JOURNAL OF COMPUTATIONAL PHYSICS 78, 350–368 (1988)

Contour Dynamics for the Euler Equations: Curvature Controlled Initial Node Placement and Accuracy

Q. ZOU,* E. A. OVERMAN II, H.-M. WU,† AND N. J. ZABUSKY

University of Pittsburgh, Department of Mathematics and Statistics,
Institute for Computational Mathematics and Applications,
Pittsburgh, Pennsylvania 15260

Received March 6, 1987; revised November 2, 1987

We have performed a systematic study of several contour dynamical algorithms for the Euler equations for short times. We have used the Kirchhoff elliptical vortex alone and subject to weak perturbations. We have found that if the initial placement of nodes is such that the internodal distance is proportional to (curvature)$^{-p}$ where $p \approx \frac{1}{3}$, then errors in short time calculations are minimized. This follows because the node density is invariant in time. © 1988 Academic Press, Inc.

1. INTRODUCTION

1.1. *Contour Dynamics*

Contour dynamics, a generalization of the water-bag model, was first applied to a plasma physics problem by Berk and Roberts [1]. They investigated numerically the evolution of the two-dimensional distribution function $f(x, v, t)$ where x is a spatial dimension and v is a corresponding velocity dimension. Contour dynamics in two spatial dimensions was first investigated by Zabusky *et al.* [2] for the incompressible Euler Equations. Contour dynamics is a boundary-integral evolutionary method and is an exact representation for a large class of two-dimensional flows where the sources of the flow are piecewise constant densities [3]. That is, the velocity of every point on a contour is given as the derivative of a streamfunction or potential, which is obtained by integrating over all contours or by solving coupled integral equations associated with all contours. If we also include singular vortex or dipole sheets, then these numerical studies originate with Rosenhead's [4] investigation of the roll-up of a vortex sheet. (See the recent work of Krasny [5] and Pullin and Phillips [6].) For the Euler equations in two dimensions, contour dynamics may be viewed as complementary to the continuum algorithmic approach of finite-difference, finite element and pseudospectral methods, where the vorticity, the source of the flow, is expandable in a Taylor series.

* Present address: Department of Mathematics, Kansas State University, Manhattan, KS 66506.

† Permanent address: Computing Center, Chinese Academy of Sciences, Beijing, Peoples Republic of China.

0021-9991/88 $3.00

Copyright © 1988 by Academic Press, Inc.
All rights of reproduction in any form reserved.

In the vortex or dipole sheet representations, the source density may become singular after a finite time. To compute beyond this time, one must introduce regularization procedures. For example, Zabusky and Overman [7] introduced continuum tangential procedures to model aspects of diffusion and dispersion operators and applied these methods with great success to the ionosphere plasma cloud problem. [8] Also, Aref and Tryggvason [9] have used the dispersive regularization inherent in vortex-in-cell codes and Krasny [5] has recently investigated carefully the small-distance cutoff in a procedure called the "blob" method. For these and the Euler equations with piecewise-constant vorticity, one must also deal with numerical inaccuracies introduced when contours develop high curvature regions (or possibly), corners or cusps, or when contours approach closely. Recently, Dritschel [10] has introduced an adaptive contour "surgery" algorithm which cuts and rejoins approaching contours to maintain the accuracy of integration at times beyond the early phase discussed in this paper. The surgery procedure avoids the computationally demanding large growth in nodes but introduces small systematic errors in conserved quantities.

In research in progress we are examining the numerical order of accuracy of algorithms discussed below when a node adjustment algorithm is a part of the evolution.

1.2. *Objectives*

There has not been a systematic study of accuracy of time-dependent CD algorithms. Wu *et al.* [11] used second-order accurate spatial discretization algorithms with node placement adapted to curvature to find V-states, stationary states of the Euler equations in rotating and translating reference frames. Overman and Zabusky [8] investigated the accuracy of time-dependent CD algorithms for plasma clouds by comparing results with growth rates of linearized perturbation analyses.

In this paper, the first in a series of two papers, we address the problem of placing nodes on the contour *initially* so that the error in short time calculations is minimized. We present a systematic study of the evolution of one contour at early times with three second-order algorithms and one third-order algorithm. An important feature is the initial placement of nodes as a function of local curvature. Our criteria for accuracy include: comparing the true and numerically computed angular velocities of Kirchhoff ellipses (containing piecewise-constant vorticity) with and without small-amplitude harmonic perturbations and comparing the evolution of invariant quantities and non-invariant quantities of the Euler equations.

In Section 2 we present the numerical algorithms and their spatial discretization. In Section 3 we discuss node placement algorithms. In Section 4 we discuss temporal discretization. In Section 5 we compare the accuracy with and without temporal evolution and show that errors are minimized if the internodal distance is proportional to (curvature)$^{-p}$ where $p \approx \frac{1}{3}$. In Section 5 we also present the result for a 4-fold symmetric V-state in which case $p \approx \frac{1}{6}$ minimizes the error. In Section 6 we discuss our results and indicate future directions.

2. Contour Dynamical Algorithms

2.1. *Continuous Representation*

Incompressible inviscid vortex motion in two dimensions can be described in vorticity-stream function form by the equations

$$d_t\omega \equiv \omega_t + u\omega_x + v\omega_y = 0, \tag{2.1}$$

$$-\omega \equiv u_y - v_x = \Delta\Psi = \Psi_{xx} + \Psi_{yy}, \tag{2.2}$$

$$(u, v) \equiv (u(x, y, t), v(x, y, t)) = (\Psi_y, -\Psi_x), \tag{2.3}$$

are ω, Ψ, and (u, v) are the vorticity, stream function, and velocities, respectively. Using the Green's function, $G \equiv -(2\pi l)^{-1}\log(r^l)$, for the unbounded domain, we obtain

$$\psi(x, y) = -(2\pi l)^{-1} \iint_{D_j} \omega(\xi, \eta)\log(r^l)\, d\xi\, d\eta, \tag{2.4}$$

where

$$r^2 = (x-\xi)^2 + (y-\eta)^2.$$

We assume that ω is constant in the domain D_j, differentiate the streamfunction, apply Green's theorem to convert a domain integral to a line integral, or obtain

$$(u, v) = -(2\pi l)^{-1} \sum_j [\omega]_j \int_{\partial D_i} \log(r^l)(d\xi, d\eta). \tag{2.5}$$

The sum in (2.5) is taken over all boundaries, ∂D_j, of piecewise constant vorticity regions, D_j, and $[\omega]_j$ denotes the difference of vorticity densities (inside-outside) associated with contour ∂D_j.

We integrate by parts and obtain

$$(u, v) = -(2\pi l)^{-1} \sum_j [\omega]_j \int_{\partial D_j} (x-\xi, y-\eta)\, r^{-l}\, dr^l. \tag{2.6}$$

The discretized version of (2.5) with $l=1$ and of (2.6) with $l=1$ and $l=2$ is designated as the LG$_2$, R1, and R2 algorithms, respectively. Note that the effective integrand, $(x-\xi, y-\eta)\, r^{-l}$ of (2.6) is nonsingular, whereas, the integrand of (2.5) has a logarithmic singularity. This result is important for numerical and theoretical use.

If a point in the field is on a vorticity contour it always remains on the contour. Thus the equations of motion of the contours are

$$\frac{dx}{dt} = u(x, y), \qquad \frac{dy}{dt} = v(x, y), \tag{2.7}$$

where $(x, y) \in \partial D$.

2.2. *Spatial Discretization*

For our second-order algorithms, the contour, ∂D, is approximated by an N-polygon with N nodes $\mathbf{x}_1, ..., \mathbf{x}_N$ ($\mathbf{x}_1 = \mathbf{x}_{N+1}$) on the contour. The integral for (u, v) over ∂D are then replaced by a sum of N integrals, one for each segment $\mathbf{x}_k\mathbf{x}_{k+1}$ ($1 \leqslant k \leqslant N$). When this integration is done exactly, the LG_2 algorithm is obtained. Dritschel [10] has obtained a third-order which we call LG_3, by generalizing this method. He introduced the local cubic polynomial $\eta_n(p)$ to interpolate contours between node $\mathbf{x}_n$ and node $\mathbf{x}_{n+1}$. Thus

$$\mathbf{x}(p) = (x_n, y_n) + p(\Delta x_n, \Delta y_n) + \eta_n(p)(-\Delta y_n, \Delta x_n), \tag{2.8}$$

where

$$\eta_n(p) = \alpha_n p + \beta_n p^2 + \gamma_n p^3, \qquad 0 \leqslant p \leqslant 1. \tag{2.9}$$

The coefficients α_n, β_n, and γ_n, and y_n are determined from

$$\eta_n(1) = 0, \qquad \kappa(0) = \kappa_n, \qquad \kappa(1) = \kappa_{n+1}, \tag{2.10}$$

where κ_n is the curvature at node $\mathbf{x}_n$, approximated by the inverse radius of a circle fitted to the nodes $n-1$, n, and $n+1$. The result is given by the following collection of formulas

$$(u, v) = -(2\pi)^{-1} \sum_j [\omega]_j \sum_{n=1}^{N_j} (G_n^{(2)} + G_n^{(3)})(\Delta x_n, \Delta y_n) + \tilde{G}_n^{(3)}(-\Delta y_n, \Delta x_n), \tag{2.11}$$

where $[\omega]_j$ is the vorticity jump (inside–outside) of the jth contour. For $\mathbf{x} \neq \mathbf{x}_n$ or $\mathbf{x}_{n+1}$,

$$G_n^{(2)} = 1 - \log h_n - \tfrac{1}{2} a_n d_n - \tfrac{1}{2} b_n (1 - d_n) - c_n^2 z_n, \tag{2.12a}$$

$$G_n^{(3)} = c_n [\beta_n + (\tfrac{1}{2} + 2d_n)\gamma_n + \tfrac{1}{2} f_n (b_n - a_n) + g_n z_n], \tag{2.12b}$$

$$\tilde{G}_n^{(3)} = \eta_n(d_n)/d_n + \tfrac{1}{2}\beta_n + (\tfrac{1}{3} + \tfrac{1}{2} d_n - c_n^2)\gamma_n + \tfrac{1}{2} g_n (b_n - a_n) - c_n^2 z_n f_n, \tag{2.12c}$$

where

$$a_n = \log(c_n^2 + d_n^2), \tag{2.13}$$

$$b_n = \log(c_n^2 + (1 - d_n)^2), \tag{2.14}$$

$$c_n = [(y - y_n)\, \Delta x_n - (x - x_n)\, \Delta y_n]/h_n^2, \tag{2.15}$$

$$d_n = [(x - x_n)\, \Delta x_n + (y - y_n)\, \Delta y_n]/h_n^2, \tag{2.16}$$

$$\Delta x_n = x_{n+1} - x_n, \; \Delta y_n = y_{n+1} - y_n, \qquad h_n^2 = (\Delta x_n)^2 + (\Delta y_n)^2, \tag{2.17}$$

$$f_n = (\partial\eta_n/\partial p)|_{d_n} - c_n^2\gamma_n, \tag{2.18}$$

$$g_n = \eta_n(d_n) - c_n^2(\beta_n + 3d_n\alpha_n), \tag{2.19}$$

$$z_n = (1/c_n)\left[\tan^{-1}\left(\frac{1-d_n}{c_n}\right) + \tan^{-1}\left(\frac{d_n}{c_n}\right)\right]. \tag{2.20}$$

For $\mathbf{x} = \mathbf{x}_n$,

$$G_n^{(2)} = 1 - \log h_n, \qquad G_n^{(3)} = 0, \qquad \tilde{G}_n^3 = \alpha_n + \tfrac{1}{2}\beta_n + \tfrac{1}{3}\gamma_n, \tag{2.21}$$

and for $\mathbf{x} = \mathbf{x}_{n+1}$,

$$G_n^{(2)} = 1 - \log h_n, \qquad G_n^{(3)} = 0, \qquad \tilde{G}_n^{(3)} = \tfrac{1}{2}\beta_n + \tfrac{5}{6}\gamma_n. \tag{2.22}$$

If $G_n^{(3)} = \tilde{G}_n^{(3)} = 0$, we recover the second-order accurate algorithm, LG_2, introduced by Overman and Zabusky [12]. When $\mathbf{x}$ is inside the circle whose diameter is the segment connecting two adjacent nodes $\mathbf{x}_n$ and $\mathbf{x}_{n+1}$, the formula is not valid in (2.12a) and Dritschel shifts the segment $(\mathbf{x}_n, \mathbf{x}_{n+1})$ parallel to itself by $\eta(p^*)$, where $p^* = d_n$. In our simulations we have also used his modified formulas, when required.

We now study two other second-order accurate algorithms, R1 and R2. These are faster then LG_2 because they do not involve any transcendental functions. Furthermore, R2 does not even involve square roots. These are obtained from (2.6) with a midpoint integration rule as

$$(u, v) = (2\pi l)^{-1} \sum_j [\omega]_j \sum_{n=1}^{N_j} (\Delta u_n, \Delta v_n)\, \Delta(r_n^l), \tag{2.23}$$

where the j-sum is over all the contours and

$$(\Delta u_n, \Delta v_n) = (x - x_{n+1/2},\ y - y_{n+1/2})\, r_{n+1/2}^{-l}, \tag{2.24}$$

$$(x_{n+1/2},\ y_{n+1/2}) = \tfrac{1}{2}(x_n + x_{n+1},\ y_n + y_{n+1}), \tag{2.25a}$$

$$r_n^2 = = (x - x_n)^2 + (y - y_n)^2, \tag{2.25b}$$

$$r_{n+1/2}^2 = (x - x_{n+1/2})^2 + (y - y_{n+1/2})^2, \tag{2.25c}$$

$$\Delta(r_n^l) = r_{n+1}^l - r_n^l. \tag{2.25d}$$

In Section 5 we compare properties of these algorithms. Although it seems that the argument of the sum in (2.23) is singular if $\mathbf{x}$ lies on the line between $\mathbf{x}_{n+1}$ and $\mathbf{x}_n$, the singularity is only apparent. For $l = 1$, the argument may be written as the product of $r_{n+1/2}^{-1}(r_{n+1} - r_n)(x - x_{n+1/2},\ y - y_{n+1/2})$. As $\mathbf{x}$ approaches $\mathbf{x}_{n+1/2}$, the pair approaches $(\cos\alpha, \sin\alpha)\ (r_{n+1} - r_n)$, where α is the angle between the x-axis and the line between $\mathbf{x}$ and $\mathbf{x}_{n+1/2}$. For $l = 2$, $(\Delta u_n, \Delta v_n)\, \Delta r_n^l$ is a product of a finite number and a number $O(h_n)$. Hence these formulas are nonsingular.

All algorithms lose accuracy as different contours or different regions of the same contour approach closely. However, R1 and R2 lose accuracy much faster than LG_2 and LG_3.

3. Initial Node Placement

3.1. *Introduction*

The study of accuracy of numerical discretization procedures usually begins with equally spaced meshes. However, the motion of nodes on a contour is usually nonuniform and the internodal distances are constantly changing. Thus it is obvious that to avoid very large and very small intervals one must employ an adaptive node placement algorithm. A step in this direction was taken by Zabusky and Overman [13]. They developed an algorithm where the internodal distance was inversely proportional to the absolute value of the local curvature κ and satisfied constraint conditions. The curvature was obtained by fitting cubic splines to the coordinates of the curve $(x(s), y(s))$, and then differentiating. Note that adaptive techniques are also being considered for finite difference methods by Sanz-Serna and Christie [14]. However, the idea of using a function of the curvature as the internodal distance was not employed in their work. We will not consider the most general theory for this process at the present time, for the precise location of the nodes is not essential.

We now generalize the work in [13] by relating the internodal distance to a power of the curvature. We motivate this procedure by considering the Kirchhoff *ellipse* and find in a time-dependent sensitivity study (described in Section 5) that $p=\frac{1}{3}$ gives the best results for it maintains a *stationary* distribution of nodes on the rotating contour.

3.2. *Initial Node Placement on Convex Curves*

Our internodal spacing algorithm for convex curves is based on the formula

$$\frac{ds}{d\mu}=\frac{c}{\kappa^p}, \tag{3.1}$$

where s is the arc length, κ is the curvature, and c and p are constants. μ is a parameter such that the points are uniformly distributed in μ.

We first consider the ellipse $(x, y)=(a\cos\eta, b\sin\eta)$, where η is the angular variable in elliptical coordinates. Hence, the curvature and differential arc length are

$$\kappa=\frac{x'y''-x''y'}{(x^2+y^2)^{3/2}}=\frac{ab}{(a^2\sin^2\eta+b^2\cos^2\eta)^{3/2}}, \tag{3.2}$$

$$(ds)^2\equiv(dx)^2+(dy)^2=(a^2\sin^2\eta+b^2\cos^2\eta)(d\eta)^2, \tag{3.3}$$

and

$$\frac{ds}{d\eta}=\frac{(ab)^{1/3}}{\kappa^{1/3}}. \tag{3.4}$$

From (3.2)–(3.4), we obtain the relation between η and μ

$$d\mu=\frac{(ab)^{1/3}}{c}\kappa^{p-1/3}\,d\eta/[b^2+(a^2-b^2)\sin^2\eta]^{(3p-1)/2}$$

or

$$\mu_1-\mu_0=c_3\int_{\eta_0}^{\eta_1}\frac{d\eta}{[b^2+(a^2-b^2)\sin^2\eta]^{(3p-1)/2}}, \tag{3.5}$$

Thus, $p=\frac{1}{3}$ yields an equal spacing in η.

To initialize the ellipse, we distribute points on the first quarter of the ellipse according to (3.5). We take $\eta_0=\mu_0=0$ and, to determine c_3, we take $\mu=N/4$ at $\eta_1=\pi/2$, corresponding to N points on the entire ellipse. We evaluate (3.5) numerically and determine c_3, with the trapezoidal rule using very many ($\approx 10^3$) equally spaced points in η. Then we interpolate linearly to obtain the value of η_k corresponding to integer values of μ_k. After obtaining the point distribution on the first quarter, we can obtain the point distribution on the remainder of the ellipse by reflection. A similar procedure is used for figures with higher m-symmetries, e.g., $m=4$ in Section 7.

4. Temporal Discretization

The motion of points on the contour is given by the pair

$$\frac{dx}{dt}=u(x,y,t) \qquad \text{and} \qquad \frac{dy}{dt}=v(x,y,t), \tag{4.1}$$

where (u, v) are obtained as line integrals discussed in Section 2. Thus at any time step we solve $2N$ ordinary differential equations where (u, v) are given in the discrete representation (2.11), (2.23), etc. In this study we are interested only in errors arising from spatial discretization and node placement methods. To minimize temporal discretization errors we choose the fourth-order Runge–Kutta method and a sufficiently small time step so that all figures shown are significant.

5. Static and Time-Dependent Accuracy Comparisons

5.1. *Introduction*

In this section we compare the accuracy of the integration and node placement algorithms for both static and time-dependent runs. We introduce the analytical formulas in Section 5.2 and present the comparisons in Sections 5.3 and 5.4. The term "relative error" for a scalar s throughout the comparison is defined as $(s_c - s_*)/s_*$, where s_c is the computed value of s and s_* is the true value of s. The term "l_2 error" for a vector $\mathbf{v}$ (the discrete representation of a function) is defined as $\|\mathbf{v}_c - \mathbf{v}_*\|/\|\mathbf{v}_*\|$, where $\| * \|$ indicates the l_2-norm.

5.2. *Analytical Results*

An elliptical distribution of vorticity is a steady-state solution of the Euler equations with angular velocity

$$\Omega = \omega_0 ab/(a+b)^2, \tag{5.1}$$

where a and b are the semi-major and semi-minor axes, respectively, and ω_0 is the vorticity [16]. The path of the particles on the ellipse in the laboratory frame at time t is given by

$$\begin{aligned} x &= \tfrac{1}{2}(a+b)\cos(2\Omega t + \sigma_0) + \tfrac{1}{2}(a-b)\cos\sigma_0, \\ y &= \tfrac{1}{2}(a+b)\sin(2\Omega t + \sigma_0) - \tfrac{1}{2}(a-b)\sin\sigma_0, \end{aligned} \tag{5.2}$$

where σ_0 is a parameter. The velocity (u, v) if a particle on the ellipse is given by

$$u = -\frac{\omega_0 a}{a+b}\, y, \qquad v = \frac{\omega_0 b}{a+b}\, x. \tag{5.3}$$

Next we summarize Love's result [17] for the linear theory of perturbations on an ellipse. Elliptical coordinates (ξ, η) are related to Cartesian coordinates (x, y) by

$$(x, y) = c(\cosh\xi\cos\eta, \sinh\xi\sin\eta), \qquad c^2 = a^2 - b^2, \tag{5.4}$$

and the ellipse is

$$\xi \equiv \xi_0 = \tfrac{1}{2}\log[(a+b)/(a-b)], \qquad 0 \leqslant \eta \leqslant 2\pi. \tag{5.4}$$

Note, (x, y) are the coordinates relative to a frame fixed in the ellipse. In the linear perturbation analysis, ξ_0 is replaced by $\xi_0 + \xi_1(\eta, t)(\xi_1/\xi_0 \ll 1)$, where

$$\xi_1 = h_0^2 \left[\sum_m \alpha_m(t)\cos m\eta + \beta_m(t)\sin m\eta) \right] \equiv h_0^2 \xi_2, \tag{5.6}$$

and

$$h_0^2 = (a^2\sin^2\eta + b^2\cos^2\eta)^{-1} = (2/c^2)(\cosh 2\xi_0 - \cos 2\eta)^{-1} \tag{5.7}$$

is the inverse of the Jacobian of the transformation. According to the linear analysis, $\alpha_m(t)$, $\beta_m(t)$ are proportional to $e^{i\sigma t}$ or $e^{\pm\alpha t}$, where

$$\sigma^2 = c_m d_m \ (c_m d_m > 0) \qquad \text{or} \qquad \alpha^2 = -c_m d_m \ (c_m d_m < 0), \tag{5.8}$$

$$c_m = \frac{\omega_0}{2}\left[\frac{2mab}{(a+b)^2} - 1 + \left(\frac{a-b}{a+b}\right)^m\right],$$

$$d_m = \frac{\omega_0}{2}\left[\frac{2mab}{(a+b)^2} - 1 - \left(\frac{a-b}{a+b}\right)^m\right]. \tag{5.9}$$

When $m = 1$, which corresponds to a displacement of the ellipse, $c_m d_m > 0$, the motion is stable. When $m = 2$, the contour remains elliptic, $c_m d_m = 0$. When $m = 3$, $c_m d_m > 0$ for $a/b < 3.0$ and $c_m d_m < 0$ for $a/b > 3.0$. When $m > 3$, $c_m d_m > 0$ for all a, b. Thus, for $m = 3$ and $a/b > 3.0$, the motion is unstable.

If we choose $m = 3$ and $\xi_2 = \varepsilon \cos m\eta$ initially, then at any time t,

$$\xi_2 = \varepsilon\left(\cos\sigma t \cos m\eta + \sqrt{\left(\frac{d_m}{c_m}\right)^{1/2}} \sin\sigma t \sin m\eta\right), \qquad \text{when} \quad c_m d_m > 0, \tag{5.10}$$

or

$$\xi_2 = \varepsilon\left(\cosh\alpha t\, m\eta - \sqrt{\left(\frac{-d_m}{c_m}\right)^{1/2}} \sinh\alpha t \sin m\eta\right), \qquad \text{when} \quad c_m s_m < 0. \tag{5.11}$$

To determine the quantity $\xi_2 = \xi_1/h_0^2$ from the numerical values of (x, y) at later times, we first find the major axis of the ellipse (perturbed or not) by fitting a spline to all points and then calculating second-order moments. We find the coordinates $(\hat{x}, \hat{y})$ by rotating our coordinate system to coincide with the major axis. (This compensates for the error in the rotation of the corresponding unperturbed ellipse.) We use formula (5.4) to find the corresponding elliptical coordinates $(\hat{\xi}, \hat{\eta})$. Thus,

$$\hat{\xi}_2 = (\hat{\xi} - \xi_0)/h_0^2(\hat{\eta}). \tag{5.12}$$

5.3. *Static Comparisons*

In the study of steady-state solutions [11], we showed that rotating V-states satisfy the nonlinear integrodifferential equation

$$-u\, dy + v\, dx = \Omega R\, dR, \tag{5.13}$$

where all variables u, v, x, y, and the polar radius of the contour R are functions of the polar angle θ and Ω is the derived angular velocity. For our purpose we arrange (5.13) and define the local angular velocity as

$$\Omega_\theta \equiv 2\left(v - u\frac{dy}{dx}\right)\Big/(dR^2/dx). \tag{5.14}$$

For the (a, b) ellipse this is

$$\Omega_\theta = (a^2 yv + b^2 xu)/[(a^2 - b^2)\, xy]. \tag{5.15}$$

We then compute Ω_θ for the values of (x, y) used, and obtain Ω_θ for $\theta = 30°$ and $\theta = 60°$ by cubic spline interpolation. To obtain a global angular velocity, we integrate (5.13) in a quarter ellipse and obtain

$$\Omega = \frac{2}{a^2 - b^2} \int_{\theta=0}^{\theta=\pi/2} u\, dy - v\, dx. \tag{5.16}$$

The global or "average" angular velocity, Ω_{gl}, is obtained by summing the discretized form of (5.13) from 0° to 90°, or

$$\Omega_{\text{gl}} \equiv (a^2 - b^2)^{-1} \sum_k (u_{k+1} + u_k)\, \Delta y_k - (v_{k+1} + v_k)\, \Delta x_k. \tag{5.17}$$

In Fig. 1, we compare $\Delta\Omega_{\text{gl}}/\Omega = (\Omega_{\text{gl}} - \Omega)/\Omega$ vs p for the three algorithms in static case (without time evolution). The comparisons are made with two ellipses, 2 : 1 and 4 : 1, and two discretizations, $N = 32$ and 64. Here we see a monotonic variation in all the curves. For LG_2 and R1, $\Delta\Omega_{\text{gl}} = 0$ for $p = p_*$, where $\frac{1}{4} \leqslant p_* \leqslant \frac{1}{3}$

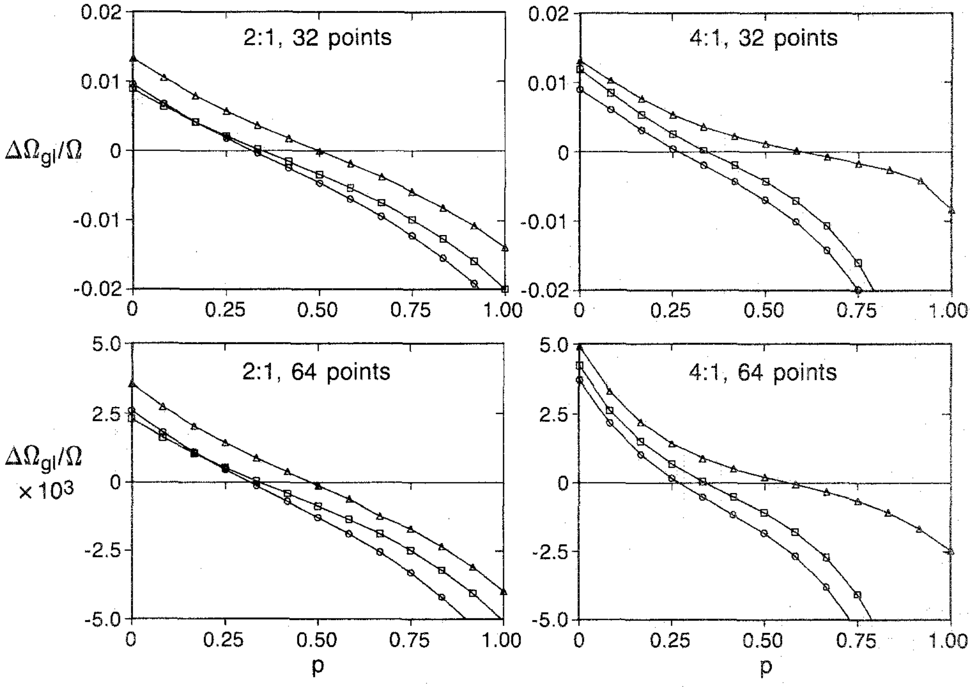

FIG. 1. $\Delta\Omega_{\text{gl}}/\Omega$ vs p for 2:1 and 4:1 ellipses with the LG_2, R1, R2 methods, $N = 32$ and 64. Square symbol represents the result of the LG_2 method, circle the R1 method and triangle the R2 method.

and $\frac{1}{2} \leqslant p_* \leqslant \frac{3}{5}$ for R2. We conclude that we can always find p_*, where $\Delta\Omega_{\text{gl}} = 0$. Note that p_* is insensitive to N, weakly dependent on a/b, and somewhat more dependent on the integration algorithm. We note but do not show that the relative errors of Ω_θ for $\theta = 30°$ and $60°$ have the same features as $\Delta\Omega_{\text{gl}}/\Omega$. The l_2 error in u, v has a minimum at the place close to p_*. The relative error of Ω_{gl} and the l_2 error of (u, v) are shown in Table I for the cases of a 2 : 1 ellipse with $N = 64$, and the LG_2 and LG_3 methods. The ratio of CPU times for the LG_3, LG_2, R1, and R2 methods is 6.7:2.2:1.6:1.0 on the VAX 8600 (using double precision) and 6.2:1.3:1.1:1.0 on the CRAY XMP-48.

Comparable results for time-dependent runs of an unperturbed ellipse are also shown in Table I. Errors are evaluated after one revolution, $T = 9\pi = 28.27$

TABLE I

Comparison Study for a 2:1 Unperturbed Ellipse with $N = 64$, Using the LG_2 and LG_3 Algorithms

		Static		Dynamic ($\Delta t = 0.141$)		
	$12p$	$\Delta\Omega_{\text{gl}}/\Omega$	l_2 error in (u, v)	$\Delta\Omega_A/\Omega$	$\Delta\Omega_0/\Omega$	Mean error in ξ_2
LG_2	0	0.230(−2)	0.119(−2)	0.112(−3)	−0.144(−2)	0.703(−4)
	1	0.162(−2)	0.105(−2)	0.812(−4)	−0.124(−2)	0.450(−4)
	2	0.104(−2)	0.959(−3)	0.549(−4)	−0.109(−2)	0.252(−4)
	3	0.521(−3)	0.905(−3)	0.317(−4)	−0.985(−3)	0.100(−4)
	4	0.455(−4)	0.888(−3)	0.105(−4)	−0.920(−3)	0.125(−5)
	5	−0.412(−3)	0.905(−3)	−0.969(−5)	−0.889(−3)	0.921(−5)
	6	−0.872(−3)	0.955(−3)	−0.298(−4)	−0.887(−3)	0.143(−4)
	7	−0.136(−2)	0.103(−2)	−0.508(−4)	−0.913(−3)	0.166(−4)
	8	−0.189(−2)	0.115(−2)	−0.736(−4)	−0.969(−3)	0.162(−4)
	9	−0.249(−2)	0.129(−2)	−0.993(−4)	−0.105(−2)	0.130(−4)
	10	−0.320(−2)	0.148(−2)	−0.129(−3)	−0.118(−2)	0.703(−5)
	11	−0.404(−2)	0.171(−2)	−0.165(−3)	−0.134(−2)	0.766(−5)
	12	−0.507(−2)	0.199(−2)	−0.208(−3)	−0.155(−2)	0.234(−4)
LG_3	0	0.763(−4)	0.910(−4)	0.243(−4)	0.102(−3)	0.173(−4)
	1	0.521(−4)	0.723(−4)	0.197(−4)	0.789(−4)	0.981(−5)
	2	0.368(−4)	0.584(−4)	0.173(−4)	0.632(−4)	0.472(−5)
	3	0.274(−4)	0.482(−4)	0.165(−4)	0.523(−4)	0.996(−6)
	4	0.219(−4)	0.409(−4)	0.168(−4)	0.449(−4)	0.217(−5)
	5	0.193(−4)	0.358(−4)	0.184(−4)	0.402(−4)	0.519(−5)
	6	0.189(−4)	0.327(−4)	0.213(−4)	0.382(−4)	0.859(−5)
	7	0.205(−4)	0.313(−4)	0.258(−4)	0.388(−4)	0.128(−4)
	8	0.241(−4)	0.316(−4)	0.326(−4)	0.424(−4)	0.185(−4)
	9	0.303(−4)	0.336(−4)	0.425(−4)	0.498(−4)	0.264(−4)
	10	0.400(−4)	0.375(−4)	0.572(−4)	0.621(−4)	0.377(−4)
	11	0.545(−4)	0.436(−4)	0.789(−4)	0.815(−4)	0.542(−4)
	12	0.763(−4)	0.524(−4)	0.111(−3)	0.111(−3)	0.785(−4)

Note. $(-\alpha) \equiv 10^{-\alpha}$.

(200 time steps), of a 2:1 ellipse. In Table I we present the following temporally varying quantities:

(1) Relative error of Ω_A, where Ω_A is the angle of the major axis divided by time. To obtain this angle, we use the spline-second moment procedure described above.

(2) Relative error of Ω_0, where Ω_0 is the angle of the point originally at $(a, 0)$ divided by time.

(3) The mean error in $\hat{\xi}_2$ (see (5.12)), which represents the distortion of the ellipse. The mean error here is defined as $(\sum_{k=1}^{N} \hat{\xi}_{2,k}^2)^{1/2}/N$, $\hat{\xi}_{2,k}$ is the value of $\hat{\xi}_2$ at the kth point, and its true value is zero for an unperturbed ellipse. We note that the LG_2 algorithm gives dynamic results that are more consistent with static results than the LG_3 algorithm, i.e., $p_* \approx \frac{1}{3}$ minimizes most errors. Note, for the R1 or R2 algorithms $p_* \approx \frac{1}{3}$ minimizes dynamical errors (not shown).

Table II shows the static l_2 error of (u, v) for a 2:1 ellipse, which corresponds to column 2 in Table I, all discretization methods are compared for $N = 64$, 128, 256, 512 and the normalized ratio, e_N/e_{512}, clearly indicates that LG_3 is third order and the others are second-order accurate.

TABLE II

Study of the Order of Accuracy for a 2:1 Unperturbed Ellipse with $p = \frac{1}{3}$

	Number of nodes N	32	64	128	256	512
LG_2	l_2 error e_N in u, v	0.369(−2)	0.888(−3)	0.217(−3)	0.538(−4)	0.134(−4)
	e_N/e_{512}	276	66	16	4	1
R1	l_2 error e_N in u, v	0.365(−2)	0.883(−3)	0.217(−3)	0.538(−4)	0.134(−4)
	e_N/e_{512}	273	66	16	4	1
R2	l_2 error e_N in u, v	0.288(−2)	0.655(−3)	0.155(−3)	0.377(−4)	0.928(−5)
	e_N/e_{512}	310	71	17	4	1
LG_3	l_2 error e_N in u, v	0.347(−3)	0.409(−4)	0.501(−5)	0.623(−6)	0.776(−7)
	e_N/e_{512}	4476	526	65	8	1

5.4. *Temporal Comparisons*

We present here a comparison of temporal errors obtained with the fourth Runge–Kutta method. We use a 4:1 unperturbed ellipse, the R1 algorithm, $N = 128$, and 60, 120, 240, 480 time steps in one revolution. The upper half shows the forward evolution and the lower half the return evolution. In Table III, the quantities shown are

(1) error in rotation angle of major axis,

(2) l_2 error in (x, y),

TABLE III

Temporal Study for a 4:1 Ellipse with $N = 128$, $p = \frac{1}{3}$, the R1 Algorithm, Forward One Revolution and Return to Zero

Number of time steps (Δt)	60(0.655)	120(0.327)	240(0.164)	480(0.0818)
		Forward to $T = 39.27$		
Error in rotation angle of major axis (degrees)	−0.101(0)	−0.538(−1)	−0.509(−1)	−0.507(−1)
$(e_n - e_{480})/(e_{240} - e_{480})$	286	17	1	
l_2 error in (x, y)	0.476(−2)	0.336(−2)	0.327(−2)	0.327(−2)
	289	17	1	
Relative error in area	−0.639(−4)	−0.187(−5)	−0.832(−7)	−0.285(−7)
	1167	34	1	
Relative error in maximum curvature	−0.536(−3)	−0.493(−3)	−0.489(−4)	−0.489(−4)
	141	14	1	
		Return to Zero		
Error in rotation angle of major axis	0.852(−2)	0.247(−3)	0.757(−5)	0.235(−6)
$(e_n - e_{480})/(e_{240} - e_{480})$	1161	34	1	
l_2 error in (x, y)	0.257(−3)	0.745(−5)	0.228(−6)	0.709(−8)
	1161	34	1	
Relative error in area	−0.127(−3)	−0.369(−5)	−0.113(−6)	0.350(−8)
	1166	34	1	
Relative error in maximum curvature	0.180(−3)	0.522(−5)	0.160(−6)	0.495(−8)
	1163	34	1	

(3) relative error in area,

(4) relative error in maximum curvature.

For each error e_n, the following row is the quantity $(e_n - e_{480})/(e_{240} - e_{480})$, where n is the corresponding number of time steps. We know that the R1 method is second-order accurate in space discretization. If the time discretization error is of order l, then we expect an error e as $e = a(\Delta x)^2 + b(\Delta t)^l$, where a and b are constants, Δx is the typical space step and Δt the time step. Thus if $l = 4$ for $n = 60$, 120, 240. $(e_n - e_{280})/(e_{240} - e_{480}) = 273$, 17, 1, respectively, and if $l = 5$, $(e_n - e_{480})/(e_{240} - e_{480}) = 1057$, 33, 1, respectively. If we examine the quantities $(e_n - e_{480})/(e_{240} - e_{480})$ in Table III, we see after one revolution that the error in the rotation angle and the l_2 error in (x, y) are fourth-order accurate in time, the relative error in area is fifth-order accurate, and the relative error in maximum curvature is approximately fourth-order accurate. When we return to $t = 0$, *all quantities are fifth-order accurate*, which is the one-step error of the Runge–Kutta scheme [15].

5.5. *Time Dependent Comparison for a Perturbed Ellipse* (*TDP*)

We use 2:1 and 2:0.5 (4:1) ellipses perturbed with $m = 3$, $\varepsilon = 0.01$, and 128 points and compute with the R1 method. The 2:1 ellipse is stable and the 4:1 ellipse is

expansion to be valid and large enough to be greater than the discretization error.

In Section 5.2 we described how we found the numerical value of ξ_2. To validate this approach of finding the major axis of the ellipse, we changed the angle of major axis by small amount (e.g., 0.01°) in both directions and found the error in ξ_2 increases. This gave us the confidence that we were using an optimum method for finding the effective extra angle of the major axis.

The results in Table IV are:

(1) Stable (2:1): The relative error in average wave speed after one revolution. The average wave speed is obtained in the following way: after obtaining the numerical value ξ_2, we determine the six extrema in ξ_2 by spline interpolation, the travelled distance of the extrema from $t = 0$ to the present time divided by time gives the average wave speed c_k $(k = 1–6)$ of the perturbation wave, and we then obtain the mean of those wave speeds and compute the relative error of the mean wave speed. Note that the true values are found from (5.10) and (5.11) in same way.

Unstable (4:1): The relative error in the average wave amplitude after 1/6 revolution. We find the six extrema and find the mean of the six absolute values of these extrema (the amplitude of the wave) and compute its relative error.

(2) l_2 error in ξ_2.

Because of the exponential increase of the wave amplitude, the linear theory is not valid for a long time in an unstable case. From Table IV, we see the minimum error occurs at p_*, when $\frac{2}{12} \leqslant p_* \leqslant \frac{7}{12}$.

Figure 2 shows the perturbation wave ξ_2 (2:1 ellipse) and Fig. 3 shows the

TABLE IV

Comparison of 2:1 and 4:1 Ellipses with $N = 128$ Nodes, the R1 Algorithm and a Perturbation $\xi_2(0) = 0.01 \cos 3\eta$

	2:1		4:1	
$12p$	Relative error in wave speed	l_2 error in ξ_2	Relative error in wave amplitude	l_2 error in ξ_2
0	0.209(−3)	0.569(−1)	0.637(−1)	0.129
1	0.186(−3)	0.494(−1)	0.366(−1)	0.895(−1)
2	0.164(−3)	0.449(−1)	0.201(−1)	0.666(−1)
3	0.146(−3)	0.425(−1)	0.144(−1)	0.534(−1)
4	0.132(−3)	0.414(−1)	0.185(−1)	0.456(−1)
5	0.121(−3)	0.411(−1)	0.262(−1)	0.414(−1)
6	0.114(−3)	0.414(−1)	0.358(−1)	0.405(−1)
7	0.111(−3)	0.418(−1)	0.482(−1)	0.449(−1)
8	0.112(−3)	0.425(−1)	0.656(−1)	0.576(−1)
9	0.116(−3)	0.436(−1)	0.911(−1)	0.823(−1)
10	0.125(−3)	0.452(−1)	0.130	0.124
11	0.139(−3)	0.479(−1)	0.191	0.190
12	0.158(−3)	0.523(−1)	0.287	0.294

Note. For 2:1, $\Delta t = 0.141$ and $T_F =$ one revolution. For 4:1, $\Delta t = 0.164$ and $T_F = \frac{1}{6}$ revolution.

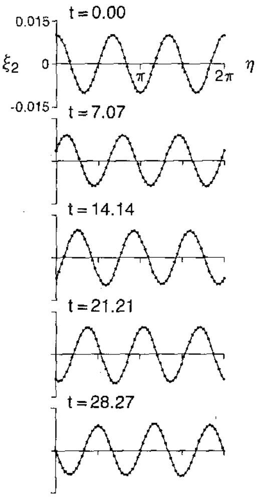

FIG. 2. The perturbation wave ξ_2 vs η for a 2:1 ellipse, with the R2 method, $N = 64$, $p = \frac{1}{3}$, $\Delta t = 0.14137$, $\varepsilon = 0.01$.

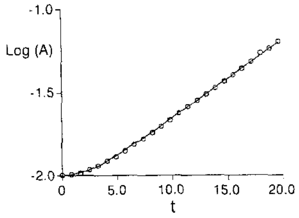

FIG. 3. The logarithm of the mean amplitude for 4:1 ellipse with the R1 method, $N=128$, $p=0$, $\Delta t=0.16362$, $\varepsilon=0.01$.

logarithm of the mean amplitude of the wave (4:1 ellipse), where the solid line represents the analytical solution and the circles are the numerical solution.

5.6. *Proper Configuration for Ellipse*

From the results shown above, we see the case $p=\frac{1}{3}$ has a special advantage, especially for the unperturbed ellipse. The reason is that when $p=\frac{1}{3}$, the node configuration is stationary relative to the ellipse. This can be seen from (3.5), since the relation between μ and η is linear when $p=\frac{1}{3}$, and the points are equally spaced in η. The particle path relative to the *rotating* ellipse is given by

$$x' = a\cos(\Omega t+\eta_0),$$
$$y' = b\sin(\Omega t+\eta_0), \tag{5.18}$$

where η_0 is a parameter. At any time t, $(\Omega t+\eta_0)$ serves as the elliptical coordinate η. Thus, the difference of η for adjacent points is invariant with time. That is, if the

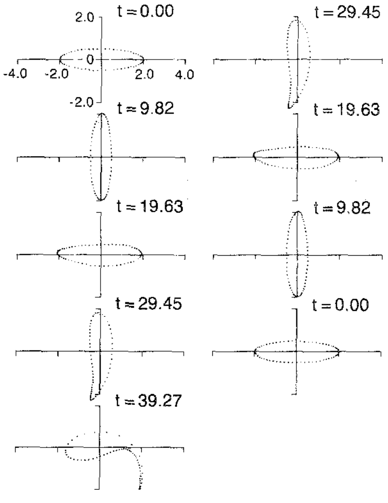

FIG. 4. The forward-and-return evolutions for a 4:1 ellipse with the R1 method, $N=64$, $p=\frac{1}{3}$, $\Delta t=0.16362$, $\varepsilon=0.01$.

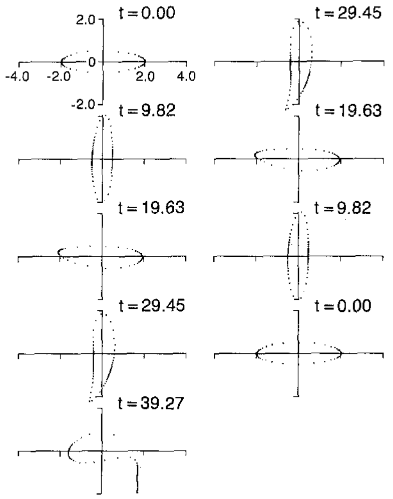

FIG. 5. The forward-and-return evolutions for a 4:1 ellipse with the R1 method, $N=64$, $p=1$, $\Delta t=0.16362$, $\varepsilon=0.01$.

points are equally spaced in η at the initial time, they will keep this configuration for a time in which an ellipse or near-ellipse represents the curve as seen in Fig. 4. In Figs. 4 and 5, we compare forward-and-return evolutions for a perturbed 4:1 ellipse with $p=\frac{1}{3}$ and $p=1$, respective. In the former case the node distribution is nearly invariant (e.g., for $t<19.63$). While in the latter case it is evident that the dense grouping of points moves around the ellipse. For example, at $\frac{1}{4}$ revolution, the high curvature region is poorly resolved, thus enhancing error growth.

6. 4-Fold Symmetric V-State–Initial Node Placement

We now examine the generality of the preceding result on stationarity of the nodal distributions by studying 4-fold symmetric V-state (4FSV). We obtain such a state with an aspect ratio 1.24418 ($\Omega=0.36620$) using iteration method of Wu *et al.* in [11] and approximate it by

$$R(\theta)=\sum_{n=0}^{15} a_n \cos(4n\theta). \tag{6.1}$$

We distribute the points at $t=0$ according to formula (3.1) as with the ellipse case. We compute the solution for one revolution ($T=17.16$) using R1 with $N=128$ and $\Delta t=0.143$ and the fourth-order Runge–Kutta method.

Table V gives the results, including:

(1) relative error of "global angular velocity" obtained from the major axis of the ellipse. To obtain this major axis, we first rotate the coordinate system by the

TABLE V

Comparison of 4FSV, R1, 128 Points, One Revolution

$12p$	$\Delta\Omega_e/\Omega$	l_2 error in R	$\Delta A/A$
0	0.903(−3)	0.641(−3)	0.149(−6)
1	0.718(−3)	0.323(−3)	−0.265(−6)
2	0.623(−3)	0.101(−3)	0.468(−6)
3	0.597(−3)	0.229(−3)	0.105(−5)
4	0.637(−3)	0.498(−3)	−0.187(−7)
5	0.753(−3)	0.841(−3)	−0.554(−5)
6	0.964(−3)	0.129(−2)	−0.215(−4)
7	0.130(−2)	0.188(−2)	−0.607(−4)
8	0.178(−2)	0.260(−2)	−0.147(−3)
9	0.243(−2)	0.342(−2)	−0.318(−3)
10	0.323(−2)	0.430(−2)	−0.610(−3)
11	0.417(−2)	0.521(−2)	−0.104(−2)
12	0.521(−2)	0.610(−2)	−0.157(−2)

amount Ωt and compute $R=R(\theta)$ in this system. We then calculate the Fourier expansion of $R(\theta)$ and change the rotation angle, and thus the coordinate system, by suppressing the $\sin(4\theta)$ term in the new system.

(2) l_2 error in R, which is the polar distance of the curve.

(3) relative error in the area.

From Table V we see the minimum error occurs about $\frac{1}{6}\leqslant p_*\leqslant\frac{1}{4}$, close to the ellipse case. Looking at the plots, we found that $p=\frac{1}{6}$ corresponds approximately to a stationary configuration. Figure 6 shows the case with $p=\frac{1}{6}$ for one revolution.

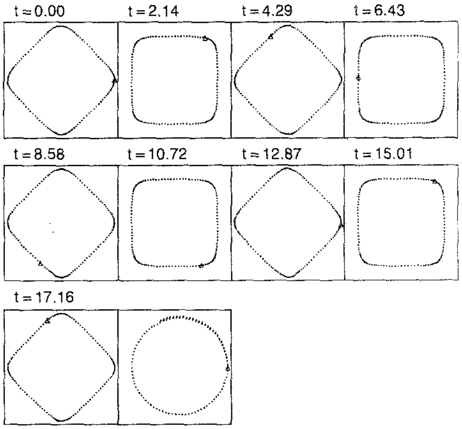

FIG. 6. The rotation of a 4FSV with aspect ratio 1.24418 ($\Omega=0.36620$), the R1 method, $N=128$, $p=\frac{1}{6}$, $\Delta t=0.143$. The triangle symbol denotes the position of the point initially located at $\phi=0$. The last panel shows the trajectory of the point initially at $\phi=0$ for $0\leqslant t\leqslant 17.16$.

The triangle symbol denotes the position initially located at $\phi = 0$ (ϕ is the polar angle) and the last plot shows its locus, which is a circle. In one revolution of the curve this point has traveled about $8\pi/3$ radians.

6. Conclusion

We have made a systematic study of spatial discretization errors of several contour dynamical algorithms for the Euler equations and established their order accuracy. As benchmarks, for these calculations we have chosen mainly the 2:1 and 4:1 Kirchhoff elliptical vortices and such vortices subject to small (linear) perturbations. We have carefully investigated the effect on errors of *initial* placement of [redacted] verified for the $m = 4$ V-state, where $\frac{1}{6} \leqslant p \leqslant \frac{1}{3}$. We believe this idea will be useful for adaptive (time-dependent) node-placement algorithms and we will explore these in a future paper.

References

1. H. Berk and K. V. Roberts, *Methods Comput. Phys.* **9**, 87 (1970).
2. N. J. Zabusky, M. H. Huges, and K. V. Roberts, *J. Comput. Phys.* **30**, 96 (1979).
3. M. V. Melander, E. A. Overman II, and N. J. Zabusky, *Appl. Numer. Math.* **3**, 59 (1987).
4. L. Rosenhead, *Proc. R. Soc. London A* **134**, 170 (1931).
5. R. Krasny, *J. Comput. Phys.* **65**, 292 (1986).
6. D. I. Pullin and W. R. C. Phillips, *J. Fluid Mech.* **104**, 45 (1981).
7. N. J. Zabusky and E. A. Overman II, *J. Comput. Phys.* **52**, 351 (1983).
8. E. A. Overman II, N. J. Zabusky, and S. L. Ossakow, *Phys. Fluids* **26**, 1139 (1983).
9. H. Aref and G. Tryggvason, preprint, 1986.
10. D. G. Dritschel, *J. Comput. Phys.* **77**, (1988) in press.
11. H. M. Wu, E. A. Overman II, and N. J. Zabusky, *J. Comput. Phys.* **53**, 42 (1984).
12. E. A. Overman and N. J. Zabusky, *J. Fluid Mech.* **125**, 187 (1982).
13. N. J. Zabusky and E. A. Overman II, *J. Comput. Phys.* **52**, 351 (1983).
14. J. M. Sanz-Serna and I. Christie, *J. Comput. Phys.* **67**, 348 (1986).
15. M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions* (Dover, New York, 1965).
16. H. Lamb, *Hydrodynamics* (Dover, New York, 1932), 6th ed., Section 159, p. 232.
17. A. E. H. Love, *Proc. London Math. Soc.* **35**, No. 1, **18** (1893).