INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2008; **58**:161–181
Published online 7 January 2008 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/fld.1715

A Lagrangian vortex method for unbounded flows

C. Moussa and M. J. Carley*, †

Department of Mechanical Engineering, University of Bath, Bath BA2 7AY, U.K.

SUMMARY

A technique is presented for velocity calculations on the highly distorted node distributions typical of those found in Lagrangian vortex methods. The method solves the partial differential equation for streamfunction directly on the nodes, via a sparse, symmetric system of equations that can be solved using standard iterative solvers. When implemented in a triangulated vortex method, the technique gives computation times which scale as $N^{1.23}$, where N is the number of nodes. The computation scheme is derived for two-dimensional problems and applied to the prediction of the evolution of perturbed multipolar vortices. Due to the numerical performance of the method, it has been possible to examine such evolution at higher and lower Reynolds numbers than have been considered in published numerical studies. Copyright © 2008 John Wiley & Sons, Ltd.

Received 30 April 2007; Revised 6 November 2007; Accepted 6 November 2007

KEY WORDS: meshed vortex methods; least squares; vortex instability

1. INTRODUCTION

Lagrangian vortex methods are a powerful class of adaptive techniques for the prediction of unsteady flows [1, 2]. They work by computing the movement of vorticity, tracking marker nodes that are transported at the local velocity. This velocity is itself a function of the global vorticity distribution and one of the central parts of any Lagrangian vortex method—and the most computationally intensive—is the computation of the nodal velocities. This paper presents a velocity computation method that can handle the highly distorted meshes typical of vortex methods and that can be implemented using standard solvers for sparse, symmetric linear systems.

The two fundamental components of a Lagrangian vortex method are a discretization or interpolation scheme to represent the vorticity distribution in terms of the nodal values, and a velocity

*Correspondence to: M. J. Carley, Department of Mechanical Engineering, University of Bath, Bath BA2 7AY, U.K.
†E-mail: m.j.carley@bath.ac.uk

evaluation method to compute the node motion. To some degree, the choice of one conditions the choice of the other but, on the whole, they can be considered independently. Two main approaches exist for the calculation of velocities: evaluation of a Biot–Savart integral and solution of the partial differential equation for streamfunction. The integral approach, which becomes a summation in 'particle' or 'blob' discretizations, is probably the most widely used, being conceptually simple and fast. In its standard form, the computational effort is $O(N^2)$ where N is the number of nodes in the problem, but the use of fast multipole methods [3–5] can reduce this to $O(N)$.

The most common method of directly solving the partial differential equation for the streamfunction is to interpolate the vorticity distribution onto a fixed grid and solve a Poisson equation on the grid. Nodal velocities can then be found by differencing. Two main approaches have been used: the vortex-in-cell method [6], which interpolates onto a regular grid and the vortex-in-element method [7] which interpolates onto an unstructured mesh. In the first case, Fourier transform-based Poisson solvers can be used to speed the calculation, while in the second, a finite element solution can be pre-computed and the velocity is found using a matrix multiplication.

An approach that does not seem to have been used in Lagrangian vortex methods is direct solution of the partial differential equation on the nodes. The obvious method, a finite element solution on the mesh formed by the nodes, would require a high-quality triangulation, which conflicts with the adaptive nature of the solution method. Some other approaches have been developed, which can be used for calculations on highly distorted, deforming, meshes, such as the natural element method of Braun and Sambridge [8], the Voronoi cell finite difference method of Sukumar [9] and the least-squares method of Baty and Wolfe [10], which has similarities to the method of this paper. The approach was to solve, in the least-squares sense, for the coefficients of a Taylor expansion of velocity potential about each nodal point, using an iterative scheme. There was no requirement that the nodal distribution be high quality, making it a candidate for adaptive vortex methods, but the computational effort was $O(N^2)$, a drawback in applications.

In this paper, we develop a method that calculates the streamfunction and its derivatives at each nodal point, using a standard linear solver for large sparse systems. The method is symmetric and can be implemented for quite general partial differential equations with very few modifications. The velocity computation is used in a triangulated vorticity scheme that has already been used in a variety of problems [11–14]. The general approach is described below, as well as the implementation of viscous effects and the practicalities of mesh generation and maintenance. Results are given for the algorithm performance and accuracy and it is then applied to the problem of the dynamics of a class of vortex instabilities.

2. NUMERICAL FORMULATION

In this section, the numerical method used to compute the streamfunction and nodal velocities is developed. The computation is performed using a least-squares method based on the three requirements of a valid solution to the partial differential equation for the system: the solution must be continuous between nodes, must satisfy the equation at nodes and must meet appropriate boundary conditions. The first of these requirements is imposed by computing the solution as the coefficients of a Taylor series expansion about each node and requiring that the expansion about one node match that about nearby nodes, to some order. The second requirement is imposed via a relationship between coefficients of the power series at each node and the third is imposed by fixing power series coefficients on boundary nodes.

2.1. *Plane flow*

The well-known Helmholtz equation [15, p. 14] states that vorticity is conserved in two-dimensional inviscid, incompressible flow:

$$\frac{\mathrm{d}\omega}{\mathrm{d}t}=0 \tag{1}$$

where the material derivative in a plane flow is

$$\frac{\mathrm{d}}{\mathrm{d}t}\equiv\frac{\partial}{\partial t}+u\frac{\partial}{\partial x}+v\frac{\partial}{\partial y}$$

For plane flow, position is given by coordinates (x, y), velocity is (u, v) and we adopt a Lagrangian approach, tracking the movement of markers that carry vorticity ω.

In order to compute the motion of the markers, we must compute their velocities. The most common method is to use the streamfunction ψ, which is a solution of

$$\psi_{xx}+\psi_{yy}=-\omega \tag{2}$$

where subscripts denote differentiation and the velocities are

$$u=\psi_y, \quad v=-\psi_x \tag{3}$$

Two main approaches are possible: direct solution of Equation (2) or evaluation of the Biot–Savart integral:

$$\psi(x, y)=\int\int G(x, y; x_1, y_1)\omega(x_1, y_1)\,\mathrm{d}x_1\,\mathrm{d}y_1 \tag{4}$$

where Green's function G is, for plane flow,

$$G(x, y; x_1, y_1)=-\frac{1}{4\pi}\log((x-x_1)^2+(y-y_1)^2) \tag{5}$$

In this paper, we adopt the approach of solving Equation (2) directly at the nodes of the vorticity distribution, using a least-squares formulation similar to that of Baty and Wolfe [10]. The solution $\mathbf{\Psi}_i$ at node i is expressed as a vector of the derivatives of the streamfunction:

$$\mathbf{\Psi}_i=[\psi_0^{(i)}\ \psi_x^{(i)}\ \psi_y^{(i)}\ \psi_{xx}^{(i)}\ \psi_{xy}^{(i)}\ \psi_{yy}^{(i)}]^{\mathrm{T}} \tag{6}$$

For a second-order partial differential equation in two dimensions, $\mathbf{\Psi}_i$ has six components.

A solution of the streamfunction equation is made up of $\mathbf{\Psi}_i$ for each node subject to continuity of the solution between nodes and satisfaction of the partial differential equation. Considering a node i and a neighbour node j (defined in Section 2.2), these requirements give two matrix equations:

$$T_{ij}^{(0)}\mathbf{\Psi}_i=\psi_0^{(j)} \tag{7}$$

$$D_i\mathbf{\Psi}_i=-\omega_i \tag{8}$$

where the left-hand side matrices are

$$T_{ij}^{(0)} = [1\ x_{ij}\ y_{ij}\ x_{ij}^2/2\ x_{ij}y_{ij}\ y_{ij}^2/2] \tag{9}$$

$$D_i = [0\ 0\ 0\ 1\ 0\ 1] \tag{10}$$

$$x_{ij} = x_j - x_i, \quad y_{ij} = y_j - y_i \tag{11}$$

The system of equations will be solved in a least-squares sense for all of the nodes simultaneously (unlike the method of Baty and Wolfe [10] who iterate over nodes).

The system of equations is expressed as

$$\mathbf{A}\psi = b \tag{12}$$

where $\mathbf{A}$ is a large, sparse matrix made up of a combination of D_i and $T_{ij}^{(0)}$, ψ contains $\mathbf{\Psi}_i$ for $i = 0 \ldots N-1$ and b contains the nodal vorticities and any boundary conditions to be applied.

Considering the interaction of a node i with a neighbour node j and ignoring, for the moment, any boundary condition at node i or j, Equations (7) can be combined to yield the single equation:

$$G_{ij}\mathbf{\Psi}_i + H_{ij}\mathbf{\Psi}_j = b_{ij} \tag{13}$$

$$G_{ij} = \begin{cases} T_{ij}^{(0)} - \dfrac{x_{ij}^2}{2} D_i, & j \text{ even} \\ T_{ij}^{(0)} - \dfrac{y_{ij}^2}{2} D_i, & j \text{ odd} \end{cases} \tag{14}$$

$$H_{ij} = [-1\ 0\ 0\ 0\ 0\ 0] \tag{15}$$

and

$$b_{ij} = \begin{cases} \dfrac{x_{ij}^2}{2}\omega_i, & j \text{ even} \\ \dfrac{y_{ij}^2}{2}\omega_i, & j \text{ odd} \end{cases} \tag{16}$$

The definition of G_{ij} alternates between the two forms given to avoid inserting a row of all zeros in the main matrix.

The matrix $\mathbf{A}$ is made up of 6×6 blocks so that its total size is $6N \times 6N$. Equation (13) can be expressed as a symmetric system and inserted as follows:

$$\begin{pmatrix} \vdots & \vdots & \vdots & \vdots & \vdots \\ \cdots & G_{ij}^{\mathrm{T}}G_{ij} & \cdots & G_{ij}^{\mathrm{T}}H_{ij} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \cdots & H_{ij}^{\mathrm{T}}G_{ij} & \cdots & H_{ij}^{\mathrm{T}}H_{ij} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{pmatrix} \begin{pmatrix} \vdots \\ \mathbf{\Psi}_i \\ \vdots \\ \mathbf{\Psi}_j \\ \vdots \end{pmatrix} = \begin{pmatrix} \vdots \\ b_{ij}G_{ij}^{\mathrm{T}} \\ \vdots \\ b_{ij}H_{ij}^{\mathrm{T}} \\ \vdots \end{pmatrix} \tag{17}$$

where the 6×6 block $G_{ij}^{\mathrm{T}}G_{ij}$, for example, is inserted in rows $6i+1$–$6i+6$ and columns $6i+1$–$6i+6$ of the left-hand side matrix.

To generate the system of Equations (12), we loop over all of the nodes $i=0,1,\dots,N-1$. At each node, we loop over the neighbours j and add Equation (17) to the overall system. In practice, we can also multiply Equation (17) by a weighting factor w_{ij}, if required.

In order to incorporate boundary conditions, the procedure is modified. As an example, we consider the inclusion of a prescribed velocity. In such a case, the vorticity at the boundary forms part of the solution and so the differential operator is not incorporated into the system. Instead the stream function components are used in place of the differential operator so that G_{ij} and b_{ij} are defined:

$$G_{ij}=T_{ij}^{(0)}-[0\ x_{ij}\ y_{ij}\ 0\ 0\ 0] \tag{18}$$

$$H_{ij}=[1\ 0\ 0\ 0\ 0\ 0] \tag{19}$$

and

$$b_{ij}=-x\psi_x-y\psi_y \tag{20}$$

where ψ_x and ψ_y are defined using the prescribed velocity on the boundary. A similar approach can be used to impose a streamfunction on the boundary as required in unbounded flow calculations.

After the system of equations has been assembled, the final part of imposing the boundary conditions is to modify the rows of the matrix corresponding to boundary conditions. These rows are cleared (set to zero) and, in the case of a velocity boundary condition, replaced with the trivial equations:

$$[0\ 1\ 0\ 0\ 0\ 0]\mathbf{\Psi}_i=\psi_x^{(i)} \tag{21a}$$

$$[0\ 0\ 1\ 0\ 0\ 0]\mathbf{\Psi}_i=\psi_y^{(i)} \tag{21b}$$

The alternative approach of deleting rows and columns corresponding to boundary nodes has not been adopted because this leads to variations in the size of the system of equations during a calculation, which makes memory management and matrix updating quite awkward.

The assembly of the system of equations can be summarized in the following algorithm:

```
For each node $i=0\dots N-1$:
If $i$ is not a boundary node:
Calculate $D_i$.
For each neighbour node $j$:
Calculate $T_{ij}^{(0)}$.
Calculate $G_{ij}$.
If $i$ is not a boundary node:
Evaluate Equation (13) and add to the system of equations.
Else  Evaluate Equation (18) and add to the system of equations.
For each node $i=0\dots N-1$:
If $i$ is a boundary node:  Insert Equations (21a) and (21b).
```

This sequence of steps yields a symmetric $6N\times 6N$ system of equations that can be solved by a standard iterative technique. In this work, the stabilized biconjugate gradient method was used [16].

2.2. *Vorticity triangulation and neighbour identification*

The velocity calculation method of Section 2.1 is used in a triangulated vorticity method. This is a technique used by a number of researchers to compute planar, axisymmetric and three-dimensional flows [11–14, 17]. In it, a Delauanay triangulation connects the vorticity nodes and the triangulation can then be used in extracting the required information from the vorticity distribution. In particular, the triangulation is used in applying the boundary conditions and in extracting information about the neighbourhood of a node.

The first part of applying boundary conditions is to identify the boundary. When a triangulation is available, boundary nodes can be identified quite easily. The second part of applying boundary conditions is to compute the value of the solution at boundary nodes. This is done using multipole expansions, with the multipole coefficients c_{ij} about some point (x_0, y_0) computed using the triangulation:

$$c_{ij} = \int\int (x-x_0)^i (y-y_0)^j \omega(x,y)\,\mathrm{d}x\,\mathrm{d}y \tag{22}$$

the integration being performed over triangular elements whose nodes have vorticities above some threshold value. The streamfunction outside the region of finite vorticity can then be computed using the expansion:

$$\begin{aligned} -4\pi\psi &= c_{00}\log R^2 - 2(c_{10}(x-x_0)+c_{01}(y-y_0))/R^2 - c_{20}((x-x_0)^2-(y-y_0)^2)/R^4 \\ &\quad - c_{02}((y-y_0)^2-(x-x_0)^2)/R^4 - 4c_{11}(x-x_0)(y-y_0)/R^4 \\ R^2 &= (x-x_0)^2+(y-y_0)^2 \end{aligned} \tag{23}$$

The triangulation also provides useful information about the neighbourhood of a node. To identify the neighbours of a node i, we begin with a list of nodes connected to node i and add nodes connected to these 'first-level' neighbours. If necessary, we add nodes connected to these nodes. The list of neighbour nodes is then sorted by distance from node i and the n closest are chosen as the neighbours. This approach avoids using nodes that are directly connected to node i but are not geometrically close to it.

A similar approach is used in determining the weighting function w_{ij}. This is a Gaussian:

$$w_{ij} = \mathrm{e}^{-(x_{ij}^2+y_{ij}^2)/\sigma^2} \tag{24}$$

where the length scale σ depends on the local mesh geometry and is a function of the edge lengths connected to node i. A robust value for σ is the arithmetic or geometric mean of the edge lengths.

2.3. *Viscosity*

Another component of a Lagrangian vortex method is the computation of viscous effects. In a plane flow, Equation (1) must be modified:

$$\frac{\mathrm{d}\omega}{\mathrm{d}t} = \nu\nabla^2\omega \tag{25}$$

where v is the kinematic viscosity, equal to $1/Re$ in non-dimensional variables. To compute the development of vorticity, a moving least-squares method has been used to calculate $\nabla^2\omega$. This is a technique that has proved robust and reliable in previous work [14, 18]. A polynomial of the form

$$\omega_j - \omega_i = \omega_x x_{ij} + \omega_y y_{ij} + \omega_{xx} x_{ij}^2/2 + \omega_{xy} x_{ij} y_{ij} + \omega_{yy} y_{ij}^2/2 \tag{26}$$

is least-squares fit to the vorticity at node i and its neighbours j and the coefficients of the polynomial are then used to estimate $\nabla^2\omega$. In practice, as in the work of Marshall and Grant [18], the polynomial is fit to $\log|\omega|$ and the derivatives of ω extracted using the chain rule for differentiation. This avoids numerical problems in regions of large vorticity gradient. When ω is small or changes sign, Equation (26) is applied directly. We note at this point that in computing a viscous flow, the prescribed velocity boundary condition of Equation (18) could be used to implement a no-slip condition.

2.4. Node motion and mesh updating

Given the computed velocities, the motion of the nodes and the evolution of their vorticities can be computed. The node motion is computed using a standard low-storage Runge–Kutta rule [19], while the vorticity is updated using a forward Euler step as in the work of Russo [17].

The stability of the viscous computation depends on the time step Δt and the characteristic length scale. For simplicity, the stability criterion for a square grid [17],

$$v\Delta t \leqslant \tfrac{1}{2}h^2 \tag{27}$$

where h is the grid spacing, has been used. In this case, h is the minimum permissible edge length in the mesh. As the vorticity evolves, it is periodically remeshed, using the basic operations shown in Figure 1.

The first of these operations, edge collapsing, replaces a short edge with a single node to stabilize the viscous computation. The second operation, edge splitting, divides a long edge in two by inserting a new node at its midpoint. This maintains spatial resolution of the flow. The third operation, boundary expansion, adds a new point outside the mesh boundary on the normal to a boundary edge. This expands the mesh as viscous diffusion increases the vorticity support.

2.5. Vorticity initialization and meshing

The initial condition for the computation is generated using an iterative method that gives a specified vorticity distribution on a mesh of given quality to within some tolerance. The process is shown in Figure 2 for the Lamb–Oseen vortex, whose evolution is computed in Section 3.1.

In the first stage, Figure 2(a), an initial mesh is generated using a circular boundary. The triangles are quite poor quality so the next step is to refine the mesh using a point insertion technique to give specified maximum and minimum triangle areas and qualities (Figure 2(b)). Finally, the vorticity is computed at each node. In an iterative procedure, the vorticity is then interpolated at the centroid of each triangle and compared with the exact value. If they differ by more than a specified tolerance, a new node is inserted at the centroid with the exact value of vorticity. The procedure is repeated until the tolerance has been met on each triangle. Since the Lamb–Oseen

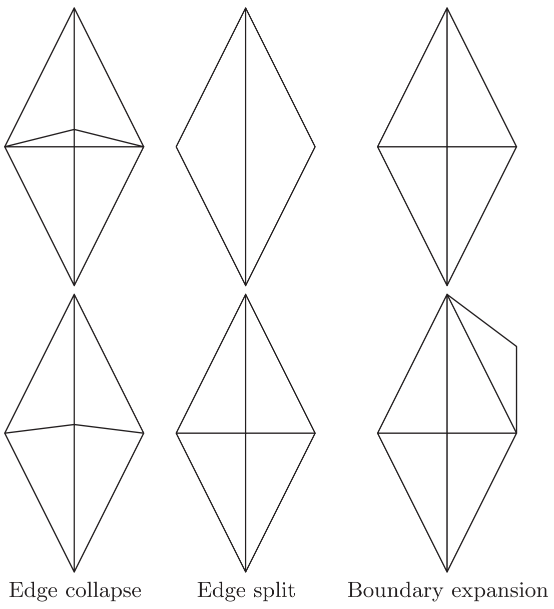

Figure 1. Basic mesh quality operations.

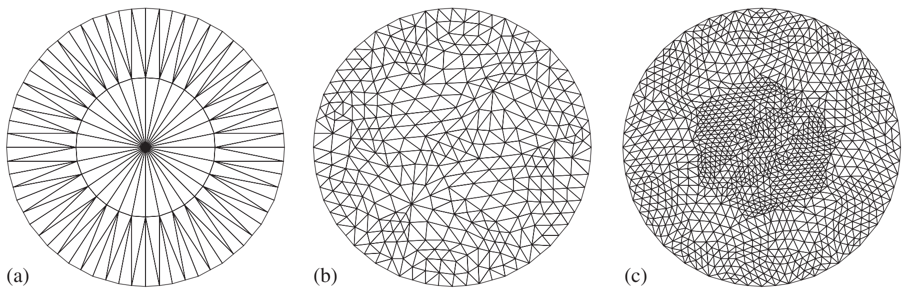

Figure 2. Iterative mesh refinement for problem initialization: (a) initial mesh; (b) mesh after Delaunay refinement; and (c) mesh after vorticity refinement.

vortex has high vorticity in the centre of the mesh, the node density is higher here, as can be seen in Figure 2(c).

3. PERFORMANCE

The performance of the calculation method is assessed using standard test cases for planar viscous and inviscid flows. The criteria used are accuracy and computation time. In each case, Equation (12) for the streamfunction was solved using a stabilized biconjugate gradient method for sparse matrices [16] with a solution tolerance $\varepsilon = 10^{-6}$, where the stopping criterion is

$$|r| \leqslant \varepsilon |b| \tag{28}$$

with $|r|$ and $|b|$ the norms of the residual and the right-hand side, respectively.

3.1. Plane flow accuracy

For plane flows, there exist exact analytical solutions that can be used to check the accuracy of the computation method. We use the radially symmetric patch described by Perlman [20], for which

$$\omega(x, y) = (1-R)^7, \quad R = [x^2 + y^2]^{1/2} \tag{29}$$

inside the circle $R = 1$ and zero outside. The velocity field is

$$(u, v) = f(R)(y, -x) \tag{30}$$

with

$$f(R) = -(1-(1-R^2)^8)/16R^2, \quad R \leqslant 1 \tag{31}$$

$$= -1/16R^2, \quad R > 1 \tag{32}$$

The vorticity distribution remains constant over time, notwithstanding the motion of the nodes, so that this is a simple check on the accuracy of the calculation. As an error measure, we employ $\max|\Delta\omega|/\max\omega$, the maximum difference between the computed and exact vorticities scaled on the maximum vorticity. The evolution of the flow is computed for 200 time steps with $\Delta t = 25 \times 10^{-3}$.

Figure 3 shows the error at the end of the calculation as a function of the number of nodes. Note that due to the remeshing of the system, the number of nodes is not constant during the calculation, although it does not vary much. The figure also shows the linear fit to the error, showing an error that reduces near linearly with N, i.e. almost to second order with discretization length.

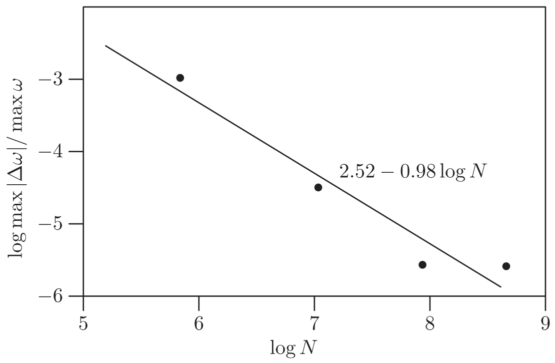

Figure 3. Evolution of Perlman's polynomial vorticity patch: maximum error in vorticity against number of points at the end of calculation.

The second plane flow test is performed to check the behaviour of the viscous calculations. Again, an exact solution is available, the Lamb–Oseen vortex described by Lamb [21, Section 334a] and by Saffman [15, p. 253]. The vorticity begins as an impulsive source of circulation Γ_0,

$$\omega(x, y, 0) = \Gamma_0 \delta(x)\delta(y) \tag{33}$$

and it evolves over time according to

$$\omega(x, y, t) = \frac{\Gamma_0}{4\pi \nu t} \mathrm{e}^{-R^2/4\nu t} \tag{34}$$

where $R^2 = x^2 + y^2$. Defining a Reynolds number $Re_\Gamma = \Gamma_0/\nu$, the evolution of the vortex is computed from times $\nu t = 0.01$ to 0.02, so that the initial peak vorticity decays to half its original value. Figure 4 shows the exact and computed evolution of the vorticity distribution for $\nu = 0.01$ ($Re_\Gamma = 100$) and Figure 5 shows the same data for $\nu = 0.001$ ($Re_\Gamma = 1000$). The comparison between the two sets of results is seen to be good in both cases.

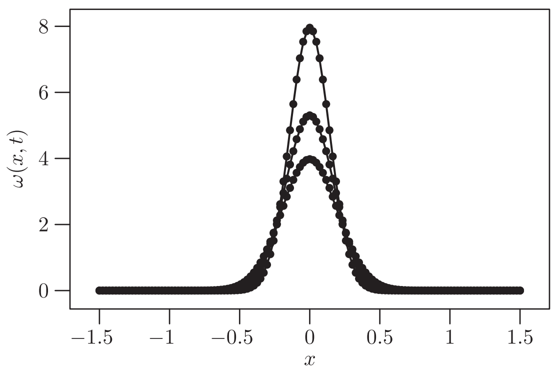

Figure 4. Evolution of Lamb–Oseen vortex $Re_\Gamma = 100$: exact (line) and computed (circles) solution at $\nu t = 0.01, 0.015, 0.020$.

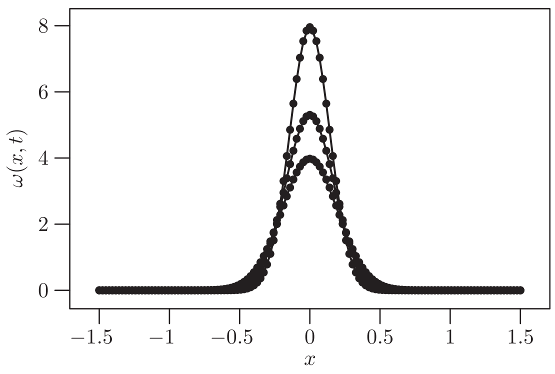

Figure 5. Evolution of Lamb–Oseen vortex $Re_\Gamma = 1000$: exact (line) and computed (circles) solution at $\nu t = 0.01, 0.015, 0.020$.

Figure 6 shows the variation in error during the computation of decay of a Lamb–Oseen vortex with $Re_\Gamma = 300$ and varying mesh resolution. The four curves show the error with different mean numbers of nodes during the calculation and the improvement in accuracy with increasing node number is clear. This point is reinforced by Figure 7, which shows the error as a function of node number at the end of the calculation. The linear fit to the error shows it reducing as $N^{-0.76}$, corresponding to a variation with discretization length between first and second order.

3.2. *Computation time*

Results are now presented for the computation time required to perform a calculation. We use Perlman's test case [20] defined in Equation (29) and examine the time required to solve once for streamfunction and that required to compute the evolution of a flow. Computations were performed on an otherwise unused node of a multiprocessor cluster.

The first set of results, Figure 8, shows the computation time required to compute the streamfunction without an initial guess, equivalent to the first time step of a calculation. The best fit line has a slope of 1.12 but this does not seem to be a very good fit to the data. Comparison with

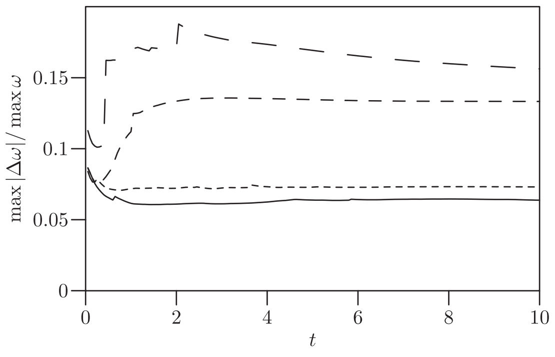

Figure 6. Evolution of Lamb–Oseen vortex, $Re_\Gamma = 300$: error during evolution, from the top mean number of nodes $\bar{N} = 2424, 3697, 6041, 9348$.

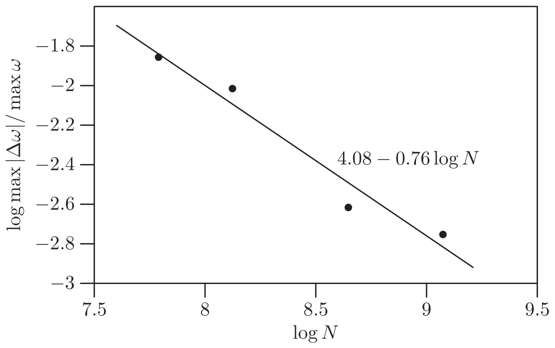

Figure 7. Evolution of Lamb–Oseen vortex, $Re_\Gamma = 300$: error as a function of number of nodes at the end of calculation.

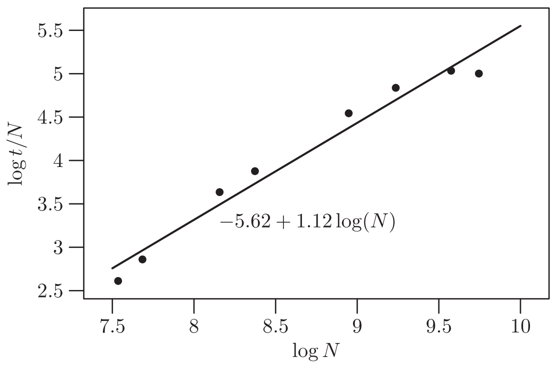

Figure 8. Execution time for first time step *versus* number of points for Perlman's polynomial vorticity patch.

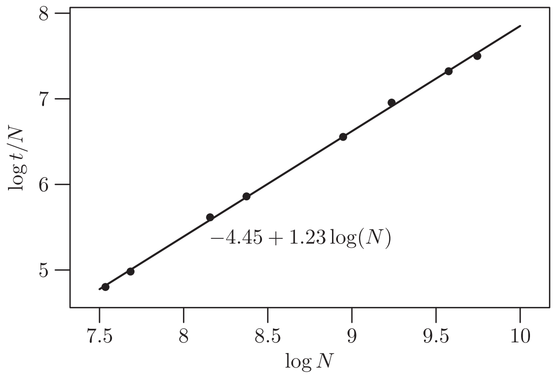

Figure 9. Execution time per point for evolution of Perlman's polynomial vorticity patch.

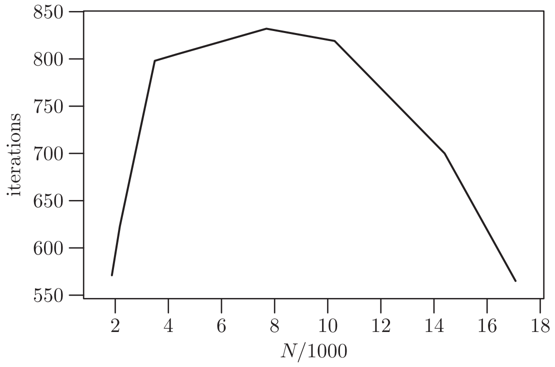

Figure 10. Number of iterations for solution at first time step.

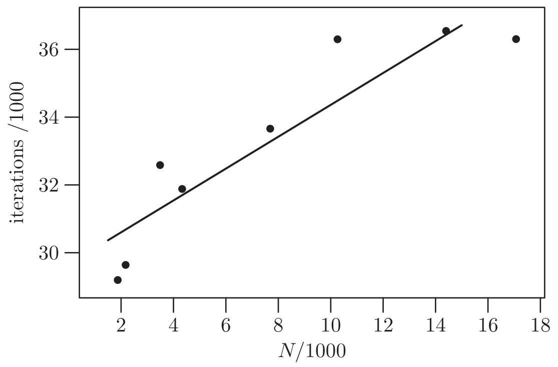

Figure 11. Total number of iterations for solution.

Figure 9 shows that the computation time for multiple time steps scales as $N^{1.23}$ for this problem, rather worse performance than might be expected from the 'cold start' data.

The reason for this discrepancy can be seen in Figure 10. This shows the number of iterations required by the linear solver in computing the streamfunction for a single calculation. This shows no clear trend with N, reflecting the scatter in Figure 8. Comparison with Figure 11 shows that the total number of iterations in computing the evolution of the flow is more closely linked to the number of nodes, noting that the computation time for the sparse matrix operations is roughly linear in N. We conclude that the computation time scales as $N^{1.23}$ and that the better performance seen in Figure 8 is a matter of chance, related to the particular configuration of the points in this case.

4. EVOLUTION OF PERTURBED MULTIPOLAR VORTICES

The method developed in the previous sections is now applied to the practical problem of computing the evolution of perturbed Lamb–Oseen vortices.

Unstable vortices exist in large-scale geophysical flows where they play an important role in the transport of mass and energy. Because of their practical and theoretical importance, they have been the subject of investigation by many researchers. In particular, the shielded or isolated vortex has been of some interest [22–25]. This is a monopole vortex surrounded by a region of vorticity of opposite sign with the whole structure having zero total circulation.

The evolution of unshielded vortices is still poorly understood, although it has been established that an unbounded structure decays to axisymmetry [26]. This decay can be slow, however, and the question of the behaviour of the structure during its evolution is important. It is known that a stage of the evolution is the transition of a perturbed structure to a multipolar form [27–29]. Using the method of this paper, a study of the development of these perturbed vortices has been conducted. In the following section, to demonstrate the utility of the new computational method, the development of a Lamb–Oseen vortex perturbed by a dipolar distortion, a case not studied before, is presented.

4.1. Initial conditions

In the current study, a perturbed Lamb–Oseen vortex is allowed to decay freely. The initial vorticity field ω is the superposition of a Gaussian distribution ω_G and a disturbance ω', not necessarily small:

$$\omega = \omega_G + \omega' \tag{35}$$

where

$$\omega_G = \frac{\Gamma}{2\pi\sigma^2} e^{-r^2/2\sigma^2} \tag{36}$$

$$\begin{aligned} \omega' &= \frac{\delta\Gamma}{2\pi\sigma^2} r^2 e^{-r^2/2\sigma^2} \cos m\theta \\ r^2 &= (x-x_0)^2 + (y-y_0)^2 \\ \theta &= \tan \frac{y-y_0}{x-x_0} \end{aligned} \tag{37}$$

with Γ the total circulation of the vortex, $\sigma = 0.6308 r_c$ [30] and m the order of the perturbation. Figure 12 shows the contours of the base Lamb vortex ω_G, a quadrupolar and a hexapolar ω'.

Simulations have been carried out for different values of δ, m and Re. The flow field is represented as a circular domain of radius $r = 8$ and discretized initially with the following criteria: minimum and maximum triangle areas are set to $A_{\min} = 0.003$ and $A_{\max} = 0.03$, respectively, and the error tolerance between the specified and the computed values of the vorticity is 10^{-4}. A small tolerance is chosen in order to have a smooth initial vorticity distribution, which leads to a finer meshing of the non-zero vorticity area compared with the rest of the flow. These parameters were found, after several trials, to give a satisfactory first triangulation, leading to good results. The choice of the initial values is critical in order to obtain decent results, because a coarse mesh will not have a smooth vorticity distribution, which will affect the evolution and decay of the vortex, while a very fine mesh with small grid spacing will lead to numerical instability if any edge length happens to be smaller than the characteristic length scale. The initial discretization for a vortex

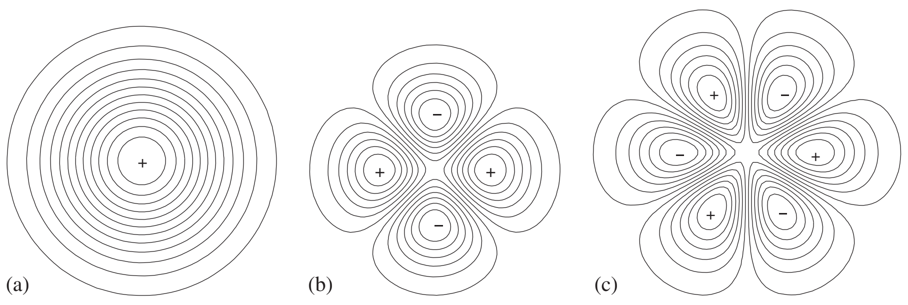

Figure 12. Vorticity contours of the (a) base Lamb vortex; (b) quadrupolar perturbation; and (c) hexapolar ω'.

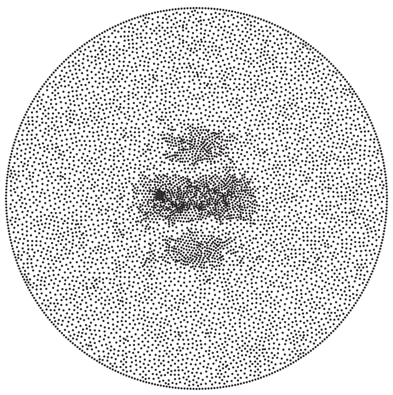

Figure 13. Initial discretization of ω, $m=2$.

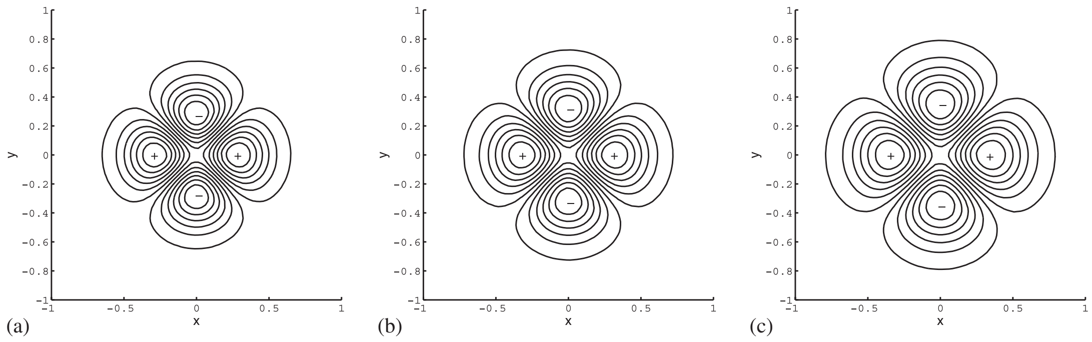

Figure 14. Perturbation vorticity ω', $m=2$ at times (a) $t=0$; (b) $t=50$; and (c) $t=100$.

with a quadrupolar perturbation is presented in Figure 13. The time step is set to $\Delta t=0.2$ and after every six time steps the domain is remeshed with the condition that the edge length lies between $h_{\min}=0.07$ and $h_{\max}=0.3$. These values are chosen to keep the triangle areas within the initial range. In all cases, the boundary streamfunction was kept constant to simulate the wall of the circular tank in which experimental studies have been conducted.

4.2. *Decay of an isolated perturbation*

Prior to running the model for the perturbed Lamb vortex, the decay of a quadrupolar isolated perturbation centred at the origin of the domain, $x_0=y_0=0$, was simulated in order to analyse its evolution and decay. The Reynolds number for this case was $Re=10^4$, the circulation $\Gamma=1$, the amplitude $\delta=0.05$ and the standard deviation $\sigma=2^{1/2}$.

The contour plots of ω' are displayed in Figure 14 and the vorticity distribution in Figure 15 for three different times. It is observed that the disturbance has an equilibrium state so that it decays while conserving the symmetry of its four components, both in their shapes and their strengths, while, due to the shear forces that the poles exert on each other, their expansion is slow. Close

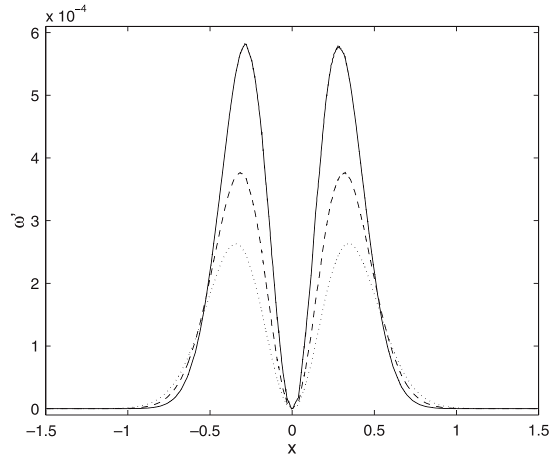

Figure 15. Vorticity distribution of the dipolar perturbation at $y=0$, $t=0$ (solid line), $t=50$ (dashed line) and $t=100$ (dotted line).

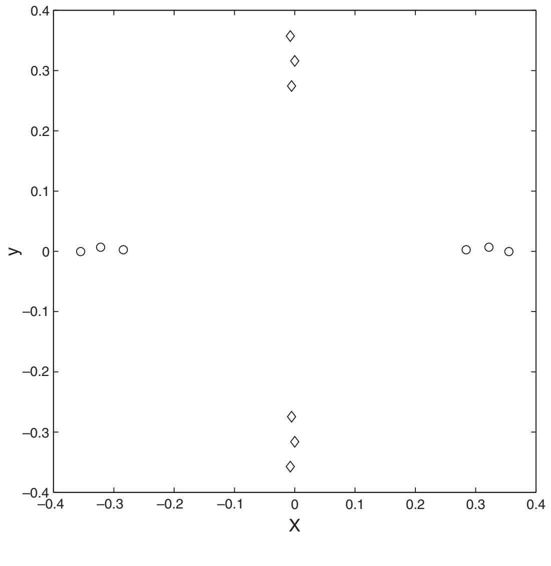

Figure 16. Position of poles of perturbation vorticity, ∘ positive and ◇ negative during the decay, time step $\Delta t=50$.

inspection of the positions of the centres shows that while expanding the poles push each other and move away, the positive ones horizontally and the negative ones vertically at equal speed. This can be clearly seen in Figure 16. The positions of the poles are symmetrical with respect to the centre of the domain $(x_0, y_0)=(0,0)$, and the small vibrations observed in the x and y positions are due to the discretization.

4.3. *Evolution of a Lamb vortex with a dipolar perturbation*

An $m=1$ perturbation with strength $\delta=0.25$ was superimposed on a Lamb–Oseen vortex with a standard deviation $\sigma=2^{1/2}$ and unit circulation ($\Gamma=1$) for a high Reynolds number $Re=10^4$. The number of elements in the domain at $t=0$ was $N_0=6677$ but during the simulation due to remeshing and vorticity expansion it was increased so that at $t=1425$ there were $N_f=8330$ points. Figure 17 illustrates the evolution of the initial vorticity field.

The first frame shows a dipolar state: the distorted positive Lamb–Oseen vortex was flanked by a smaller and weaker vortex of opposite sign. Due to the difference in intensities, the shear exerted by the dominant vortex was stronger so the satellite stretched and orbited counterclockwise around the centre while having a clockwise rotation itself. This meant that its angular velocity was slower than that of the base vortex, which helped the axisymmetrization process. At the start, the strong vortex was egg shaped but during the evolution its free elliptical circumference bent to one side and at some point, due to the difference in rotation speeds, was squeezed between the centre and the encircling vortex, as seen in frame 7 of Figure 17 ($T=400$). Then the vorticity was compressed into a filament that decayed, turning the Lamb–Oseen vortex back to its base state. The axisymmetrization of vortices through filamentation was numerically studied by Melander *et al.* [31]. They showed that the edges of a smoothed Kirchhoff vortex developed during its rotation into filaments that broke the elliptical symmetry of the vortex field and activated an 'axisymmetrization principle' on the structure. In the current study, the core of the main vortex

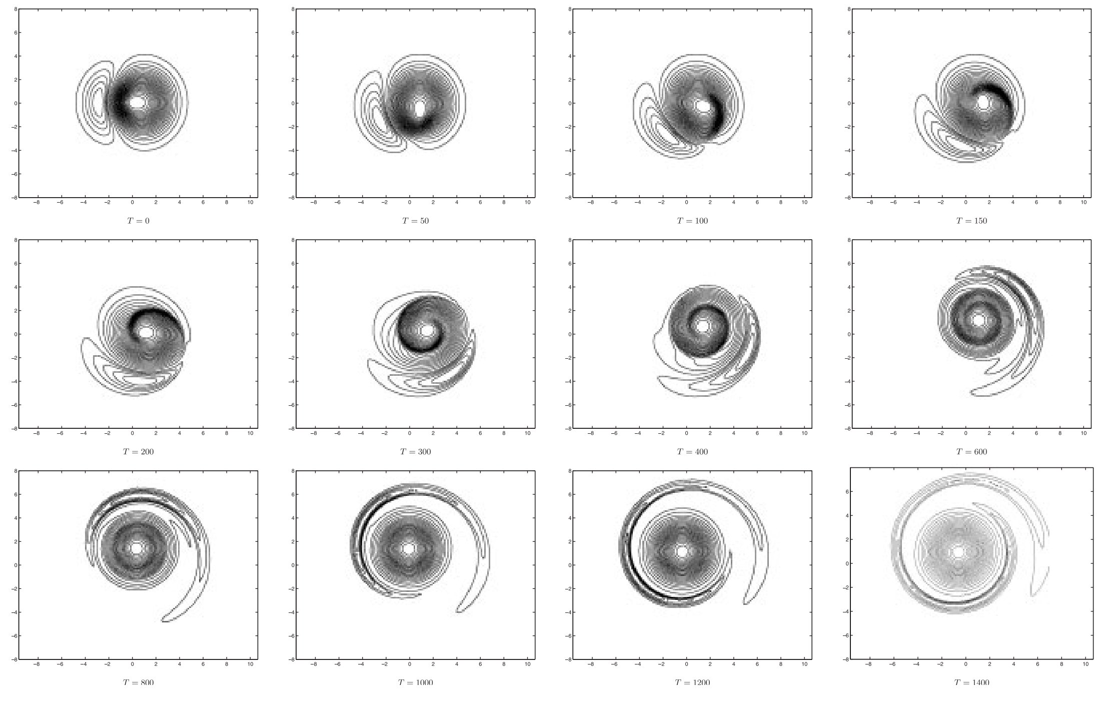

Figure 17. Evolution of a Lamb–Oseen vortex with a dipolar perturbation: $m=1$, $Re=10^4$ and $\delta=0.25$.

started to become circular even before the formation of the filament but axisymmetry was not reached until its total decay.

It is well known that an unperturbed Gaussian monopole pirouettes with zero linear momentum. However, in this case the shear exerted by the companion vortex while stretching affected the position of the principal one as shown in Figure 18. Primarily, the base vortex drawn by the weak one travelled downward at a fast pace, then after a reorganization phase rotated slowly on a circular course. This movement matched that of the satellite, which first moved down on an elliptical trajectory due to the shape of the main vortex, then assumed a circular path due to the effect of the axisymmetrization mechanism on the centre before being drawn closer to it as the

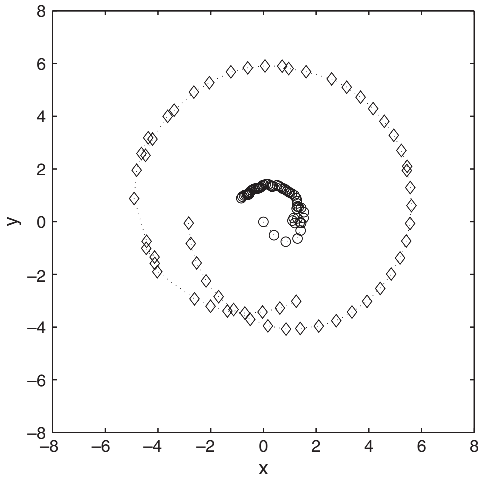

Figure 18. Position of maximum ω in the main vortex ($\circ$) and minimum ω in the satellite ($\diamond$) during evolution of the perturbed Lamb–Oseen ($m = 1$) at different time steps $\Delta t = 25$.

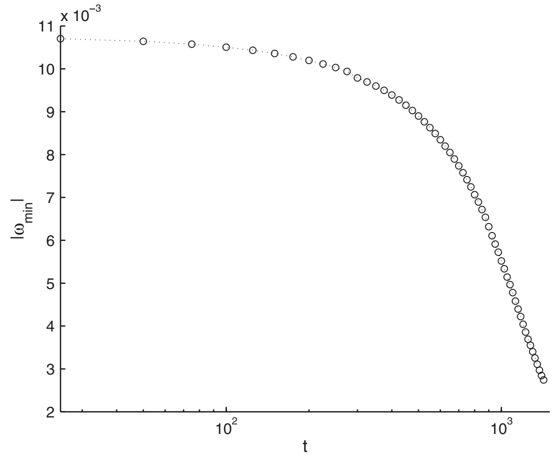

Figure 19. Decay of the satellite ($m = 1$) on a logarithmic time scale.

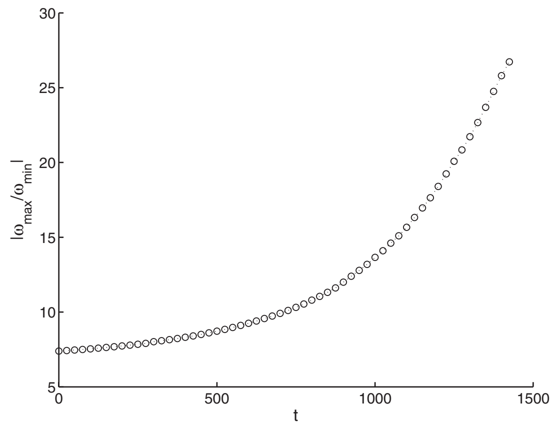

Figure 20. Absolute value of $\omega_{\max}/\omega_{\min}$ during evolution of the perturbed vortex, $m=1$, $\Delta t=25$.

filament separating both vortices diminished and disappeared. This is manifested by the sudden displacement of the centre of the satellite as observed on the left-hand side of Figure 18.

The change in trajectory and speed of the positive vortex might be due to the decreasing intensity of the small vortex during its revolution. The decay of the negative vorticity is displayed in Figure 19 on a logarithmic time scale, which helps to see clearly the early stages of the process. Consequently, its effect is apparent from the development of the ratio $\omega_{\max}/\omega_{\min}$ in Figure 20. Initially, the magnitude of the side vortex was strong enough to draw its partner towards it ($|\omega_{\max}/\omega_{\min}|=7.4$), and the slow diminution phase observed in the decay curve coincides with the early fast displacement of the main vortex. But as the slope of the declination steepened, the effect of the negative satellite weakened, which slowed down the rotation of the central vortex. Finally, it is worth remarking that during the evolution the satellite degenerated without mixing with the Lamb vortex and that, based on its rotation and decay rate, it is expected that it will stretch even further, becoming a thin filament that will gradually vanish due to viscous effects, leaving the axisymmetric monopole as the end state.

5. CONCLUSIONS

A novel approach for the computation of velocities in two-dimensional Lagrangian vortex methods has been presented and assessed. The method is a general approach to the solution of partial differential equations on highly distorted meshes that can be solved using a standard iterative solver applied to a sparse, symmetric system of equations. The method gives computation times that scale as $N^{1.23}$ in computing unsteady flows and demonstrates good error behaviour.

The utility of the method was demonstrated by computing the evolution of a Lamb–Oseen vortex subject to a dipolar perturbation. In this case, the base vortex decays back to its original state, while its satellite rotates around it stretching into a filament.

ACKNOWLEDGEMENTS

The work presented in this paper has benefitted from discussions with GertJan van Heijst and Lorena Barba. The development of the code described in this paper relied heavily on free software. The authors wish to acknowledge, in particular, the GNU Triangulated Surface Library of Popinet [32] and the GNU Scientific Library [33].

REFERENCES

1. Leonard A. Vortex methods for flow simulation. *Journal of Computational Physics* 1980; **37**:289–335.
2. Sarpkaya T. Computational methods with vortices—the 1988 Freeman scholar lecture. *Journal of Fluids Engineering* (ASME) 1989; **111**:5–52.
3. Greengard L, Rokhlin V. A fast algorithm for particle simulations. *Journal of Computational Physics* 1987; **73**:325–348.
4. Carrier J, Greengard L, Rokhlin V. A fast adaptive multipole algorithm for particle simulations. *SIAM Journal on Scientific and Statistical Computing* 1988; **9**(4):669–686.
5. Strickland JH, Amos DE. Fast solver for systems of axisymmetric ring vortices. *AIAA Journal* 1992; **30**(3): 737–748.
6. Christiansen JP. Numerical simulation of hydrodynamics by the method of point vortices. *Journal of Computational Physics* 1973; **13**(3):363–379.
7. Sweeney C, Meskell C. Fast numerical simulation of vortex shedding in tube arrays using a discrete vortex method. *Journal of Fluids and Structures* 2003; **18**(5):501–512.
8. Braun J, Sambridge M. A numerical method for solving partial differential equations on highly irregular evolving grids. *Nature* 1995; **376**:655–660.
9. Sukumar N. Voronoi cell finite difference method for the diffusion operator on arbitrary unstructured grids. *International Journal for Numerical Methods in Engineering* 2003; **57**:1–34.
10. Baty RS, Wolfe WP. Least-squares solutions of a general numerical method for arbitrary irregular grids. *International Journal for Numerical Methods in Engineering* 1997; **40**:1701–1717.
11. Carley M. A triangulated vortex method for the axisymmetric Euler equations. *Journal of Computational Physics* 2002; **180**:616–641.
12. Marshall JS, Grant JR, Gossler AA, Huyer SA. Vorticity transport on a Lagrangian tetrahedral mesh. *Journal of Computational Physics* 2000; **161**(1):85–113.
13. Russo G, Strain JA. Fast triangulated vortex methods for the 2D Euler equations. *Journal of Computational Physics* 1994; **111**:291–323.
14. Huyer SA, Grant JR. Solution of two-dimensional vorticity equation on a Lagrangian mesh. *AIAA Journal* 2000; **38**(5):774–783.
15. Saffman PG. *Vortex Dynamics*. Cambridge Monographs on Mechanics and Applied Mathematics. Cambridge University Press: Cambridge, 1992.
16. Barrett R, Berry M, Chan TF, Demmel J, Donato J, Dongarra J, Eijkhout V, Pozo R, Romine C, Van der Vorst H. *Templates for the Solutions of Linear Systems*: *Building Blocks for Iterative Methods*. SIAM: Philadelphia, PA, U.S.A., 1994.
17. Russo G. A deterministic vortex method for the Navier–Stokes equations. *Journal of Computational Physics* 1993; **108**:84–94.
18. Marshall JS, Grant JR. A Lagrangian vorticity collocation method for viscous, axisymmetric flows with and without swirl. *Journal of Computational Physics* 1997; **138**:302–330.
19. Williamson JH. Low storage Runge–Kutta schemes. *Journal of Computational Physics* 1980; **35**:48–56.
20. Perlman M. On the accuracy of vortex methods. *Journal of Computational Physics* 1985; **59**:200–223.
21. Sir Horace Lamb. *Hydrodynamics* (6th edn). Cambridge Mathematical Library, Cambridge University Press: Cambridge, 1997.
22. Kloosterziel RC, van Heijst GJF. An experimental study of unstable barotropic vortices in a rotating fluid. *Journal of Fluid Mechanics* 1991; **223**:1–24.
23. Carton XJ. On the merger of shielded vortices. *Europhysics Letters* 1992; **18**(8):697–703.
24. Flor JB, Govers WSS, van Heijst GJF, van Sluis R. Formation of a tripolar vortex in a stratified fluid. *Applied Scientific Research* 1993; **51**:405–409.

25. Flor JB, van Heijst GJF. Stable and unstable monopolar vortices in a stratified fluid. *Journal of Fluid Mechanics* 1996; **311**:257–287.
26. Rossi LF, Graham-Eagle J. On the existence of two-dimensional, localized, rotating, self-similar vortical structures. *SIAM Journal on Applied Mathematics* 2002; **62**(6):2114–2128.
27. Rossi LF, Lingevitch JF, Bernoff AJ. Quasi-steady monopole and tripole attractors for relaxing vortices. *Physics of Fluids* 1997; **9**(8):2329–2338.
28. Barba LA, Leonard A. Emergence and evolution of tripole vortices from net-circulation initial conditions. *Physics of Fluids* 2007; **19**(017101):1–16.
29. Barba LA. Nonshielded multipolar vortices at high Reynolds number. *Physical Review E* 2006; **73**(6):065303(R).
30. Türk L, Coors D, Jacob D. Behavior of wake vortices near the ground over a large range of Reynolds numbers. *Aerospace Science and Technology* 1999; **3**(2):71–81.
31. Melander MV, McWilliams JC, Zabusky NJ. Axisymmetrization and vorticity-gradient intensification of an isolated two-dimensional vortex through filamentation. *Journal of Fluid Mechanics* 1987; **178**:137–159.
32. Popinet S. GTS: GNU Triangulated Surface Library. http://gts.sourceforge.net/, 2000–2004.
33. Galassi M, Davies J, Theiler J, Gough B, Jungman G, Booth M, Rossi F. *GNU Scientific Library Reference Manual*. Network Theory Ltd.: Bristol, U.K., 2005.