JOURNAL OF COMPUTATIONAL PHYSICS **133,** 222–234 (1997)
ARTICLE NO. CP975668

Contour Dynamics with Symplectic Time Integration

P. W. C. Vosbeek*,† and R. M. M. Mattheij*

**Department of Mathematics and Computing Science and †Department of Physics, Eindhoven University of Technology, Eindhoven, 5600 MB, The Netherlands*
E-mail: wsanpv@win.tue.nl; mattheij@win.tue.nl

Received December 18, 1995; revised January 29, 1997

In this paper we consider the time evolution of vortices simulated by the method of contour dynamics. Special attention is being paid to the Hamiltonian character of the governing equations and in particular to the conservational properties of numerical time integration for them. We assess symplectic and nonsymplectic schemes. For the former methods, we give an implementation which is both efficient and yet effectively explicit. A number of numerical examples sustain the analysis and demonstrate the usefulness of the approach. © 1997 Academic Press

1. INTRODUCTION

In this paper we will discuss some aspects of the *contour dynamics* method, in particular the time integration. This well-known method is a useful tool for simulating vortices in two-dimensional flows of an incompressible, inviscid fluid. The method and many improvements thereof, has been brought to full growth by the pioneering work of Dritschel [1, 2]. Contour dynamics is based on the idea that the evolution of a patch of uniform vorticity is fully determined by the evolution of its boundary contour. The method is not limited to just one region of uniform vorticity; indeed, several contours can be nested in order to obtain an approximation of a patch of distributed vorticity (see [1, 2, 12]).

Two-dimensional flows of an incompressible, inviscid fluid can be described by Euler's equation, which expresses balance of linear momentum, and the continuity equation, which expresses conservation of mass. Regarding the latter conservation law we remark that, for an incompressible fluid, the velocity field is divergence free and thus, a stream function ψ can be introduced in the usual way

$$\begin{aligned} \dot{x}(t) &= u(x, y, t) = \frac{\partial \psi(x, y, t)}{\partial y}, \\ \dot{y}(t) &= v(x, y, t) = -\frac{\partial \psi(x, y, t)}{\partial x}, \end{aligned} \tag{1}$$

Here u is the component of the velocity field in the x-direction and v the component in the y-direction. From this we can see that we are dealing with a *Hamiltonian system* with *Hamiltonian* $-\psi$. A very important property of such a system is the concept of preservation of area (see [10]) which in our case is equivalent to conservation of mass. Operators which have this property, are called *symplectic*. The solution operator of a Hamiltonian system is thus a symplectic operator. Since we like to solve this Hamiltonian system numerically, it is important, especially for long-time calculations, to preserve the area. This is possible if a so-called *symplectic integration scheme* is used (see [10]).

In contour dynamics, we are in fact dealing with two types of discretizations, viz. one in space and one in time. In this paper, we will show that the spatially discretized problem is also a Hamiltonian system, and therefore a symplectic time integration scheme is to be preferred to ordinary integration methods. Furthermore we shall outline how such a scheme can be applied to the contour dynamics method and we show some results.

2. THE GOVERNING EQUATIONS

The vorticity vector $\boldsymbol{\omega}$ is defined as the curl of the velocity field $\mathbf{u}$. Since we consider a two-dimensional flow in the (x, y)-plane, this vorticity vector points in a direction perpendicular to the (x, y)-plane; so we can write

$$\boldsymbol{\omega} = \omega \mathbf{e}_z.$$

By defining the stream function as in (1) and taking the curl of the linear momentum equation, we obtain the vorticity equation

$$\frac{\partial \omega}{\partial t} + \frac{\partial \psi}{\partial y}\frac{\partial \omega}{\partial x} - \frac{\partial \psi}{\partial x}\frac{\partial \omega}{\partial y} = 0, \tag{2}$$

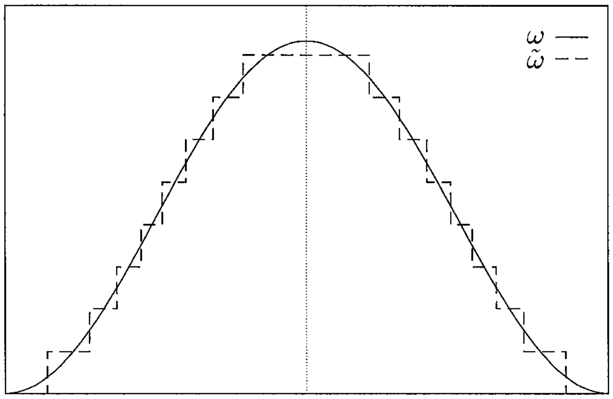

FIG. 1. A cross section of the continuous vorticity profile and the piecewise constant vorticity profile.

which expresses conservation of vorticity of a fluid particle. Furthermore, a relation between ω and ψ can be derived from their definitions

$$\nabla^2\psi = -\omega. \tag{3}$$

By solving (3) using Green's function, we find an expression for ψ

$$\psi(\mathbf{x}, t) = -\iint_{\mathbb{R}^2} G(\mathbf{x}; \mathbf{x}')\omega(\mathbf{x}', t)\, dx'\, dy', \quad t \geq 0,$$

where $\mathbf{x} := (x, y)^{\mathrm{T}}$ and $G(\mathbf{x}; \mathbf{x}') := (1/2\pi) \ln \|\mathbf{x} - \mathbf{x}'\|$ is Green's function in two dimensions. The norm $\|\cdot\|$ is defined by $\|\mathbf{x}\| := \sqrt{x^2 + y^2}$ for each $\mathbf{x} \in \mathbb{R}^2$.

The initial continuous vorticity distribution ω is now replaced by a piecewise constant distribution $\tilde{\omega}$ like in Fig. 1. This is not a severe limitation of the method, as shown by Legras and Dritschel in [7], where comparisons between piecewise constant and continuous distributions are presented. Conservation of vorticity of a fluid particle now ensures that the distribution remains piecewise constant throughout time. For $\tilde{\psi}$ we then find

$$\tilde{\psi}(\mathbf{x}, t) := -\sum_{m=1}^{M} \omega_m \iint_{G_m(t)} G(\mathbf{x}; \mathbf{x}')\, dx'\, dy', \quad t \geq 0,$$

where the $G_m(t)$ are the regions of uniform vorticity ω_m at time t (see, e.g., Fig. 2).

By applying Stokes' theorem for a scalar field, we can derive an expression for the velocity field,

$$\tilde{\mathbf{u}}(\mathbf{x}, t) = -\sum_{m=1}^{M} \omega_m \oint_{C_m(t)} G(\mathbf{x}; \mathbf{x}')\, d\mathbf{x}', \tag{4}$$

where ω_m is the jump of vorticity when crossing the contour $C_m(t)$ outward. Note that this system of equations is also Hamiltonian, with Hamiltonian $-\tilde{\psi}$.

3. SPATIAL DISCRETIZATION

3.1. *Discretization of the Contours*

From (4), we see that the velocity $\tilde{\mathbf{u}}$ at any point of the two-dimensional plane is determined by a sum of contour integrals. To calculate these contour integrals numerically, we discretize each contour C_m into a finite but adjustable number of nodes N (N, of course, depends on m and also on time t). Between two adjacent nodes, the contours are approximated by so-called *elements*. These elements can, e.g., be linear (in this case, the two adjacent nodes are simply connected by a straight line segment), or quadratic, cubic, etc. The parameterisation $\mathbf{x}_n(\xi)$ of an element e_n with nodes $\mathbf{x}_n$ and $\mathbf{x}_{n+1}$, is chosen such that $\mathbf{x}_n(-1) = \mathbf{x}_n$, $\mathbf{x}_n(1) = \mathbf{x}_{n+1}$ and $\mathbf{x}_N(1) = \mathbf{x}_1(-1)$. In the case of linear elements, this parameterisation is given by

$$\mathbf{x}_n(\xi) = \tfrac{1}{2}(1 - \xi)\mathbf{x}_n + \tfrac{1}{2}(1 + \xi)\mathbf{x}_{n+1}. \tag{5}$$

The interpolated version of contour C_m will be called $\hat{C}_m$.

The velocity $\hat{\mathbf{u}} := (\hat{u}, \hat{v})^{\mathrm{T}}$ at a point $\mathbf{x}$ anywhere in the flow field of the spatially discretized problem is then given by

$$\begin{aligned}\hat{\mathbf{u}}(\mathbf{x}, t) &= -\sum_{m=1}^{M} \frac{\omega_m}{2\pi} \oint_{\hat{C}_m(t)} \ln \|\mathbf{x} - \mathbf{x}'\|\, d\mathbf{x}' \\ &= -\sum_{m=1}^{M} \frac{\omega_m}{2\pi} \sum_{n=1}^{N} \int_{-1}^{1} \ln \|\mathbf{x} - \mathbf{x}_n(\xi)\|\, \dot{\mathbf{x}}_n(\xi)\, d\xi.\end{aligned} \tag{6}$$

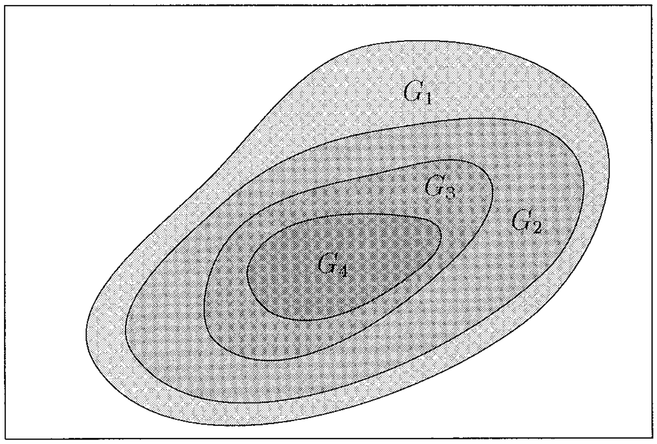

FIG. 2. An arbitrary patch of piecewise constant vorticity distribution.

Note that here the overdot denotes a ξ-derivation. In all numerical examples, we shall use linear elements. Of course, higher order interpolation gives better results with respect to contour shapes (see [2]); however, since we are mainly interested in the behaviour of the area as a result of the time integration, linear elements are adequate here. The integrals in (6) along the elements can be determined using Gaussian quadrature. Only when $\mathbf{x}$ is equal to (or lying close to) one of the element nodes, an analytical solution of this integral is used; this is needed because the logarithm is (almost) singular in that case.

We will now prove the following property:

PROPERTY 3.1. *The velocity field $\hat{\mathbf{u}}$ of the spatially discretized problem is divergence-free.*

Proof. The partial derivatives of the velocity field are given by

$$\frac{\partial \hat{u}}{\partial x} = -\sum_{m=1}^{M} \frac{\omega_m}{2\pi} \sum_{n=1}^{N} \int_{-1}^{1} \frac{(x - x_n(\xi))\dot{x}_n(\xi)}{\|\mathbf{x} - \mathbf{x}_n(\xi)\|^2}\, d\xi,$$

$$\frac{\partial \hat{v}}{\partial y} = -\sum_{m=1}^{M} \frac{\omega_m}{2\pi} \sum_{n=1}^{N} \int_{-1}^{1} \frac{(y - y_n(\xi))\dot{y}_n(\xi)}{\|\mathbf{x} - \mathbf{x}_n(\xi)\|^2}\, d\xi.$$

So, for the divergence we find

$$\begin{aligned}(\nabla, \hat{\mathbf{u}}) = \frac{\partial \hat{u}}{\partial x} + \frac{\partial \hat{v}}{\partial y} &= -\sum_{m=1}^{M} \frac{\omega_m}{2\pi} \sum_{n=1}^{N} \int_{-1}^{1} \\ &\quad \frac{((x - x_n(\xi))\dot{x}_n(\xi) + (y - y_n(\xi))\dot{y}_n(\xi))}{\|\mathbf{x} - \mathbf{x}_n(\xi)\|^2}\, d\xi \\ &= \sum_{m=1}^{M} \frac{\omega_m}{2\pi} \sum_{n=1}^{N} \int_{-1}^{1} d \ln \|\mathbf{x} - \mathbf{x}_n(\xi)\| \\ &= \sum_{m=1}^{M} \frac{\omega_m}{2\pi} \sum_{n=1}^{N} [\ln \|\mathbf{x} - \mathbf{x}_n(1)\| - \ln \|\mathbf{x} - \mathbf{x}_n(-1)\|] \\ &= 0,\end{aligned}$$

since $\mathbf{x}_n(1) = \mathbf{x}_{n+1}(-1)$ for all n, $1 \leq n < N$, and $\mathbf{x}_N(1) = \mathbf{x}_1(-1)$. This even holds when $\mathbf{x}$ is equal to one of the nodes on the contour. ∎

From this theorem it follows that the spatially discretized problem also has a Hamiltonian; it is given by

$$\begin{aligned}\hat{\psi}(\mathbf{x}) = \sum_{m=1}^{M} \frac{\omega_m}{4\pi} \sum_{n=1}^{N} \int_{-1}^{1} [f(\mathbf{x}, \mathbf{x}_n(\xi))\dot{x}_n(\xi) \\ + g(\mathbf{x}, \mathbf{x}_n(\xi))\dot{y}_n(\xi)]\, d\xi,\end{aligned} \tag{7}$$

where

$$\begin{aligned}f(\mathbf{x}, \mathbf{x}_n(\xi)) &:= (x - x_n(\xi)) \arctan\left(\frac{y - y_n(\xi)}{x - x_n(\xi)}\right) \\ &\quad - (y - y_n(\xi)) \ln \|\mathbf{x} - \mathbf{x}_n(\xi)\|, \\ g(\mathbf{x}, \mathbf{x}_n(\xi)) &:= -(y - y_n(\xi)) \arctan\left(\frac{x - x_n(\xi)}{y - y_n(\xi)}\right) \\ &\quad + (x - x_n(\xi)) \ln \|\mathbf{x} - \mathbf{x}_n(\xi)\|.\end{aligned}$$

If we use linear elements for the interpolation of the contours and use h_n to denote the length of element e_n, we find the following property for the discretization error.

PROPERTY 3.2. *If $h_{\max}$ is defined as the maximum of all h_n, then*

$$\|\tilde{\mathbf{u}} - \hat{\mathbf{u}}\| = \left\| \sum_{m=1}^{M} \frac{\omega}{2\pi} \oint_{\hat{C}_m - C_m} \ln \|\mathbf{x} - \mathbf{x}'\|\, d\mathbf{x}' \right\| = \mathcal{O}(h_{\max}^2).$$

Here, the integration over the difference of the two contours $\hat{C}_m$ and C_m is defined as the difference of the integration over $\hat{C}_m$ and over C_m.

Proof. For the sake of simplicity, only the situation of one contour C is considered. However, the more general case (with more than one contour) can be treated in the same way. We assume that the nodes $\mathbf{x}_n$ of contour $\hat{C}$ are lying on the exact contour C. Then we find

$$\begin{aligned}\tilde{\mathbf{u}} - \hat{\mathbf{u}} &= \frac{\omega}{2\pi} \int_{\hat{C} - C} \ln \|\mathbf{x} - \mathbf{x}'\|\, d\mathbf{x}' \\ &= \frac{\omega}{2\pi} \sum_{n=1}^{N} \oint_{e_n - C_n} \ln \|\mathbf{x} - \mathbf{x}'\|\, d\mathbf{x}',\end{aligned}$$

where e_n is the straight line segment (with length h_n) connecting two adjacent nodes $\mathbf{x}_n$ and $\mathbf{x}_{n+1}$ and C_n is the part of contour C that connects these points also (see Fig. 3).

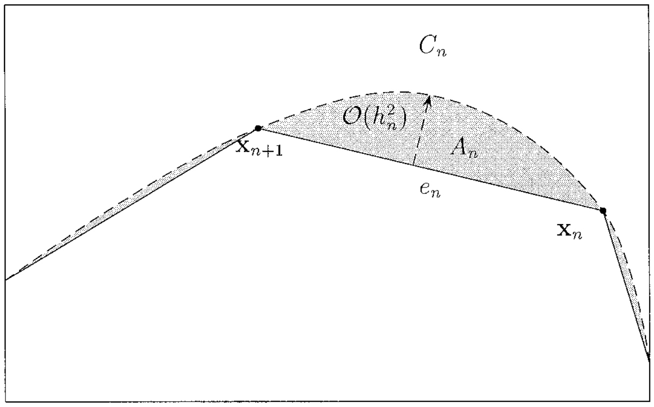

FIG. 3. The region A_n enclosed by e_n and C_n.

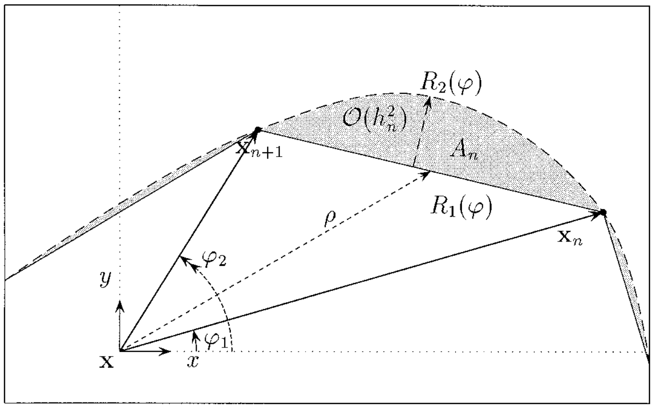

FIG. 4. The situation of Fig. 3 in polar coordinates.

We now consider the contribution vector $\boldsymbol{\delta}_n$ of one contour $e_n - C_n$ (i.e., the "local geometrical error") to the discretization error. We use Stokes' theorem for a vector field to obtain a surface integral over A_n (see Fig. 3) from the contour integral and find

$$\begin{aligned}\boldsymbol{\delta}_n &:= \oint_{e_n - C_n} \ln \|\mathbf{x} - \mathbf{x}'\|\, d\mathbf{x}' \\ &= \iint_{A_n} \mathbf{e}_z \times \nabla_x \ln \|\mathbf{x} - \mathbf{x}'\|\, dx'\, dy' \\ &= \iint_{A_n} \left(\frac{y - y'}{\|\mathbf{x} - \mathbf{x}'\|^2} \mathbf{e}_x - \frac{x - x'}{\|\mathbf{x} - \mathbf{x}'\|^2} \mathbf{e}_y \right) dx'\, dy'.\end{aligned}$$

We now introduce polar coordinates in the following way:

$$\begin{aligned} x - x' &:= r\cos(\varphi), \\ y - y' &:= r\sin(\varphi).\end{aligned}$$

Then, the surface integral becomes

$$\boldsymbol{\delta}_n = \int_{\varphi_1}^{\varphi_2} \int_{R_1(\varphi)}^{R_2(\varphi)} (\sin(\varphi)\mathbf{e}_x - \cos(\varphi)\mathbf{e}_y)\, d\varphi\, dr.$$

Here φ_1, φ_2, $R_1(\varphi)$, and $R_2(\varphi)$ are as in Fig. 4. For the x-component we thus find

$$\begin{aligned} |(\boldsymbol{\delta}_n, \mathbf{e}_x)| &\le \int_{\varphi_1}^{\varphi_2} |R_2(\varphi) - R_1(\varphi)|\, |\sin(\varphi)|\, d\varphi \\ &\le (\varphi_2 - \varphi_1) \max_{\varphi_1 \le \varphi \le \varphi_2} |R_2(\varphi) - R_1(\varphi)|, \end{aligned} \tag{*}$$

and similarly for the y-component

$$\begin{aligned} |(\boldsymbol{\delta}_n, \mathbf{e}_y)| &\le \int_{\varphi_1}^{\varphi_2} |R_2(\varphi) - R_1(\varphi)|\, |\cos(\varphi)|\, d\varphi \\ &\le (\varphi_2 - \varphi_1) \max_{\varphi_1 \le \varphi \le \varphi_2} |R_2(\varphi) - R_1(\varphi)|. \end{aligned} \tag{**}$$

Because of the linear interpolation, we have $|R_2(\varphi) - R_1(\varphi)| \le Ch_n^2$ for $\varphi_1 \le \varphi \le \varphi_2$.

Furthermore, we can derive an expression for the angle $\vartheta := \varphi_2 - \varphi_1$. If we define ρ by $\rho := \|\mathbf{x} - \frac{1}{2}(\mathbf{x}_n + \mathbf{x}_{n+1})\|$, then

$$\begin{aligned} \|\mathbf{x} - \mathbf{x}_n\|^2 &= \rho^2 + h_n\rho\cos(\vartheta_1) + \tfrac{1}{4}h_n^2, \\ \|\mathbf{x} - \mathbf{x}_{n+1}\|^2 &= \rho^2 - h_n\rho\cos(\vartheta_1) + \tfrac{1}{4}h_n^2, \end{aligned}$$

where θ_1 is the angle between $\mathbf{x} - \frac{1}{2}(\mathbf{x}_n + \mathbf{x}_{n+1})$ and $\frac{1}{2}(\mathbf{x}_{n+1} - \mathbf{x}_n)$. For the inner product $(\mathbf{x} - \mathbf{x}_n, \mathbf{x} - \mathbf{x}_{n+1})$ we simply have

$$(\mathbf{x} - \mathbf{x}_n, \mathbf{x} - \mathbf{x}_{n+1}) = \rho^2 - \tfrac{1}{4}h_n^2.$$

Since

$$\cos(\vartheta) = \frac{(\mathbf{x} - \mathbf{x}_n, \mathbf{x} - \mathbf{x}_{n+1})}{\|\mathbf{x} - \mathbf{x}_n\|\, \|\mathbf{x} - \mathbf{x}_{n+1}\|},$$

we find after some calculation that

$$\begin{aligned} \sin^2(\vartheta) &= 1 - \cos^2(\vartheta) \\ &= \frac{(h_n/\rho)^2 \sin^2(\vartheta_1)}{1 - \frac{1}{2}(h_n/\rho)^2\cos(2\vartheta_1) + \frac{1}{16}(h_n/\rho)^4}. \end{aligned}$$

For $\rho > h_n$, we find from this that $\vartheta = \mathcal{O}(h_n/\rho)$.

Using this result in (*) and (**), we have

$$\|\boldsymbol{\delta}_n\| = \begin{cases} \mathcal{O}(h_n^3/\rho), & \text{if } \rho > h_n, \\ \mathcal{O}(h_n^2), & \text{if } \rho \le h_n. \end{cases}$$

Now, by combining the contributions from all regions A_n and by assuming that only few elements e_n are lying close (i.e., closer then h_n) to the point $\mathbf{x}$, we may conclude that the overall discretization error is $\mathcal{O}(h_{\max}^2)$, where $h_{\max}$ is the maximum element length on contour $\hat{C}$. This completes the proof. ▌

Note that this type of discretization error is not necessarily made every time step during the calculations. The first time this error is made when the initial contour grid is calculated. After that, such spatial errors only occur when nodes are added or removed; if this is done properly the errors might be smaller.

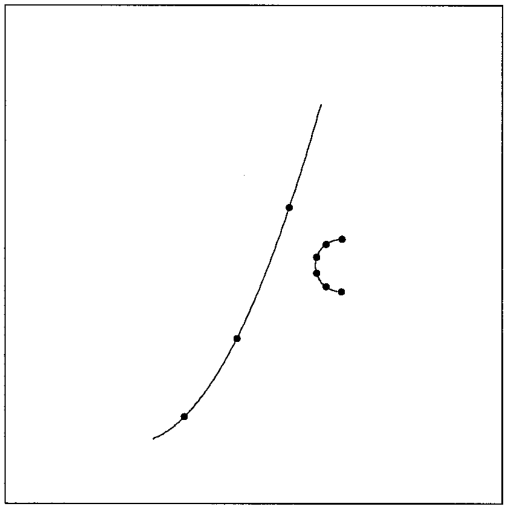

FIG. 5. A small-scale feature encountering a large-scale one (from [1, 2]).

3.2. *Node Distribution*

Since, in general, the shape of the contours becomes increasingly complex when time proceeds, the number of nodes initially placed on the contours, will not be enough to approximate the contours nicely at later time points. Therefore, the number of nodes on a contour may change during the calculations.

Several situations may occur where one has to add nodes to, or remove nodes from, a contour. In general problems may arise when a small scale feature with a high density of nodes, encounters a larger scale feature with a lower node density (see Fig. 5). In this case, the two parts of the contours may intersect, unless nodes are added properly to the large scale feature. To prevent this intersection of two (parts of) contours, the following technique is implemented: we say a node $\mathbf{x}_i$ is lying *opposite to* an element e_n with nodes $\mathbf{x}_n$ and $\mathbf{x}_{n+1}$, if the line through $\mathbf{x}_i$, perpendicular to the line l through $\mathbf{x}_n$ and $\mathbf{x}_{n+1}$ (see Fig. 6) intersects l in between $\mathbf{x}_n$ and $\mathbf{x}_{x+1}$. When no nodes are added between $\mathbf{x}_n$ and $\mathbf{x}_{n+1}$ and the local curvature at $\mathbf{x}_i$ is higher than the curvature at $\mathbf{x}_n$ or $\mathbf{x}_{n+1}$, such a point $\mathbf{x}_i$ may cause trouble. This can be avoided by properly adding a node between $\mathbf{x}_n$ and $\mathbf{x}_{n+1}$ each time the *distance* between the node and the element is becoming smaller than a given critical value. This distance is defined through the length of the vector $\mathbf{v}$ (see Fig. 6). It is also possible that oppositely situated points do not cause trouble and are even such that the local curvature is low enough to allow some nodes to be removed (for example, on a filament); here, the local curvature is found from differentiation of a quadratic polynomial through three consecutive points. Alternatively, nodes may have to be added at places where the local curvature is high in order to approximate the contour well enough. This can be formulated as follows: if h_n is the length of element e_n, and $\kappa(\mathbf{x}_n)$ is the local curvature at node $\mathbf{x}_n$, then a node has to be added between $\mathbf{x}_n$ and $\mathbf{x}_{n+1}$ if

$$\tfrac{1}{2}(|\kappa(\mathbf{x}_n)| + |\kappa(\mathbf{x}_{n+1})|)h_n > \delta_1,$$

and node $\mathbf{x}_n$ has to be removed if

$$\tfrac{1}{2}(|\kappa(\mathbf{x}_n)| + |\kappa(\mathbf{x}_{n+1})|)h_n < \delta_2,$$

where δ_1 and δ_2 are given and $\delta_1 > \delta_2$. Furthermore, we require the element length to become not smaller than a minimum $h_{\min}$ and not larger than a maximum $h_{\max}$. The former requirement is made because we want to control the number of nodes on a contour in view of the CPU-time. The latter requirement is made to prevent the element length to become too large at filaments (where the curvature is very low at some places). The last condition the node distribution is required to satisfy, is that of *quasi-uniformity*. This can be formulated as

$$\frac{h_{n-1}}{K} \leq h_n \leq K h_{n-1},$$

where K is a constant sufficiently larger than 1, which ensures that the element length of two neighboring elements is not changing too much. This is essential, since the local curvature, which depends on three successive nodes, cannot be calculated very accurately otherwise.

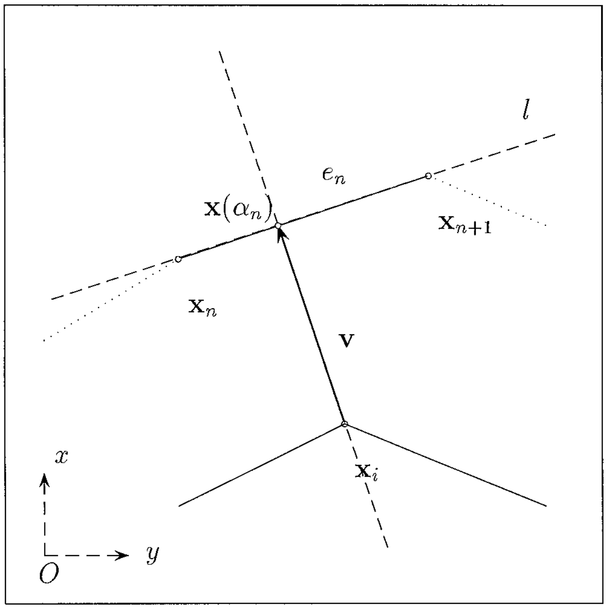

FIG. 6. Node $\mathbf{x}_i$ is lying opposite to e_n.

All together, we have four criteria for adding nodes and three for removing. It is obvious that the actual removal of nodes is very simple. For adding a node, however, one has to decide where the new node has to be placed. To this end, we fit a quadratic polynomial through three successive nodes and place the new node on this polynomial. The velocity at the new node is also determined by quadratic interpolation. Here, the quasi-uniformity of the nodes also is of importance for accuracy.

In the numerical examples in this paper, no surgery (see [1]) is applied. This is because we focus on the time integration part of the method. Surgery would, of course, improve the method, especially in case of complex long time calculations.

4. TIME INTEGRATION

As we can see from (6), the velocity field $\hat{\mathbf{u}}$ depends on the position of every node on every contour. Let $\mathbf{X}(t)$ be the vector of x- and y-coordinates of all nodes at a certain time t,

$$\mathbf{X}(t) := (x_1(t), x_2(t), ..., x_N(t), y_1(t), y_2(t), ..., y_N(t))^{\mathrm{T}}.$$

Denote the velocity in the x-direction at node $\mathbf{x}_n$ by $\hat{u}_n(\mathbf{X})$ ($:= \hat{u}(\mathbf{x}_n)$) and in the y-direction by $\hat{v}_n(\mathbf{X})$ ($:= \hat{v}(\mathbf{x}_n)$). Furthermore, let $\mathbf{U}$ be the vector of velocities in x- and y-directions:

$$\mathbf{U}(\mathbf{X}) := (\hat{u}_1(\mathbf{X}), \hat{u}_2(\mathbf{X}), ..., \hat{u}_N(\mathbf{X}), \hat{v}_1(\mathbf{X}), \hat{v}_2(\mathbf{X}), ..., \hat{v}_N(\mathbf{X}))^{\mathrm{T}}.$$

For the time evolution of the contours, we now have to solve the following initial value problem:

$$\begin{aligned} \dot{\mathbf{X}} &= \mathbf{U}(\mathbf{X}), \quad t > 0, \\ \mathbf{X}(0) &= \mathbf{X}_0. \end{aligned} \tag{8}$$

4.1. *Symplectic and Nonsymplectic Runge–Kutta Methods*

As we have pointed out in Section 3.1, the spatially discretized problem, of which we want to know the time evolution, is Hamiltonian. This means that the solution operator is symplectic (see [10]). In a numerical time integration, this solution operator is replaced by an approximate one. If we wish the latter to retain the Hamiltonian character of the former, we should insist that the approximate solution operator is symplectic as well. However, most standard numerical integrators replace the solution operator by a nonsymplectic mapping. This is illustrated by the following simple example. Consider the time evolution of a circular vortex patch (initial radius equal to $r(0)$) of uniform vorticity ω. The velocity field inside the patch and on its boundary, is clearly given by (see [6])

$$\left.\begin{aligned} u(\mathbf{x}) &= \dot{x} = -\frac{\omega}{2} y, \\ v(\mathbf{x}) &= \dot{y} = \frac{\omega}{2} x, \end{aligned}\right\} \quad t > 0, \quad \|\mathbf{x}\| \leq 1, \tag{9}$$

By applying the Euler forward scheme to (9), we find

$$\begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = \begin{pmatrix} 1 & -\dfrac{\omega\Delta t}{2} \\ \dfrac{\omega\Delta t}{2} & 1 \end{pmatrix} \begin{pmatrix} x(t-\Delta t) \\ y(t-\Delta t) \end{pmatrix},$$

where Δt is the time step. For the length $r(t)$ of the vector $\mathbf{x}$ after $t/\Delta t$ time steps, we thus find

$$\begin{aligned} r^2(t) &= x^2(t) + y^2(t) \\ &= \left(1 + \left(\frac{\omega\Delta t}{2}\right)^2\right)(x^2(t-\Delta t) + y^2(t-\Delta t)) \\ &= \left(1 + \left(\frac{\omega\Delta t}{2}\right)^2\right)^{t/\Delta t} r^2(0) \\ &= r^2(0)\exp\left(t\Delta t\left(\frac{\omega}{2}\right)^2 + \mathcal{O}\left(t\Delta t^3\left(\frac{\omega}{2}\right)^4\right)\right). \end{aligned}$$

The radius of the patch thus grows exponentially with time (see Fig. 7, where we used $\omega = 2\pi$ and $\Delta t = 0.05$). Since

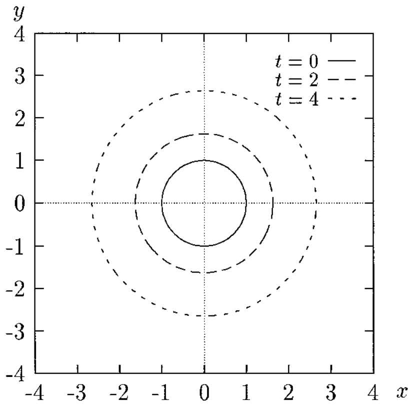

FIG. 7. The evolution of a circular patch (initial radius equal to 1) of uniform vorticity ($\omega = 2\pi$) using the Euler forward scheme with $\Delta t = 0.05$.

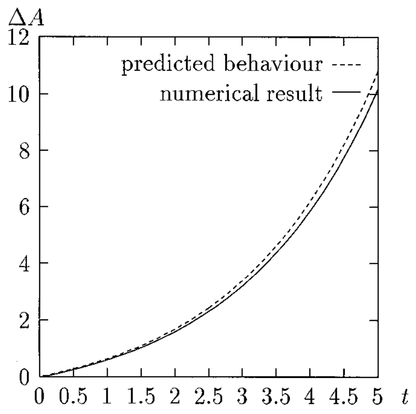

FIG. 8. The predicted behaviour of ΔA and the numerically obtained ΔA as a function of time t. The area of the patch is calculated each time step using a contour integral expression.

the patch remains circular for all time, we find for the area $A(t)$ of the patch after $t/\Delta t$ time steps

$$\begin{aligned} A(t) &= \pi r^2(t) \\ &= \pi r^2(0) \exp\left(t\Delta t\left(\frac{\omega}{2}\right)^2 + \mathcal{O}\left(t\Delta t^3\left(\frac{\omega}{2}\right)^4\right)\right). \end{aligned} \tag{10}$$

The interesting feature here is that

$$A(t) \sim A(0)\exp(t\alpha^2\Delta t), \tag{11}$$

where $\alpha = \omega/2$; i.e., it becomes unbounded as $t \to \infty$, although it can be kept close to $A(0)$ on any finite interval by choosing Δt small enough. In Fig. 8 we have plotted $\Delta A := |A(t) - A(0)|/A(0)$ versus time t (where the area $A(t)$ was calculated at each time step by a contour integral expression) and also the behaviour of ΔA (predicted by (11)). Clearly, (11) predicts the behaviour of the area very well. Furthermore, it may be clear that the Euler forward scheme is not symplectic and one may be forced to take many steps to stay close to conservation. In fact, this applies to any explicit method. Note that if we would apply the Euler backward scheme to this problem (although this would hardly be feasible in practice, since implementation would require the solution of a large linear system), we would find shrinking patches instead. For higher order (explicit) RK-schemes, a similar result applies, albeit with a more moderate growth. In particular, for the classical explicit fourth-order RK-method (see [3, 4]), one finds by straightforward expansion

$$A(t) \sim A(0)\exp(-t\alpha^6\Delta t^5/72). \tag{12}$$

In Fig. 9 we have plotted both the predicted and the numerically obtained values of ΔA as a function of time for the same problem as before (circular vortex patch). Here we omitted taking the absolute value of $(A(t) - A(0))/A(0)$ in order to show the decrease of the area. Again, the results agree remarkably well with the theory. Although the variation of the area is much smaller in this case, compared to the Euler forward scheme, it still can be of significant importance for large time intervals or problems with larger values of α.

Next, we will pay attention to a second-order symplectic *Runge–Kutta* (RK) scheme, the *midpoint rule.*

4.2. *Implementation of the Symplectic Midpoint Rule*

Symplectic RK-methods are implicit, in general (see [10]). Therefore, the implementation of a symplectic method can be rather complex. If the problem were stiff, Newton-like iteration methods would be needed to solve the linear systems involved. In our case, however, these systems normally are not stiff. Indeed, in order to follow the contour properly, we have to resolve regions which are moving relatively fast at all steps (probably at different places during the evolution); i.e., the "smaller" time scales dictate the step size at any time. Therefore, we can use a *predictor–corrector scheme* ($\mathbf{P}(\mathbf{EC})^I\mathbf{E}$) for the implementation of the symplectic scheme. This means that the method will be explicit after all, but conservation can be achieved by choosing I large enough. For this scheme, we shall choose the midpoint rule (which is only second order), but a similar implementation can be used for higher order symplectic methods.

Denote the vector of approximate x- and y-coordinates of the positions of the nodes after k time steps Δt by $\mathbf{X}_k$. Furthermore, denote the vector of approximate velocities

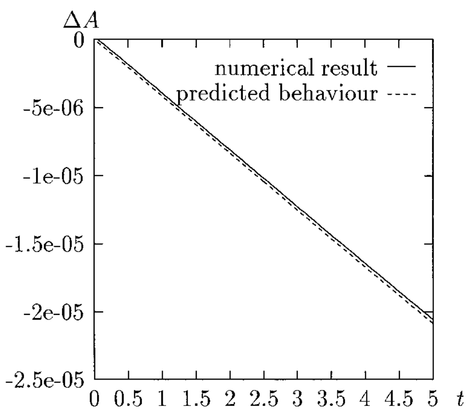

FIG. 9. Similar to Fig. 8 but for RK4 and $\Delta A := (A(t) - A(0))/A(0)$ (not the absolute value). Note the decrease of the area in this case.

in x- and y-direction by $\mathbf{U}(\mathbf{X}_k)$. Then the midpoint rule can be represented as

$$\mathbf{X}_{k+1} = \mathbf{X}_k + \Delta t \mathbf{U}\left(\frac{\mathbf{X}_k + \mathbf{X}_{k+1}}{2}\right),$$

or

$$\frac{\mathbf{X}_k + \mathbf{X}_{k+1}}{2} = \mathbf{X}_k + \frac{\Delta t}{2}\mathbf{U}\left(\frac{\mathbf{X}_k + \mathbf{X}_{k+1}}{2}\right).$$

Introducing vector

$$\overline{\mathbf{X}} := \frac{\mathbf{X}_k + \mathbf{X}_{k+1}}{2},$$

we apparently need to solve

$$\overline{\mathbf{X}} = \mathbf{X}_k + \frac{\Delta t}{2}\mathbf{U}(\overline{\mathbf{X}}) \tag{13}$$

for each time step. This is done by the following ($\mathbf{P(EC)}^I\mathbf{E}$) method:

$$\begin{aligned} &\textbf{P}\text{redict: } \overline{\mathbf{X}}^0 = \mathbf{X}_k + \frac{\Delta t}{2}\mathbf{U}(\mathbf{X}_k),\\ &\left.\begin{aligned} &\textbf{E}\text{valuate: } \mathbf{U}(\overline{\mathbf{X}}^{i-1}),\\ &\textbf{C}\text{orrect: } \overline{\mathbf{X}}^i = \mathbf{X}_k + \frac{\Delta t}{2}\mathbf{U}(\overline{\mathbf{X}}^{i-1}),\end{aligned}\right\} \text{ for } i = 1, ..., I,\\ &\textbf{E}\text{valuate: } \mathbf{U}(2\overline{\mathbf{X}}^I - \mathbf{X}_k).\end{aligned}$$

The predictor step of this method is equivalent to the Euler forward scheme for obtaining a first approximation to $\mathbf{X}_{k+1}$. At the last evaluation step (i.e., after I cycles), the velocities at the (approximate) new positions

$$\mathbf{X}_{k+1} := 2\overline{\mathbf{X}}^I - \mathbf{X}_k,$$

are calculated and used for the next time interval.

In general, the number of cycles to be performed depends on both the order of the predictor scheme and the order of the corrector scheme. If the corrector is of order p and the predictor of order q ($p \geq q$), then the local discretization error after i cycles, $\delta^{(i)}$, is

$$\delta^{(i)} = \mathcal{O}(\Delta t^p) + \mathcal{O}(\Delta t^{i+q}).$$

As far as standard accuracy arguments are concerned, we would not need to do more than $p - q + 1$ cycles; so in our case, two cycles would be enough. However, a finite number of iterations implies an explicit integration, after all, and loss of its symplectic property. In order to obtain an (almost) symplectic scheme, we therefore have to do more cycles, basically until we have solved (13) within machine precision (see Fig. 10).

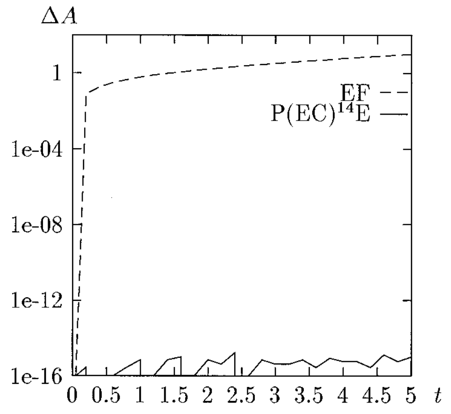

FIG. 10. The same problem as in Fig. 7; 14 cycles were needed to reach machine precision.

One should realize, though, that each cycle requires the calculation of the velocity field, so this is very time consuming. To overcome this problem, we use an extrapolation method to accelerate the convergence of the iteration process. The method we actually used is the so-called *minimal polynomial extrapolation* (MPE) method (see [11]). This MPE method is very suitable for our problem, since it is based on differences and does not need additional information about the Jacobian matrix. Before explaining the idea of this method, we will prove the following property.

PROPERTY 4.1. *For $i \geq 0$,*

$$\overline{\mathbf{X}}^i = \overline{\mathbf{X}} - \left(\frac{\Delta t}{2}\right)^{i+1}\mathbf{J}^i\,\mathbf{U}(\overline{\mathbf{X}}) + \mathcal{O}(\Delta t^{i+2}), \tag{14}$$

where $\mathbf{J}$ is the Jacobian matrix defined by

$$\mathbf{J} := \frac{\partial \mathbf{U}(\overline{\mathbf{X}})}{\partial \overline{\mathbf{X}}}.$$

Proof. The proof is by induction. For $i = 0$, the result follows immediately from (13) since

$$\overline{\mathbf{X}}^0 = \mathbf{X}_k = \overline{\mathbf{X}} - \frac{\Delta t}{2}\mathbf{U}(\overline{\mathbf{X}}),$$

Assume the property holds for i. Then we find with (13), the corrector part of the ($\mathbf{P(EC)}^I\mathbf{E}$) scheme and a Taylor expansion of $\mathbf{U}$ around $\overline{\mathbf{X}}$,

$$\begin{aligned}\overline{\mathbf{X}}^{i+1} &= \mathbf{X}_k + \frac{\Delta t}{2}\mathbf{U}(\overline{\mathbf{X}}^i)\\ &= \mathbf{X}_k + \frac{\Delta t}{2}\mathbf{U}(\overline{\mathbf{X}}) - \left(\frac{\Delta t}{2}\right)^{i+2}\mathbf{J}^{i+1}\mathbf{U}(\overline{\mathbf{X}}) + \mathcal{O}(\Delta t^{i+3})\\ &= \overline{\mathbf{X}} - \left(\frac{\Delta t}{2}\right)^{i+2}\mathbf{J}^{i+1}\mathbf{U}(\overline{\mathbf{X}}) + \mathcal{O}(\Delta t^{i+3}).\end{aligned}$$

This completes the proof. ▮

From this property it follows that

$$\overline{\mathbf{X}}^i - \overline{\mathbf{X}} = \frac{\Delta t}{2}\mathbf{J}(\overline{\mathbf{X}}^{i-1} - \overline{\mathbf{X}}) + \mathcal{O}(\Delta t^{i+2}) \qquad \text{for } i \geq 1, \quad (15)$$

$$\overline{\mathbf{X}}^i - \overline{\mathbf{X}}^{i-1} = \frac{\Delta t}{2}\mathbf{J}(\overline{\mathbf{X}}^{i-1} - \overline{\mathbf{X}}^{i-2}) + \mathcal{O}(\Delta t^{i+1}) \quad \text{for } i \geq 2, \quad (16)$$

and the sequence $\{\overline{\mathbf{X}}^i\}$ is linearly convergent. Since MPE is based on differences, it will be convenient to have short notations for these. We define

$$\mathbf{a}^i := \overline{\mathbf{X}}^i - \overline{\mathbf{X}}^{i-1} \quad \text{for } 1 \leq i \leq I,$$

and

$$\mathbf{A} := (\mathbf{a}^1, \mathbf{a}^2, \ldots, \mathbf{a}^{I-1}).$$

Now MPE calculates the fixed point $\overline{\mathbf{X}}$ as a weighted average of the iterates with weights determined by the coefficients of the minimal polynomial $P(\lambda)$ of $\mathbf{J}$ with respect to $\mathbf{a}^1$. We take $I - 1$ to be the degree of $P(\lambda)$. Then, the minimal polynomial can be written as

$$P(\lambda) = \sum_{i=0}^{I-1} c^{(i)}\lambda^i, \quad c^{(I-1)} = 1.$$

Let the vector $\mathbf{c} := (c^{(0)}, c^{(1)}, \ldots, c^{(I-2)})^{\mathrm{T}}$ be the vector of the unknown coefficients of the minimal polynomial. Then $\mathbf{c}$ is the solution of the system of equations

$$\mathbf{A}\mathbf{c} = -\mathbf{a}^{I-1}. \quad (17)$$

In general, I will be much smaller than the number of nodes. Thus, the system (17) has more equations than unknowns, but consistency can be proven (see [11]). Calculations of $\mathbf{c}$ requires only an LU-decomposition of A and the solution of the upper triangular system, which is cheap compared to the calculation of the velocities. Once the vector $\mathbf{c}$ has been found, the fixed point can be calculated from

$$\left(\sum_{i=0}^{I-1} c^{(i)}\right)\overline{\mathbf{X}} = \sum_{i=0}^{I-1} c^{(i)}\overline{\mathbf{X}}^{i+1}. \quad (18)$$

Of course, we do not know the degree of the minimal polynomial. But this is not a problem in practice. If $I - 1$ is larger than the degree of $P(\lambda)$, then there is no problem at all. If it is smaller, then instead of achieving equality in (17), the least squares solution gives coefficients of an "almost annihilating" polynomial that is the "best" monic polynomial of degree $I - 1$ for eliminating the influence of $I - 1$ dominant components of the error (see [11]). These dominant components of the error are generated by the absolutely largest eigenvalues.

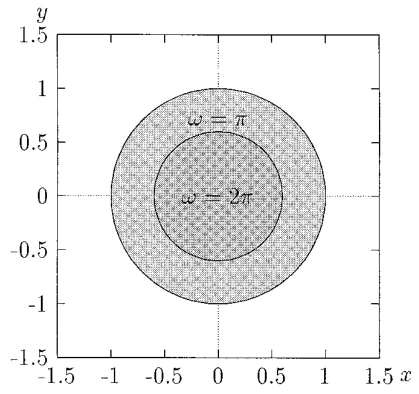

FIG. 11. The situation at $t = 0$: the outer ring has vorticity $\omega = \pi$ and the region inside the inner circle has vorticity $\omega = 2\pi$.

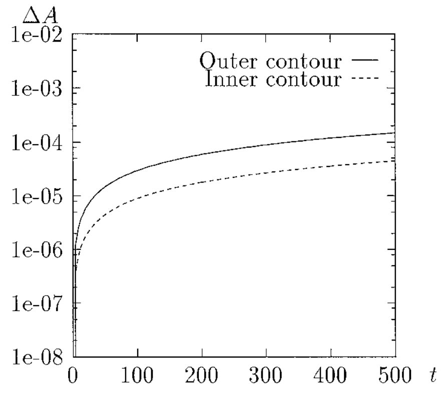

FIG. 12. The variation ΔA of the area as a function of time t. The time step is taken to be $\Delta t = 0.05$, the number of cycles is equal to 3 and MPE-extrapolation is performed.

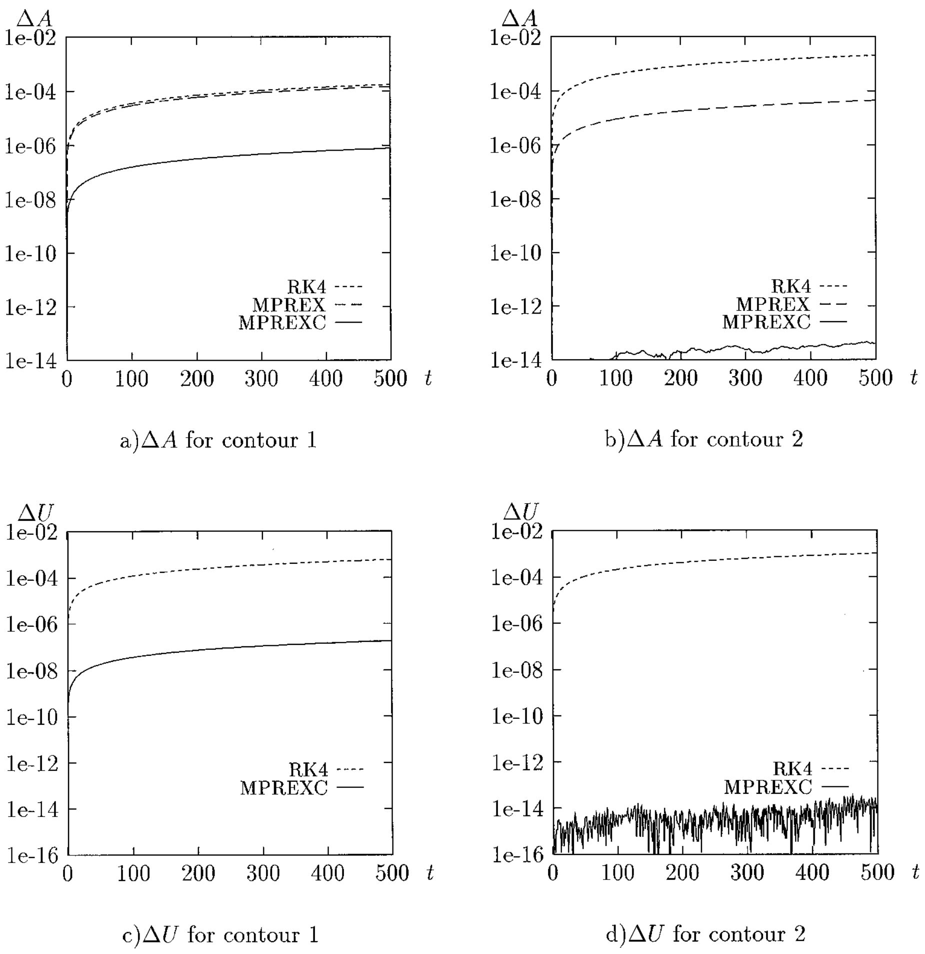

FIG. 13. (a) and (b) show the variation of the area ΔA calculated with the midpoint rule with 3 cycles, where the extrapolation part is split up over the contours (MPREXC), with the midpoint rule with 3 cycles and global extrapolation (MPREX) and with a fourth-order RK-method (RK4). (c) and (d) show the variation of the velocities ΔU for MPREXC and RK4. In all cases $\Delta t = 0.05$.

Consider now a vortex patch consisting of two concentric circular contours, where the outer contour rotates slower than the inner (the fluid enclosed by the inner contour has a larger uniform vorticity than the fluid enclosed by the outer and inner contours). Then the eigenvalues belonging to the outer contour are smaller (in an absolute sense) than those of the inner one. So we might expect that, in the case where $I - 1$ is smaller than the degree of the minimal polynomial, the area of the inner contour is better conserved than that of the outer contour. This is exactly what happens in Figs. 11 and 12. Here the evolution of a circular vortex patch with two contours has been calculated. The outer ring has vorticity $\omega = \pi$ and the inner circle has vorticity $\omega = 2\pi$. So the outer contour has eigenvalues which are smaller (in absolute value) than those of the inner. In Fig. 12 we see that the area enclosed by the inner contour is better conserved than that of the outer, as expected. This suggests that it might be better to split up the extrapolation process over the contours; instead of calculating one set of coefficients $c^{(i)}$ for the whole system, a different set of coefficients is calculated for each contour in order to obtain the dominating terms for each contour. We have implemented this, and the results for the same vortex patch as in Fig. 11 are shown in Figs. 13a (outer contour) and 13b (inner contour). The results for both the outer and the inner contours have improved. This is due to the fact that larger ω makes the extrapolation more accurate (dominance of the eigenvalue is more pronounced). In the next section we will show some more results. In Fig. 13, also, results are shown of the evolution of the same vortex patch but now with the classical fourth-order explicit RK method. The results are in agreement with (12). It might be clear, that the results of the midpoint rule are much better than those of the RK method. This is while the same effort was needed for both methods; we used $I = 3$, so the velocities had to be calculated four

TABLE I

The Number of Rotations at $t = 500$ of Contour 1 and Contour 2 for the Three Different Integrators and the Exact Values

	MPREXC	MPREX	RK4	Exact
Contour 1	169.6	169.6	169.8	170
Contour 2	249.3	249.3	249.8	250

times per time step which is the same as for the fourth-order RK method. In addition, Fig. 13 contains two plots (Figs. 13c and d), where the behaviour of the error ΔU in the velocities (which is defined as $\Delta U := |(U(t) - U(0))/U(0)|$, where $U(t) = \sum_n \sqrt{\hat{u}_n^2 + \hat{v}_n^2}$) as a function of time is shown, both for RK4 and MPREXC. Apparently, MPREXC also gives better results with respect to the velocities.

Another error, the phase shift, may also be investigated (although it is often less interesting in practical situations). A way to do this is to follow the point on a contour which initially is placed at the positive part of the x-axis. The solution of the problem can be determined analytically, and we find that such a point of the outer contour passes the positive x-axis exactly 170 times and that of the inner one passes 250 times. For all numerical methods, this turns out to be slightly less than these analytical values, as can be seen in Table I. However, the differences between the investigated methods are very small, so we may conclude that all numerical schemes produce a phase shift of comparable magnitude.

5. FURTHER NUMERICAL RESULTS

In the previous section, we have compared some numerical results obtained by the midpoint rule with a fourth-order explicit RK method. However, that test problem was rather simple. Therefore, we will consider two more complex problems in this section.

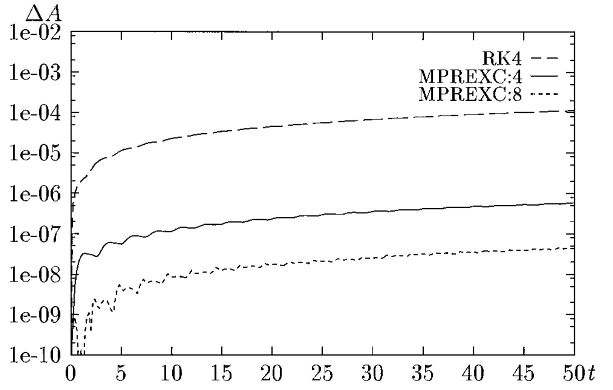

FIG. 14. ΔA as a function of time t for the Kida-vortex.

EXAMPLE 5.1. This example concerns the evolution of a so-called Kida-vortex (see [5]). Initially, an elliptical patch, with aspect ratio equal to 0.75 and of uniform vorticity $\omega = 2\pi$, is placed in the centre of a strain flow $\mathbf{u}_s$, given by

$$u_s = ex$$
$$v_s = -ey.$$

Here, e is the strain rate which we have chosen to be $e = 0.5$. In this case, the motion of the vortex is periodic: the vortex rotates around its centre while it remains elliptic and the aspect ratio changes periodically with time. We calculated the evolution of the vortex three times: once with the midpoint rule with 3 cycles (i.e., four calculations of the velocities per time step) and extrapolation (MPREXC:4), once with the fourth-order Runge–Kutta method (RK4), and finally, once with the midpoint rule with 7 cycles (i.e., eight velocity calculations per time step) and extrapolation (MPREXC:8). In Fig. 14, ΔA is plotted as a function of time t for all three calculations. We see from this figure that MPREXC:4 conserves the area of the

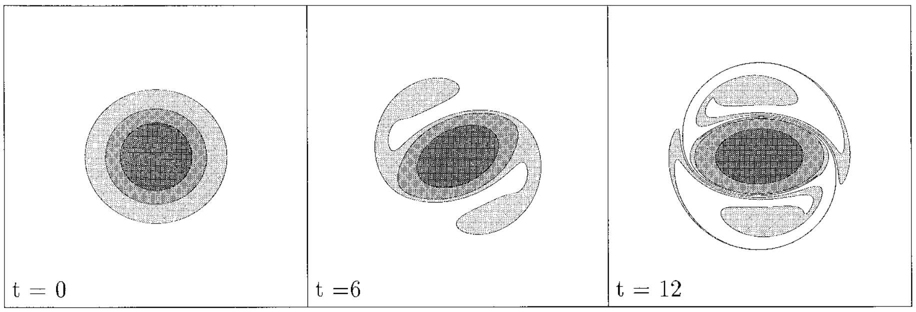

FIG. 15. The evolution of a monopolar vortex into a tripolar vortex.

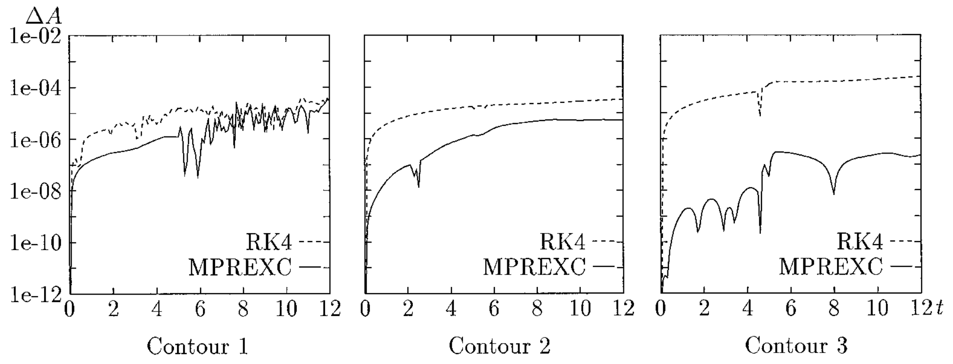

FIG. 16. ΔA plotted as a function of time for contours 1 (outer contour), 2 and 3 (inner contours).

patch better than RK4 with the same effort. Furthermore, we see that with more effort (e.g., 7 cycles), the area is even better conserved. Note that again the behaviour of the area of the patch using RK4 agrees with (12). Furthermore, we should remark that we forced the number of nodes on the contour to be constant. Of course, the variation of the aspect ratio of the ellipse then causes a variation of the area, but this is the same for all calculations. If we had applied node redistribution, we would not have been sure that the effect this had on the area would have been the same for all calculations; this would have hampered our assessment. We shall encounter a similar problem in the next example.

Example 5.2. This example concerns the evolution of a monopolar vortex into a tripolar vortex which is a vortex consisting of an elliptic core with two satellites with vorticity of opposite sign (see, e.g., [8, 9]). The initial configuration consists of three concentric slightly elliptically disturbed contours (aspect ratio equals 0.95). The outer ring has negative vorticity, while the core (consisting of the area enclosed by the second contour) has positive vorticity. Due to the elliptical disturbance, the monopole deforms and becomes a tripole. The evolution is shown in Fig. 15. As we can see from this figure, the outer contour deforms dramatically and this will have an influence on the area enclosed by the outer contour. Although the two inner contours deform much less, nodes are added here too (in regions where the curvature became larger) which also effects the area enclosed. Since this node redistribution might be different for calculations with different time integrators (because of the growing or shrinking of the area caused by the integrator), it is hard to compare the various results. Nevertheless, we again have plotted ΔA for all three contours as a function of time both for RK4 and for MPREXC. The results are shown in Fig. 16.

For contour 1, both methods seem to produce comparable results from time $t = 6$ on. However, this may be caused by the node redistribution, since for $t < 6$, MPREXC gives better results. For the other two contours, MPREXC appears to be better on the entire time interval. This may be very important for long-time integration, since the expansion of the areas of the inner contours can affect the dynamics of the whole vortex. Note that for long-time integration, surgery is essential here.

6. CONCLUSIONS

In this paper, we have considered some aspects of the contour dynamics method. We focussed on the Hamiltonian form of the spatially discretized problem and the consequences this has for the numerical time integration. We have demonstrated that symplectic integration conserves the area enclosed by a contour better than an ordinary integration method. However, a problem with symplectic integration schemes, is that they usually are implicit. This can make implementation rather difficult. Since, in general, the system of equations of the contour dynamics method is not stiff, one may use a predictor–corrector scheme for the implementation. To obtain a symplectic integrator this way, one should perform enough corector steps, i.e., basically until machine precision is reached. However, every corrector step requires the calculation of the velocities, which is rather time consuming. Therefore, we have chosen to use an extrapolation method to accelerate the iteration process. The MPE method turns out to be very suitable for contour dynamics. It is based on the use of differences and does not need additional information about the Jacobian matrix of the system. Performing the extrapolation for each of the contours separately, appears to work even better in practice.

ACKNOWLEDGMENTS

We thank Professor G. J. F. van Heijst who introduced us to the subject of contour dynamics.

REFERENCES

1. D. G. Dritschel, Contour surgery: A topological reconnection scheme for extended integrations using contour dynamics, *J. Comput. Phys.* **77,** 240 (1988).
2. D. G. Dritschel, Contour dynamics and contour surgery: Numerical algorithms for extended, high-resolution modelling of vortex dynamics in two-dimensional, inviscid, incompressible flows, *Comput. Phys. Rep.* **10,** 77 (1989).
3. E. Hairer, S. P. Nørsett, and G. Wanner, *Solving Ordinary Differential Equations 1* (Springer-Verlag, New York/Berlin, 1987).
4. E. Hairer and G. Wanner, *Solving Ordinary Differential Equations 2* (Springer-Verlag, New York/Berlin, 1991).
5. S. Kida, Motion of an elliptic vortex in a uniform shear flow, *J. Phys. Soc. Jpn* **50,** 3517 (1981).
6. H. Lamb, *Hydrodynamics,* 6th ed. (Dover, New York, 1932).
7. B. Legras and D. G. Dritschel, A comparison of the contour surgery and pseudo-spectral methods, *J. Comput. Phys.,* **104,** 287 (1993).
8. S. J. Lin, Contour dynamics of tornado-like vortices, *J. Atmos. Sci.* **49,** 1745 (1992).
9. P. Orlandi and G. J. F. van Heijst. Numerical simulation of tripolar vortices in 2D flow. *Fluid Dyn. Res.,* **9,** 179 (1992).
10. J. M. Sanz-Serna and M. P. Calvo. *Numerical Hamiltonian Problems.* Chapman & Hall, first edition, 1994.
11. D. A. Smith, W. F. Ford, and A. Sidi. Extrapolation methods for vector sequences. *SIAM Rev.,* **29,** 199 (1987).
12. N. J. Zabusky, M. H. Hughes, and K. V. Roberts. Contour dynamics for the Euler equations in two dimensions. *J. Comput. Phys.,* **30,** 96 (1979).