
Article:
Peterson, R.C., Jimack, P.K. and Kelmanson, M.A. (1999) The solution of two-dimensional free-surface problems using automatic mesh generation. International Journal for Numerical Methods in Fluids, 31 (6). pp. 937-960. ISSN: 0271-2091

DOI:10.1002/(SICI)1097-0363(19991130)31:6<937::AID-FLD906>3.0.CO;2-P

The Solution of Two-Dimensional Free-Surface Problems Using Automatic Mesh Generation

Richard C Peterson[1,2], Peter K Jimack[1] and Mark A Kelmanson[2]

[1]School of Computer Studies,
University of Leeds,
Leeds LS2 9JT,
England.

[2]Department of Applied Mathematics,
University of Leeds,
Leeds LS2 9JT,
England.

Submitted to *Int J Num Meth Fluids*
19th June 1998.

AMS Subject classifications: 65M50, 65M60, 76D07, 76M10.

Key words:
Free-surface flow, mixed finite-element method,
Stokes flow, surface tension, automatic mesh generation, viscous sintering.

Running head:
FREE-SURFACE AUTOMATIC MESH GENERATION

Summary

We describe a new method for the iterative solution of two-dimensional free-surface problems, with arbitrary initial geometries, in which the interior of the domain is represented by an unstructured, triangular, Eulerian mesh and the free surface is represented directly by the piecewise-quadratic edges of the isoparametric quadratic-velocity, linear-pressure Taylor-Hood elements. At each time step, the motion of the free surface is computed explicitly using the current velocity field and, once the new free-surface location has been found, the interior nodes of the mesh are repositioned using a continuous-deformation model which preserves the original connectivity. In the event that the interior of the domain must be completely remeshed, we employ a standard Delaunay triangulation algorithm which leaves the initial boundary discretisation unchanged.

Our algorithm is validated via the benchmark viscous-flow problem of the coalescence of two infinite cylinders of equal radius, in which the motion is due entirely to the action of capillary forces on the free surface. This problem has been selected for a variety of reasons: the initial and final (steady-state) geometries differ considerably; in the passage from the former to the latter, large free-surface curvatures — requiring accurate modelling — are encountered; an analytic solution is known for the location of the free surface, and; there exists a large body of literature on alternative numerical simulations.

A novel feature of the present work is its geometric generality and robustness; it requires *a priori* knowledge of neither the evolving domain geometry nor the solution contained therein.

1 Introduction

An understanding of the dynamics of free-surface phenomena is of significant importance in a wide variety of scientific and engineering disciplines [1, 2] and, to this end, the broad subject area has yielded to a number of experimental [3, 4], theoretical [5, 6] and numerical [7, 8] investigations. It is with reference to the last of these that the present work is motivated. Specifically, our objective has been to develop an algorithm which is able to deal both accurately and robustly with the apparently-conflicting factors of: the need for geometric generality — including problems comprising regimes of widely-varying free-surface curvature — and; the avoidance of excessive user interaction and problem-specific code. Our desire for generality is driven by the immense range of free-surface phenomena arising in practical applications: e.g. overturning waves [9], droplet and bubble formation [10, 11], viscous sintering [12, 13, 14] and coating flows [15, 16, 17]. The disparate nature of the problems modelled in each of the above areas has led to the successful development of wide variety of simulation software, usually comprising variations on either the boundary-element method (BEM) [12, 13] or the finite-element method (FEM) [18, 19].

Where only the motion of the free-surface is required, the BEM [20] is often preferred, since it results in systems of equations involving only free-surface unknowns. Where an interior flow solution is also required, the advantages of the BEM are less obvious, since the interior solution can only be post-computed at considerable expense once the free-surface motion has been found. Furthermore the BEM Green's-function-based formulation is applicable to only those problems for which the governing PDE(s) and boundary conditions admit a suitable transformation into integral form. Given our objective of algorithmic generality, we therefore opt to use the FEM.

In the context of free-surface simulation via the FEM, two main interrelated practical aspects are evident: first, an interior mesh must be maintained throughout so that the interior flow may be calculated; second, as the domain evolves, an efficient means must be found to modify this mesh at each iteration. Typically, existing FEM strategies for achieving this employ a fixed mesh topology together with some form of continuous mesh deformation [19, 21, 22]. Both the mesh topology and the particular form of continuous deformation to be employed must be chosen in advance for every problem. Continuous-deformation techniques have the important practical advantage that, where relevant, the effects of motion of the mesh can be accounted for analytically thus avoiding the need to perform interpolation of the solution between meshes at each time step [23]. Moreover, coupled with the problem of updating the mesh is that of updating the representation of the free surface itself which, ideally, should be achieved through the use of the kinematic boundary condition alone. Frequently, other aspects, such as conservation of mass [22], are also explicitly taken into consideration, despite the fact that this presupposes incompressibility, thereby eroding the generality of the method.

Within our algorithm the free surface is represented in a natural way by the edges of the isoparametric elements forming the boundary, and evolution of this free surface is driven purely by the kinematic boundary

condition. At each iteration, the perturbation to the current boundary is used to modify the interior mesh as though it were deforming in a continuous fashion.

The quality of this evolving mesh is continually monitored using a variety of geometric indicators and, when mesh quality falls below prescribed thresholds, the mesh is regenerated anew, preserving only the locations of the boundary nodes. Full details relating to the initial generation, and subsequent modifications, of the mesh are presented in Section 2.

Throughout the remainder of the paper, application of our method is facilitated by focussing on a specific two-dimensional problem, described in Section 3, namely the Stokes-flow coalescence of two parallel infinite fluid cylinders of unit radius, in which the motion is due entirely to the action of capillary forces at the free surface. The problem chosen has been the subject of a number of theoretical [24] and numerical [13, 14, 18] investigations, and so many results are available for the purpose of comparison. In Section 4 we describe in detail the flow solver used for the test problem, and the results obtained are discussed with regard to both accuracy and efficiency.

2 Mesh generation

The primary difficulty in applying the finite element method to free-surface problems is the need to specify a procedure for constructing meshes over a domain, the shape of which is *a priori* unknown. For those FEM problems which involve free surfaces that move little during the computation, an initial mesh can be updated simply by using a continuous-deformation algorithm which maintains the same mesh connectivity throughout the problem. In preserving the connectivity, we also preserve the sparsity patterns of the various matrices that arise in the discretisation and solution of the problem e.g. stiffness and mass matrices, Jacobians, preconditioners etc. with obvious efficiency advantages. Although for more general problems we cannot avoid the need to remesh completely, our aim is to do so as infrequently as possible.

To set the methods described below in context, Figure 1 is included, illustrating the semi-explicit scheme employed in the current work. The use of an explicit free-surface update step in this scheme places limitations on the size of time step which may be employed if stability is to be ensured. For this reason fully-implicit methods are often preferred. There are however many problems for which the time step restrictions are not so severe and in these cases semi-explicit methods can be cost-effective, particularly where time accuracy is important. The advantages of employing the semi-explicit method are: that the systems of differential algebraic equations (DAEs) which must be solved at each time step are linear — if the underlying CFD problem is itself linear; that the resulting matrices are symmetric; and that the locations of the free-surface nodes need not be considered as variables during the flow solution phase — resulting in a substantial reduction in computational cost. While we restrict our discussion here to the semi-explicit case, many of the techniques described below are equally applicable to fully implicit schemes such as those employed by the DAE solvers (such as DDASPK [25]) commonly used to solve the stiff systems of equations frequently arising from incompressible-flow problems.

Mesh quality

For those problems in which the free-surface boundary conditions depend on the shape of the free-surface, e.g. when surface tension is involved, the overall accuracy of the solution depends critically on the resolution of the mesh at the boundary. This is because the accuracy of the free-surface boundary conditions depends directly on the accuracy of the free-surface representation. Thus the convergence rate of the solution is limited to the rate of convergence of the discrete boundary conditions, potentially presenting great difficulties due to practical constraints on computational resources. Fortunately however, for 2-D problems, while the number of interior unknowns increases quadratically with resolution, the number of boundary unknowns increases only linearly. Thus the use of a finer discretisation on the boundary than in the interior of the domain, may be feasible, whereas uniform refinement, to obtain boundary conditions of comparable accuracy, would not be. Unstructured meshes of six-node, isoparametric, quadratic triangular elements [26] are employed,

allowing us to obtain far more accurate representations of curved boundaries than could be achieved using linear elements.

The first step in meshing a domain Ω is the discretisation of the boundary $\partial\Omega$. For simplicity the discussion here is restricted to the case in which the boundary can be represented by a closed curve parameterised by arc length s. For a piecewise-quadratic boundary representation it is natural to wish to discretise so that the derivative of curvature with respect to arc length is taken into account. Unfortunately, finite numerical precision makes this approach problematic [13], and we therefore employ boundary discretisations chosen such that for each element edge $\partial\Omega^e$ in $\partial\Omega$, the curvature $k(s)$ is equidistributed according to

$$\int_{\partial\Omega^e} k ds \leq k_{tol}, \tag{1}$$

where k_{tol} is a prescribed parameter. We further require that edge length be constrained so that

$$\int_{\partial\Omega^e} ds \leq h_{max} \tag{2}$$

where h_{max} is a prescribed parameter. In addition, we impose the following constraints on boundary edge length:

$$\frac{1}{\alpha}\int_{\partial\Omega^{e+1}} ds \leq \int_{\partial\Omega^e} ds \leq \alpha \int_{\partial\Omega^{e+1}} ds \tag{3}$$

and

$$\frac{1}{\alpha}\int_{\partial\Omega^{e-1}} ds \leq \int_{\partial\Omega^e} ds \leq \alpha \int_{\partial\Omega^{e-1}} ds, \tag{4}$$

where α is a *mesh smoothness* parameter, chosen to prevent the ratio of lengths of adjacent boundary edges being too large. Our numerical experiments suggest that a value of $\alpha = 2.5$ is satisfactory for most problems.

Interior meshes are generated using Triangle [27], Shewchuk's automatic two-dimensional Delaunay mesh generating package. Based upon Ruppert's Delaunay refinement algorithm [28], Triangle can selectively refine an initial mesh, deciding whether to split each triangle according to a set of area constraints associated with the triangles of the original mesh: in so doing it guarantees to produce a mesh with no interior angle less than 20.7^o. This property is, however, potentially compromised when Triangle is used to generate a boundary conforming mesh, i.e. is not allowed to split boundary edges. To prevent this difficulty arising, in particular when boundary discretisations with large variations in edge length are employed, the interior mesh must be *graded* so that edge length does not differ too greatly between neighbouring elements. This is achieved by associating with each element i, of some initial mesh, a length l_i given by

$$l_i = \min\left(h_{max}, \min_j (h_j \left|\mathbf{m}_j - \mathbf{c}_i\right|)\right), \tag{5}$$

where $\mathbf{m}_j$ is the location of the midpoint of edge j and h_j is its length, where $\mathbf{c}_i$ is the centroid of element i, and where we minimize over the boundary nodes $j = 1...N_B$. This translates into a corresponding maximum area constraint a_i given by

$$a_i = \frac{\sqrt{4}}{3} {l_i}^2. \tag{6}$$

In other words, a_i is chosen to be the area of the equilateral triangle with side l_i. While the choice of the above grading is entirely motivated by the need to ensure adequate mesh quality, i.e. for essentially geometric reasons, in practice for surface-tension-driven flows at least, the patterns of local refinement that arise are similar to those that might be chosen when adaptively refining with respect to stress gradients.

Time-stepping

For free-surface fluid-flow problems, the motion of the free surface is described by the kinematic boundary condition. With a first-order explicit free-surface advection scheme, such as the one described below, the kinematic boundary condition, together with the solution at the current time level and the currently-selected time step, are used to extrapolate the new location of the free surface from the old.

At the end of a typical time step, the mesh of the interior of the domain is updated using an elastic-mesh model based upon that described by Lynch in [29]. This involves solving a linear-elasticity problem for interior-vertex displacements, using boundary-vertex displacements as boundary conditions. The linear elasticity model takes the form of a Poisson problem

$$\nabla \cdot (\mathbf{C} \nabla \mathbf{x}) = \mathbf{f}, \tag{7}$$

where $\mathbf{x}$ is a vector of interior vertex displacements, $\mathbf{f}$ a vector representing a body force and $\mathbf{C}$ is the elasticity tensor. For simplicity we have taken $\mathbf{f}$ to be zero and $\mathbf{C}$ to be the identity tensor.

Discretising an appropriate weak-form of (7) using linear elements by applying the Galerkin method, a symmetric positive-definite system of linear equations is obtained. This auxiliary system is typically much smaller than the system arising from the main problem and need only be solved approximately. This can be achieved cost-effectively using an iterative method; in practice the cost of such a solution is small, typically much less than 0.1% of the cost of the solution of the main system of equations. The perceived advantage of Lynch's method is that, for sufficiently small time steps at least, the continuous nature of the boundary displacement field results in a continuous displacement field for the interior nodes and thus tangling of the mesh, due to the crossing over of vertices into adjacent elements, cannot occur.

To allow ourselves further generality in our experiments a simple pointwise weighted-Jacobi mesh-smoothing operator is applied to the mesh after each elastic-mesh solve. This amounts to updating the position $\mathbf{r}_i$ of each interior node according to the following iterative scheme

$$\mathbf{r}_{i+1} = (1 - \omega)\mathbf{r}_i + \frac{\omega}{N} \sum_{j=1}^{N} \mathbf{r}_j, \tag{8}$$

where j sums over the N neighbouring vertices of vertex i, and ω is a relaxation parameter, typically taken to be 0.1. The smoothing operator (8) is applied a small number of times (≤ 40) after every elastic-mesh solve, rather than iterating to full convergence, the Jacobi-type approach being preferred since it does not introduce any effects dependent upon node ordering.

Remeshing criteria

Remeshing of the domain is inefficient for two reasons. Firstly, it forces the DAE or ODE solver to restart, typically necessitating a reduction in time step. Secondly, and equally importantly, for many PDEs it forces us to do expensive and potentially inaccurate interpolations of solutions between meshes. In order to minimize the frequency of remeshes a strategy of *periodic incremental boundary modification* is adopted, involving the splitting and/or merging of boundary edges as necessary. Whenever boundary edges are split or merged at least part of the domain must be remeshed (for simplicity we always remesh the entire domain) and consequently our algorithm is designed to postpone any free-surface edge merge or split operations for as long as is reasonable, so as to be able to perform a set of these operations at the same time. Postponing an edge merge operation is not so problematic since it merely reduces efficiency. The splitting of edges is, however, more urgent, since we wish always to keep the error in the solution bounded. An attempt is made to reconcile these conflicting requirements by adopting the strategy of *merging edges cautiously* but *splitting edges aggressively*, as reflected in the choices of the constants discussed in the remainder of this section.

A full remesh is carried out when dictated by one or more of the following criteria:

1. a boundary edge is too long, i.e.

$$h_i > h_{max}, \tag{9}$$

   where h_i is the length of edge i;

2. the integral of the modulus of the curvature along an edge is too great, i.e.

$$\int_{\partial\Omega^i} |k| \, ds > k_{tol}; \tag{10}$$

3. the minimum internal angle has fallen below a prescribed tolerance, i.e.

$$\theta_{min} < \theta, \tag{11}$$

   where θ_{min} is the minimum internal angle in the current mesh, and θ is the prescribed tolerance;

4. an edge node is located too far from its corresponding edge midpoint, i.e.

$$|\mathbf{e}_i - \mathbf{m}_i| > \beta \, h_i, \tag{12}$$

   where $\mathbf{e}_i$ is the position of edge node i, $\mathbf{m}_i$ is the midpoint of the chord associated with the edge and β is a parameter chosen so as to bound the displacements of free-surface edge nodes from edge midpoints.

The last two criteria reflect constraints that must be applied if the optimum asymptotic rate of convergence of the FEM is to be achieved [30]. In particular, the third reflects the constraint that the maximum interior angle in any element must be bounded away from 180^o. In practice we instead bound the minimum interior

angle away from zero, taking $\theta = 10^o$, in order to avoid problems that can arise when small angles occur in elements adjacent to the free-surface.

The last criterion is used to trigger a remesh operation whenever a free-surface edge node is found to be displaced too far from its corresponding edge midpoint. In such (rare) situations the edge node must be *adjusted* i.e. moved closer to the edge midpoint. A value of $\beta = 1.1$ has been found to be satisfactory in practice.

Remeshing

Once the decision to perform a remesh has been taken, any necessary splitting, merging and adjustment of boundary edges takes place in three successive phases: first any merges are performed, then any splits and finally any adjustments. Note that an edge resulting from a merge operation in the first phase may be split in the second phase. The criteria used to decide whether to split or merge edges are detailed below.

Criteria for merging boundary edges

The adjacent edges $\partial\Omega^i$ and $\partial\Omega^{i+1}$ are merged in either of the following circumstances:

1. the curvature of two adjacent edges is of the same sign and

$$\int_{\partial\Omega^i} |k|\, ds + \int_{\partial\Omega^{i+1}} |k|\, ds < \mu\ k_{tol}; \tag{13}$$

   where the constant $\mu = 0.7$ is chosen so that edges are merged cautiously.

2. the combined lengths of a pair of adjacent edges is below a given tolerance i.e.

$$\int_{\partial\Omega^i} ds + \int_{\partial\Omega^{i+1}} ds < 2\ h_{min}, \tag{14}$$

   where h_{min} is a prescribed parameter chosen to limit the minimum boundary edge length.

Criteria for splitting boundary edges

There are three circumstances in which a boundary edge must be split:

1. the integral of the modulus of the curvature along the side becomes too great, i.e.

$$\int_{\partial\Omega^e} |k|\, ds > \delta\ k_{tol}, \tag{15}$$

   where the constant $\delta = 0.9$ is selected so that edges are split aggressively, preventing forced remeshes from occurring too often.

2. an edge is too long, i.e.

$$h_i > \delta\, h_{max}; \tag{16}$$

3. the ratio of adjacent edge lengths is too large, i.e.

$$h_i > \rho\, min(h_{i+1}, h_{i-1}). \tag{17}$$

The value of the constant ρ is chosen as a compromise between maintaining acceptable grading of the mesh's boundary and allowing the merging of edges to occur unhindered. The interaction of criteria (13) and (15) controls the level of refinement of the mesh boundary, k being the error indicator employed here. In general problem specific error indicators could replace k.

Procedures for splitting, merging and adjustment of edges

Once the decision to split or merge edges has been taken, the question arises of how this can be done with the introduction of the minimum error. The case of splitting an edge might appear to be the least problematic; in principle any piecewise-quadratic boundary edge can be split into two exactly equivalent piecewise-quadratic edges by appropriate choices of the new boundary node locations. Unfortunately the isoparametric constraint that edge nodes must lie close to edge midpoints means that this cannot, in general, be achieved. Similar problems arise when edges must be merged. In order to prevent velocity transients being introduced, it is important to ensure that the tangents at the ends of the new edge are preserved. In general however, one cannot expect to be able to choose a new midpoint that simultaneously lies on the edge bisector, preserves both end-point tangents and conserves mass.

With these difficulties in mind, the following compromise strategies are adopted when locating edge nodes. When merging edges we proceed by fitting a quadratic Lagrange interpolation curve through the three vertices making up the pair of edges. We then find the intersection of this curve with the perpendicular bisector of the chord joining the endpoints of the new edge, and take this to be the new edge node. Similarly, when splitting an edge, we employ the current Lagrange interpolation curve defined on the edge, and find its intersection with the perpendicular bisector of the chord joining the endpoints, taking this to be the new vertex. The two new edge nodes are located by similarly bisecting the two halves of the old edge. Finally, adjustment of edges is performed by finding the intersection of the current Lagrange interpolation curve with the perpendicular bisector of the chord joining the endpoints of the edge and taking this as the new edge-node position.

Table 1 summarizes the various constants employed in the automatic mesh adaptation scheme. Suggested ranges of values for each constant are listed in the figure, together with the values we typically employ. While the number of constants employed appears at first sight to be a drawback, in practice the values of most are not critical to the operation of the method.

3 A model problem

In order to validate our method we consider here one of the few free-surface problems for which an exact analytic solution is known — the Stokes-flow coalescence of two parallel, infinite cylinders of unit radius. Hopper [24] describes an analytic solution for the evolution of the boundary for this problem. His method does not however give quantitative information about the tangential velocity at the surface nor about the flow in the interior of the domain. The problem has become a standard benchmark problem for free-surface methods and has been solved previously using the BEM [13, 14], the FEM on a continuously-deforming mesh [18] and the FEM on a continuously-deforming mesh with local mesh repair [31]. An important practical advantage of employing a Stokes flow as a test problem is that it frees us, at this stage, from having to consider methods for the interpolation of solutions between meshes when remeshing.

In the model problem two parallel cylinders of unit radius and infinite length are brought together so that at time $t = 0$ they make contact. As the two cylinders coalesce under the influence of surface tension, a neck forms between the two cylinders, the curvature of which is initially unbounded but which decreases rapidly in magnitude as the configuration evolves. We avoid the difficulties associated with the initial singularity, at $t = 0$, by taking as our starting point the configuration that occurs at the dimensionless time $t = 0.2825$ (corresponding to a value $\nu = 0.7$ in van der Vorst's formulation [14]). Taking this as our initial configuration, requires us to discretise accurately a boundary along which the magnitude of the curvature varies between approximately 1 and 100, necessitating a variation in boundary edge lengths of approximately two orders in magnitude. Figure 2 shows three of the initial meshes employed to represent the domain at $t = 0.2825$. The grading of the mesh described in Section 2 can clearly be observed in the figure.

Standard six-node Taylor-Hood elements are employed, with pressure degrees of freedom at the vertices and velocity degrees of freedom at all six nodes, giving a quadratic velocity interpolant and a linear pressure interpolant. The Taylor-Hood element is used since it is known to be stable in the LBB sense [32] for Stokes and Navier-Stokes problems.

Equations and boundary conditions

The dimensional Navier-Stokes equations for an incompressible Newtonian fluid in the absence of body forces take the form

$$\rho \left[\frac{\partial \widetilde{\mathbf{u}}}{\partial \widetilde{t}} + (\widetilde{\mathbf{u}}.\nabla)\widetilde{\mathbf{u}} \right] = \eta \nabla^2 \widetilde{\mathbf{u}} - \nabla \widetilde{p} \tag{18}$$

and

$$\nabla . \widetilde{\mathbf{u}} = 0 \tag{19}$$

where η and ρ are respectively the dynamic viscosity and density of the fluid. Following [18], we non-dimensionalise the problem by choosing a length scale L_0 and defining characteristic scales for velocity U_0,

time T_0 and stress σ_0, using:

$$U_0 = \frac{\gamma}{\eta} \tag{20}$$

$$T_0 = \frac{\eta L_0}{\gamma} \tag{21}$$

$$\sigma_0 = \frac{\gamma}{L_0} \tag{22}$$

where γ is the surface tension. Thus the dimensionless equations

$$Su\left[\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}.\nabla)\,\mathbf{u}\right] = \nabla^2\mathbf{u} - \nabla p \tag{23}$$

and

$$\nabla.\mathbf{u} = 0 \tag{24}$$

are obtained, where Su is the dimensionless Suratman number,

$$Su = \frac{\rho\gamma L_0}{\eta^2}. \tag{25}$$

When the Suratman number is vanishingly small the Stokes approximation is applicable and the equations reduce to the form:

$$\nabla^2\mathbf{u} - \nabla p = 0 \tag{26}$$

$$\nabla.\mathbf{u} = 0. \tag{27}$$

The motion of the boundary is given by the kinematic boundary condition, i.e. a material point on the boundary remains on the boundary. Thus, if the free surface is represented by a curve $\mathbf{s}(s)$ parametrized by arc length s, the kinematic boundary condition can be expressed as

$$\mathbf{n}\cdot\mathbf{u} = \mathbf{n}\cdot\dot{\mathbf{s}}\ , \tag{28}$$

where $\mathbf{n}$ is the outward free-surface normal. The absence of a material derivative in the momentum equation (26) allows the problem to be solved using quasi-steady-state methods. Thus only a single linear, steady-state fluid dynamics problem need be solved at each time step to give the velocities required to update the free-surface location, which can then be done explicitly using the kinematic boundary condition.

For a free-surface problem in which the viscosity of the surrounding fluid is vanishingly small, the externally imposed tangential stress can be taken to be zero. The normal stress on the free surface has two components, the first of which corresponds to the pressure of the surrounding fluid (here taken to be zero), while the second is only present when the free surface has non-zero surface tension and curvature associated with it. In these circumstances the stress on the free surface is given by

$$\hat{\sigma} = -\gamma k\mathbf{n}, \tag{29}$$

where k is the mean curvature of the free surface.

Finite element implementation

Following [33] the viscous term in (26) can be rewritten in the equivalent stress-divergence form to give

$$\nabla^2\mathbf{u} + \nabla(\nabla.\mathbf{u}) - \nabla p = 0. \tag{30}$$

If the Galerkin method [26] is now applied to the appropriate weak form of equation (30) the discrete momentum equations take the form

$$\int_\Omega \nabla q_i \cdot \mathbf{T}\, d\Omega = -\int_{\partial\Omega} q_i \hat{\sigma}\, ds, \tag{31}$$

where q_i is the corresponding quadratic test function, $\hat{\sigma}$ is the imposed stress and $\mathbf{T}$ is the stress tensor for a Stokes flow:

$$\mathbf{T} = T_{ij} = -p\delta_{ij} + \left(\frac{\partial u_j}{\partial x_i} + \frac{\partial u_i}{\partial x_j}\right). \tag{32}$$

With the stress boundary conditions given by (29) the right-hand side of the momentum equation has the form

$$\int_{\partial\Omega} q_i \gamma k \mathbf{n}\, ds. \tag{33}$$

Unfortunately the direct computation of the curvature of a boundary represented by a piecewise-discontinuous curve is difficult to achieve in an accurate and reliable manner, chiefly due to the problems of differentiation in finite-precision arithmetic. Computation is facilitated by using an equivalent form of (33) obtained by integration by parts [34]. Since

$$k\mathbf{n} = \frac{d\mathbf{t}}{ds}, \tag{34}$$

where $\mathbf{t}$ is the tangent, for a free-surface edge $\vec{AB}$ we have:

$$\int_A^B q_i \gamma k \mathbf{n}\, ds = \int_A^B q_i \gamma \frac{d\mathbf{t}}{ds}\, ds = [q_i \gamma \mathbf{t}]_A^B - \int_A^B \gamma \mathbf{t} \frac{dq_i}{ds}\, ds. \tag{35}$$

This new boundary integral can be evaluated exactly using a 3-point quadrature rule, where $\mathbf{t}$ is given by

$$\mathbf{t} = \sum_{i=1}^{3} \mathbf{s}_i \frac{dq_i}{ds}, \tag{36}$$

where i ranges over the three basis functions active on any particular edge and the $\mathbf{s}_i$ are the positions of the relevant boundary nodes.

The free-surface boundary location is updated using a first-order explicit scheme. The kinematic boundary condition (28) is solved for $\dot{\mathbf{s}}$ using *mass consistent normals*, as described by Gresho *et al.* [35], with the time step being selected using the CFL condition

$$\delta t \leq min\left\{\frac{1}{4}\frac{l_{min}}{|\mathbf{v}_{max}|}, \delta_{max}\right\} \tag{37}$$

where l_{min} is the length of the shortest edge in the mesh, $\mathbf{v}_{max}$ is the largest velocity found from the most recent solution, δ_{max} is a prescribed maximum time step, and the constant $\frac{1}{4}$ arises since the elements are

quadratic. While (37) is generally found to be sufficient, it is sometimes necessary to employ smaller values of the constant in order to ensure stability.

Finally we note that, in isolation, the system of equations (30,27) is singular. With purely natural boundary conditions, the null space corresponds to the rigid-body translations and rotations of the domain. The system is made non-singular by imposing the two additional constraints

$$\int_\Omega \mathbf{u}\, d\Omega = \mathbf{0} \tag{38}$$

and

$$\int_\Omega |\, \mathbf{u} \times \mathbf{r}\, |\; d\Omega = 0, \tag{39}$$

where the right-hand sides can in principle be replaced by any values for the total momentum and angular momentum of the system. In order to apply the constraint (39), an origin must be chosen from which $\mathbf{r}$ is to be measured; this is typically taken to be the center of mass of the system. The main system of equations, arising from the finite-element discretisation of the CFD problem, is modified to reflect these constraints, giving a symmetric-indefinite system of the form

$$\begin{pmatrix} A & B^T & C^T \\ B & 0 & 0 \\ C & 0 & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \mathbf{p} \\ \lambda \end{pmatrix} = \begin{pmatrix} \mathbf{f} \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix} \tag{40}$$

where $\mathbf{f}$ corresponds to the stress boundary conditions and $\lambda = (\lambda_1, \lambda_2, \lambda_3)$, is a vector of Lagrange multipliers.

4 Results and discussion

Perhaps the most important test for any FEM implementation is that the solutions obtained converge as the mesh is refined, and that they do so at the expected rate. For the later stages of the coalescence our method gives excellent results at moderate cost. However, as earlier and earlier starting times are considered, the initial neck curvature increases rapidly, and the size of elements in the neck region must be correspondingly decreased. Thus, eventually, explicit time-stepping schemes become prohibitively expensive. For this reason we have chosen to work only with problems for which the maximum initial curvature is not too great.

In general we can only expect our solutions to converge as fast as our boundary conditions converge. When the correct boundary conditions are known analytically or when there are no large variations in curvature, mesh independence can be conveniently demonstrated through a process of uniform refinement applied to an initial quasi-regular mesh. However when dealing with geometries with large variations in boundary curvature this approach presents difficulties, since a quasi-regular mesh that is sufficiently fine to adequately resolve the boundary (conditions) can easily give rise to a problem that is too large to solve in a reasonable time. With this in mind we have selected a family of five *quasi-similar* meshes by making appropriate choices of the parameters h_{max}, h_{min} and k_{tol}. A selection of these meshes is shown in Figure 2, while Table 2 summarizes the values of the mesh parameters employed, together with the numbers of elements, unknowns and boundary vertices for each initial mesh. Notice that we have chosen to take

$$k_{tol} \propto {h_{max}}^3 \tag{41}$$

in each case. Thus when k_{tol} is halved the number of boundary nodes doubles but the number of interior nodes increases only by a factor of $2^{2/3} \approx 1.59$. The choice of the relationship (41) is made in the light of the theoretical convergence rate of the boundary conditions and allows us to employ much finer boundary discretisations than would otherwise be feasible. In all cases we have taken $h_{min} = 0.0001$. Figure 3 illustrates the evolution of mesh 3 with a series of snapshots taken at the dimensionless times shown after the initial contact of the two cylinders. As can be seen, both the number of elements and their connectivity change considerably as the cylinders merge.

The large, sparse systems of linear equations arising from the FEM are solved using the Orthomin form of the Conjugate Residuals method [36, 37, 38], employing only the simplest of preconditioners, namely diagonal scaling. Since the condition number associated with the problems is typically large (10^6 – 10^8), each linear system must be solved to a high accuracy. Each linear solve is terminated when the L_2 norm of the residual has been reduced below an absolute tolerance of 10^{-10}. The second column in Table 3 lists the numbers of Orthomin iterations required to solve the initial linear problem for each of the meshes considered. It can be seen that the number of iterations is roughly proportional to N, the number of unknowns, for each problem, and since the cost of each iteration is dominated by a single sparse-matrix product, the asymptotic work rate is $O(N^2)$. The third column gives the number of steps required to integrate the system forward to a time $t = 4.0$, the fourth column the number of remesh operations required, and the final column the average

processor time per time step for each run on a 180MHz SGI R5000 workstation. As these figures show, even the moderately sized meshes employed here require considerable computational resources.

In order to reduce the number of iterations required for each linear solve a second-order explicit predictor is computed using previous solutions. The order of this predictor is reduced to one for the second step after a remesh and immediately after a remesh a zero vector is employed as the initial guess. Figure 4 shows how the number of unknowns varies during the solution of the problem when mesh 3 is employed, while Figure 5 shows the number of Orthomin iterations required for each time step, a spike occurring whenever the domain is remeshed. Figure 5 clearly demonstrates the effectiveness of our second-order predictors, in reducing by up to 50% the number of iterations required, particularly in the later stages of the problem.

Convergence rates

As a convenient way of assessing the accuracy of the method we consider the velocity of the free surface in the neck region. We reason that this is a good test of overall accuracy since the neck is the region of maximum activity, both physically and in terms of the number of mesh–boundary modifications carried out. The *neck radius* is defined here as half the minimum distance between nodes lying on opposite sides of the neck, while *neck velocity* is defined as the rate of change of neck radius. In the later stages, when the neck has nearly disappeared, the choice of *neck nodes* is frozen for the remainder of the computation.

Figure 6 shows both the exact and computed neck velocities for mesh 3 as it evolves with time. The computed solution is in good agreement with the exact solution, with relative velocity errors of less than 2%, even for a mesh as coarse as mesh 3. The largest error in velocity is always observed to occur during the initial stages of the evolution when the neck curvature is at its greatest and the driving forces are largest. Figure 7 shows the computed neck velocities during the early stages of the problem, for each of the meshes employed. For the coarser meshes particularly, we observe a certain amount of *noise* superimposed on the computed neck velocity, which is perhaps understandable for a discrete simulation. In particular we note that the largest components of the noise arise when the domain is remeshed. It is clear however, that the level of noise decreases rapidly as we refine the mesh, and that its amplitude is a fraction of the error in the velocity, for all the meshes considered. Linear approximations to the neck velocity between times $t = 0.3$ and $t = 0.32$, were found using a least-squares fit and used to interpolate the neck velocities at the time $t = 0.31$. The resulting velocity errors are shown Table 3 and plotted against k_{tol} in Figure 8, from which it can be seen that the velocity errors converge with rate $O(k_{tol})$.

To understand why we achieve only an $O(k_{tol})$ rate of convergence (equivalently an $O(h)$ rate of convergence if the curvature k is understood to be constant) we must consider the way in which the boundary conditions are computed from the piecewise-discontinuous boundary representation employed. Since (35) is an identity in exact arithmetic, its use cannot affect the asymptotic convergence rate of the boundary conditions. Thus

boundary conditions computed using (35) can only ever have the same order of accuracy as the value of $k(s)$ that could be computed using the standard formula for the curvature, i.e.

$$k(s) = \frac{y_{ss}x_s - x_{ss}y_s}{({x_s}^2 + {y_s}^2)^{\frac{3}{2}}}. \tag{42}$$

Since the free surface is represented by the piecewise quadratic function $\mathbf{s}(s)$, accordingly y_{ss} is represented by a piecewise constant function. Thus even if x_s, y_s and x_{ss} are known exactly, the computation of $k(s)$ can never be better than $O(h)$ accurate for an edge of length h.

Higher-order schemes for computing approximations to k and $\mathbf{n}$ such as those employed in [14], which involve finite-difference stencils encompassing boundary nodes belonging to adjacent elements, are of course a possible alternative. In the absence of the necessary smoothing however, such formulas typically result in much less accurate values for k and $\mathbf{n}$ than the direct approach employed here.

A further potential problem that results from the low-order accuracy of the computed free-surface velocities is the tendency for free-surface instabilities to arise. Since our boundary conditions are $O(h)$ accurate, the computed free-surface velocities are similarly only $O(h)$ accurate. This causes problems since we are relying on these velocities to update an ideally $O(h^3)$–accurate boundary representation. While the effects of such errors in the boundary conditions tend to cancel out over longer ranges, locally they can lead to the free surface developing oscillatory structures, resulting in edges with incorrect curvatures and unnecessary discontinuities in the tangent at free-surface vertices. Where similar oscillatory phenomena are observed by other authors [9, 12] additional smoothing of the free-surface is commonly carried out. Since our method makes no *a priori* assumptions about the smoothness of the free-surface, it automatically refines the mesh in such regions, leading to the employment of unacceptably short time steps. Although this difficulty does not arise in the solution of the problems described here, there are some problems for which this is a serious limitation, where the semi-explicit method is employed, due to the effects on the time step.

Tangential stress error

As a further check on the accuracy of the method we consider the free-surface tangential stress as recovered from the discrete solution obtained using a mesh similar to mesh 3 at a time $t = 0.29$. If the boundary conditions described in Section 3 are being respected exactly, then the tangential stress component should be zero everywhere on the free surface. Figure 9 shows the recovered normal and tangential stresses in the neck region, as functions of arc length, the maximum dimensionless normal stress being approximately 85. As the figure shows, the tangential-stress error is not negligible, though it is everywhere a small fraction of the imposed normal stress.

The tangential-stress errors observed are a direct consequence of the piecewise-discontinuous nature of the free-surface representation employed. In order to understand the source of these errors, for simplicity, we focus on a single free-surface vertex on the boundary of a mesh that is symmetric about the normal at

the vertex, as shown in Figure 10. Since the mesh is symmetric about B, both the geometric and the mass-consistent normals coincide. The crucial point is that even if we start out with a mesh with no discontinuities in the tangent at vertices, we have no guarantee that such discontinuities will not arise as the free surface evolves. If there is continuity of the tangent at B then the normal $\mathbf{n}$ will be orthogonal to both the tangents $\mathbf{t}_1$ and $\mathbf{t}_2$. Thus the imposed normal stress will have no component parallel to the tangent on either adjacent edge. However, when the tangent is discontinuous at B, the imposed normal stress will have a non-zero component parallel to each of the two tangents at B, and tangential-stress errors will result. In the configuration shown, the resulting tangential stresses, on sides $\vec{AB}$ and $\vec{BC}$, resulting from an applied normal stress $\hat{\sigma}_n$ will be

$$\hat{\sigma}_n(\mathbf{n}\cdot\mathbf{t}_1) = \hat{\sigma}_n \sin(\varphi) \tag{43}$$

$$-\hat{\sigma}_n(\mathbf{n}\cdot\mathbf{t}_2) = -\hat{\sigma}_n \sin(\varphi) \tag{44}$$

i.e. the tangential-stress errors are of equal magnitude but opposite sign. At the vertex B the two errors cancel out, but along the two adjacent edges this does not happen. When the mesh is not symmetric about B the situation is more complicated and a net tangential-stress error can result. As Figure 11 shows, as the mesh is refined the observed tangential-stress error is reduced considerably. Theoretical considerations suggest that the tangential-stress error should depend linearly on k_{tol} and Figure 11 appears to confirm this.

Conservation of mass

As our final test we consider mass conservation. Figure 12 illustrates how the domain area varies with time in the case of mesh 3. The area changes slightly during the course of the problem, but never by more than 0.01%. For the present semi-explicit method the rate of mass loss is found to depend in an approximately linear fashion on: time step, free-surface velocity and the value of k_{tol}. Since the time step is typically constrained by (37) to be proportional to k_{tol}, the overall rate of mass loss for the semi-explicit method is thus inversely proportional to k_{tol}^2.

Three sources of mass conservation error are readily identifiable: errors due to inaccuracies in the solution of the linear systems, errors due to boundary modifications and errors due to the free-surface advection scheme. The first of these is easily dealt with by increasing the accuracy to which the linear systems are solved, resulting in a more exact imposition of the incompressibility constraint. The errors due to boundary modifications are also easily understood. Since conservation of mass is not directly enforced when we split, merge and adjust edges, there are necessarily small errors introduced whenever we modify the boundary. We could, of course, elect to impose mass conservation through our choice of midpoints when merging/splitting elements, but for the present we prefer the approach of trying to represent the boundary as accurately as possible, due to potential effects on the accuracy of boundary conditions. This source of error can only be reduced by taking a finer discretisation of the boundary. Contrasting the piecewise boundary-update strategy described in Section 2 with the alternative approach of periodically remeshing the entire boundary [14], we

note that our method has the advantage that it introduces mass conservation errors only at those points where the boundary is actually modified. A further interesting comparison can be made with methods, such as that described in [22], which impose conservation of mass directly by replacing the kinematic boundary condition with a free-surface update procedure based on volume-of-fluid considerations. These methods, in principle, impose conservation of mass exactly and must, as a consequence, impose the kinematic boundary condition only approximately. Furthermore, volume-of-fluid methods are applicable only to problems in which an appropriate conservation law, such as incompressibility, applies. The final source of mass conservation error results from the time-stepping scheme employed. It is clear that for an explicit scheme, even if the solution at the start of a time step satisfies the incompressibility constraint, it will not in general do so at the end of the time step. This source of error appears unavoidable, though it can be reduced by employing finer meshes and shorter time steps. For an implicit time-stepping scheme, on the other hand, the linearity of the incompressibility constraint means that this source of error is absent and consequently the only significant source of mass conservation error will be due to the occasional splitting, merging and adjustment of edges.

A further example

In principle there is nothing to prevent the methods described in Section 2 from being applied to a broad range of free-surface problems. Our initial experimental investigations suggest that the inclusion of inflow, outflow and no-slip boundary conditions is unproblematic, so long as the constraints (38) and (39) are appropriately modified. When selecting initial free-surface configurations, care must however be taken when the derivative k_s is large, or k is discontinuous.

As a further example of the general applicability of the current method, we consider the evolution of the domain shown in Figure 13, starting at the dimensionless time $t = 0.0$. Note that for convenience in the generation of the initial mesh we have used a smaller value of h_{max} than that used for the remainder of the problem. The boundary has been constructed by taking a cross shape and replacing the interior and exterior corners with short circular arcs of radius R, which in principle could be made arbitrarily small. While the curvature along the straight sides is zero, at the rounded corners it suddenly jumps to the value $\pm\frac{1}{R}$. Thus, initially at least, $\frac{dk}{ds}$ is unbounded at the points where straight sides meet circular arcs. As the mesh evolves, the initial discontinuities in k smear out, resulting in extended regions of the free surface where $\frac{dk}{ds}$ is large. We could, of course, avoid the initial discontinuities entirely by employing fourth-order splines rather than circular arcs. This problem illustrates well the sort of configuration which would be difficult to model using the method of spines [1], but which our new method handles with ease and with minimal set-up time.

Conclusions

The strategy of using an elastic deformation of the mesh whenever possible and performing a full remesh only when necessary appears to work satisfactorily; in practice the technique is robust and the choice of the constants employed is not critical. The approach is economical of the user's time since the setting up of a new problem involves only the selection of an initial boundary mesh. The semi-explicit method would appear to be an attractive solution strategy for the type of free-surface problem considered here — at least in the absence of extremes of curvature.

Experiment confirms the theoretical prediction that for piecewise-quadratic isoparametric elements the stress boundary conditions are $O(h)$ accurate and consequently so are the computed boundary velocities. In our searches of the available literature we have been unable to find numerical evidence which contradicts this finding.

The methods described here constitute a general-purpose tool for investigating free-surface flow problems with arbitrary geometry, complementing the existing BEM and FEM techniques, where they are applicable, and opening up lines of investigation for problems to which existing methods are not applicable.

Acknowledgement

RCP gratefully acknowledges the financial support of both the School of Computer Studies and the Department of Applied Mathematics at the University of Leeds.

References

[1] S.F. Kistler and L.E. Scriven. Coating flows. In J.R.A. Pearson and S.M. Richardson, editors, *Computational Analysis of Polymer Processing*, chapter 8, pages 243–299. Applied Science Publishers, London, 1983.

[2] R.M.M. Mattheij and G.A.L. van de Vorst. Mathematical modelling and numerical simulations of viscous sintering processes. Technical Report RANA 95-14, Eindhoven University of Technology, 1995.

[3] A.V. Anilkumar T.G. Wang and C.P. Lee. Oscillations of liquid drops: Results from USML-1 experiments in space. *Journal of Fluid Mechanics*, 308:1–14, 1996.

[4] H.A. Stone. Dynamics of drop deformation and breakup in viscous fluids. *Annual Review of Fluid Mechanics*, 26:65–102, 1994.

[5] Jae-Tack Jeong and H.K. Moffatt. Free-surface cusps associated with flow at low Reynolds number. *Journal of Fluid Mechanics*, 241:1–22, 1992.

[6] A. Prosperetti. Free oscillations of drops and bubbles: The initial-value problem. *Journal of Fluid Mechanics*, 100:333–347, 1980.

[7] H.M. Ettouney and R.A. Brown. Finite-element methods for steady solidification problems. *Journal of Computational Physics*, 49:118–150, 1983.

[8] R.A. Brown. Finite-element methods for the calculation of capillary surfaces. *Journal of Computational Physics*, 33:217–235, 1979.

[9] M.S. Longuet-Higgins and E.D. Cokelet. The deformation of steep surface waves on water 1. a numerical method of computation. *Proceeding of the Royal Society of London, Series A*, 350:1–26, 1976.

[10] T.B. Benjamin and A.T. Ellis. The bifurcation of liquid bridges. *Journal of Fluid Mechanics*, 212:25–39, 1990.

[11] Xiaofan Li and C. Pozrikidis. Shear flow over a liquid drop adhering to a solid surface. *Journal of Fluid Mechanics*, 307:167–190, 1996.

[12] H.K. Kuiken. Viscous sintering: The surface-tension-driven flow of a liquid form under the influence of curvature gradients at its surface. *Journal of Fluid Mechanics*, 214:503–515, 1990.

[13] G.A.L. van de Vorst and R.M.M. Mattheij. Numerical analysis of a 2-D viscous sintering problem with non-smooth boundaries. *Computing*, 49:239–263, 1992.

[14] G.A.L. van de Vorst. *Modelling and Numerical Simulation of Viscous Sintering.* PhD thesis, Eindhoven University of Technology, 1994.

[15] E.B. Hansen and M.A. Kelmanson. Steady viscous free surface flow on a rotating cylinder. *Journal of Fluid Mechanics*, 272:91–107, 1994.

[16] M.S. Carvalho and L.E. Scriven. Multiple states of a viscous free surface flow: Transition from pre-metered to a metering inflow. *International Journal for Numerical Methods in Fluids*, 24:813–831, 1997.

[17] J.L. Summers P.H. Gaskell, M.D. Savage and H.M. Thompson. Modelling and Analysis of Meniscus Roll Coating. *Journal of Fluid Mechanics*, 298:113–137, 1995.

[18] J.I. Martinez-Herrera and J.J. Derby. Analysis of capillary-driven viscous flows during the sintering of ceramic powders. *AIChE Journal*, 40(11):1794–1803, 1994.

[19] O.A. Basaran. Nonlinear oscillations of viscous liquid drops. *Journal of Fluid Mechanics*, 241:169–198, 1992.

[20] R.W. Yeung. Numerical methods in free-surface flows. *Annual Review of Fluid Mechanics*, 14:395–442, 1982.

[21] W.A. Miller J.W. Ross and G.C. Weatherly. Dynamic computer simulation of viscous flow sintering kinetics. *Journal of Applied Physics*, 52(6):3884–3888, 1981.

[22] F. Mashayek and N. Ashgriz. A spine-flux method for simulating free surface flows. *Journal of Computational Physics*, 122:367–399, 1995.

[23] P.K. Jimack and A.J. Wathen. Temporal derivatives in the finite-element method on continuously deforming grids. *SIAM Journal on Numerical Analysis*, 28(4):990–1003, 1991.

[24] R.W. Hopper. Plane Stokes flow driven by capillarity on a free surface. *Journal of Fluid Mechanics*, 213:349–375, 1990.

[25] A.C. Hindmarsh P.N. Brown and L.R. Petzold. Using Krylov methods in the solution of large-scale differential-algebraic systems. *SIAM Journal on Scientific Computation*, 15(6):1467–1488, 1994.

[26] J.N. Reddy. *An Introduction to the Finite Element Method*. McGraw-Hill Book Company, 1984.

[27] J.R. Shewchuk. Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. In *First Workshop on Applied Computational Geometry*, pages 124–133. Association for Computing Machinery, May 1996.

[28] J. Ruppert. A Delaunay refinement algorithm for quality 2-dimensional mesh generation. *Journal of Algorithms*, 18(3):548–585, 1995.

[29] D.R. Lynch. Unified approach to simulation on deforming elements with application to phase change problems. *Journal of Computational Physics*, 47:387–411, 1982.

[30] G. Strang and G.J. Fix. *An Analysis of the Finite Element Method.* Prentice-Hall, Inc., New Jersey, 1973.

[31] A. Jagota and P.R. Dawson. Simulation and viscous sintering of two particles. *Journal of the American Ceramic Society*, 73(1):173–177, 1990.

[32] M.D. Gunzburger. *Finite Element Methods for Viscous Incompressible Flows.* Academic Press, San Diego, California, 1989.

[33] P.M. Gresho. Some current CFD issues relevant to the incompressible Navier-Stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 87:201–252, 1991.

[34] K.J. Ruschak. A method for incorporating free boundaries with surface tension in finite element flow simulators. *International Journal for Numerical Methods in Engineering*, 15:639–648, 1980.

[35] R.L. Sani M.S. Engelman and P.M. Gresho. The implementation of normal and/or tangential boundary conditions in finite element codes for incompressible fluid flow. *International Journal for Numerical Methods in Fluids*, 2:225–238, 1982.

[36] A. Ramage and A.J. Wathen. Iterative solution techniques for the Stokes and Navier-Stokes equations. *International Journal for Numerical Methods in Fluids*, 19:67–83, 1994.

[37] A. Wathen and D. Silvester. Fast iterative solution of stabilised Stokes systems. Part 1: Using simple diagonal preconditioners. *SIAM Journal on Numerical Analysis*, 30(3):630–649, 1993.

[38] T.A. Manteuffel S.F. Ashby and P.E. Saylor. A taxonomy for conjugate gradient methods. *SIAM Journal on Numerical Analysis*, 27(6):1542–1568, 1990.

List of Tables

List of Figures

Constant	Function	Minimum value	Value employed	Maximum value
α	Initial boundary smoothness parameter	1.20	1.50	2.00
β	Isoparametric displacement tolerance	1.05	1.10	1.20
δ	Aggression factor when splitting edges	0.70	0.90	0.95
μ	Caution factor when merging edges	0.70	0.90	0.95
ρ	Boundary mesh smoothness tolerance	2.20	2.50	3.00
θ	Minimum interior angle	5.00	10.00	15.00

Table 1: Constants employed in the adaptive mesh generator

Mesh	k_{tol}	h_{max}	Boundary vertices	Elements	Unknowns
1	0.2000	0.3218	112	876	4228
2	0.1414	0.2867	148	1132	5470
3	0.1000	0.2554	200	1464	7094
4	0.0707	0.2276	272	1972	9560
5	0.0500	0.2027	380	2714	13169

Table 2: The coalescence of two infinite cylinders — initial mesh statistics

1. Set up initial distribution of boundary nodes.
2. Mesh the domain.
    3. Solve for velocity and pressure fields at start of time step.
    4. If evolution complete then stop.
    5. Update the free surface explicitly using velocity field.
    6. If boundary nodes need inserting, deleting or adjusting then go to 9.
    7. Solve elastic mesh problem and relocate interior nodes accordingly.
    8. Go to 3.
9. Merge boundary edges.
10. Split boundary edges.
11. Adjust boundary edges.
12. Remesh the domain.
13. If necessary interpolate previous solution(s) onto new mesh.
14. Go to 3.

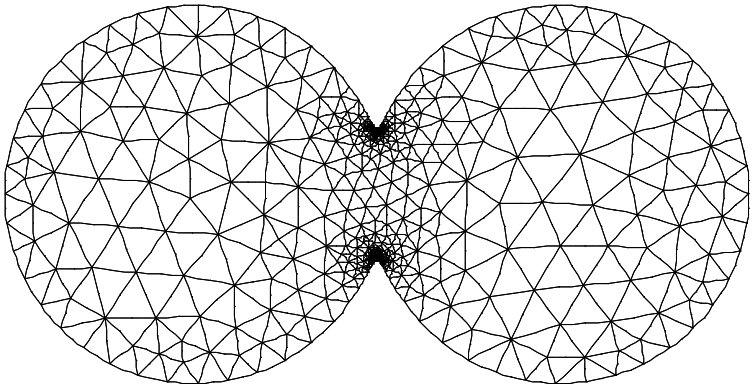
Figure 1: Outline of the semi-explicit scheme for free-surface flow problems

Mesh	Iterations	Steps	Remeshes	Average time per step
1	1549	1548	54	23s
2	1767	1766	149	44s
3	2086	2085	71	69s
4	2649	2648	77	104s
5	3384	3383	130	221s

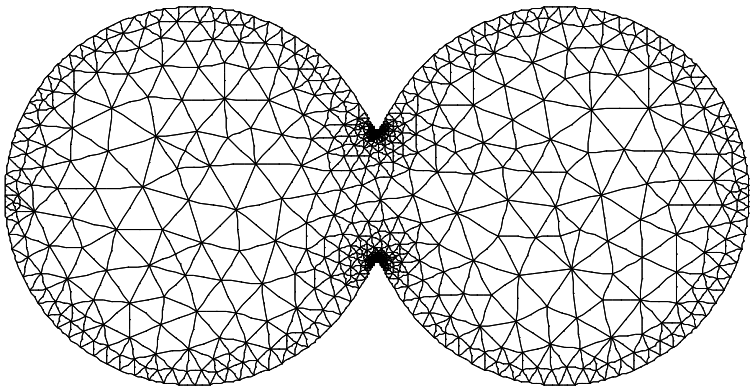
Table 3: The coalescence of two infinite cylinders — run statistics

Mesh	k_{tol}	Neck velocity error
1	0.2000	0.0272
2	0.1414	0.0189
3	0.1000	0.0127
4	0.0707	0.0089
5	0.0500	0.0061

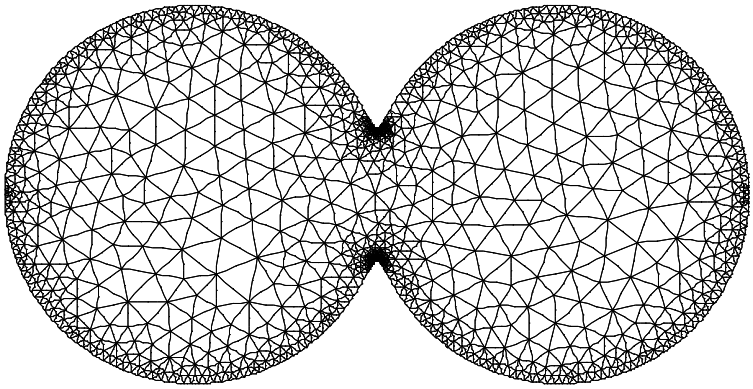
Table 4: The coalescence of two infinite cylinders — neck velocity error at $t = 0.31$



(a) Mesh 1

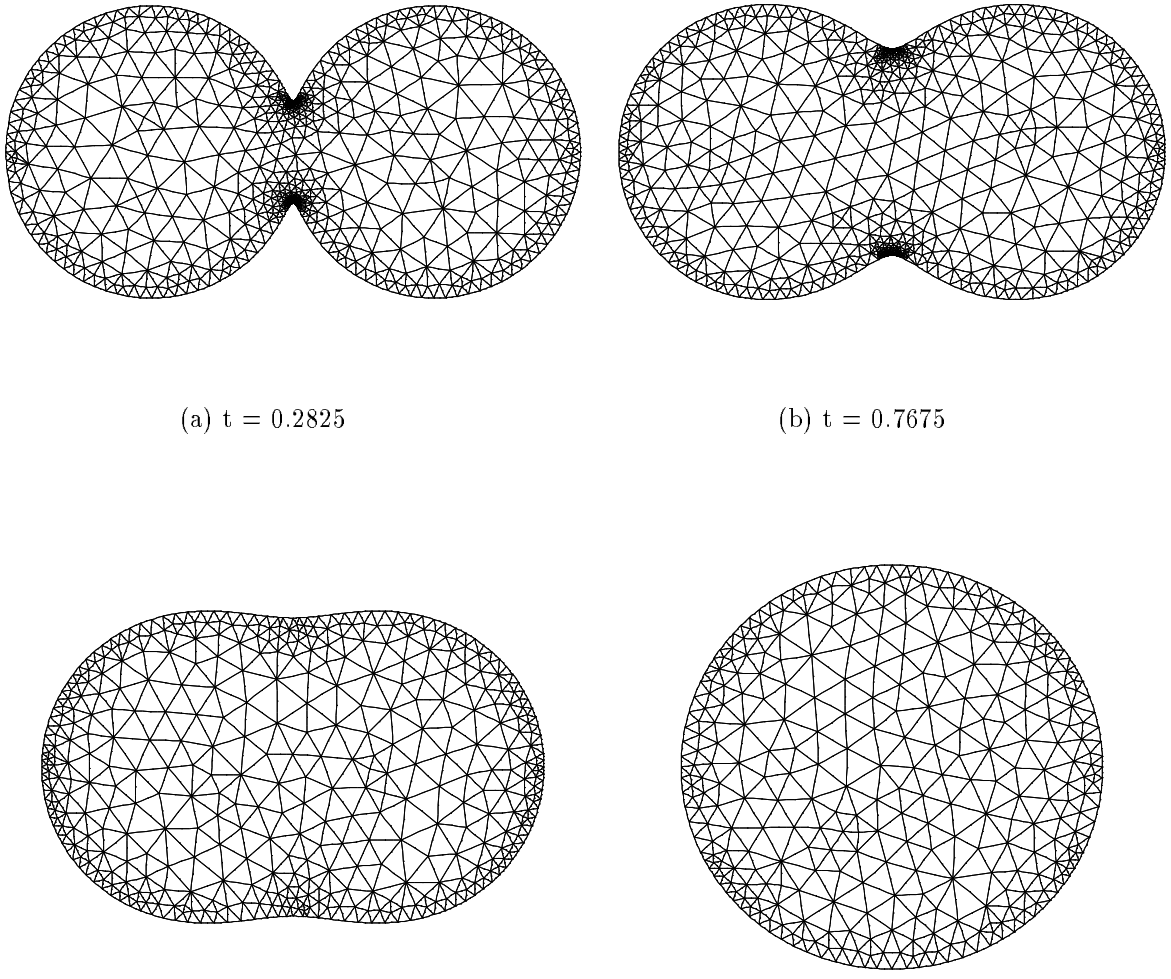


(b) Mesh 3



(c) Mesh 5

Figure 2: The coalescence of two infinite cylinders — selected initial meshes: (a) $k_{tol} = 0.2000$, $h_{max} = 0.3218$, (b) $k_{tol} = 0.1000$, $h_{max} = 0.2554$, (c) $k_{tol} = 0.0500$, $h_{max} = 0.2027$.



(a) t = 0.2825

(b) t = 0.7675

(c) t = 1.5027

(d) t = 5.0000

Figure 3: The coalescence of two infinite cylinders — evolution of mesh 3

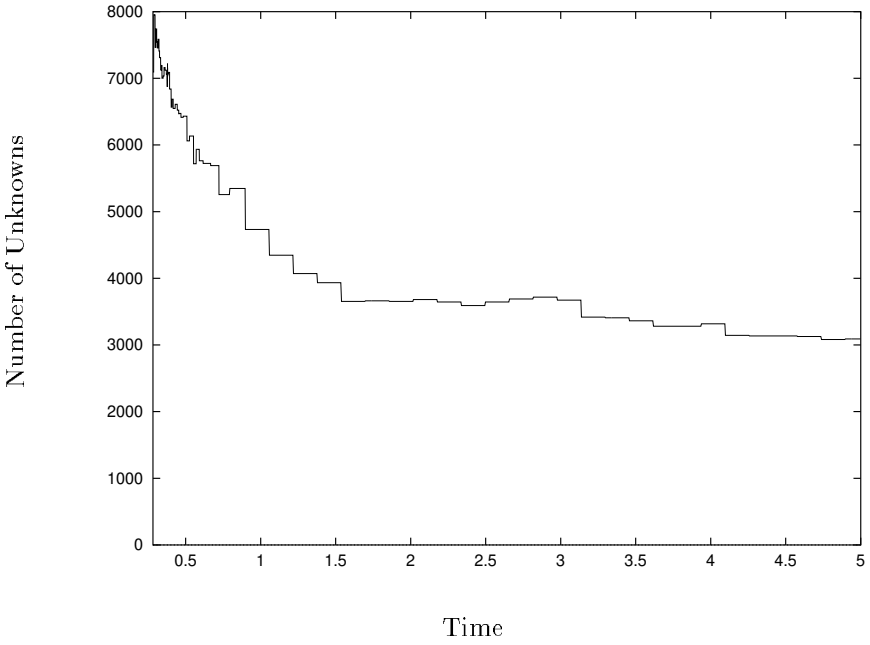


Figure 4: Number of unknowns as a function of time — mesh 3

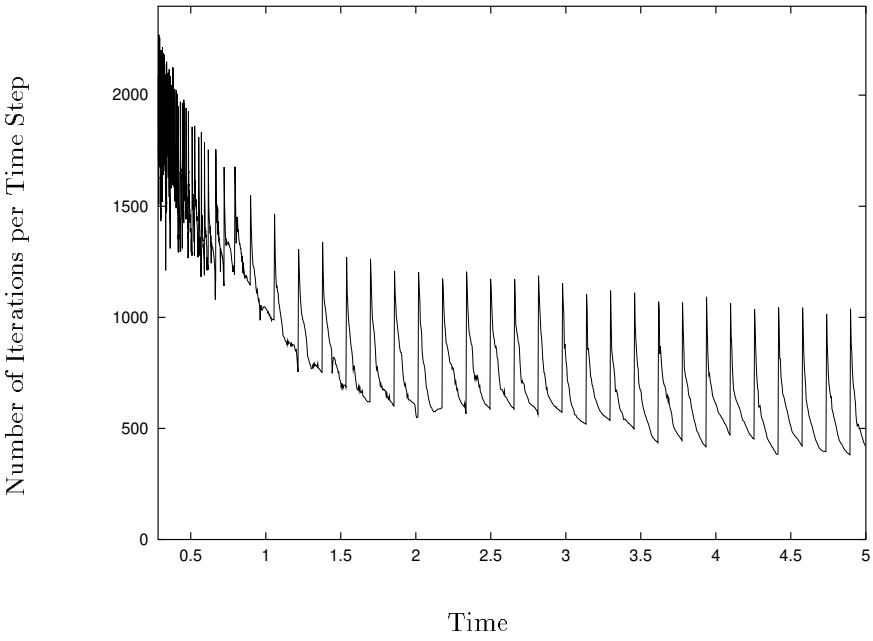


Figure 5: Number of Orthomin iterations per time step as a function of time — mesh 3

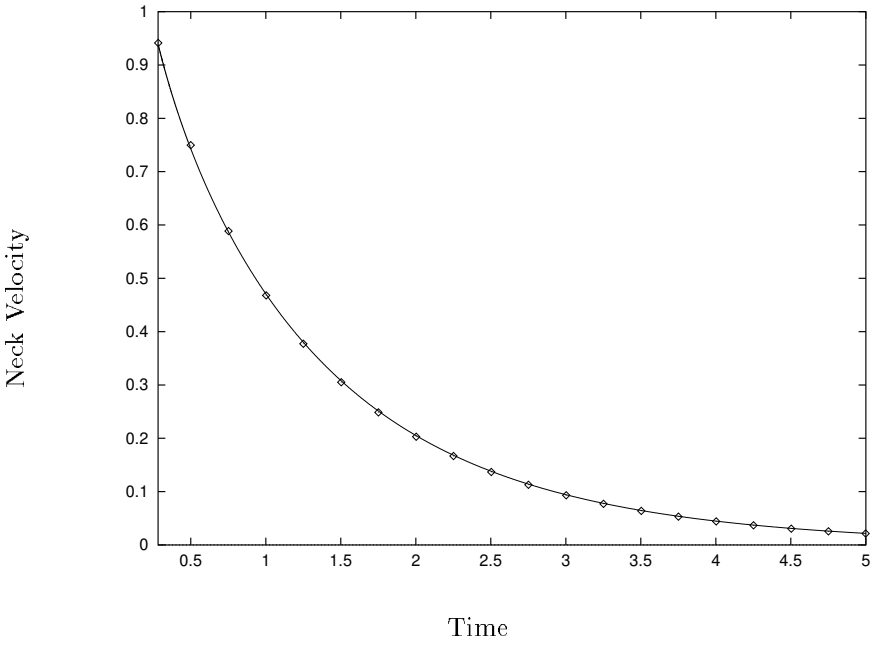


Figure 6: Neck velocity as a function of time for mesh 3: exact — , computed ◇

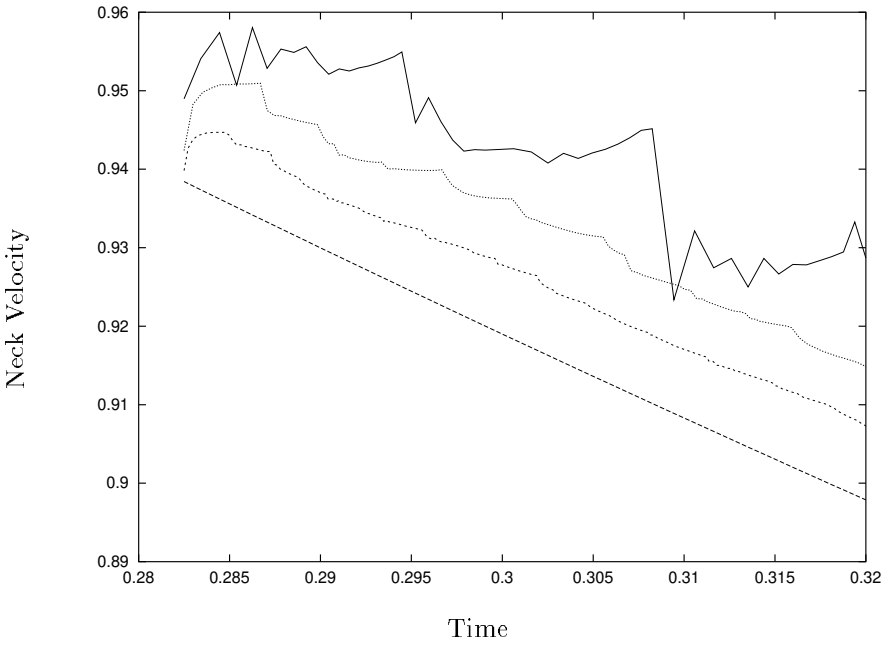


Figure 7: Initial stage velocities: mesh 1 — , mesh 3 . . . , mesh 5 - - - , exact – – –

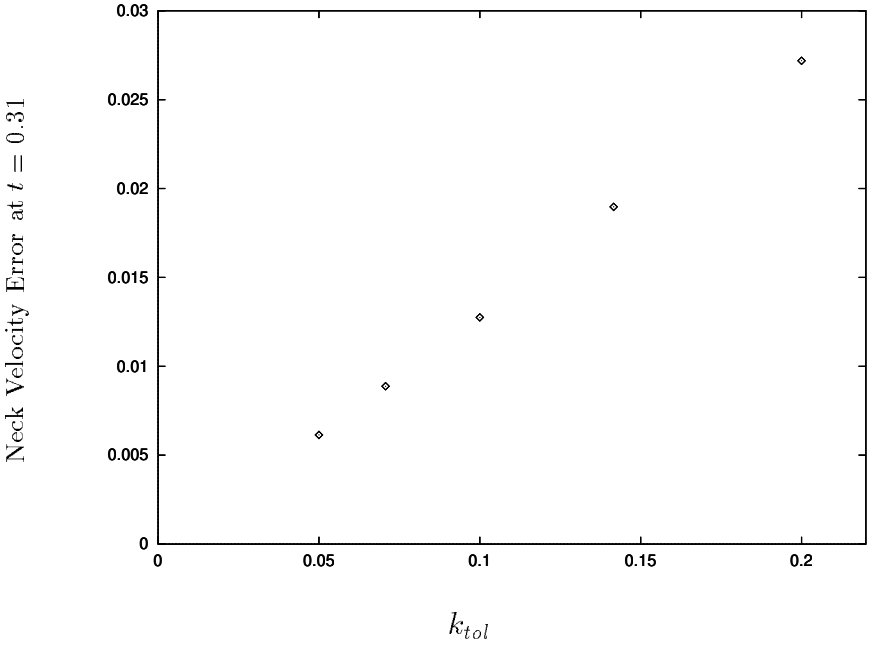


Figure 8: Convergence rate: neck velocity error at $t = 0.31$ as a function of k_{tol}

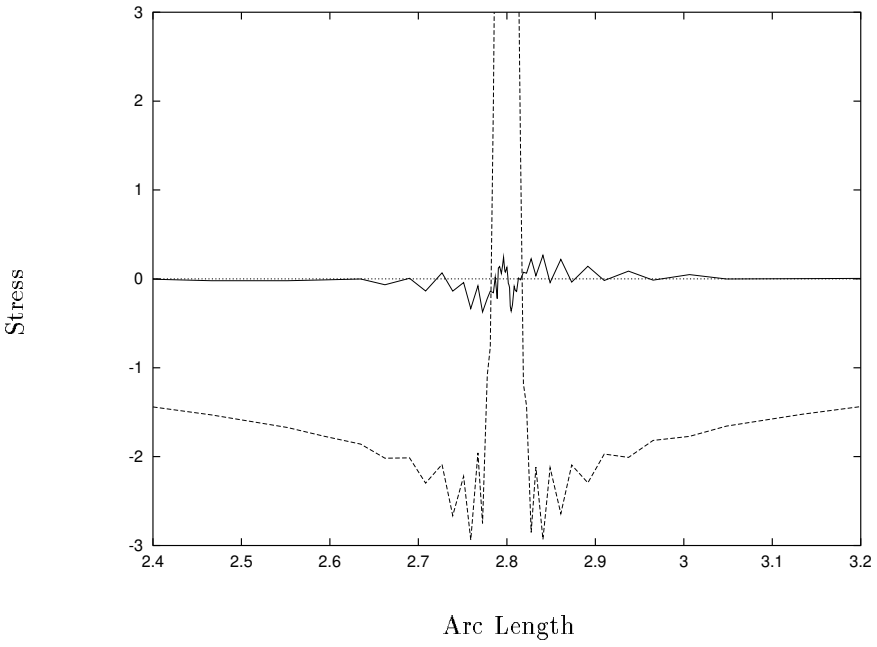


Figure 9: Normal and tangential stress in the neck region at $t = 0.29$: normal stress – – – , tangential stress —

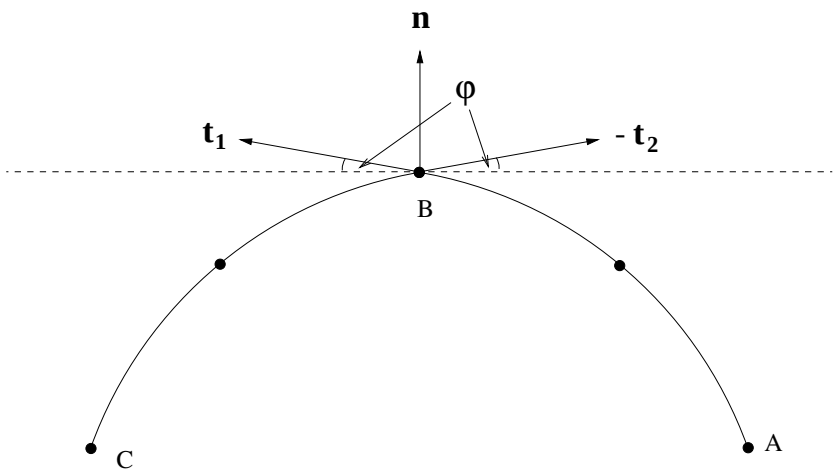


Figure 10: Discontinuity in the tangent at a free–surface vertex B: $\mathbf{n}$ = normal, $\mathbf{t_1}$ = tangent on edge $\vec{AB}$, $\mathbf{t_2}$ = tangent on edge $\vec{BC}$

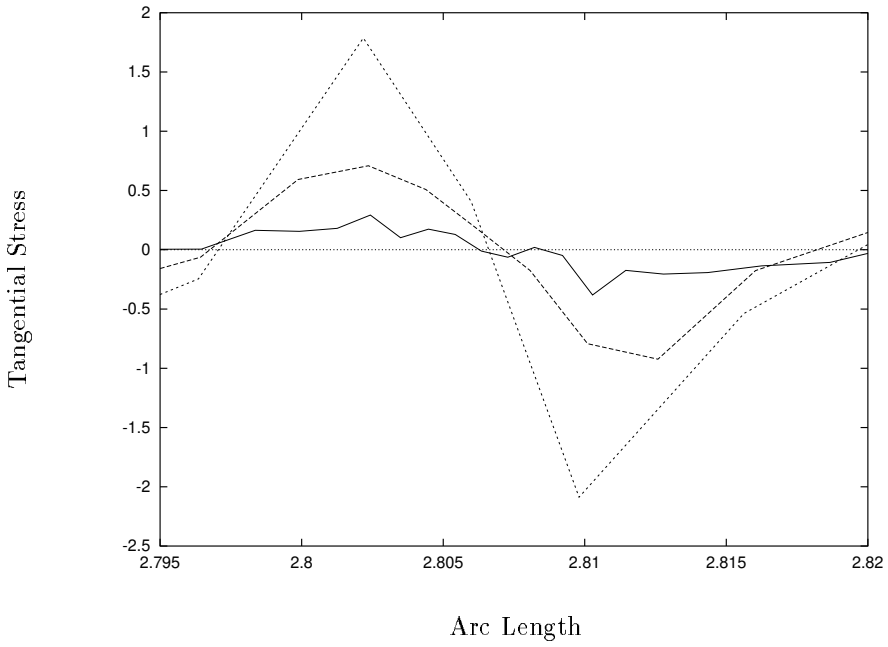


Figure 11: Tangential stress — centered on the neck region at $t = 0.29$: mesh 1 - - - , mesh 3 – – – , mesh 5 —

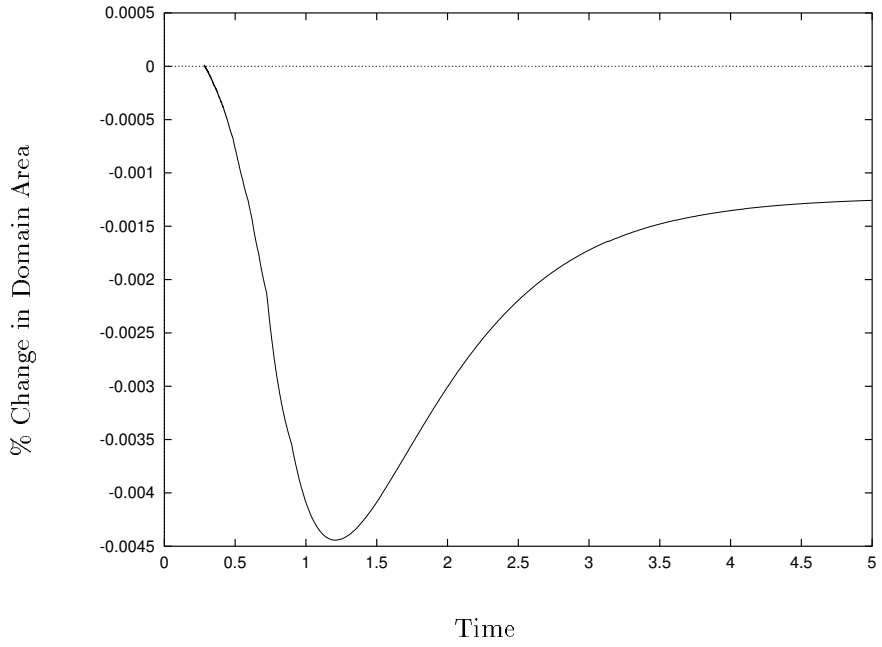
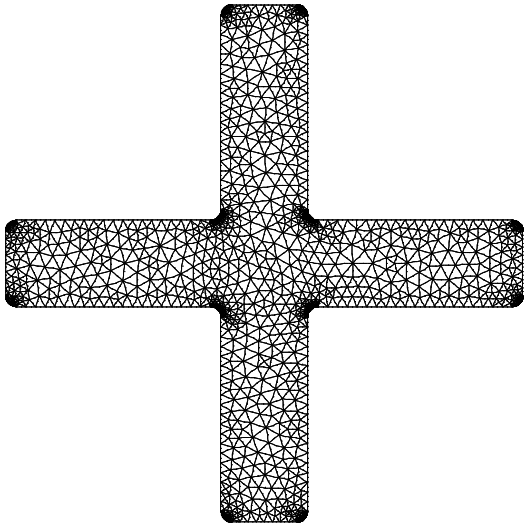
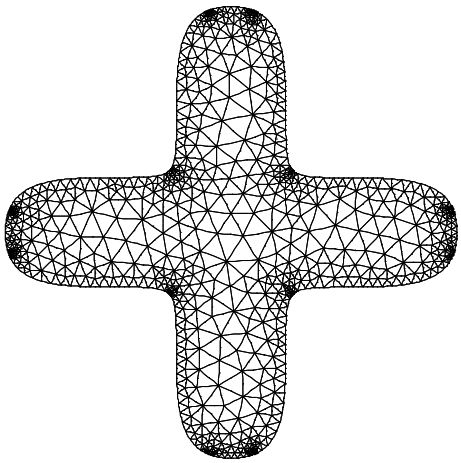


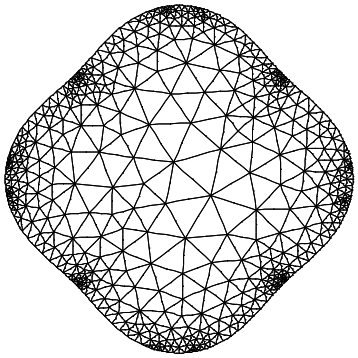
Figure 12: Conservation of mass: percentage change in domain area as a function of time — mesh 3



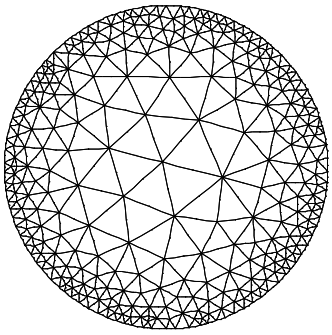
(a) t = 0.0000



(b) t = 0.1069



(c) t = 0.5041



(d) t = 1.1021

Figure 13: Stokes-flow evolution of a cross with rounded corners