By continuing this process we obtain the Taylor series $\Phi_{l_{\max},i}(z)$ for the boxes at the lowest level.

For a particle at the point z_k in the box $B_{l_{\max},i}$, with the center of the latter at $\bar{z}_i$, we find the induced velocity by first evaluating $\Phi_{l_{\max},j}(z_k - \bar{z}_i)$ and then adding (by direct calculations) the contributions of the particles inside $B_{l_{\max},i}$ and its nearest neighbours, thus completing the calculations for the case of an ensemble of point vortices in a bounded region of space.

B. *Ensemble of Vortex Blobs*

The R-G algorithm assumes point vortices, and this assumption is fundamental to the calculations of the algorithm. For

example, the translation of Eq. (3.5) is based on the fact that the quantities involved have a $1/r$ form, and calculations in the other parts of the algorithm are based on similar considerations. This fact imposes restrictions on the application of the R-G algorithm to calculations with vortex blobs.

Vortex blobs have an internal structure, so we must ensure that any pair of blobs whose interaction is calculated by the R-G algorithm are sufficiently separated from one another that—up to the accuracy of the calculations—they see each other as point vortices (see the final remarks in Section 2). For the Beale–Majda blobs with $p = 4$, for example, this will be true provided $(1 - 2e^{-r^2/2\delta^2} + e^{-r^2/2\delta^2}) \approx 1$, where δ is the shape parameter, and this condition will be satisfied if $e^{-r^2/2\delta^2} \ll 1$. In principle, then, if we want an accuracy of ε, we should require that $e^{-r^2/2\delta^2} \leq \varepsilon$ for any pair of blobs treated as point vortices in the calculations, which would mean $r \geq \delta(-2 \log \varepsilon)^{1/2}$. The largest value of $e^{-r^2/2\delta^2}$ occurs when a particle in a computational box B is separated from a particle in one of the boxes on its interaction list precisely by $s_{l_{\max}}$, a very special situation which we call the *worst case*. As δ is somewhat larger than the grid spacing h, we find that if we assume this ε to be machine epsilon for single precision accuracy on the machine used (programming was in FORTRAN on a Sun SPARC 10, Model 41, with machine epsilon of $2^{-23} \approx 1.2 \times 10^{-7}$), we would require a separation of $r \geq 5.68\delta > 5.68h$ in order that all blobs whose interactions are treated by the R-G algorithm see each other as point vortices. In practice, however, for most far-field blob–blob interactions the value of $e^{-r^2/2\delta^2}$ is much smaller than the worst case value. Thus it is overly restrictive to impose the condition $e^{-r^2/2\delta^2} \leq \varepsilon$ with ε equal to machine epsilon. We find that for single precision work it is quite satisfactory to take $\varepsilon \approx 1 \times 10^{-4}$, thus obtaining the restriction $r \geq 4.3\delta$.

For a given ε, we can connect ε, h, and $l_{\max}$ as follows: Let $\varepsilon = 10^{-k}$ and let $h = 2^{-\nu}$. Then the restriction $s_{l_{\max}} \geq (-2 \log \varepsilon)^{1/2}\delta$ may be written as

$$l_{\max} \leq q\nu - \tfrac{1}{2} \log_2 k - \alpha,$$

where $\alpha = 0.5(1 + \log_2(\log 10)) \approx 1.1016$. For application, it is convenient to plot level curves of integral values of the right-hand side of the last inequality on a graph whose axes represent k and $q\nu$.

If the limitation on level discussed in the preceding paragraph holds, so that $l_{\max} \approx -\log_2(5h)$, the velocity field computed by the R-G algorithm will agree with that computed by the direct method. While this limitation on $l_{\max}$ is specific to the Beale–Majda blobs using the fourth-order kernel, similar considerations would apply to blobs of other shapes. In Sections 4 and 5B we will see the inaccuracies which may arise if these restrictions are violated.

In order for the work involved in the R-G algorithm to be of $\mathbb{O}(n)$ floating point operations, it is necessary that there be a bound on the number of particles in each box at the lowest level in the box structure (see [20]). However, when using blobs, if we restrict the maximum level in the box structure to that permitted by the restriction $r \geq 4.3\delta$ discussed above the average number of particles per box, $(N/2^l)^2$, will be unbounded as $h \to 0$. The reason is that the restriction on r requires that $N/2^l \geq h^{q-1}(-2 \log \varepsilon)^{1/2}$, and, as $q < 1$, the right-hand side of this inequality is unbounded as $h \to 0$. Since we cannot use the levels in the box structure required by the algorithm as $h \to 0$, that is, as the total number of particles increases without bound, we would not, in principle, be able to achieve the desired $\mathbb{O}(n)$ behaviour. In [20] Greengard presents an "Adaptive Algorithm" which would avoid the difficulty mentioned above. This algorithm, however, is not available to us, as it requires that one be able to reduce the side of lowest level boxes as far as desired, and this is precluded for blobs by the restrictions on the maximum level which can be used. The difficulty discussed in this paragraph may not be a problem in practice, as the minimum value of h which can be used is restricted by the fact that for small values of h round-off errors dominate the truncation errors, so very small values of h are not acceptable for computations in any case.

Another possible problem is that even if we start with a uniform grid, and so have a uniform density of particles initially, the flow may concentrate the particles into a small region of space, so that essentially all the blobs are located in only a few boxes at the lowest available level. This, together with the inability to reduce the side of lowest level boxes as far as desired, will again prevent the achievement of the desired $\mathbb{O}(n)$ behaviour. Such a concentration of blobs occurs in the rollup of vortex sheets [26].

Subject to the restrictions noted above, the R-G algorithm provides a method which requires $\mathbb{O}(n)$ floating point operations and which produces a velocity field which agrees with that computed by the direct method.

C. *Problems Periodic in One Direction*

As described above, the method appears applicable only to problems for which the vorticity has compact support. In [20] Greengard discusses two-dimensional problems which are periodic in both directions and, in particular, gives detailed computations for an x- and y-periodic arrangement of point dipoles. He notes there that for periodic structures it is necessary to work with the series for the forces (or velocities) rather than those for the potentials, since the latter may become infinite. As we eventually will consider problems periodic in one spatial dimension, we have worked with velocities throughout.

Now we extend the method to the case of a general problem periodic in the x-direction but with vorticity bounded in the y-direction. Assume the period in the x-direction to be 1, and let the necessary extent in the y-direction be $|y| \leq K/2$ for some positive integer K. Then we can accommodate the periodicity and also provide the coverage needed in the y-direction by taking as the computational domain a $K \times K$ box. (We remark that, while the R-G algorithm can be extended to rectangles

with aspect ratio near unity, albeit with some loss of efficiency caused by the fact that more terms are necessary for the convergence of the various series, rectangles of arbitrary aspect ratio are not acceptable. The problem which arises in the latter case is that some boxes other than the nearest neighbour boxes may fail to lie entirely in the region of convergence.)

Apply the first half (the upwards pass) of R-G algorithm with the $K \times K$ box as the computational domain, thus obtaining for each box at each level of that computational domain an associated Laurent series as described in Section 3A. (In what follows, we will refer to the $K \times K$ box as the *basic domain*). We may think of each of the periodic images of the basic domain as being partitioned in the same manner as the basic domain, and by virtue of the periodicity a box at any level of an image box will have an associated Laurent series identical to that of its counterpart in the basic domain. We note here that if $K > 1$ then the computational work can be reduced by using the periodicity in x to permit calculation of the Laurent series for only a fraction of the boxes at lower levels in the box structure. In particular, it is helpful here to have K be a power of 2.

In the algorithm for the bounded problem we were able to start the downwards pass at level $l = 2$, as boxes at levels $l = 0$ and $l = 1$ have nothing but nearest neighbour boxes at their own level. In the periodic case, however, even the level 0 box—the basic domain—will be affected by particles in an infinite number of image boxes at level 0 which are not nearest neighbours of the basic domain. However, as we show below, the velocities induced on the particles in the basic domain by particles outside the basic domain and its two nearest neighbours can be written as a Taylor series centered at the center of the basic domain. Given that this can be done, thus providing the Taylor series $\tilde{\Phi}_{0,1}(z)$ about the center of the basic domain representing the velocity induced by the particles in all the periodic images other than the two adjacent to the basic domain, the balance of the algorithm proceeds as above. The only change necessary is that the computations at each level must allow for the effects of particles in the two adjacent image boxes. Since the Laurent series for any box outside the basic domain is, by periodicity, the same as the series for its periodic image inside the basic domain, this is essentially a matter of modifications in the bookkeeping involved in listing, at each level, the boxes which are nearest neighbours of or on the interaction list of a given box.

To see how to sum the effects of those periodic images of the basic computational box which are not adjacent to the basic domain we need the details of the calculation of the Taylor series induced at the center of one box by the Laurent series associated with another box at the same level, the latter not being a nearest neighbour of the first box. Suppose the Laurent expansion is centered at the point z_m and we want the Taylor series about the point z_0, given that the latter lies in the region of convergence of the Laurent series. If the Laurent series for the induced velocity has the form

$$L_m(z) = \sum_{k=1}^{\infty} a_k^{(m)} (z - z_m)^{-k}, \tag{3.6}$$

while the Taylor series which we want is to have the form

$$T_m(z) = \sum_{j=0}^{\infty} b_j^{(m)} (z - z_0)^j. \tag{3.7}$$

then by a straightforward calculation the coefficients $b_j^{(m)}$ are found in terms of the $a_k^{(m)}$ by

$$b_j^{(m)} = (z_m - z_0)^{-j} \sum_{k=1}^{\infty} (-1)^k \binom{j+k-1}{k-1} a_k^{(m)} (z_m - z_0)^{-k} \quad (j \geq 0),$$

which is essentially [20, Eq. (2.46)]. For our problem, if z_0 is the center of the basic domain and z_m is the center of the image box whose center lies at $z_0 + mK$, we just have $z_m - z_0 = mK$ and, as the $a_k^{(m)}$ are the same for each m, the coefficients $b_j^{(m)}$ are given by

$$b_j^{(m)} = (mK)^{-j} \sum_{k=1}^{\infty} (-1)^k \binom{j+k-1}{k-1} a_k (mK)^{-k} \quad (j \geq 0), \tag{3.8}$$

where we have replaced $a_k^{(m)}$ by a_k.

Now combine the effect of all the periodic image boxes other than those immediately adjacent to the basic domain by summing (3.8) over all values of m with $|m| \geq 2$ to get the coefficients of the Taylor series $T(z) = \sum_{j=0}^{\infty} b_j (z - z_0)^j$ representing the effect of the particles of all those boxes at points inside the basic domain. We thus obtain

$$\begin{aligned} b_j &= \sum_{|m| \geq 2} b_j^{(m)} = \sum_{|m| \geq 2} \sum_{k=1}^{\infty} (-1)^k \binom{j+k-1}{k-1} a_k (mK)^{-j-k} \\ &= \sum_{k=1}^{\infty} (-1)^k \binom{j+k-1}{k-1} a_k \sum_{|m| \geq 2} (mK))^{-j-k} \quad (j \geq 0). \end{aligned} \tag{3.9}$$

If $j \geq 1$ in the summations of (3.9) the series converge absolutely if they converge at all, and there is no question as to the interchange of the order of summation. The same is true for the case of $j = 0$, except when $k = 1$. The interchange of the order of summation in this case must be justified on physical grounds. Note that the system is symmetric with respect to interchange of the vorticity in the image box μ with the vorticity in the image box $-\mu$ for each integer μ. If the contribution of the odd powers of m in (3.9) were not zero, the result would not be invariant on the replacement of m by $-m$, and thus the physical symmetry would be violated. In particular, the contributions of the terms with $k = 1$ for $j = 0$ must be zero, and this is just the contribution which we obtain in the interchanged summation.

If $j + k$ is even $(mK)^{-j-k} + (-mK)^{-j-k} = 2(mK)^{-j-k}$, while

if $j + k$ is odd then $(mK)^{-j-k} + (-mK)^{-j-k} = 0$. Hence, we need consider only those k-values for which $k \equiv j \pmod 2$. For these terms we also have $(-1)^j = (-1)^k$, so for all $j \geq 0$

$$b_j = 2(-1)^j \sum_{\substack{k=1 \\ k=j \,(\mathrm{mod}\, 2)}}^{\infty} \binom{j+k-1}{k-1} a_k K^{-j-k} \sum_{m=2}^{\infty} m^{-j-k}.$$

But the last sum over m is just $\zeta(j + k) - 1$, where ζ is the Riemann zeta-function, and the latter is tabulated to high precision for relatively small even integer arguments [3] and requires only a few terms to calculate values to high precision for larger values of the argument. Thus the coefficients b_j in

$$\tilde{\Phi}_{0,1}(z) = \sum_{j=0}^{\infty} b_j (z - z_0)^j$$

are given by

$$b_j = 2(-1)^j \sum_{\substack{k=1 \\ k=j \,(\mathrm{mod}\, 2)}}^{\infty} \binom{j+k-1}{k-1} a_k K^{-j-k} [\zeta(j + k) - 1].$$

In [20] Greengard shows that the various series for the potential converge. Since the series for the conjugate velocities are just the derivatives of the series for the potential, and since the latter represent analytic functions in their regions of convergence, the series for the conjugate velocities also represent analytic functions with the same regions of convergence as the original series. As the series for the conjugate velocities are derivatives of the series for the potentials, we can expect the series for the conjugate velocities to require a larger number of terms for convergence to a given accuracy. The number of terms required for a given accuracy is problem dependent. We shall discuss the determination of appropriate values for this parameter in subsequent sections.

4. PROBLEMS IN ALL SPACE

Our goal in the next two sections is to document the performance of the R-G algorithm with blobs for problems posed in all space or periodic in one direction. In this section we concentrate on problems which are posed in all space but have vorticity with compact support. (The general all-space problem is usually reduced to this kind of problem by truncating the initial vorticity field in some manner.) We use an exact solution to verify the coding of the algorithm and to demonstrate the limitation imposed by the structure of the blobs on the largest value of the level l permissible in the R-G algorithm for production of results within a desired accuracy. All conclusions in this section also apply to periodic problems, and in Section 5 we discuss the performance of the R-G algorithm with blobs for problems which are periodic in one direction.

We are interested in comparing the results of the calculation of velocities using vortex blobs and the R-G algorithm with the results of the calculation of the same velocities using blobs and the direct method (i.e., summing the blob–blob interactions over all particles). Accordingly, we adopt as the measure of error the norm $\|u_h^g - u_h^d\|$, where $u_h^g(z)$ is the velocity calculated at z by the R-G algorithm and $u_h^d(z)$ is the velocity calculated by the direct method. We compute both the discrete L_2 and uniform norms via

$$\|u_h^g - u_h^d\|_2 = \left(\sum_i (|u_h^g(z_i) - u_h^d(z_i)|^2) h^2 \right)^{1/2} \tag{4.1a}$$

and

$$\|u_h^g - u_h^d\|_\infty = \max_i |u_h^g(z_i) - u_h^d(z_i)|. \tag{4.1b}$$

A general class of problems whose exact solution can be calculated consists of those in which the vorticity has compact support and the stream function Ψ is, in plane polar coordinates (r, θ), a function only of the radial coordinate. Assuming the velocity, and thus Ψ', is bounded as $r \to 0$, we obtain the velocity by

$$\begin{aligned}(u_1, u_2) &= \left(\frac{\partial \Psi}{\partial y}, -\frac{\partial \Psi}{\partial x} \right) \\ &= \frac{1}{r} (y\Psi', -x\Psi') = \frac{g(r)}{r^2} (y, -x),\end{aligned} \tag{4.2}$$

where

$$g(r) = -\int_0^r \rho\omega(\rho)\, d\rho. \tag{4.3}$$

Beale and Majda [14] and Perlman [36] studied several such problems using vortex blob methods, so by choosing to study one of those problems we can compare the results of calculations using the fast particle algorithm with direct calculations. The problem which we consider is this: Let the vorticity be given by

$$\omega(r) = \begin{cases} (1 - r^2)^7, & r \leq 1 \\ 0, & r > 1. \end{cases}$$

Then the velocity field is given by (4.2), with

$$g(r) = \begin{cases} -\frac{1}{16}[1 - (1 - r^2)^8], & r \leq 1 \\ -\frac{1}{16}, & r > 1. \end{cases}$$

In carrying out the calculations of the R-G algorithm it is necessary to determine a value for the number of terms to be used in the various series. Denote by $u_h^{g,p}$ the velocity calculated

TABLE I

Determination of the Number of Terms Used in the Series

h	$l_{\max}$	q	p^*
0.10	2	0.95	12
0.05	2	0.95	12
0.10	3	0.95	15
0.10	2	0.75	12

by using the R-G algorithm with p terms in each of the series. Then we determine the number of terms which is appropriate by computing, at $t = 0$, values of the quantity $E_p := \|u_h^{g,p} - u_h^{g,(p+1)}\|$ in both the L_2 and uniform norms over a range of values of p. If E_p is sufficiently small, there is no significant change in the velocities computed with $p + 1$ terms in the series as compared to those computed with p terms. As expected, E_p decreases as p increases, with the rate of decrease being rapid for smaller values of p. While for a given p the values of E_p are not independent of the parameters $l_{\max}$, h, and q, the error E_p as measured in the L_2 norm was reduced to a value of less than 1.04×10^{-8} (or about machine epsilon multiplied by the average velocity over the unit disk) for approximately the same value of p independent of the other parameters. (See Table I, where p^* represents the value of p such that in the L_2 norm $E_p < 1.04 \times 10^{-8}$ whenever $p \geq p^*$.) Based on the data of Table I and similar calculations for other triplets $(h, l_{\max}, q)$, 15 terms were used in each of the series of the R-G algorithm.

Perlman was interested in the study of certain errors which arise in the use of vortex blobs for the solution of these problems and did not actually integrate the equations of motion. Instead, she used knowledge of the exact solutions to determine the locations of the vortex blobs at later times and then used this blob distribution in (2.12) to calculate the velocity field at these times. In our calculations of this section we have used the same method; i.e., we used the knowledge of the exact solution for the determination of the blob locations at later times and then used these locations in the calculation of both $u_h^g(z_i)$ and $u_h^d(z_i)$ for $t > 0$.

For a given distribution of blobs and for some point z the agreement between $u_h^g(z)$ and $u_h^d(z)$ depends on two things:

1. The validity of the approximation of the blobs by point vortices, which is in turn dependent on the maximum bisection level $l_{\max}$, and

2. The internal accuracy of the computations of the R-G algorithm, which is determined by the number of terms p used in the various series.

Since $l_{\max}$ and p do not change with time, the values of $\|u_h^g - u_h^d\|_2$ should be approximately constant in time, a fact which we verify below. It follows that computation of $\|u_h^g - u_h^d\|_2$ at $t = 0$ is sufficient to approximate the values of $\|u_h^g - u_h^d\|_2$ at all times.

For a blob at position (R, θ_0) at $t = 0$ the streamline is a circle of radius R, and the blob moves with angular velocity $d\theta/dt = |u(R)|/R = -g(R)/R^2$. Hence its position at time t is $(R, \theta_0 - g(R)t/R^2)$. We use these positions at later times to calculate the velocity field given by (2.12). Denoting the exact solution by $u(z)$, we compute the error in velocity in the discrete 2-norm, $\|u_h^g - u\|_2$, using expressions analogous to (4.1a) above. These results are shown in Fig. 2, and are in excellent agreement with [36, Fig. 3.1], which shows $\|u_h^d - u\|_2$ as a function of time.

We now use the triangle inequality to obtain some relationships among the quantities $\|u_h^d - u_h^g\|$, $\|u_h^d - u\|$, and $\|u_h^g - u\|$. First note that

$$\big|\,\|u_h^d - u\| - \|u_h^g - u_h^d\|\,\big| \leq \|u_h^g - u\| \leq \|u_h^d - u\| + \|u_h^g - u_h^d\|,$$

so if $\|u_h^g - u_h^d\| \ll \|u_h^d - u\|$ then $\|u_h^g - u\| \approx \|u_h^d - u\|$. Hence, if the R-G algorithm produces a velocity field which closely approximates the one produced by direct calculations, then the velocity fields produced by the R-G and direct methods approximate the exact velocity field with errors of the same order of magnitude. However, these inequalities do not prevent $\|u_h^d - u_h^g\|$ from being large even though $\|u_h^g - u\|$ is small.

Another simple application of the triangle inequality gives

$$\big|\,\|u_h^d - u\| - \|u_h^g - u\|\,\big| \leq \|u_h^d - u_h^g\|.$$

Thus, the velocity fields produced by the R-G algorithm and the direct method can each approximate the exact velocity field even if $\|u_h^d - u_h^g\|$ is large.

These simple estimates guide our presentation in this section and in Section 5B.

We now compare the velocities calculated by the R-G algorithm to those calculated with the direct method. First consider Table II, where we give the values of the quantity $s_{l_{\max}}/\delta$ for various values of h and $l_{\max}$ and for two values of the shape parameter q. (Recall that $s_{l_{\max}}$ is the side of the box at the lowest level, and note that for the problem under consideration the computational domain is 2×2, so $s_l = 2 \times 2^{-l}$.) We showed in Section 3B that for the R-G algorithm to be accurate with vortex blobs it is necessary that the quantity $s_{l_{\max}}/\delta$ exceed some minimum value. We suggested there that for kernel (2.15) a minimum value of 4.3 for this ratio might prove satisfactory, and it may be seen from Table II that the range of values for h and q shown there provides a range of values for $s_{l_{\max}}/\delta$ which will allow us to test this conjecture.

Tables III and IV give values of $\|u_h^g - u_h^d\|$ at $t = 0$ for the L_2 and uniform norms, respectively, for selected values of h and $l_{\max}$. The velocities involved have magnitudes ranging from 0 to $\frac{1}{16}$, so round-off error in a single floating point operation can be expected to contribute an absolute error of approximately 3.7×10^{-9} and, assuming that round-off errors may be treated

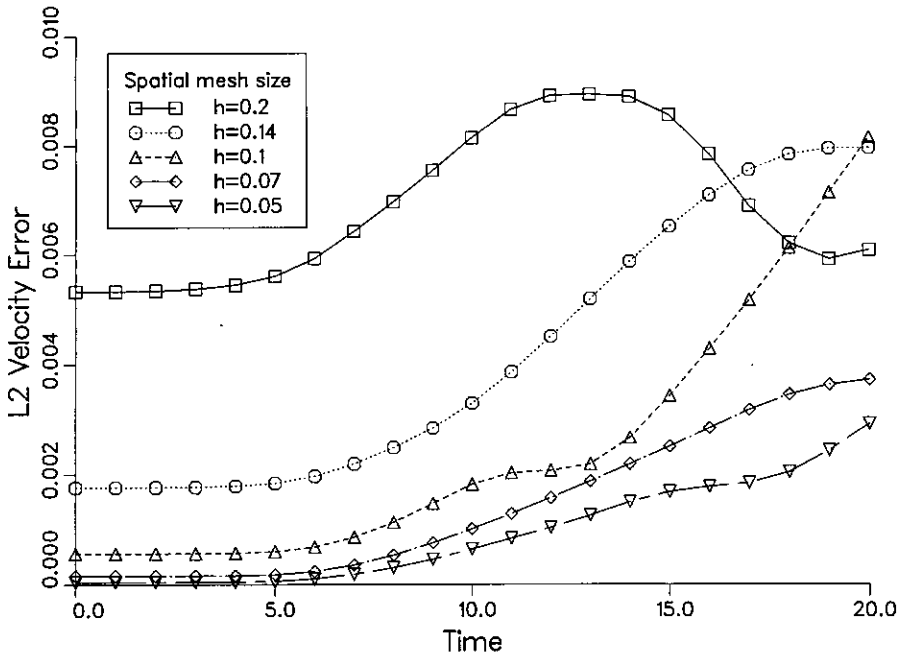

FIG. 2. Variation of $\|u_h^\varepsilon - u\|_2$ and $\|u_h^\varepsilon - u\|_\infty$ with t for various values of the step size h in the problem of Section 4.

TABLE II

Values of $s_{l_{\max}}/\delta$ for Selected h and $l_{\max}$

	$q = 0.95$				$q = 0.75$			
		$s_{l_{\max}}/\delta$				$s_{l_{\max}}/\delta$		
h	δ	$l = 2$	$l = 3$	$l = 4$	δ	$l = 2$	$l = 3$	$l = 4$
0.2000	0.2168	2.31	1.15	0.58	0.2991	1.67	0.84	0.42
0.1414	0.1560	3.21	1.60	0.80	0.2306	2.17	1.08	0.54
0.1000	0.1122	4.46	2.23	1.11	0.1778	2.81	1.41	0.70
0.0707	0.0807	6.19	3.10	1.55	0.1371	3.65	1.82	0.91
0.0500	0.0581	8.61	4.30	2.15	0.1057	4.73	2.36	1.18
0.0353	0.0418	11.97	5.98	2.99	0.0815	6.13	3.07	1.53
0.0250	0.0301	16.63	8.32	4.16	0.0629	7.95	3.98	1.99

TABLE III

$\|u_h^\varepsilon - u_h^q\|_2$ for Selected h and $l_{\max}$ at $t = 0$

	$q = 0.95$			$q = 0.75$		
h	$l = 2$	$l = 3$	$l = 4$	$l = 2$	$l = 3$	$l = 4$
0.2000	2.2E-04			1.1E-03		
0.1414	7.0E-06	1.6E-03		3.5E-04		
0.1000	4.7E-08	1.5E-04	1.7E-03	1.5E-05	1.8E-03	
0.0707	7.0E-08	5.7E-06	7.7E-04	4.8E-07	5.6E-04	
0.0500	9.4E-08	9.7E-08	1.0E-04	9.7E-08	5.7E-05	1.5E-03
0.0353	1.5E-07	1.6E-07	4.5E-06	1.5E-07	3.9E-06	6.3E-04
0.0250		2.8E-07	2.7E-07	2.8E-07	2.8E-07	1.2E-04

TABLE IV

$\|u_h^\varepsilon - u_h^q\|_\infty$ for Selected h and $l_{\max}$ at $t = 0$

	$q = 0.95$			$q = 0.75$		
h	$l = 2$	$l = 3$	$l = 4$	$l = 2$	$l = 3$	$l = 4$
0.2000	4.5E-04			2.9E-03		
0.1414	1.8E-05	3.7E-03		8.3E-04		
0.1000	1.1E-07	3.8E-04	2.8E-03	3.5E-05	3.7E-03	
0.0707	1.5E-07	1.7E-05	2.6E-03	1.3E-06	1.5E-03	
0.0500	2.0E-07	2.1E-07	3.0E-04	2.3E-07	1.4E-04	3.5E-03
0.0353	3.6E-07	3.8E-07	1.7E-05	3.1E-07	1.2E-05	1.9E-03
0.0250		6.9E-07	5.0E-07	6.6E-07	6.0E-07	3.6E-04

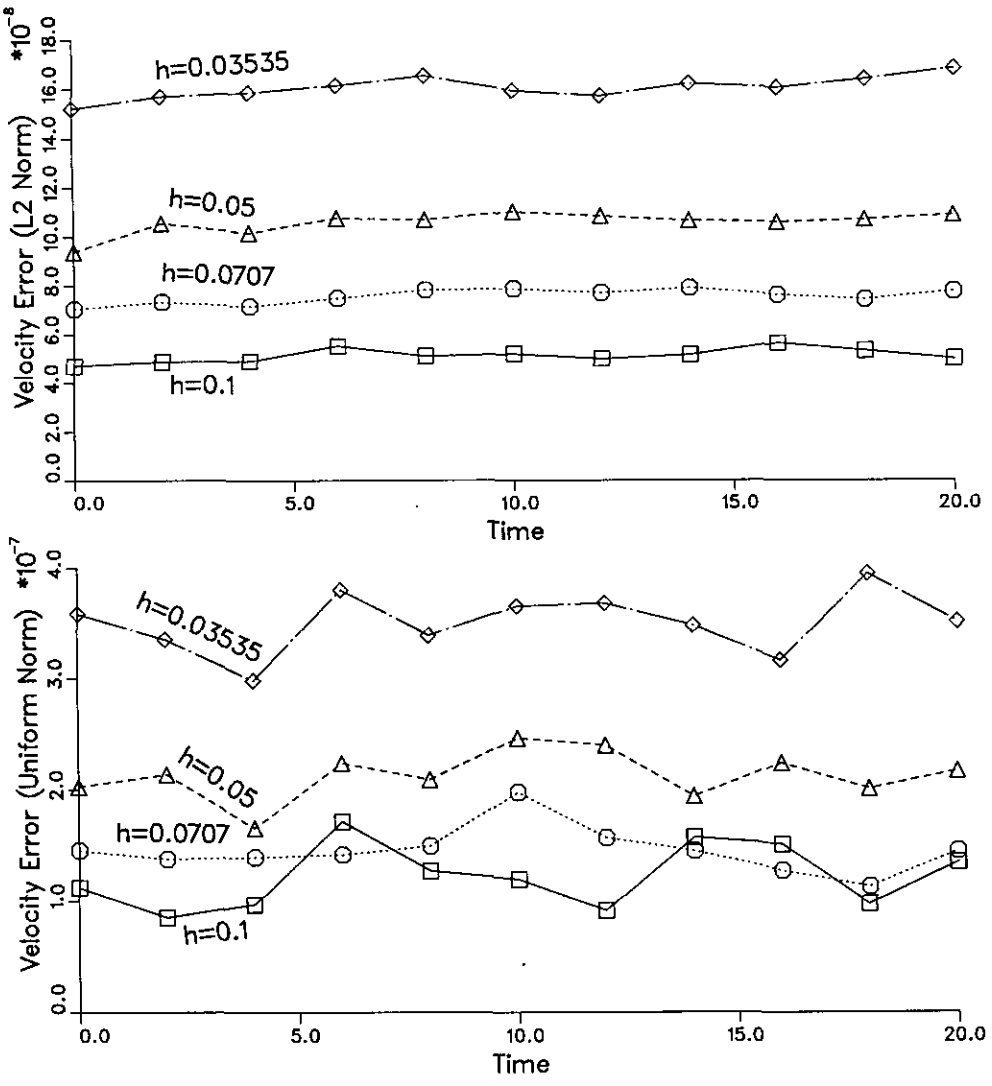

FIG. 3. Variation of $\|u_h^x - u_h^d\|_2$ with time for various values of the step size h in the problem of Section 4.

by using the truncated Laurent series of the R-G algorithm instead of direct calculations is small, so the main source of error is round-off error. To explain this assertion, note that as h decreases by a factor of $1/\sqrt{2}$, the number of particles, n, increases by a factor of 2. Since the number of floating point operations should lie between $\mathbb{O}(n)$ and $\mathbb{O}(n^2)$, as n is doubled we expect the number of floating point operations to increase by a factor between 2 and 2^2. Assuming that round-off errors may be treated as normally distributed random variables, we thus expect the round-off error to increase by a factor between $\sqrt{2}$ and 2. The results shown in Fig. 3 are consistent with this calculation, and this result confirms that our choice of the number of terms used in the various series of the R-G algorithm was appropriate.

We thus see that, with appropriate choices for the parameters of the method, the R-G algorithm can be used with vortex blobs and yields results essentially identical to those which result from direct calculations. If $l_{\max}$ is too large the accuracy deteriorates.

5. PERIODIC PROBLEMS

In this section we apply the method developed in Section 3C to two problems, each of which is periodic in one of the spatial dimensions in the plane. Each problem has a steady state solution for which the exact solution for the velocity field is known as a function of position, so we have a standard with which to compare our numerical solutions. We also examine the running time of this implementation of the R-G algorithm as a function of the number of blobs, and we present a Hamiltonian for this class of problems.

A. *Infinite Row of Vortices*

Our first example is that of an infinite row of point vortices, each of strength γ, evenly spaced along the x-axis at the points $0, \pm a, \pm 2a, \ldots$. This problem is one in which the actual vorticity distribution is composed of point vortices, so the use of point vortices in a numerical method should provide an exact solution of the problem up to round-off errors. Errors which might arise in this computation should provide a lower bound to errors in more general problems which are periodic in x.

This problem provides a useful test of coding and of convergence of the vortex blob method as the shape parameter $\delta \to 0$. The advantage of this problem is that no blob in the basic domain except the one at (0, 0) contributes to the flow, so we can carry out the computations for any value of δ with just one particle in the basic domain. In a more general problem δ and the grid spacing h are connected, and carrying out any computation with small δ requires that h also be small, so that the number of blobs is large, this number being $\mathbb{O}(h^{-2}) = \mathbb{O}(\delta^{-2/q})$, where $0 < q < 1$. The problem may thus also be used as a very simple test of the regularization of a problem with discontinuous vorticity distribution by the use of vortex blobs.

The vorticity in this problem is concentrated in the particle at the origin and its periodic images, so no other grid point need be considered in the computation of the flow. We will evaluate the accuracy of the computations by computing, in the L_2 and uniform norms, the quantity $\|u^\delta - u\|$ defined as in Eqs. (4.1), but with the points z_i in this case being those of a uniform grid of spacing $\tilde{h}$ placed on the basic domain. Here $u^\delta(z)$ is the velocity at z by the R-G algorithm, and $u(z)$ is the exact velocity at that point.

This problem is discussed by Lamb [31], where it is shown that the velocity field is (u_1, u_2) with

$$u_1 = -\frac{\gamma}{2a}\left[\frac{\sinh(2\pi y/a)}{\cosh(2\pi y/a) - \cos(2\pi x/a)}\right], \tag{5.1a}$$

$$u_2 = \frac{\gamma}{2a}\left[\frac{\sin(2\pi x/a)}{\cosh(2\pi y/a) - \cos(2\pi x/a)}\right]. \tag{5.1b}$$

We will study this problem with $a = 1$ and $\gamma = 1$, taking as the basic domain a square of side 1 centered at (0, 0). We use the method of Section 3C to obtain, for each box at the lowest level, the Taylor series which represents the velocity induced in that lowest level box by all vortices other than one in that box or one of its nearest neighbours. If the vortex at (0, 0) is in the lowest level box under consideration or one of its nearest neighbours, the computed velocity at a point in the box is obtained by adding the effect of that vortex to the value computed by the Taylor series. Now superpose a grid of spacing $\tilde{h} = N^{-1}$, where $N = 2M$ for some $M \in \mathbf{Z}^+$, with the nodes of this grid located at the points $(i\tilde{h}, j\tilde{h})$, with $-M \le i < M$ and $-M \le j \le M$, a total of $N(N+1)$ nodes. The velocities at each of these nodes are compared with the exact values.

i. *Point vortex calculations.* As in Section 4, the number of coefficients required in the series of the R-G algorithm, p, was determined by examining the velocity errors at $t = 0$ calculated using various values of p. These calculations were carried out in both the L_2 and uniform norms for various values of $\tilde{h}$. There is no significant difference in the results for different values of $\tilde{h}$. The errors are quite small over a range of values of p, and in this case the random nature of round-off errors makes it difficult to pinpoint a "best" value of p. It is clear from the results that for this problem the value $p = 15$ used in Section 4 would be too small. It is also clear that there is no reason to use a value larger than $p = 21$, and this value is used for the calculations of this section.

Using point vortices with $l_{\max}$ having the values 2, 3 and 4, we find that the error in velocity at the point (0, 0) is very small. Hence the point vortex remains at the origin aside from small random errors and, again as in Section 4, the errors at later times are approximately the errors observed at $t = 0$. Figure 4 shows the change in the L_2 error $\|u^\delta - u\|_2$ with time from $t = 0.0$ to $t = 1.0$, where the position of the point vortex is computed at later times by integrating the computed velocity field at (0, 0) with a fourth-order Runge–Kutta scheme [39] with $\Delta t = 0.025$. It thus appears satisfactory to approximate

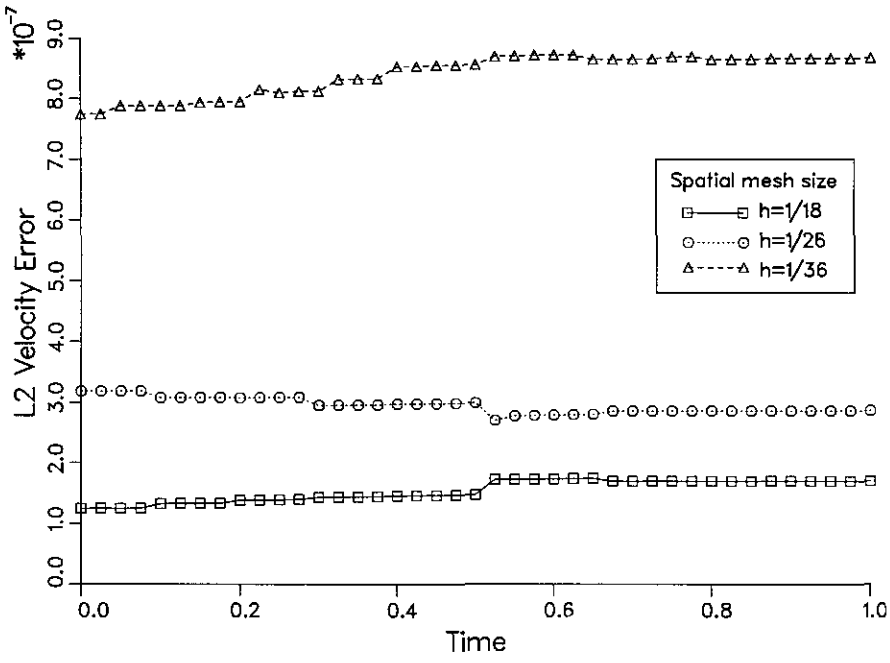

FIG. 4. Variation of $\|u^s - u\|_2$ with time for various values of the step size h in the problem of Section 5A.

the errors at later times by the error at $t = 0$. Accordingly, all further computations in this subsection (Section 5A) are carried out at $t = 0$.

In Table V we give values of the errors $\|u^s - u\|$ at $t = 0$ for the L_2 and uniform norms for two values of the spacing $\tilde{h}$ of the test grid and for $l_{\max} = 2, 3, 4$. In addition, we compute the maximum of the relative velocity error, E_{rel}, defined as the maximum over the grid of the quantity $|(u^s - u)/u|$, with points at which $u = 0$ ignored. We note that the errors do not change with $l_{\max}$ (in fact, the agreement holds in all significant figures of our single precision computations), but that the errors increase as the grid spacing decreases. These results are consistent with the absence of any significant truncation error, with the observed errors being caused by round-off. Assuming that the increased error in the calculations with the finer test grid is indeed caused by increased round-off errors, the values of the errors with $\tilde{h} = \frac{1}{26}$ may be taken as an upper bound for the error in velocity in the method as applied to this problem.

TABLE V

Velocity Errors at $t = 0$ (Point Vortices)

	$\tilde{h} = \frac{1}{26}$			$\tilde{h} = \frac{1}{52}$		
$l_{\max}$	E_∞	E_{rel}	E_2	E_∞	E_{rel}	E_2
2	6.2E-06	9.2E-07	3.2E-07	5.9E-05	7.1E-06	1.7E-06
3	6.2E-06	9.2E-07	3.2E-07	5.9E-05	7.1E-06	1.7E-06
4	6.2E-06	9.2E-07	3.2E-07	5.9E-05	7.1E-06	1.7E-06

ii. *Vortex blob calculations.* Next we carried out calculations for the same problem using, instead of point vortices, vortex blobs located at the point (0, 0) and its periodic images. Using point vortices the spacing $\tilde{h}$ of the test grid was arbitrary, but when using blobs there are certain limitations and relations among the parameters. If we write $\kappa := \sqrt{14 \log 10} \approx 5.6777$, the considerations of Section 3B (specifically, a condition that $e^{-s_{l_{\max}}^2/2\delta^2} \leq 10^{-7}$) dictate that we require $\kappa\delta \leq 2^{-l_{\max}}$ in order to guarantee that the blob calculations agree with direct calculations to 1 part in 10^7. On the other hand, if the test grid is too coarse, the vortex blob at the origin will appear as a point vortex at all points of the test grid other than the point at (0, 0), and the results will thus be indistinguishable from those obtained using point vortices. This restriction requires that $\tilde{h} < \kappa\delta$. Using $l_{\max} = 2$, together with $\tilde{h} = \frac{1}{26}$ or $\tilde{h} = \frac{1}{52}$ we find the limitations $0.0068 \leq \delta \leq 0.0440$ if $\tilde{h} = \frac{1}{26}$ and $0.0034 \leq \delta \leq 0.0440$ if $\tilde{h} = \frac{1}{52}$.

In Table VI we give the errors in the uniform and L_2 norms, as well as the maximum of the relative velocity error, for $\tilde{h} = \frac{1}{26}$ and $\tilde{h} = \frac{1}{52}$ at time $t = 0$. The number of terms used in each series is $p = 21$. For each of these values of $\tilde{h}$ we use a range of values of δ which satisfy the inequalities of the preceding paragraph. First, note that both for $\tilde{h} = \frac{1}{26}$ and $\delta = 0.006$ and for $\tilde{h} = \frac{1}{52}$ and $\delta = 0.003$ the errors agree with those computed using point vortices (this agreement is actually to seven significant figures, not just the three significant figures shown in the table). This justifies the assertion above that for values of $\delta < \tilde{h}/\kappa$ the results will be indistinguishable from those obtained using point vortices. Except for values of δ near the lower limits for δ of $\tilde{h}/\kappa$ (where round-off errors dominate those due

TABLE VI

Velocity Errors at $t = 0$ (Vortex Blobs, Except P.V. = Point Vortices)

	$\tilde{h} = \frac{1}{26}$			$\tilde{h} = \frac{1}{52}$		
δ	E_∞	E_{rel}	E_2	E_∞	E_{rel}	E_2
0.040	6.77E-1	1.64E-1	6.01E-2			
0.030	3.47E-1	1.19E-1	3.22E-2			
0.025	4.91E-1	1.19E-1	4.15E-2			
0.020	4.46E-1	1.08E-1	3.47E-2			
0.015	1.43E-1	3.47E-2	1.10E-2			
0.010	2.54E-3	6.16E-4	1.95E-4	8.93E-1	1.08E-1	3.47E-2
0.009	4.54E-4	1.09E-4	3.45E-5	6.72E-1	8.13E-2	2.59E-2
0.008	4.58E-5	9.84E-6	3.12E-6	4.09E-1	4.95E-2	1.57E-2
0.007	7.15E-6	9.22E-7	3.41E-7	1.81E-1	2.19E-2	6.97E-3
0.006	6.20E-6	9.22E-7	3.18E-7	4.81E-2	5.82E-3	1.85E-3
0.005				5.09E-3	6.15E-4	1.94E-4
0.0045				9.12E-4	1.10E-4	3.36E-5
0.004				9.54E-5	1.15E-4	2.64E-6
0.0035				5.72E-5	6.91E-6	1.67E-6
0.003				5.91E-5	7.14E-6	1.71E-6
P.V.	6.20E-6	9.22E-7	3.18E-7	5.91E-5	7.14E-6	1.71E-6

to truncation), the errors for $\tilde{h} = \frac{1}{52}$ and a given value of δ are of the same order of magnitude as those corresponding to $\tilde{h} = \frac{1}{26}$ and a value of δ twice as large, as would be expected from scaling.

The convergence of the computed velocities to the exact values is clear for both values of $\tilde{h}$, understanding that the accuracy which can be achieved is bounded below in each case by the accuracy of the point vortex calculations. While the convergence should be $\mathcal{O}(\delta^4)$ [14] if the initial conditions were smooth, the observed rate of convergence is much faster than order 4 for smaller values of δ. In fact, for $\tilde{h} = \frac{1}{26}$ in the range $\delta = 0.007$ to 0.01 the convergence is approximately exponential. (See the semi-log plots of Fig. 5, where the data of Table VI is plotted and on which the order of convergence is given locally by the slope of the graph. To provide a reference point, one may note that on the graph for $\tilde{h} = \frac{1}{26}$ the convergence from $\delta = 0.02$ to 0.015, or $\log_{10} \delta \approx -1.70$ to -1.82, is almost exactly order 4.) An explanation for this high rate of convergence may be the following: As δ is reduced to the smaller values studied, more and more of the test particles see the vortex at the origin as a point vortex and thus see a value of $\delta = 0$. Hence the effective value of δ over all the test particles is less than its true value. As $\delta = h^q$, a small value of δ corresponds to a large value of q, and thus a higher order of convergence.

The results of this subsection show that our approach to a periodic problem is viable and indicates that the implementation is correct. We may also take the errors found here to be lower bounds on the errors which may be expected in more complicated problems.

B. *An Example of Stuart of Periodic Flow*

Our second example of a problem periodic in one direction is somewhat more interesting than the row of vortices of Section 4A and, in fact, includes that example as a limiting case. After defining the problem, we determine the parameters in both the problem and the numerical method so as to obtain results with reasonable computational effort. We then demonstrate that if the practical restriction on l_{max} is observed the numerical solution converges to the exact solution as the number of blobs is increased. Finally, we show that the R-G algorithm becomes increasingly inaccurate as l_{max} exceeds the maximum practical bisection level.

If we set $\omega_t = 0$ in (2.3) and use $\omega = \Delta\Psi$ and $u = (u_1, u_2) = (\Psi_y, -\Psi_x)$ we obtain the two-dimensional steady state vorticity equation

$$\frac{\partial(\Delta\Psi)}{\partial x}\frac{\partial\Psi}{\partial y} - \frac{\partial(\Delta\Psi)}{\partial y}\frac{\partial\Psi}{\partial x} = 0, \tag{5.2}$$

which is satisfied if $\Delta\Psi = g(\Psi)$ for some function g. A periodic example of such a stream function is given in [40], where Stuart considers the flow with stream function given by

$$\Psi = \log(C\cosh y + A\cos x), \tag{5.3}$$

where C and A are real constants with $C \geq 1$, $A \geq 0$, and $C^2 - A^2 = 1$. Here $\Delta\Psi = e^{-2\Psi}$. The vorticity is

$$\omega = -\Delta\Psi = -e^{-2\Psi} = -\frac{1}{(C\cosh y + A\cos x)^2}, \tag{5.4}$$

and the velocities are given by

$$u_1 = \frac{C\sinh y}{C\cosh y + A\cos x}, \quad u_2 = \frac{A\sin x}{C\cosh y + A\cos x}. \tag{5.5}$$

Except for an inconsequential translation of the origin along the x-axis, the limiting case $C \to \infty$ yields Eqs. (5.1) with $a = 2\pi$ and $\omega = -4\pi$. The other limiting case, with $C = 1$, $A = 0$, reduces to a hyperbolic tangent velocity profile with $\Psi = \log\cosh y$ and velocity field given by $(u_1, u_2) = (\tanh y, 0)$, parallel to the x-axis. This can be regarded as a model of a two-dimensional shear layer, with the velocity changing from $+1$ on one side of the layer to -1 on the other.

In this problem the support of the vorticity is periodic in x and unbounded in the y-direction. In order to apply the methods of Section 3B we need to truncate the support of the vorticity to a domain which is bounded in the y-direction. In practice, by using a suitable basic computational domain, we may proceed as though the support were bounded in that direction. To see why this should be so, consider the vorticity in the rectangle R_α bounded by $x = 0$, $x = 2\pi$, and $y = \pm\alpha$ for some $\alpha > 0$. The total vorticity in R_α is given by the circulation $\Omega_\alpha =$

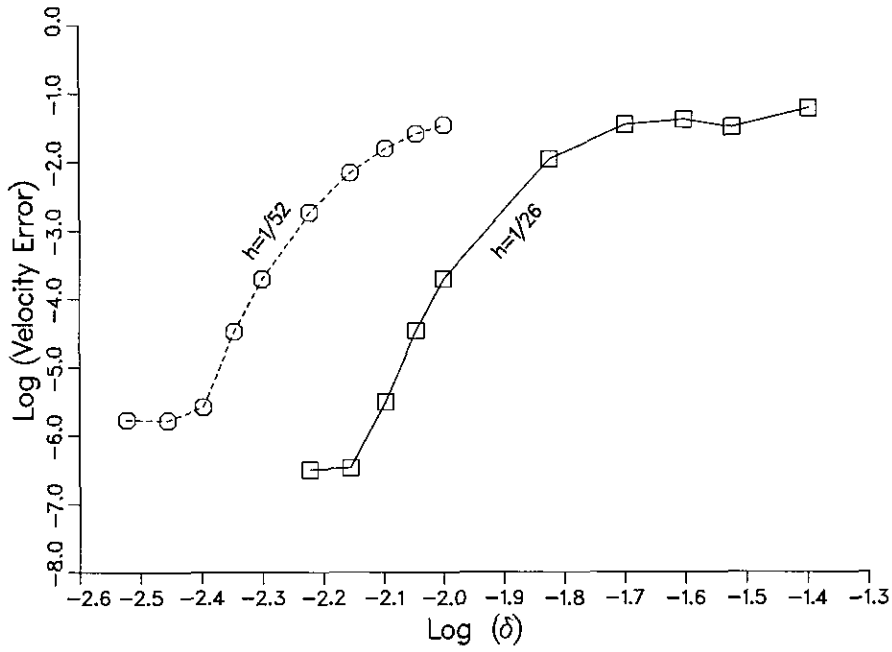

FIG. 5. Log_{10} of $\|u_{\text{calc}} - u_{\text{exact}}\|_2$ vs $\log_{10}(\delta)$ for the problem of Section 5A.

$\int_{R_\alpha} \omega \, dx \, dy = \int_{R_\alpha} \nabla \times u \, dx \, dy = \oint_{\partial R_\alpha} u_1 \, dx + u_2 \, dy$, and by a residue calculation we find

$$\Omega_\alpha = -2 \int_0^{2\pi} u_1(x, \alpha) \, dx = -4\pi \frac{\sinh \alpha}{(\sinh^2 \alpha + 1/C^2)^{1/2}}.$$

Note that as $\alpha \to \infty$ we find $\Omega_\alpha \to \Omega_\infty = -4\pi$, independent of the value of C.

For a given α, $|\Omega_\alpha|$ has a minimum value of $4\pi \tanh \alpha$ when $C = 1$. Thus $\tanh \alpha \leq |\Omega_\alpha/\Omega_\infty| \leq 1$, so when $\tanh \alpha$ is near unity most of the vorticity is contained in the rectangle R_α and its periodic images. If $\alpha = \pi$ we find that if $C \geq 2$ the magnitude of the vorticity outside R_π is less than $0.001|\Omega_\infty|$. The effect of ignoring the vorticity outside R_π is also lessened by the facts that (i) the velocity at one point induced by the vorticity at another point falls off as $1/r$, where r is the distance between the points and (ii) the vorticity is an even function of y, so at points inside the square the velocity induced by an element at some point (x, y) with $y > \pi$ and that induced by the element at $(x, -y)$ tend to counteract one another. It thus appears that computations using as the basic computational domain the compact set $[0, 2\pi] \times [-\pi, \pi]$ should provide, in many cases, a reasonable approximation to the results which one would obtain if all the initial vorticity were to be considered.

To test this last conjecture we calculate $\|u - u_h\|_\infty$ at $t = 0$ using basic computational domains of sizes $2\pi \times 2\pi$ and $4\pi \times 4\pi$. When $C = 2$ we find $1 - |\Omega_{2\pi}/\Omega_\infty| < 2 \times 10^{-6}$, so in this case the square $R_{2\pi}$ contains essentially all of the vorticity and calculations with that domain should be accurate. Table VII gives the results of calculating $\|u - u_h\|_\infty$, using expressions analogous to Eqs. (4.1), where $u(z)$ is the velocity described by equations (5.5) and $u_h(z)$ is the velocity calculated by the R-G algorithm. (Here and henceforth the symbol $u_h(z)$ should be interpreted as u_h^δ, as all subsequent calculations on the blobs will be carried out using the R-G algorithm, and no further direct calculations will be made. The latter symbol may still be used for clarity.) For $C = 2$, there is no significant change (in single precision) in the calculations using the $2\pi \times 2\pi$ domain as compared to those using a domain of $4\pi \times 4\pi$. Since for a given α the value of $|\Omega_\alpha/\Omega_\infty|$ increases with C, the results obtained by using $[0, 2\pi] \times [-\pi, \pi]$ as the basic computational domain should provide even better approximations to the exact results for values of $C > 2$ than for $C = 2$. Hence, we may take a basic computational domain of $2\pi \times 2\pi$ as adequate for all values of $C \geq 2$.

Contrary to the approach used in the calculations of Section 4 and of Section 5A, in this problem we will calculate the later

TABLE VII

Errors in $\|u - u_h\|_\infty$ at $t = 0$ ($C = 2$)

	Basic computational domain	
h	$2\pi \times 2\pi$	$4\pi \times 4\pi$
$2\pi/20$	2.30580E-01	2.30580E-01
$2\pi/30$	1.02949E-01	1.02950E-01
$2\pi/40$	4.59672E-02	4.59685E-02

TABLE VIII

Order of Convergence β in $h = 2\pi/N$ at $t = 0$ (See Text)

	Uniform norm		L_2 norm	
N	$\|u - u_h\|$	β	$\|u - u_h\|$	β
20	2.3058E-01		1.7177E-01	
30	1.0295E-01	1.99	6.1943E-02	2.52
40	4.5967E-02	2.80	2.6227E-02	2.99
50	2.3275E-02	3.05	1.2800E-02	3.21
60	1.3259E-02	3.09	6.9380E-03	3.36
80	5.2371E-03	3.23	2.5446E-03	3.49

positions of the test particles by integrating the equations of motion (2.8) with velocities calculated by (2.12). For a particle with a given velocity, the angular velocity of the particle in its orbit will vary inversely with its distance from the center of the core, while for particles at the same distance from the core the angular velocity varies directly with the magnitude of the velocity. An increase in angular velocity of the particles will necessitate a smaller time step if the computational errors in the later positions of the particles are to be sensibly unchanged. For this problem the maximum magnitude of the velocity has the value C, so for larger values of C the velocities will be generally larger. Since this is true in particular for the particles at $t = 0$, along with the rates of convergence in the mesh size. Here we have assumed that the error can be written as $\|u - u_h\| = \text{const}(h)^\beta$ and have calculated the value of β appropriate to successive values of h by $\beta_{i+1} = \log(E_i/E_{i+1})/\log(h_i/h_{i+1})$. For the Beale–Majda kernel of order $\hat{p}$ and with the shape parameter δ given by $\delta = h^q$ the theoretical order of convergence is $\mathbb{O}(h^{\hat{p}q})$ [13]. Since we are using $\hat{p} = 4$ and $q = 0.95$, this theoretical order of convergence is 3.80. While this order of convergence has not yet been achieved by the values in Table VIII, the order of convergence continues to increase as h decreases, and the values of the computed order of convergence at $N = 80$ are not too far under the theoretical value.

It is now necessary to determine optimum values of Δt for the integration. To find these values (which, as indicated earlier, may depend on h), we use a fourth-order Runge–Kutta method to integrate the equations of motion from $t = 0$ to $t = 0.1$, using time steps of 0.1, 0.05, 0.025, 0.0125, and 0.00625 and various values of h. From these results we compute, for each value of h, an integration error E_k associated with the time step Δt_k via

$$E_k = \|z^{(\Delta t_k)} - z^{(\Delta t_{k+1})}\| := \max_i |z_i^{(\Delta t_k)}(0.1) - z_i^{(\Delta t_{k+1})}(0.1)|. \quad (5.7)$$

When this quantity is small, there is little change in the integrated positions of the particles using a time step Δt_{k+1} as

TABLE X

Integration Errors in $\|u - u_h\|_\infty$ at $t = 0.1$

	Time step Δt_k				
N	$\Delta t = 0.1$	$\Delta t = 0.05$	$\Delta t = 0.025$	E_0	μ
20	2.2590E-01	2.2590E-01	2.2590E-01	2.2590E-01	1.624
30	1.0016E-01	1.0010E-01	1.0010E-01	1.0010E-01	4.292
40	4.7037E-02	4.6888E-02	4.6886E-02	4.6886E-02	6.301
50	2.4725E-02	2.4435E-02	2.4430E-02	2.4430E-02	5.501
60	1.3718E-02	1.3311E-02	1.3302E-02	1.3302E-02	5.588
80	5.8341E-03	5.2394E-03	5.2268E-03	5.2265E-03	5.560

Note. For significance of E_0 and μ, see text.

bounded above and the required time step is bounded away from zero. However, it must be noted that to determine this fact we had to use the exact solution (5.5) for the velocities. In general, such a solution would not be available, and the procedure carried out above to obtain the data of Table IX would be necessary. Since round-off errors are significant in comparison with truncation errors for $\Delta t < 0.05$, the only meaningful order of convergence calculations are those connecting $\Delta t = 0.1$ and $\Delta t = 0.05$. These give an order of convergence ranging from 3.92 to 4.03, in excellent agreement with the theoretical value.

In Table X we give the errors $\|u - u_h\|$ in the uniform norm and in the 2-norm at $t = 0.1$ with $C = 2$ for various values of Δt and N. Assuming that for each value of N the errors $\|u - u_h\|$ satisfy a relation of the form $E(\Delta t) = E_0 + \alpha(\Delta t)^\mu$, we can estimate the error E_0 appropriate to $\Delta t = 0$ and also the order of convergence μ. These values are also given in Table X. The data of Table X indicate that with $C = 2$ a time increment of $\Delta t = 0.05$ would prove satisfactory for values of $N \leq 50$, while for $C = 2$ and $50 < N \leq 80$ an increment of $\Delta t = 0.025$ would be preferable. Thus the results of these calculations are in reasonable agreement with the more precise calculations reflected in Table IX. In problems of interest, of course, the exact solution would not be available, so the method of determining an appropriate time step used in constructing the data of Table IX would be essential. It may be noted that the values of E_0 in Table X are in good agreement with the errors $\|u - u_h\|_\infty$ at $t = 0$ which appear in Table VIII. While the integration routine being used is fourth order, the convergence as computed using the values of μ is better than expected, being approximately 5.5.

With $\Delta t = 0.025$ the system was integrated from $t = 0$ to $t = 2.0$ using various spatial mesh sizes ($N = 20$, 28, 40, 56, 80). Velocity errors $\|u - u_h\|$ in the uniform norm and in the 2-norm are shown in Fig. 6. As the time step used is thought to be optimum, we would expect that the errors in position due to time integration errors would be small and that as the calculations using the fast method appear to agree with those of direct calculations the remaining errors would be those due to the vortex blob approximation. Thus, we expect these errors to go to zero as $\mathbb{O}(\delta^r)$ or, using the Beale–Majda blobs with $p = 4$, as $\mathbb{O}(h^{4q})$. Convergence in the spatial increment h is apparent in both the uniform norm and the 2-norm, although it is better in the 2-norm. It does not reach the theoretical value of 3.8, appropriate to our value of $q = 0.95$, but it increases as h is reduced. (See Fig. 7.) Note that the rate of convergence deteriorates for larger values of t. Perlman's results [36] indicated that a smaller value of q, so a larger δ, is necessary to improve the accuracy for longer running times. While we have not investigated this question systematically in this paper, partial results we have obtained are in agreement with this result of Perlman.

We also monitor a Hamiltonian for the system (see Section 5D). In Fig. 8 we show values of the Hamiltonian at every tenth time step. At the three finest mesh sizes the variation in the Hamiltonian is observed to be less than three parts in 10^3, which is of the same order of magnitude as the accuracy expected in the calculations of the Hamiltonian.

We now return to a point first raised in Section 4, namely, the relations among $\|u_h^d - u_h^s\|$, $\|u_h^d - u\|$, and $\|u_h^s - u\|$. The first of these represents the error in induced velocity which occurs by virtue of the replacement of direct calculations of blob–blob interactions by calculations using the R-G algorithm. The second may be taken as a measure of the error inherent in vortex blob calculations of velocities, while the third represents the error in computed velocities which results from the use of vortex blobs in conjunction with the R-G algorithm. In using the R-G algorithm we want our results to be the same as those which would be obtained by direct calculations, and we can ensure this by requiring that $\|u_h^d - u_h^s\| \ll \|u_h^d - u\|$. That is, up to the precision of the velocity calculations, $\|u_h^d - u_h^s\|$ is to be essentially zero. For single precision on the Sun we suggested in Section 3B that this accuracy in $\|u_h^d - u_h^s\|$ can be achieved, provided we chose the maximum level $l_{\max}$ in the box structure of the R-G algorithm so as to satisfy the inequality $s_{l_{\max}}/\delta \geq 4.3$. In Section 4 we showed that, at least for the problem of that section, satisfactory values of this ratio were $s_{l_{\max}}/\delta \geq 3.98$.

We mentioned in Section 4 that it might be possible to have the difference $\|u_h^d - u\| - \|u_h^s - u\|$ small in magnitude even though $\|u_h^d - u_h^s\|$ was not negligible. That is, it might be possible to use a maximum level in the box structure which does not satisfy the earlier inequalities and still to get results whose accuracy is satisfactory for some particular purpose. Heretofore we have generally used the value $q = 0.95$ in our calculations. For this value we have found that, by coincidence, the value of $l_{\max}$ which minimizes the computational time also satisfies the inequalities on $s_{l_{\max}}/\delta$. We now choose a value of q for which this is not the case. For our illustration we calculated velocities for the problem of this section, still with $C = 2$, using $q = 0.60$, $h = 2\pi/40$ (so a total of 1640 blobs), and using values $l_{\max} = 2$, 3, 4, and 5. We integrated the system from $t = 0.0$ to $t = 1.0$ using, as before, an integration step $\Delta t = 0.025$. The values of $s_{l_{\max}}/\delta$ and the errors $\|u_h^s - u\|_2$ and $\|u_h^s - u\|_\infty$ at $t = 1.0$ are shown in Table XI.

For this example, the optimum choice of level for minimiza-

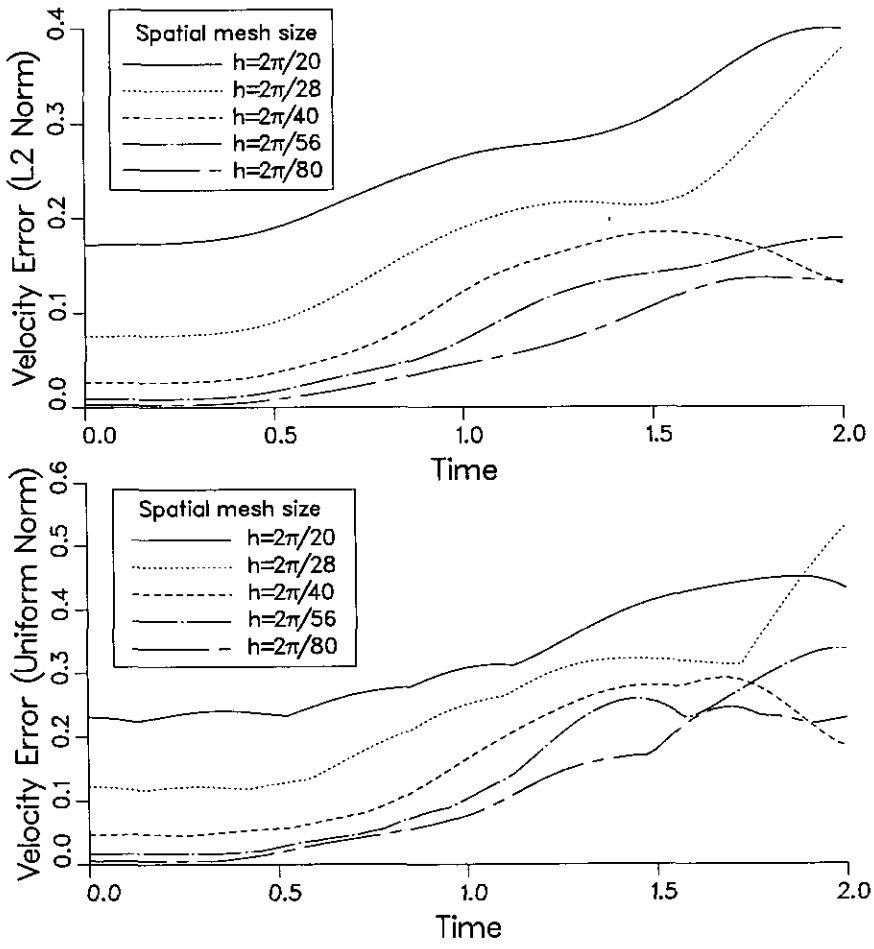

FIG. 6. Velocity errors $\|u - u_h\|$ in L_2 and uniform norms for various values of the step size h in the problem of Section 5B.

tion of computational time is $l_{\max} = 3$, although $l_{\max} = 2$ is not far from optimum (see Fig. 9). We assume that the results with $l_{\max} = 2$ (where $s_{l_{\max}}/\delta > 4.3$) are essentially those which would be obtained using the direct method for calculation of induced velocities. With $l_{\max} = 3$, where $s_{l_{\max}}/\delta \approx 2.4$, the relative error in $\|u_h^\varepsilon - u\|_\infty$ as compared with calculations using $l_{\max} = 2$ is approximately 7.4×10^{-4} at $t = 1.0$ and never exceeds 2.7×10^{-3} for $0.0 \leq t \leq 1.0$. An error of this magnitude might be acceptable in practice in exchange for the smaller computational time required.

On the other hand, if one were to use $l_{\max} = 4$ the relative error in $\|u_h^\varepsilon - u\|_\infty$ as compared with direct calculations (as represented by the calculations with $l_{\max} = 2$), is approximately 7.6% at $t = 1.0$ and exceeds 10% at some values of t in (0.0, 1.0). For the same value of $l_{\max}$ the relative errors in the L_2 norm as compared to direct calculations are never less than 15% and often are near 25%. Results with $l_{\max} = 5$ are worse yet.

TABLE XI

Errors $\|u_h^\varepsilon - u\|$ in L_2 and Uniform Norms at $t = 1.0$ for the Problem of Section 5B with $q = 0.6$, $h = 2\pi/40$, and Various Values of $l_{\max}$

$l_{\max}$	$s_{l_{\max}}/\delta$	$\|u_h^\varepsilon - u\|_2$	$\|u_h^\varepsilon - u\|_\infty$
2	4.77	0.2407390	0.1761023
3	2.38	0.2409178	0.1760369
4	1.19	0.2590528	0.21355111
5	0.60	0.3168969	0.2519368

C. *Variation of Computational Time with the Number of Blobs*

The point of using the R-G algorithm instead of the direct method for calculation of the velocity field is that for large numbers of blobs the $\mathcal{O}(n)$ R-G algorithm will require less computational time than would the $\mathcal{O}(n^2)$ direct method. We must verify that the performance of the R-G algorithm is indeed $\mathcal{O}(n)$.

Running time of the routine was determined for various values of N at various maximum levels of the box structure.

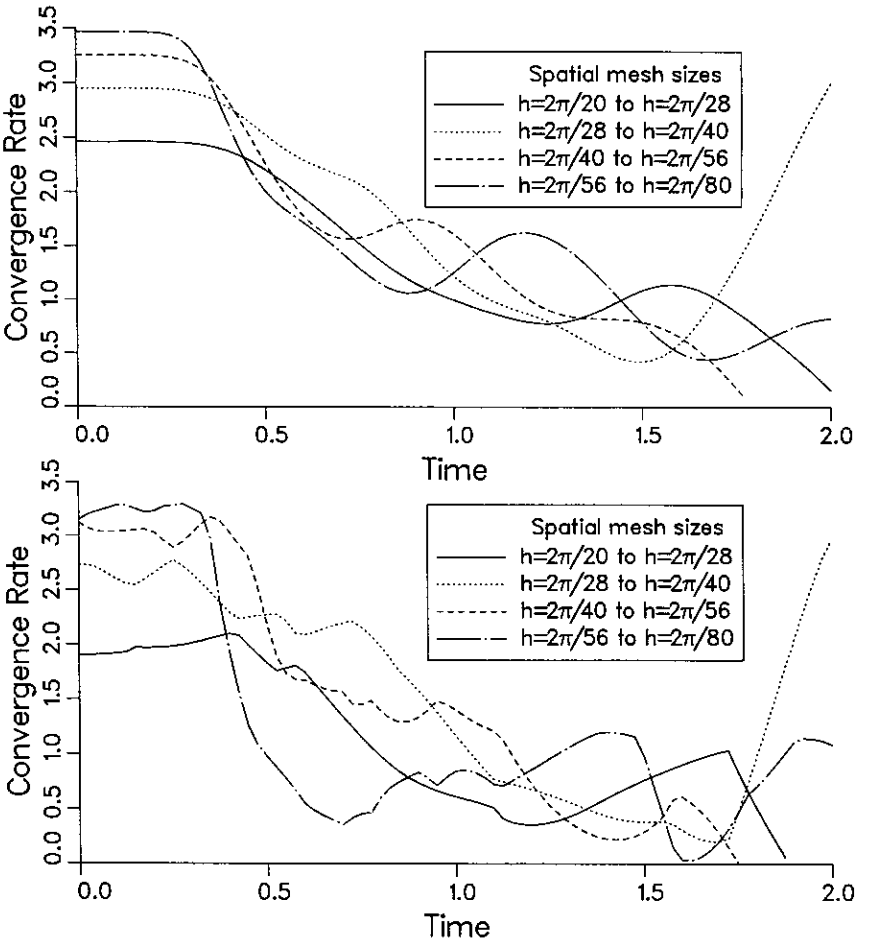

FIG. 7. Fate of convergence in spatial mesh size vs time for various values of the step size h in the problem of Section 5B.

(Note that because of the finite extent of the blobs not all pairs of values of N and l would be suitable in practice. But the fact that the *results* of the computations have no significance doesn't preclude timing them.) In order to get good comparisons with the theoretical predictions for the R-G algorithm we used here the same time increment ($\Delta t = 0.05$) for all values of N, and the Hamiltonian of Section 5D was not computed. We computed the running time in seconds required for 20 integration steps with a fourth-order Runge–Kutta routine. This requires $1 + 4 \times 20 = 81$ evaluations of the velocity field at each blob location. These results are plotted in Fig. 9, which displays a log–log plot of time versus number of blobs $n = N(N + 1)$. To interpret this graph we superpose two constant slope lines with slopes 1 and 2 (dotted in Fig. 9) and also a line which fits the lower envelope of the timing curves. This last line (dashed in Fig. 9) is calculated to have slope ≈ 1.066. These results verify that the R-G algorithm requires—at least approximately—$\mathbb{O}(n)$ floating point operations for the computations of the velocities induced by an ensemble of n vortex blobs. Figure 9 is in excellent agreement with the results of [6]. In particular, see [6, Fig. 9], with the difference in time scales explained by the facts that the evaluation method and computer used by Anderson were different from those used here.

D. *A Hamiltonian for Vortex Blobs with Periodic Structure*

Using vortex blobs of the Beale–Majda form of order $\hat{p} = 4$ on a periodic problem (assumed to have periodicity 1 in the x-direction), the equations of motion are

$$\dot{x}_i = -\frac{1}{2\pi}\sum_{k=-\infty}^{\infty}\sum_{j\neq i}\frac{\omega_j(1 - 2e^{-r_{ijk}^2/\delta^2} + e^{-r_{ijk}^2/2\delta^2})}{r_{ijk}^2}(y_i - y_j) \tag{5.9a}$$

$$\dot{y}_i = \frac{1}{2\pi}\sum_{k=-\infty}^{\infty}\sum_{j\neq i}\frac{\omega_j(1 - 2e^{-r_{ijk}^2/\delta^2} + e^{-r_{ijk}^2/2\delta^2})}{r_{ijk}^2}(x_i - x_j - k), \tag{5.9b}$$

where $r_{ijk} = |z_i - z_j - k| = [(x_i - x_j - k)^2 + (y_i - y_j)^2]^{1/2}$.

If we define $\xi_i = \alpha_i x_i$ and $\eta_i = \alpha_i y_i$, where each α_i is some constant, it is easily verified that

$$\frac{\partial \dot{\xi}_i}{\partial \xi_i} = -\frac{\partial \dot{\eta}_i}{\partial \eta_i}.$$

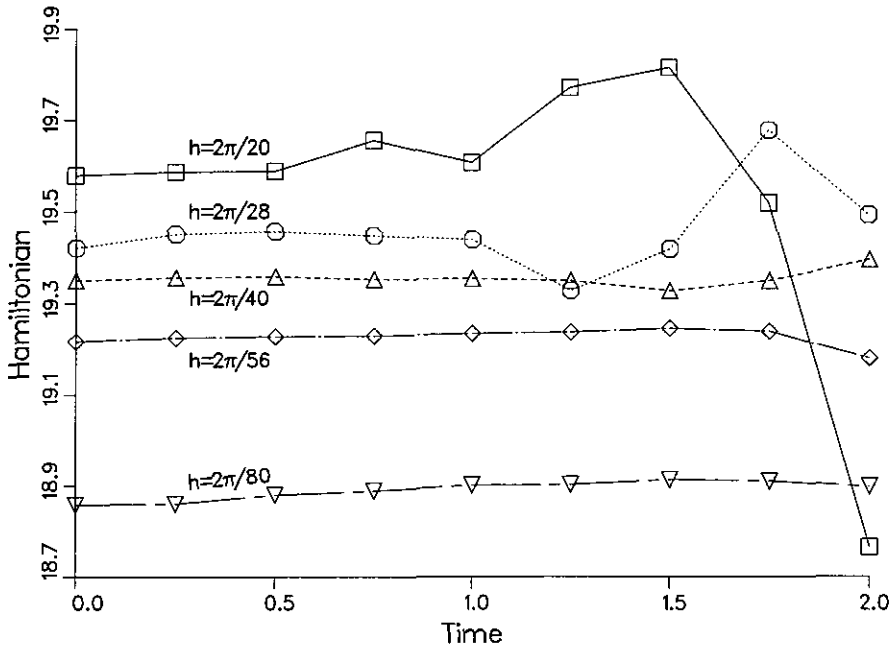

FIG. 8. Hamiltonian vs time for various mesh sizes h for the problem of Section 5B. The Hamiltonian is computed every 10th integration step.

Hence, (5.9a) and (5.9b) define a Hamiltonian system for any choice of the constants α_i. To obtain a Hamiltonian H such that $\dot{\xi}_i = -\partial H/\partial \eta_i$ and $\dot{\eta}_i = \partial H/\partial \xi_i$ we take $\alpha_i = \sqrt{\omega_i}$, and a suitable Hamiltonian function is then given by

$$\frac{1}{4\pi}\sum_{j=1}^{n}\sum_{i>j}^{n}\omega_i\omega_j\left[\log\left(\cosh 2\pi(y_i-y_j)-\cos 2\pi(x_i-x_j)\right)\right. \tag{5.10}$$
$$\left.+\,2\sum_{k=-\infty}^{\infty}\left(E_1[(r_{ijk}/\delta)^2]-\frac{1}{2}E_1[(r_{ijk}/\sqrt{2}\delta)^2]\right)\right],$$

where the function E_1 is defined by

$$E_1(\lambda)=\int_{\lambda}^{\infty}\frac{e^{-\sigma}}{\sigma}\,d\sigma.$$

Here the log term inside the first two summations corresponds to the 1 in the parentheses in the numerators of (5.9a) and (5.9b), while the sum over k corresponds to the exponential terms.

Since the function E_1 is quite small for larger values of its argument, evaluation of (5.10) is not as difficult or time-consuming as it might first appear. Suppose that it is satisfactory to ignore terms of the Hamiltonian smaller than 2×10^{-7}. If the side of the square used as the computational domain is 1, we see that for any pair (i, j) we will have $r_{ijk} < 0.5$ for at most one value of k. As $E_1(\lambda) < e^{-\lambda}/\lambda$ when $\lambda \geq 1$, and as $e^{-13}/13 \approx 1.74 \times 10^{-7}$, $r_{ijk} \geq 0.5$ will imply $E_1[(r_{ijk}/\sqrt{2}\delta)^2] \leq 2 \times 10^{-7}$, provided $\delta \leq 1/\sqrt{104}$. As in Section 3B we now write $N = h^{-1} = 2^{\nu}$ and use the fact that $\delta = h^q$ to rewrite the last inequality for δ as $q\nu \geq 0.5 \log_2(104)$, or, approximately, $q\nu > 3.35$. This calculation shows that if this last inequality is satisfied we are justified in including, for each pair (i, j), at most one of the terms in the summation over k in (5.10). Typical values used in our calculations are $q = 0.95$ with $N \geq 20$, in which case $q\nu \geq 4.1$ and the inequality is satisfied.

Now we may assume, consistent with the assumption above, that we need compute the terms involving the function E_1 only for those r_{ijk} for which $E_1[(r_{ijk}/\sqrt{2}\delta)^2] > 2 \times 10^{-7}$. By the calculations above, we need only calculate these terms when $r_{ijk} \leq \sqrt{26}\delta$. This can be quite small. For example, if $h = \frac{1}{40}$ and $q = 0.95$ then $\delta \approx 0.0301$ and we need consider only $r_{ijk} \leq 0.1533$. Similar considerations apply if the side of the basic computational domain is some value other than unity (as in the Stuart problem of Section 5B). In particular, if the side of the basic computational domain is s_0, the inequality $q\nu \geq 0.5 \log_2(104)$ is replaced by $q\nu + (1 - q)\log_2 s_0 \geq 0.5 \log_2(104)$.

For computations E_1 can be approximated to the accuracy which we desire. If $\lambda \geq 1$ a rational function approximation is available of the form [3, Eq. (5.1.56)]

$$\lambda e^{\lambda}E_1(\lambda)=\frac{\lambda^4+a_1\lambda^3+a_2\lambda^2+a_3\lambda+a_4}{\lambda^4+b_1\lambda^3+b_2\lambda^2+b_3\lambda+b_4}+\varepsilon(\lambda),$$

where $|\varepsilon(\lambda)| < 2 \times 10^{-8}$ and where the a_l and b_l are constants (given in [3]). When $0 < \lambda \leq 1$ we may write E_1 as [3, Eq. (5.1.53)]

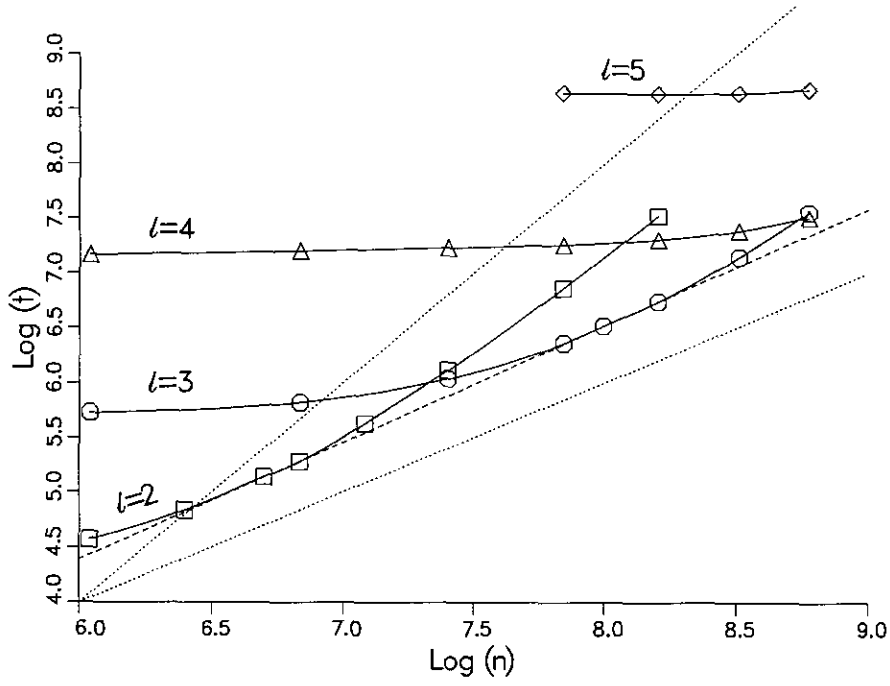

FIG. 9. A log–log plot of computation time t vs number of vortex blobs n for the problem of Section 5B. The dotted lines are lines of slopes 1 and 2, corresponding to $\mathcal{O}(n)$ and $\mathcal{O}(n^2)$, respectively. The dashed line is an empirical fit to the lower envelope of the timing curves and has a slope of ≈ 1.066.

$$E_1(\lambda) = -\log \lambda - \gamma + c_1\lambda + c_2\lambda^2 + c_3\lambda^3 + c_4\lambda^4 + c_5\lambda^5 + \tilde{\varepsilon}(\lambda),$$

where $|\tilde{\varepsilon}(\lambda)| < 2 \times 10^{-7}$, where γ is the Euler–Mascheroni constant, and where the c_l are constants given in [3].

The form of (5.10) makes it clear that even with the above analytic simplifications the computation of the Hamiltonian requires $\mathcal{O}(n^2)$ floating point operations, where n is the number of blobs. As n increases this computation can dominate the calculations. For this reason, we have, in the Stuart problem of Section 5B, calculated the Hamiltonian only every tenth time step. With this limitation, the calculation of the Hamiltonian added only about 25% to the computational time required for even the largest number of blobs used (6480). For larger numbers of blobs, it might be preferable to calculate the Hamiltonian only at the beginning and the end of the computation. It would be desirable to have a computational scheme for the Hamiltonian which requires fewer than $\mathcal{O}(n^2)$ floating point operations. For work along these lines see [43] and the references therein.

6. SUMMARY AND CONCLUSIONS

We have considered the Rokhlin–Greengard fast multipole algorithm when it is used to evaluate vortex blob interactions in a two-dimensional incompressible inviscid fluid. The advantage of the R-G method over the direct evaluation procedure is that the former requires $\mathcal{O}(n)$ floating point operations to evaluate the velocity field induced by the interaction of n blobs, whereas the direct evaluation procedure requires $\mathcal{O}(n^2)$ floating point operations.

We used an exact solution of the incompressible Euler equations in all space to demonstrate that the R-G algorithm with blobs can compute a velocity field which is essentially identical to that obtained by the direct method provided the maximum bisection level in the R-G algorithm satisfies a practical limitation which depends on the structure of the blobs. If this restriction is violated, then the errors which arise from the differences between the R-G evaluation of the velocity field and that of the direct method can be of the same order of magnitude as the errors which arise from the vortex blob method itself, or even larger.

We have also extended the R-G algorithm with blobs to two-dimensional problems which are periodic in one direction and unbounded in the other direction. (The vorticity field in these problems is either bounded or falls off sufficiently rapidly to be treated as bounded.) The same practical restriction on the bisection level as for the R-G algorithm with blobs in all space is required here. When this restriction is observed and the other parameters such as the time step and the number of coefficients used in the various series of the R-G algorithm are chosen appropriately, the method appears to converge at a rate predicted by theory. On the other hand, if the practical restriction on the maximum bisection level is violated, then the errors which arise from the differences between the evaluation of the velocity field by the R-G and direct methods again can be of the same order of magnitude as the errors which arise from the

yortex blob method itself or larger. We have also demonstrated

12. J. T. Beale and A. Maida, *Math. Comput.* **39**, 1 (1982).